A tool for holistic optimization of mechatronic design concepts

Daniel Malmquist

Doctoral Thesis
KTH Royal Institute of Technology
School of Industrial Engineering and Management
Department of Machine Design
SE -100 44 Stockholm, Sweden
Academic thesis, which with the approval of KTH Royal Institute of Technology, will be presented for public review in fulfillment of the requirements for a Doctorate of Technology in Machine Design. The public review is held in Gladan, Brinellvägen 85, KTH Royal Institute of Technology, 100 44, Stockholm on 2015-01-30 at 13:00

Cover illustration by Pietro Baioni

© Daniel Malmquist, December 2014

Print: Universitetsservice US AB
Abstract

Designing mechatronic systems is challenging. Even so, mechatronic products are regarded as one of the most important means in order to innovate in many large industries, e.g. automobile and aerospace. The inherent difficulty lies in the multiple engineering domains involved and how these are treated during the development process. A holistic approach, which treats these domains concurrently, is needed in order to find hidden synergies between domains and in the end find an optimal product design based on given requirements.

Traditional product development methodologies, even ones with a mechatronic design focus, rely on treating the individual domains separately and only integrating them at a point in time rather far into the detailed design phase of the development process. In addition, in order to solve problems that arise in later design stages, the design engineers need to backtrack to earlier stages and in the worst case repeat substantial parts of the work. This is both time and cost inefficient.

This dissertation summarizes and extends previously published research by the author about a novel mechatronic design methodology and a supporting software tool. The goal of the design methodology is to enable design of better products, in terms of e.g. development cost, size, and sustainability, by finding synergies and reducing backtracking through better analysis of system concepts. The methodology relies on using an optimizer to efficiently determine the potential of a system concept, described by combining a number of component models, e.g. motors, transmissions, and structural ones, from a component library and specifying their parameters.

A number of design cases have been evaluated, some of which are presented in this dissertation, including both one where a physical prototype is built, and a few virtual ones. These design cases have been instrumental in evaluating the methodology as well as the software tool.

It is concluded that the design methodology is able to properly evaluate a concept against competing concepts, and that it is a useful addition to existing methodologies. However, as always, a number of improvements are possible, some of which are presented in a concluding section. After all, as Leonardo da Vinci once said: “Art is never finished, only abandoned”.
Acknowledgements

Looking back I realize that at the time I decided to pursue a Ph.D. I had no clue of what it would entail. I spent a good part of the first year trying to figure it out. It has at times been challenging, and at times been great fun. Nevertheless, I am glad I decided to start this journey; I have gained invaluable experiences and I have had a great time doing so.

I would like to thank my advisor, Jan, for giving me this opportunity as well as for patiently providing feedback on my research and publications during these years. I would also like to thank Martin G who has been a good mentor and sounding board to me since even before I started my doctoral studies.

I started at machine design together with four other guys, Daniel, Mohammad, Fredrik, and Sagar, and I would like to express my gratitude to them for putting up with my sometimes odd sense of humor, our discussions about everything and nothing, and not least for providing valuable feedback on my research. I’m happy to have had the opportunity to work with so many of my colleagues at the Mechatronics division.

During the third year of my doctoral studies I spent five months at Stanford University. I’m grateful for how welcoming everyone at the Dynamics Design Lab was towards me, especially Avinash and Steve who I spent a lot of my time working with.

These five years would never have been possible without the support from all my friends outside academia. I’m truly grateful for all the great memories!

Finally, I want to express how thankful I am for the unconditional love and support from my family. Jessica, Mum, Dad, Johan, Fredrik, Andréa, and Sidney, You are all amazing!

Daniel Malmquist
Stockholm, 10/12 – 2014
List of appended publications

Paper A

Paper B

Paper C

Paper D

Paper E
List of additional publications


Terminology

Some of the terms used in this dissertation might differ depending on the context they are used in and who they are used by, or might not be common knowledge. Hence, in order to form a common starting point for the reader, a number of important terms are introduced here.

**By-wire** – A term commonly used in automotive and aerospace industry to describe a system which uses electromechanical actuators controlled over an electric signal wire, in contrast to conventional hydraulic or mechanical systems. A drive-by-wire system, for instance, uses an electromechanical actuator to steer a wheel according to a control signal from e.g. a haptic steering wheel.

**CAD** – Computer Aided Design. A collection term for software tools which can be used in order to aid design. Different CAD tools could e.g. support the user while designing mechanical components or circuit boards.

**Design methodology** – A process, usually consisting of a number of distinct design stages (phases), used to design a product. Design methodologies commonly include stages for concept development, detailed design, testing, and production ramp up.

**Design variable** – A design variable is a parameter, e.g. a dimension or a gear ratio, which the optimizer controls in order to find the optimal solution. A design variable is usually constrained to bounds determining which values are allowed.

**Engineering domain** – An engineering discipline, e.g. mechanical engineering, control engineering, or electrical engineering.

**FEM** – Finite Element Method. Numerical method commonly used to analyze for instance structural characteristics and fluid dynamics.

**Fitness function** – See objective function.
**Fitness value** – The resulting value from an objective function; i.e. a value describing how well the evaluated system fulfills the optimization objective, e.g. the size of a concept system design.

**Genetic algorithm** – An optimization / search heuristic that mimics natural selection. Sometimes also referred to as evolutionary algorithm.

**Haptic device** – Device providing feedback to its user through touch by e.g. applying a force. For instance used for surgical simulators or by-wire vehicles.

**Harmonic drive** – A type of gearing distinguished by very high single stage gear ratios and close to zero backlash.

**Mechatronics** – A multidisciplinary engineering field integrating electrical, mechanical, and control engineering.

**Objective Function** – A function describing the goal of an optimization and whose output is the fitness value. Could for instance be a mathematical function or an algorithm.
## Contents

Abstract V  
Acknowledgements VII  
List of appended publications IX  
List of additional publications XI  
Terminology XIII  
Contents XVII  

1. Introduction 1  
  1.1. Motivation 1  
    1.1.1. Application domain example 3  
  1.2. Problem formulation 5  
    1.2.1. Research questions 6  
  1.3. Research approach 6  
    1.3.1. Phase 1 6  
    1.3.2. Phase 2 8  
    1.3.3. Phase 3 8  
    1.3.4. Phase 4 8  
    1.3.5. Phase 5 8  
  1.4. Summary of appended papers 9  
    1.4.1. Paper A 9  
    1.4.2. Paper B 9  
    1.4.3. Paper C 10  
    1.4.4. Paper D 10  
    1.4.5. Paper E 10  
  1.5. Dissertation outline 10  

2. Background and related research 13  
  2.1. General product development 13  
  2.2. Mechatronic product development 14  
  2.3. Optimization methods 19
3. Design methodology

3.1. Context of methodology

3.2. System modeling

3.3. Component Modeling
   3.3.1. Static property models
   3.3.2. Dynamic behavior models

3.4. Component model examples
   3.4.1. PMDC-Motor
   3.4.2. Planetary gear
   3.4.3. Harmonic drive
   3.4.4. Solid shaft
   3.4.5. Linear timing belt drive
   3.4.6. Load
   3.4.7. Full state feedback controller

3.5. Software tool

4. Design examples

4.1. Haptic steering wheel

4.2. Two-axis gantry system

5. Discussion and conclusions

5.1. Conclusions

5.2. Discussion and future work
   5.2.1. Methodology
   5.2.2. Component models
   5.2.3. Design cases

References

Appended papers
Chapter 1
Introduction

“A great product isn't just a collection of features.
It’s how it all works together.”
— Tim Cook, 2014

It is difficult to design mechatronic systems. Even though mechatronic systems are a fundamental part of a large fraction of all modern products today, little is known about how to make the most out of them. A core advantage, and challenge, of mechatronic systems design is the possibility to map product features to the different engineering domains involved, i.e. electrical, mechanical, and control engineering. However, in order to do this, and make use of possible synergies between domains, a holistic design approach, where these are treated concurrently, is needed. Traditional mechatronic design methodologies, such as the VDI 2206 [1] V-Model, rely on treating the involved engineering disciplines separately, hence only integrating them in late design stages. It is easy to see why this generally results in suboptimal products.

This dissertation presents a new mechatronic design methodology, as well as a supporting software tool, capable of concurrent model and optimization based mechatronic systems design in early development process stages.

This chapter introduces the background, problem formulation, goal, and method of the research presented in this dissertation.

1.1. Motivation

In order for a company to stay competitive and assure its survival within a given field it needs to continuously evolve its products through new innovations. Mechatronics is widely regarded as an important
Innovation by itself is nonetheless not sufficient to assure a successful product. If the product cannot be developed and manufactured at a reasonable cost, it will not be competitive on the market. To achieve success, most conventional design methodologies make use of iterations, i.e. backtracking from later design stages to earlier ones if and when problems arise. This is a time, and hence cost, expensive practice. Minimizing the amount of iterations needed could significantly decrease the time to market for a product.

In addition, it is common that as much as 60-80% of the total product cost is committed already during the concept stage of the design process [4], [5]. This is especially problematic since this is the point of time in the design process where the least information about the design task is known. Figure 1.1 illustrates this paradoxical fact, i.e. the design freedom decreases throughout the design process while the knowledge about the design task increases. Due to these facts, a design methodology focusing on this early concept stage, front-loading certain design activities to it, has a much larger impact potential than the ones focusing mainly on the
detailed design stage.

It is important to note that many different factors matter while designing modern day products, not only functionality, performance and cost. Examples of important factors are for instance:

- Eco-sustainability
- User-friendliness
- Lifespan
- Size

It can be challenging to intuitively understand the dependencies between different design goals. Hence, a design method needs to not only support and handle multiple engineering domains concurrently, but also multiple design objectives.

Mørkeberg Torry-Smith et al. [6] presents a study on the challenges of designing mechatronic systems. Their findings back the statements made in this section by concluding that current attempts at supporting assessment of multi-disciplinary concepts are insufficient and that new, modern mechatronic design methods with supporting software tools are needed to find cross-domain synergies while taking all the important factors into account.

1.1.1. Application domain example

As mentioned above, the application of mechatronics is widespread in many industries, e.g. aeronautics, automotive, automation, robotics, and home appliances. The vehicle industry for instance has for a long time been moving towards an increasing amount of electromechanical systems in cars and trucks. This change is motivated by, among other things, demands on fuel efficiency, and the move towards alternative drive lines, such as hybrid, electrical, and fuel cell based ones. Traditional auxiliary systems, for instance, often require the engine to be running in order to operate; an assumption which cannot be made in modern vehicles. An example of such a system is hydraulic power steering, commonly substituted by electric power steering (EPS) in modern vehicles. Another example is electronic brake systems where an electric signal has been added in addition to the conventional pneumatic or hydraulic one in order to improve the dynamic response. Some manufacturers have even taken one step further, removing the conventional systems still used for
backup all together, replacing them with purely electro-mechanical drive-by-wire systems. Many modern vehicles, for instance, make use of throttle by wire systems. Examples of advantages with these new by-wire systems are increased design freedom and higher performance. The Nissan Motor Company introduced the first steer-by-wire production vehicle through its luxury brand Infiniti in 2013. The car, however, relies on a mechanical backup that can be enabled or disabled through a clutch, hence limiting some of the advantages with by-wire systems.

The research presented in this dissertation relate to the automotive industry in the sense that two of the design cases investigated are based on the design of haptic force-feedback steering wheels for two prototype drive-by-wire cars. The cars are named X1 and RCV and have been developed, respectively, by the Dynamic Design Lab research group at Stanford University and researchers at KTH Royal Institute of Technology, including the author of this thesis. Both vehicles have four wheel steer-by-wire control and the RCV is additionally capable of four wheel drive and wheel camber control. The prototype cars can be seen in Figure 1.2 and Figure 1.3.

Figure 1.2. Stanford University X1 drive by wire prototype vehicle.
1.2. Problem formulation

With the motivation given in the previous section in mind, the goal of this research is to:

**G1:** Support deeper analysis of conceptual mechatronic systems as a means to improve product development efficiency. In order to achieve this, a novel design methodology for mechatronic systems, with a supporting software tool, is to be developed. The methodology should treat mechanical, electrical, and control engineering domains as well as multiple design objectives in a concurrent manner. Furthermore, application of the method needs to be straightforward and quick, in order to facilitate the evaluation of many different concepts with small effort, such that a sufficient decision basis for further work can be efficiently produced.

**G2:** Verify the developed methodology and tool through design cases whereof at least one should include development of a physical prototype within the automotive field.
1.2.1. Research questions

A number of research questions with their foundation in the given goals, form the basis of this dissertation.

Q1: How can mechanical, electrical, and control engineering domains be treated concurrently in order to evaluate the potential of a mechatronic system without increasing the design effort to an unmanageable level?

Q2: How can a software tool be designed and how can an optimizer be used to support efficient evaluation?

Q3: How can the static properties and the dynamics of a mechatronic system be concurrently evaluated in a manner orders of magnitude faster than current approaches relying on simulation based analysis?

1.3. Research approach

The overall plan for this research, as seen in Figure 1.4, was developed in order to achieve the scientific goals described in the previous section. The work was divided into five phases, each lasting approximately one year. Each phase has its own separate focus area and starts with a literature study, ends with a design case, and has one or more deliverables (i.e. publications, tool, designs, and prototype).

1.3.1. Phase 1

The first research phase focused on laying a solid foundation for the research project through a thorough state of the art analysis of mechatronic design methodologies as well as possible research demonstrators within the vehicular mechatronics field. Furthermore, engineers from the vehicle industry were interviewed in order to identify problems with the design methodologies used in practice as well as their thoughts on what needs to be included in a new methodology and tool. The deliverables of this phase were a technical report [7] and a conference publication [8], both reviewing the state of the art.
Figure 1.4. Outline of research approach.
1.3.2. Phase 2

The focus of the second phase was the static dimensioning part of the design methodology. The work of Roos [9] was, to some extent, used as a basis for this phase. The component models described in his work were applied together with the new method, allowing arbitrary system topologies and a much more flexible component configuration as opposed to previous work. A mechatronic servo design case was evaluated in order to compare the two methods. This work resulted in a conference publication, see appended Paper C.

1.3.3. Phase 3

During Phase 3 the possibility to evaluate dynamics as part of the design process was investigated. The phase resulted in the extension of the method, adding support for evaluation of dynamic constraints in a time efficient manner. One conference paper [10] and one journal paper, see appended Paper B, were published.

1.3.4. Phase 4

The goal of the fourth phase was to conduct a proper design case, from concept to physical prototype (G2) in order to validate previous work. Five months were spent at Stanford University, working in tight collaboration with researchers in the vehicular mechatronics field in order to identify a design case and its requirements. New component models were developed and a haptic force feedback steering wheel for the RCV prototype drive by wire car was later designed and built according to the design methodology. As additional deliverables, two conference publications were published; see appended papers D and E.

1.3.5. Phase 5

The main focus of Phase 5 was to develop a supporting software tool and extending the methodology with support for more complex systems, such as systems with multiple degrees of freedom. A two-axis gantry system design case was analyzed to verify the new functionality. The phase deliverables were a submitted journal paper, see appended Paper A, and this dissertation.
1.4. Summary of appended papers

This dissertation is a summary of the research presented in the appended papers. The papers reflect the major steps described in the research approach section. The presented research can be split into three major categories: methodology, component models, and design cases. Whereas all papers present design cases in more or less detail, paper D contains the most thorough one. Papers B and C focus mainly on further developing the method, whereas E presents a new component model. Paper A takes a hybrid approach, focusing on both new component models and method improvements.

1.4.1. Paper A

Title: A Holistic Optimization based Mechatronic Design Methodology with a Supporting Software Toolbox

Paper A extends the methodology and tool with support for multiple degrees of freedom systems. Component models for linear belt drives are described and a two-axis gantry system design case is presented. The design case has a larger number of components, and hence design variables, in comparison to previously presented cases, and is used to evaluate the capability of the method regarding systems with increased complexity. In addition, a new, significantly improved version of the software tool is presented.

1.4.2. Paper B

Title: Optimization of Mechatronic Systems for supporting Early Design Decisions

Paper B introduces the method used for evaluating dynamics and gives an overview of the, at the time of writing, state of the methodology. A design case is presented in which the effect of including control and dynamic constraints in the optimization loop is evident. The paper shows that it is possible to efficiently evaluate dynamic performance without simulation and to do this concurrently with dimensioning in a time efficient manner.
1.4.3. Paper C

Title: Optimization of Mechatronic Systems for supporting Early Design Decisions

Paper C contains the first published description of the design methodology and presents an early version of it, capable of static dimensioning only. A design case is presented, similar to previously published research by Roos [9], with the intention to validate correctness of the methodology and results. The result shows that the new, more general, methodology can replicate Roos’ static dimensioning results in a time efficient manner while still allowing for flexible system and component configuration.

1.4.4. Paper D

Title: Holistically Integrated Design of a Haptic Steering Wheel

This paper presents the design and implementation of a haptic force feedback steering wheel using the design methodology. Two different system concepts are evaluated whereof the best is developed further, i.e. into a physical prototype. The physical device is compared to the optimization results from the method, showing that both static dimensioning and the dynamic response approximation from the methodology correlate well with that of the physical system.

1.4.5. Paper E

Title: Optimal design of harmonic drive servo

Paper D introduces the harmonic drive component models needed for the design case presented in Paper D. The paper compares different design guidelines for harmonic drive gearing given in scientific publications in order to develop scaling models for various component properties. A simplified design case is presented in which a servo for a haptic steering wheel is designed and compared to an existing device. The results show that it is possible to improve the design of the existing device significantly in terms of size and friction.

1.5. Dissertation outline

The dissertation is divided into a number of chapters. The first introductory chapter is concluded with this paragraph, and describes the motivation, goals, and approach for the research presented. It is followed
by a chapter about the background and related research in a number of fields, but the main focus of the chapter is presenting the state of the art in holistic mechatronic design. The third chapter presents the core of this research, i.e. the design methodology and is followed up with a chapter describing two design cases. The last chapter contains conclusions, suggestions for future work and a discussion.
Chapter 2

Background and related research

We are all agreed that your theory is crazy. The question that divides us is whether it is crazy enough to have a chance of being correct.

— Niels Bohr, 1958

This chapter aims to orient the reader in, and provide a background about, important related work in the fields of product development, optimization and component modeling.

2.1. General product development

The Design Theory and Methodology, DTM, field has been actively researched since the 1970’s. Before then design was rather thought of as closer to art than engineering, mainly due to insufficient knowledge about the field [11]. In general, most widely known generic design methodologies, e.g. Pahl et al. [12], VDI 2221 [13], Ulrich and Eppinger [14], Koller [15] and Roth [16], [17], all share similarities. Even though the processes differ in their explicit specifications, they all consist of a number of distinct design phases, going from more abstract ones to more concrete ones. As an example Ulrich and Eppinger [14] splits their design process into six phases: planning, concept development, system-level design, detail design, testing, and production ramp up. An additional similarity is that they all make use of time expensive backwards iteration between phases as needed, i.e. if a problem arises in a later phase, backtracking to earlier ones might be required. See Figure 2.1 for a development process adapted from [14] with added iteration cycles

Design methodology literature presents many different methods for concept selection, e.g. Ulrich and Eppinger’s Concept scoring method [14], Pahl and Beitz’ Use Value Analysis method [12], and Pugh’s Datum
method (decision matrix method) [18]. Derelöv [19] presents a comprehensive review on evaluation and selection methods for concepts. In order to completely understand a concept, it has to go through a full development process, something which is usually not feasible for several parallel concepts due to time and cost constraints. Hence the challenge lies in finding a balance between how far to take the concept evaluation before narrowing down and deciding upon which concept(s) to further develop. The goal of concept selection is to pick the highest potential concept while using the least amount of resources in doing so.

### 2.2. Mechatronic product development

As earlier mentioned, mechatronic product development has traditionally been carried out according to conventional design methodologies where each engineering domain is treated separately and is only integrated in later phases. In most cases this will result in a sub-optimal product since each engineering domain design is optimized separately, see Figure 2.2. Da Silva et al. [20], [21] note that in order to design an optimal mechatronic system the different engineering domains, e.g. structural design and control design, need to be treated concurrently. Integrated structure and control design could in a sense be regarded a
subset to the mechatronic design problem. Even this subset, however, is difficult to handle. The focus of the research field is how to improve controller robustness and performance by integrating the structure design process in the control design process; [22]–[24] provides examples of this.

Derived partially from software design methodology, the VDI 2206 [1] V-model is one of the most widely cited mechatronic design methods. A top-down approach is suggested, where modularization with well defined component interfaces is prioritized in order to put a focus on the overall system design. As seen in the macro-level representation of the method in Figure 2.3, the development process is split into three major phases: system design, domain-specific design, and system integration. As also seen in the figure, the name of the method refers to that the design is verified/validated against specification and requirements resulting from earlier phases at each level of granularity. In reality, and as noted in the design guideline, development of mechatronic products usually requires a number of V-cycles at different abstraction level, as seen in Figure 2.4. As with most design processes, the V-model still relies on backtracking when problems arise.

Most modern mechatronic design methods are based on model based design. As an example, a commonly suggested approach, within academia, e.g. [6], [25]–[28], to treating mechatronic systems holistically throughout the whole design process is by the application of system modelling tools and languages, such as the System Modeling Language, SysML. The idea is that by modeling all requirements and dependencies between components’ behavior and properties it is possible to treat all the important factors of the design concurrently. The approach allows system modeling at both very fine and coarse detail levels. The approach is unnecessarily complex for the research presented in this dissertation and does not help solving the core problem in focus, namely holistic design optimization in an early design phase. Future iterations of the presented design methodology, however, could potentially benefit from SysML integration.

Due to the multiple engineering domains involved in mechatronics design, it is seldom possible to use one single design tool to describe the whole design in detail. A frequently suggested solution, e.g. [29]–[32], is the use of tool integration software (e.g. ModelCenter or modeFRONTIER) that allows domain specific tools, e.g. mechanical CAD tools (ProEngineer), FEM-analysis tools (e.g. Comsol), electrical CAD
Figure 2.3. Macro-level V-model adapted from [1].

Figure 2.4. Iterative V-model adapted from [1].
tools (e.g. EagleCad), electric circuit simulation tools (e.g. LTSpice), and control simulation tools (e.g. OpenModelica or Simulink), to interact with a system model in order to model and analyse specific parts of the system and the integration of these, see Figure 2.5. This allows a designer to create a detailed description of a system through system modeling and to analyze the result, or even optimize the design holistically. This approach, however, is mainly suitable for detailed design and not for concept evaluation due to the modeling effort required from the designer and the evaluation time needed for the in-depth analysis the tools provide. This is especially true if the design is to be optimized, and even more so if dynamics is involved, since this will require a large amount of time consuming evaluations of different system variations.

In his dissertation [9], Roos presents a method for evaluating mechatronic servo system concepts through optimization. Algebraic models of permanent magnet DC motors (PMDC-motors), planetary gears, and motor amplifiers are used to concurrently dimension the servo for a static load case. Roos makes use of two different types of models for static components: his PMDC-motor models rely on scaling an existing drive to fit the required load whereas his planetary gear models are derived directly from design standards. The scaling models are based on physical properties of electrical motors, resulting in a high model fidelity. After the static load dimensioning a genetic algorithm based optimizer is used to design a controller with minimized control error through time expensive simulation. This however, relies on two separate optimization
loops, and can hence not be considered fully holistic. In addition, the method is limited to a fixed servo system topology. The dissertation shows, however, the potential of using simplified models in order to evaluate concepts. Figure 2.6 shows an overview of Roos’ design methodology.

Budinger et al. [3], [33], [34] presents a model based design methodology for preliminary design of mechatronic actuators. Similar to Roos, the method relies on scaling models to determine properties such as geometrical, power, thermal, and reliability based on existing component designs. The dynamic behavior is described with a meta-modeling technique where a response surface is estimated, with a polynomial, based on a number of simulation runs. This allows fast evaluation of dynamics after the estimation simulations have finished. A design example is presented in which the method is applied to the design of an aircraft control surface actuator in order to explore the effects of for instance actuator mass and anchorage point. The design case shows that the method is able to obtain a significantly improved solution, as compared to initial sizing, after 20 minutes of computation.

Behbahani and de Silva [35], [36] presents a multi-criteria design methodology based on genetic algorithms. The method calculates a “Mechatronic Design Quotient” which reflects the global design satisfaction of a given design by aggregating a number of criteria satisfaction scores. This value is then used as an optimization objective. The design of a fish cutting machine is used as an example to illustrate the methods capabilities. The papers, however, do not provide detailed information about the models used in the design case and how these are evaluated.
Coelingh et al. [37] describes a method for evaluating dynamic performance of conceptual mechatronic systems with a minimal amount of detail design needed. The method relies on approximating the system’s dynamics with a mass-spring-damper model, which reflects the dominant behavior. The paper primarily presents a means to analyze dynamics at an early stage, and not how this fits into an overall design methodology.

In order to further automate the concept phase of a design methodology, and create a “mechatronic compiler” capable of designing a product based on given requirements, the process of concept generation needs to be considered. I.e. selection of the system topology needs to be included in the concept evaluation process. Kerzhner [28], Lin et al. [38], and Bebhabani et al. [39], [40] each give examples of how to include this topological optimization in the process.

### 2.3. Optimization methods

Optimization is commonly used in multi-disciplinary engineering in order to find the best possible design, e.g. [9], [23], [33], [35]. A large number of optimization methods exist, and it would be impossible to cover all of them in this dissertation. Instead, a number of examples will be given. Optimization methods are usually classified as either gradient-based or non-gradient-based. As the name indicates, the gradient-based methods rely on following the gradient of the solution space. This approach is efficient, which allows solving problems with many design variables. However, these methods are in general unable to solve problems with a discrete solution space or discrete design variables (e.g. integer design variables) and are generally limited to local optimization [41], see Figure 2.7. A straightforward solution to get around the latter, hence making the gradient based methods able to find global optima, is the use of multiple local searches with different starting points [42]. On the other hand, the non-gradient based methods ability to solve a multitude of optimization problems, including ones with discrete design variables, make them a good fit for mechatronic design optimization. Some of these methods are introduced below.
2.3.1. Monte Carlo simulation

Monte Carlo simulation, invented during the development of the nuclear bomb, is a method which uses repeated random sampling to approximate the solution of a function [43], see Figure 2.8 for an example. The name refers to the method’s resemblance to playing and recording results in a casino. If the sampled function is an objective function, the approach can be used for optimization [44]. The general advantage with Monte Carlo methods is that they will not as easily get stuck in local optima due to the nature of the method. However, random sampling without any kind heuristics guiding the optimization becomes exponentially less practical with an increasing amount of design

![Figure 2.7. Example of local and global optima. A gradient based method could easily get stuck at one of the local optima.](image)

![Figure 2.8. Illustration of Monte Carlo simulation. The red points represent the random sampling of the total design space represented by the surface.](image)
variables. Bellman [45] coined the term “Curse of Dimensionality”, which refers to this exponential solution space increase associated with adding extra dimensions to a design space. For instance, in order to evaluate $x$ points in each of $y$ dimensions (i.e. design variables in the optimization case) $x^y$ function evaluations are needed. This means that even though ten points in two dimensions only require one hundred evaluations, ten points in ten dimensions would require ten billion evaluations. It is easy to see that this quickly becomes unfeasible.

### 2.3.2. Simulated annealing

Simulated annealing [46] is an optimization algorithm, related to Monte Carlo optimization, and inspired by the annealing process in metallurgy, i.e. the process of heating and cooling a metal in order to change its physical properties. When the temperature is high the internal structure is loosened up, and when it cools down it returns to a fixed state. Similarly, in simulated annealing, a variable representing the temperature starts out at a large value and is then slowly decreased. The variable determines the ability to jump out of local minima by, to a larger extent, allowing moving to a worse solution when the value is high. As the temperature variable decreases, the algorithm starts to focus on a smaller and smaller solution space. In a sense, the cool down rate determines how much time should be spent in finding an as good solution as possible. The optimization method is great at finding “good enough” solutions to discrete optimization problems, which should make it well suited to the mechatronic design problem, e.g. [47], [48].

### 2.3.3. Complex method

Gavel et al. [49] describes the optimization of a conceptual aircraft fuel transfer system through the use of the Complex optimization method [50]. The method relies on (1) randomly selecting a number, larger than the amount of design variables, of points in the design space, (2) calculating the fitness of these points, and (3) repeatedly replacing the worst point with a new one until optimum has been found. The worst point is replaced with its reflection through the centroid of the other points. The reflection distance is varied in order to avoid getting stuck at a specific point.
2.3.4. Evolutionary methods

One of the most widely used categories of non-gradient based optimization methods are evolutionary methods, e.g. genetic algorithms [9], [35], [48]. An evolutionary optimization algorithm mimics the Darwinian principle of natural selection in order to find optimum. Genetic algorithms, for instance, describe an individual, i.e. possible solution, with a number of genes, i.e. design variables. A generic genetic algorithm works, as illustrated in Figure 2.9, by:

1. Generating a population of individuals, each with its own combination of gene values.
2. Determining the fitness of each individual according to an objective function.
3. Forming a new generation, i.e. population, by saving, crossbreeding and mutating the elite individuals (i.e. the ones with the best fitness value) and removing the rest.
4. Repeating steps 2 and 3 until a maximum amount of generations is reached or the change in best fitness value is below a certain threshold.

Evolutionary algorithms are well suited for mechatronic design optimization due to their capabilities in finding good solutions to a diverse range of complex optimization problems.

Figure 2.9. Overview of genetic algorithm based optimization.
Chapter 3

Design methodology

Algorithm – noun. A process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer.
— Oxford Dictionaries, 2014

This chapter describes the proposed design methodology, a number of component models, and the developed supporting software tool. Additional details and the development history can be found in the appended papers.

3.1. Context of methodology

The presented design methodology has its roots in the static component models developed by Roos [9]. The method relies on the same type of models used, e.g. scaling and design guideline based ones, but applies them in a more flexible way. As mentioned in the introduction chapter, the focus area of the method is better analysis of design concepts, see Figure 3.1. This enables development of better products by making use of synergies between engineering domains, reducing time to market and cost by lowering the amount of backtracking from later design phases. The methodology does not take concept topology generation and topology optimization into account. This could, however, be a natural extension of the method. Given specifications, requirements,
and optimization goals, as defined below, the method facilitates that the potential of a concept can be evaluated through the use of optimization. Figure 3.2 illustrates a simplified overview of the methodology.

Requirements describe what the mechatronic system needs to be capable of, i.e. the load profile it needs to handle, and the dynamic performance. The load profile could for instance be given as a motion profile and an equivalent mass or inertia, which needs to be moved according to the specified profile. In difference to other mechatronic design methodologies, the dynamic performance is not treated as an optimization objective, but rather as a requirement which needs to be fulfilled, hence constraining the solution space. A dynamic performance requirement could for instance be the closed loop pole locations, or maximum allowed control error.

Specifications, on the other hand, describe how the requirements will be fulfilled, i.e. with which system, in terms of components, topology, parameters, and design variables. For instance, the specification might say that the system consists of a motor, a gearing and some structural elements, with certain parameters given, e.g. radius of a shaft. The specification also states which parameters are to be used as design variables and their allowed ranges.

The optimization goals specify what the purpose of the optimization is. This could for instance be minimizing the mass, volume, or cost of the system, or a combination of these. If it is possible to sufficiently model a property, e.g. size, cost, or some sustainability related property, for efficient evaluation, it can be used as an optimization objective.

The system resulting from the optimization can then be used as a starting point for detailed design, or as feedback for further concept refinement.
3.2. System modeling

A system model, describing the concept to be evaluated, is one of the core input specifications needed from the design engineer. As illustrated in Figure 3.3, the system model is created by:

1) Combining a number of physical mechatronic base components, e.g. PMDC motor, gear, or shaft, from a library
2) Configuring their parameters as fixed values or optimization design variables.
3) Adding a number of dynamic components, e.g. controller and/or disturbance models

Each physical component contains a set of both static property models and dynamic behavior models whereas dynamic components contain only behavior models and stepwise instructions on how to evaluate dynamics. As an example, a non-physical component setup could be used to evaluate the maximum error of a controller. In this case the controller topology needs to be determined first, then controller parameters need to be calculated, and finally the system response to a given load profile can be evaluated.

In order to determine a system design from a given concept specification and requirements, an optimizer is used. Most non-gradient based optimization methods should work well with the method, but so far
Figure 3.4. Overview of methodology.

- **Objective**: Size, Cost, Weight, Etc...
- **Dynamics**: Poles, Maximum error, Etc...
- **Load**: Load model, Equivalent mass, Motion profile, Etc...
- **System specification**: Controller, Sensor
- **System requirements**: Optimization process (GA), Static component models, Dynamic constraint check, Crossbreed and Mutate

User interaction needed
only one based on genetic algorithms has been used. The optimizer determines the fitness value of a potential system through static dimensioning. However, if the dynamic constraints are not fulfilled, e.g. if the maximum control error is too large, or the resulting dynamic load on a component is larger than dimensioned statically for, the individual is rejected. Figure 3.4 outlines the major steps of the evaluation methodology.

### 3.3. Component Modeling

System optimization requires thousands, or tens of thousands, of evaluations of the component models involved, hence each individual model cannot take more than in the order of milliseconds of time to evaluate. For obvious reasons, this puts high requirements on model evaluation efficiency. The methodology handles this by avoiding simulation as well as using simplified models for describing static properties and dynamic behavior.

As mentioned above, the component models contain a number of static property models and dynamic behavior models. A gear component can e.g. have one model describing the dimensioning of the gears, one model describing its friction, one model describing how a load profile can be translated from output side to input side (for static dimensioning), and differential equations describing the behavior, used for validating dynamic requirements. A physical component is required to, at the very least, implement the following two models:

1. a **“static load transfer”** model, which translates the load profile from the load side to the drive side of the component for static dimensioning purposes.
2. a **“dynamic load transfer”** model, which returns the differential equations governing the component behavior.

#### 3.3.1. Static property models

Static property models are algorithms containing the steps needed to calculate a certain property. They are, in most cases, based on underlying algebraic relations describing either dimensioning standards or scaling relations, like [3], [9]. Since the latter work by scaling physical properties from existing, similar, products, they have a high fidelity in most cases, whereas the guidelines used in the former does not necessarily correlate
with what is state of practice in industry. The main approaches to developing scaling models is either basing them on the underlying physical phenomena governing the design, basing them on design guidelines for the type of component, or the use of regression analysis in order to fit a polynomial to an existing design. Careful design and the low complexity of these models, e.g. not using FEM-analysis for dimensioning, allow very time efficient implementation.

Static properties are evaluated by executing a specific component model, but this model might need input from other components. For instance, in order to determine the size of a DC-motor connected to a load through a planetary gear, the DC-motor size model will need to know the effective load it should handle, and in order to determine this, the gearing needs to be dimensioned first. Figure 3.5 illustrates the process of evaluating an example static component model; the steps shown are as follows:

1. The size of the motor component is requested by calling the “Dimension” model.
2. The “Dimension” model needs to know the load affecting the motor so this is requested from the “static load transfer” model of the gear.
3. The gear’s “static load transfer” model needs to know the load affecting the planetary gear; hence this is requested from the “static load transfer” model of the load component.
4. The load’s “static load transfer” model needs to know the load specification; therefor this is requested from the “load specification” model of the load component.
5. The “load specification” model answers request (4) with the load specification.

6. The load’s “static load transfer” model now answers request (3).

7. The gear’s “static load transfer” model now knows the load affecting the planetary gear, but it still needs to know the gear properties, e.g. inertia and efficiency; hence this is requested from the “dimension” model of the gear component.

8. The gear’s “dimension” model needs to know what load to dimension for, hence this is requested from the load’s “static load transfer”

9. The load’s “static load transfer” model has already been calculated and hence the request (8) can be returned directly.

10. The gear’s “dimension” model is now calculated and returned to answer request (7).

11. The gear’s “static load transfer” model is now calculated and returned to answer request (2).

12. The motor’s “Dimension” model now has all the information needed, hence the original request (1) is answered.

3.3.2. Dynamic behavior models

Detailed models of dynamics are usually complex and time expensive to evaluate. Janschek [51] suggest that it in many cases is possible to use reduced order, linear models, based on multiple mass oscillators to evaluate performance in early stages of design. The design methodology presented in this dissertation relies on evaluating dynamics without simulation by limiting the models to be linear time invariant. The following section describes how this is achieved.

Opposite to simulation, which treats dynamic systems in the time-domain, this method relies on approximating the time-domain behavior by evaluating the dominant harmonics of the load profile and other exogenous inputs in the frequency domain. These dominant frequencies, their amplitudes and phases, are identified by applying Fourier transforms to the respective input signals. By using strictly linear time invariant, LTI, models, transfer functions for important signal transfers can be determined and then evaluated separately for each of the individual harmonics. Examples of important transfer functions capture for instance the closed loop behavior from reference to output, used to determine the overall closed loop performance, and closed loop behavior
from reference to actuator input, which can be used to determine if the
dynamic load the actuator needs to handle is larger than what it was
statically dimensioned for. The response of these transfer functions to
individual frequencies are calculated by replacing the Laplace frequency
variable, $s$, with

$$s = 2\pi f j$$  \hspace{1cm} (1)

where $f$ is the frequency and $j$ is the imaginary unit. The gain and phase
shift for that frequency is then the magnitude and phase angle,
respectively, of the complex result, i.e.

$$A_{tf} = |G(2\pi f j)|$$ \hspace{1cm} (2)

$$\varphi_{tf} = \text{arg}(G(2\pi f j))$$ \hspace{1cm} (3)

where $A_{tf}$ and $\varphi_{tf}$ are the gain and phase shift of the transfer function for
a specific frequency, and $G$ is the transfer function. The individual
frequency domain responses can then be superposed to construct an
approximation of the time domain response, i.e.

$$\sum_{i=1}^{h} A_i \cdot A_{tf i} \cdot \cos(2\pi f_i \cdot t + \varphi_i + \varphi_{tf i})$$ \hspace{1cm} (4)

where $h$ is the number of dominant harmonics used, $A_i$ and $\varphi_i$ are the
amplitude and phase angle determined by the Fourier analysis for the
frequency, $f_i$, and $t$ is time. Figure 3.6 illustrates this superposition of
harmonics. This approach requires orders of magnitude fewer
calculations than the numeric differential equation solving methods used
for simulation.

For the sake of efficiency, all transfer functions that need to be
evaluated are determined symbolically outside the optimization loop.
This allows fast evaluation inside the loop by replacing the symbolic
parameters with numerical values. For this to be possible, it is implied
that the controller structure is determined outside the optimization loop.
Different controller designs have been tested with the method, including
a full state feedback controller adapted from Åström and Wittenmark [52], described further in appended Paper B. Frede et al. [53] extends the controller design by including noise and disturbance rejection.

As mentioned earlier, the dynamic performance is regarded as a requirement, hence if constraints such as maximum integrated square error, maximum error, and overshoot are not fulfilled, the proposed solution is rejected by the optimizer.

### 3.4. Component model examples

This section presents a number of component examples that have been used during development of the methodology. Note that the following models use the indices $in$ and $out$ for denoting the input and output of a component, where output is defined as the port on the load side.

#### 3.4.1. PMDC-Motor

The Permanent Magnet DC-Motor models used in this research have been adapted from Roos et al. [54] and are derived from electrical, magnetic, thermal and mechanical properties. The relations between the dimensions of a motor and its nominal torque handling capability can be described as

\[
C_m l_mr_m^{2.5} = \frac{1}{\tau} \int_0^T \left( (C_m l_m r_m^4 + J_m) \dot{\theta}_m + T_{m.out} \right)^2 \, dt
\]  

(5)
where $C_m$ is a constant for a specific motor type and cooling condition, $l_m$ is the rotor length, $r_m$ is the radius of the stator, $\tau$ is the cycle time of the load profile, $C_{mj}$ is a constant for a specific motor type, $J_m$ is the rotor inertia, $\theta_m$ is the angular position of the output shaft, and $T_{m,\text{out}}$ is the output torque. The volume of the motor, $V_m$, is approximated as a cylinder, i.e.

$$V_m = \pi r_m^2 l_m$$  \hspace{1cm} (6)

The motor size required can be determined through scaling by first determining the values of $C_m$ and $C_{mj}$ for a given motor type. The mass of the motor can then be determined from the scaled size and material properties.

The motor dynamics are modeled as

$$K_t i_{in} = T_{m,\text{out}} + J_m \dot{\theta}_m$$  \hspace{1cm} (7)

where $K_t$ is the motor torque constant, and $i_{in}$ is electric current.

The motor component uses the following parameters, of which any can be used as design variable (although it does not make sense to use some of them, e.g. reference motor and Max. RPM) and some can be calculated from others if not explicitly specified:

- **Axis** – Determines the rotational axis of the motor output shaft.
- **Commutation** – Determines if trapezoidal or sinusoidal commutation is used.
- **$J_o$ ratio** – Represents the ratio between motor shaft plus bearing inertia and rotor inertia.
- **Length** – Specifies motor length.
- **LoR max.** – Specifies the maximum length over radius ratio allowed.
- **LoR min.** – Specifies the minimum length over radius ratio allowed.
- **Max. RPM** – Specifies the maximum permissible RPM.
- **Radius** – Specifies motor radius.
• **Reference motor** – Specifies the motor type used as reference for scaling.
• **Reference motor index** – Specifies which specific motor of the type specified by *reference motor* should be used for scaling.

### 3.4.2. Planetary gear

In addition to the motor models in the previous section, Roos also developed models for planetary gears [55], which he based on Swedish design guides for spur gear [56], [57]. A planetary gearing contains three gear types: a sun gear, a ring gear, and planet gears. While different input/output/fixed gear configurations are possible, it is common to fix the ring gear while using the sun and planet gears as input and output. In this configuration the planet gears will revolve around the sun gear. Figure 3.7 illustrates the inner parts of a planetary gearing. As with the motor, the volume of the gearbox is approximated as

\[
V_{gp} = \pi r_{pg}^2 b_{pg}
\]  

(8)

where \( r_{pg} \) is the outer gear radius and \( b_{pg} \) is the total width of the gearbox. The design guidelines, forming the basis for the models, state that the gear size is limited by the bending stress at the root of a gear tooth, the Hertzian pressure at the teeth contact surfaces, and mechanical fatigue. However, if the gears are made of reasonably ductile steel and the number of sun gear teeth is small the Hertzian pressure is likely to be the sole constraining characteristic. Roos and Spiegelberg [55] derive the following expression, which needs to be fulfilled for each tooth surface

![Figure 3.7. Inner elements of a planetary gearing.](image-url)
pair in contact:

\[ r_{pg}^2 b_{pg} \geq Z_H^2 Z_M^2 Z_E^2 K_{H\alpha} K_{H\beta} \frac{\hat{T}_{pg,\text{out}} (n - 1)^2}{6(n - 2)\sigma_{H,\text{max}}^2} \]  \hspace{1cm} (9)

where \( Z_H \) is a form factor for Hertzian pressure, \( Z_M \) is a material factor for Hertzian pressure, \( Z_E \) is the contact ratio for Hertzian pressure, \( K_{H\alpha} \) is a factor describing the division of load between teeth, \( K_{H\beta} \) is a load distribution factor for Hertzian pressure, \( \sigma_{H,\text{max}} \) is the maximum allowed flank pressure, which can be determined for specific gear materials. The variable \( n \) is the gear ratio and \( \hat{T}_{pg} \) is the peak torque the gear is subject to. By using standard gear parameters the expressions can be simplified and written as

\[ r_{pg}^2 b_{pg} \geq 4 \cdot 10^{10} C_{gr} \frac{\hat{T}_{pg,\text{out}} (n - 1)^2}{(n - 2)\sigma_{H,\text{max}}^2} \]  \hspace{1cm} (10)

where the constant \( C_{gr} \) represents the relation between the outer radius and the pitch radius of the gear.

The gear dynamics model takes inertia and gear ratio into account, while neglecting other effects such as friction and stiffness, resulting in the following differential equations:

\[ (T_{pg,\text{in}} - J_{pg} \dot{\theta}_{pg,\text{in}}) \eta = T_{pg,\text{out}} \]  \hspace{1cm} (11)

\[ \theta_{pg,\text{in}} = n\theta_{pg,\text{out}} \]  \hspace{1cm} (12)

where \( \eta \) is efficiency, \( J_{pg} \) is inertia, and \( \theta_{pg} \) is angular position.

As with the motor, any of the parameters of the gear component can be used as design variable if needed. The following is a list of all the gear parameters:

- **Axis** – Specifies the rotational axis.
- **Efficiency** – Efficiency per stage.
- **Gear ratio** – The gear ratio of the component.
- **\( K_{r,o} \)** – Ratio between outer and inner gear radius.
- **Length** – Specifies the length of the planetary gearing.
- **Material** – Specifies material to use for gear wheels (needed since models are not scaling based).
- **Max. RPM** – Maximum permissible RPM.
- **Radius** – The radius of the component.
- **RoW Min.** – Maximum allowed radius over width ratio.
- **RoW Min.** – Minimum allowed radius over width ratio.
- **Safety factor** – Specifies the safety factor to use while dimensioning gears.
- **Stages** – Number of gear stages to use.
- **Sun teeth** – Specifies the number of teeth of the sun gear.

### 3.4.3. Harmonic drive

The harmonic drive is a type of gearing distinguished by very high single stage gear ratios and close to zero backlash, making them ideal for high precision servo applications. The main internal components of a harmonic drive are a typically elliptical wave generator, a flexible ring (a hollow cup with external teeth) and a rigid ring (a ring with internal teeth), see Figure 3.8. Any of these three components can be used as input or output while the third is fixed. However, the most common configuration is having the flexible ring as the low velocity/high torque output and the wave generator as high velocity/low torque input while the rigid ring is held fixed.

![Figure 3.8. Internal components of harmonic drive.](image)
The flexible ring is designed to have fewer teeth than the rigid ring; and due to this they won’t fully engage with each other in their normal non-stressed state. The wave generator is designed to deform the flexible ring such that it comes in contact with the rigid ring in two or more main areas called lobes, e.g., an elliptical wave generator would result in two lobes. The difference in number of teeth must be an integral multiple of the number of lobes [58]. One full rotation of the wave generator will cause a smaller counter rotation of the flexible ring proportional to the difference in number of teeth between the rings. For the standard configuration mentioned above, this results in the following gear ratio:

\[ R = \frac{N_F}{N_F - N_R} \]  

(13)

where \( N_F \) is the number of teeth on the flexible ring, and \( N_R \) is the number of teeth on the rigid ring [58].

The models described here were first presented in appended Paper E. The models were developed by studying various design guidelines for the gear type and based on that determining how to scale the models. Figure 3.9 compares scaling models based on different design guidelines [59]–[61] by showing scaled values for several existing harmonic drives based on a single drive in the series, see appended Paper E for more details. As

![Figure 3.9. Comparison of scaling models based on different design guidelines.](image)
can be seen in the figure, the scaling model based on Lelikov’s design guideline provides the best correlation with the real drive and is hence the one used for the component model. The following expression describes the relation between diameter, \( d \), gear ratio, \( n \), and output torque, \( T_{hd, out} \):

\[
d = \sqrt[3]{\frac{C_{hd} T_{hd, out} \cdot (1 + 0.001n)}{\sqrt{n}}}
\]  

(14)

where \( C_{hd} \) is constant for a specific gear model type. Other component dimensions and mass can be scaled from the diameter and the base drive. Figure 3.10 shows a 3d plot of calculated volume based on specific gear ratio and load torque compared to existing drives.

In order to determine the static load transferred from output to input side the friction needs to be known. It is approximated as

\[
T_f = f_o \cdot \text{sgn}(\theta_{hd, out}) + f_1 \cdot \theta_{hd, out}
\]  

(15)

where \( T_f \) is friction torque, \( f_o \) constitutes a Coulomb friction torque, \( f_1 \) is the viscous friction coefficient, and \( \theta_{hd, out} \) is output angular position. As with the diameter, the friction coefficients are determined through scaling.

Figure 3.10. Scaled drive size compared to real products.
The differential equations describing the behavior are approximated linearly as

\[(T_{hd,in} - J_{hd} \dot{\theta}_{pg,in})\eta = T_{hd,out}\]  

\[\theta_{hd,in} = n\theta_{hd,out}\]  

where \(T_{hd,in}\) is the input torque, \(J_{hd}\) is the inertia as seen on the input side, and \(\theta_{hd,in}\) is the input angular position. The average efficiency, \(\eta\), can be approximated from the friction model for a specific load case. A potential improvement to this model would be including the gear stiffness, which is a relatively prevalent effect in a harmonic drive. This would also imply a higher order dynamics model.

The harmonic drive component uses the following parameters:

- **Axis** – Specifies the rotational axis.
- **Gear ratio** – The gear ratio of the component.
- **Length** – Specifies the length of the planetary gearing.
- **Max. RPM** – Maximum permissible RPM.
- **Radius** – The radius of the component.
- **Reference drive** – Specifies the harmonic drive gear type to use as reference for scaling.
- **Reference drive index** – Specifies which harmonic drive of the above specified type to use as reference drive.

### 3.4.4. Solid shaft

The shaft models presented here were derived from classic solid mechanics and the following equations, taken from [62], describe some of the important properties needed for dimensioning:

\[r_s = \frac{3 \sqrt{2T_{s, out}}}{\tau_{s, max} \pi}\]  

\[J_{s,a} = \frac{\pi r_s^4}{2}\]  

\[G = \frac{E}{2(1 + v)}\]
where $r_s$ is shaft radius, $\tau_{s,\text{out}}$ is the peak torque, $\tau_{s,\text{max}}$ is the maximum permissible shear stress, $J_{s,a}$ is the area moment of inertia of the shaft, $G$ is shear modulus, $E$ is Young’s module, $\nu$ is Poisson’s ratio, $k_s$ is the shaft’s stiffness, and $l_s$ is the shaft’s length.

The dynamics of the shaft are approximated as a lumped mass-spring-damper, as seen in Figure 3.11, resulting in the following differential equations:

\[ \frac{1}{2} J_s \ddot{\theta}_{s,\text{in}} = T_{s,\text{in}} - k_s (\theta_{s,\text{in}} - \theta_{s,\text{out}}) - d_s (\dot{\theta}_{s,\text{in}} - \dot{\theta}_{s,\text{out}}) \]  
\[ \frac{1}{2} J_s \ddot{\theta}_{s,\text{out}} = k_s (\theta_{s,\text{in}} - \theta_{s,\text{out}}) + d_s (\dot{\theta}_{s,\text{in}} - \dot{\theta}_{s,\text{out}}) - T_{s,\text{out}} \]

where $J_s$ is the moment of inertia, $\theta_s$ is angular position, $T_s$ is the torque, and $d_s$ is damping.

The shaft component uses the following parameters:

- **Axis** – Specifies the rotational axis.
- **Length** – Specifies the length of the shaft.
- **Material** – Specifies the material of the shaft.
- **Radius** – The radius of the component.

### 3.4.5. Linear timing belt drive

Timing belts are commonly used as a means to transform rotational energy into translational energy. This is achieved by attaching the element which is to move translational to the belt and driving one of the pulleys. Figure 3.12 illustrates the main components, i.e. the belt, the toothed drive pulley, the un-toothed return pulley, and the carriage. The model described here was first presented in, and is described more in
detail in, appended Paper A. The model is not as complete as the other ones described in this chapter; for instance has structural components of the belt drive been omitted and pulley teeth strength is not considered.

The models described here are at large based on the work of Yang [63] who, in his dissertation, describe the modeling and control of two-axis gantry robots. Figure 3.13 illustrates the forces and torques affecting the system as well as a free body diagram.

The effects of the return pulley, e.g. inertia and friction, as well as carriage mass are likely going to be small in comparison to ones of the driving side, e.g. pulley, motor, gearing, and carriage. Due to this, the former are neglected in order to reduce the model order.

The dimensioning model focuses on belt strength. The minimum belt area, \( A \), required to handle a given peak load force, \( \tilde{F} \), is determined based on the yield stress of the belt, \( \sigma_y \), i.e.

\[
Max(\tilde{F}_1, \tilde{F}_2) = A\sigma_y
\]  

(24)
The belt forces depend on the belt elasticity, and can be determined by treating the belt as a spring with a varying free length and fixed operating length, in contrast to regular springs which have fixed free length and varying operating length. Appended Paper A derives the following equations:

\[
F_1 = \frac{L(F_0 + F_{out}) - EA \sqrt{L(L(EA + F_0)^2 + 2y(EA + F_0)F_{out} + LF_{out}^2)}}{2L} \tag{25}
\]

\[
F_2 = \frac{LF_0 - L(EA + F_{out}) \sqrt{L(L(EA + F_0)^2 + 2y(EA + F_0)F_{out} + LF_{out}^2)}}{2L} \tag{26}
\]

where \( L \) is the distance between the two pulleys, \( E \) is the Young’s module, and \( y \) is the translational position. The constant \( F_0 \) is the belt pretensioning force, which is required in order to avoid a belt slack that might result in a whip effect. The constant is recommended to be larger than the maximum tangential force, \( F_{t,\text{max}} \), acting on the pulley [64], i.e.

\[
F_0 \geq F_{t,\text{max}} \tag{27}
\]

All other dimensions, mass, and belt stiffness are derived based on belt size.

The following equations describe the dominant dynamic behavior:

\[
f_{1} \dot{\theta}_{in} = T_{in} - r_1 k(r_1 \theta_{in} - y) \tag{28}
\]

\[
F_{out} = k(r_1 \theta_{in} - y) \tag{29}
\]

where

\[
k = \frac{2 * L * (EA + F_0)}{L^2 - r_1^2 * \dot{\theta}_{in}^2} \tag{30}
\]

is a time varying belt stiffness. As mentioned earlier, dynamics models are required to be linear time invariant in order be used for the design methodology and this dynamic model does not fulfill this requirement. A first order Taylor series expansion around \( \theta_{in} = 0 \) is applied to
\[ k(r_1 \theta_{in} - y) \] (31)

in order to linearize the system, resulting in

\[ J_1 \ddot{\theta}_{in} = T_{in} - r_1 \frac{2(r\theta_{in} - y)(EA + F_0)}{L} \] (32)

\[ F_{out} = \frac{2(r\theta_{in} - y)(EA + F_0)}{L} \] (33)

which is linear time invariant.

The component uses the following parameters:

- **Axis**– Determines translational axis.
- **Belt thickness** – Specifies the belt thickness.
- **Belt width** – Specifies belt width.
- **Efficiency** – Transmission efficiency.
- **Length** – Specifies the distance between pulleys.
- **Length margin** – Pulley distance required in addition to load profile motion.
- **Pulley material** – Material to use for pulleys.
- **Pulley radius** – Radius of pulleys
- **Safety factor** – Describes belt safety factor used while dimensioning.
- **Tension member ratio** – The ratio between tension members cross-sectional area and belt cross-sectional area.
- **Tension member material** – Material of tension members.
- **WoT max.** – Maximum allowed belt width over thickness ratio.
- **WoT min.** – Minimum allowed belt width over thickness ratio.

### 3.4.6. Load

The load component models are dependent on the design case to be evaluated. One scenario is, for instance, to treat the load as an inertia or mass, i.e.

\[ J_i \ddot{\theta}_{lin} = T_{lin} \] (34)

\[ \theta_{lin} = \theta_{l.out} \] (35)

or
where $J_l$ is the load inertia, $T_l$ is the load torque, $\theta_l$ is angular position, $M_l$ is the load mass, $F_l$ is load force, and $y_l$ is position.

### 3.4.7. Full state feedback controller

The full state feedback controller component is based on the pole placement method described in Åström and Wittenmark [52] and is presented in appended Paper B. The component only contains dynamic models and does hence not affect the static dimensioning of the system directly. A two degrees of freedom controller structure, i.e. a structure containing a feed forward part and a feedback part, is used and its parameters are determined through the following steps:

1. Select feedback polynomial structure

\[
G_c(s) = \frac{S(s)}{R(s)} = \frac{S_n s^n + \cdots + S_0}{s^n + r_{n-1}s^{n-1} + \cdots + r_0} \tag{38}
\]

where $n$ is the number of poles in plant transfer function, and $S_i$ and $r_i$ are controller parameters.

2. Calculate the closed-loop polynomial

\[
A_{cl} = A \cdot R + B \cdot S \tag{39}
\]

where $A$ and $B$ are the denominator and numerator respectively of the plant transfer function.

3. Select a desired closed-loop polynomial

\[
A_d = A_m(s) \cdot A_o(s) \tag{40}
\]
where $A_m$ is a polynomial of order equal to that of $A$, and $A_o$ is a polynomial of order equal to the order of $A_{cl}$ minus the order of $A_m$.

4. Solve the Diophantine equation $A_{cl}(s) = A_d(s)$ for the controller parameters in $S$ and $R$.

5. Set the feedforward numerator polynomial to

$$T(s) = t_0 A_o(s)$$

where $t_0$ is a static gain that gives unit DC gain in the closed-loop transfer function from reference, $r$, to output, $y$, for example

$$t_0 = \frac{A_m(0)}{B(0)}$$

The transfer functions of the feedforward and feedback parts of the controller are given by the polynomials $T$, $R$, $S$, and $R$:

$$G_{ff} = T/R$$

$$G_c = S/R$$

3.5. Software tool

The methodology is supported by a software tool implemented as a MATLAB toolbox, and making use of Mathematica integration in order to symbolically manipulate equation systems. The capability of the software tool includes the following:

1. Given specifications, requirements, and a library of components, a conceptual mechatronic system can be configured through an intuitive graphical user interface.
2. Given system configuration and models of physical component properties, load based static dimensioning can be performed.
3. Given system configuration, static dimensions of components, and differential equations describing component behavior, transfer functions describing system behavior can be determined.
4. Given system configuration and system behavior a controller can be designed and closed loop performance can be evaluated.
5. Given all of the above, holistic optimization of a mechatronic system can be performed.

The tool features a graphical user interface, GUI, as seen in Figure 3.14, which in a straightforward fashion allows the user to

- define the motion profiles used as specification.
- drag, drop, and interconnect components such as motors and gears from a library to form a mechatronic system.
- configure component parameters such as dimensions, materials, and gear ratios. Each parameter can be either a fixed value or an optimization design variable. Design variables can be set to be either a floating point span, integer span, an enumerated set of possible values, or have an algebraic relation to other design variables (e.g. in order to match the radii of two components).
- specify the dynamic components and constraints.
- select and configure the optimizer parameters and objectives.

The toolbox is implemented using the object oriented programming

Figure 3.14. Screenshot of the software tool GUI.
Figure 3.15. Optimization process outline.
paradigm, hence allowing components to be implemented as classes which can be instantiated into individual objects during optimization. Component models are implemented as class methods and each component class inherits common functionality from a parent class. This could for instance be the capability to store preparation data, calculated before the optimization loop for convenient use later on, or remembering which models have already been called for a given instance of that class.

Figure 3.15 provides an outline of the optimization process. When the optimization process is started by the user, the tool runs a sanity check on the configuration in order to find out if the user has missed something. After sanity checks have passed, the static and dynamic preparation functions of each component are executed. This allows the component models to for instance load data files into memory and for the plant model to be determined. A plant model is calculated by fetching Laplace transformed differential equations describing component behavior from each individual component, adding connecting equations (i.e. setting the output of one component to equal the input of the next), and solving the equation system with respect to given input and output variables.

When sanity check and preparation functions are finished, the tool starts several parallel workers and copies the configuration and prepared data to them in order to make best use of modern multi-core and multi-processor computers.

During the optimization loop the optimizer generates a number of individuals, i.e. proposed solutions, which are evaluated and compared to each other. The objective function is used to evaluate each individual solution generated by the optimizer according to the given objectives. This is done by:

1. Instantiating the system and its components according to given design variables and parameters.
2. Evaluating the static objective fitness according to the process described in section 3.3.1. The dynamic programming approach taken prevents unnecessary model evaluations, i.e. the output of a model for a specific individual is stored and if more than one model within the same individual require the same information it does not need to be reevaluated. If the system does not give a valid static objective fitness value it is discarded.
3. Evaluating dynamic constraints as described in section 3.3.2., i.e. by replacing symbolic transfer function constants with numerical
values for the plant model as well as for controller and evaluating the resulting transfer functions for the dominant frequencies. If dynamic constraints are not fulfilled the solution is discarded.

The resulting system solution from the optimizer can be post processed according to the user needs. The tool is for instance capable of drawing a visual representation of the system.
Chapter 4
Design examples

The Coldest Winter I Ever Spent Was a Summer in San Francisco
— Unknown (commonly attributed to Mark Twain)

This chapter introduces two of the design cases studied during the research process. The first case is presented in appended Paper D, and is about the design of a haptic steering wheel including validation by building a physical prototype. The second design case, which was first presented in appended Paper A, investigates the design of a two axis gantry system. This is a more complex design problem since the system consists of more components, more design variables, and multiple degrees of freedom. The first design case, however, is more complete in the sense that the concept is actually designed, built and integrated into an operational research concept vehicle [65].

4.1. Haptic steering wheel

The haptic steering wheel case studied the design and implementation of a steering wheel for the RCV prototype vehicle described in Section 1.1.1. The RCV is a drive-by-wire vehicle, and as such it has no mechanical feedback from road wheels to driver. This feedback information is important for both safety and driver experience reasons; e.g. Liu and Chang [66] demonstrated that having steering torque feedback results in better curve negotiation and skid recovery by the driver. Due to these reasons, it is important to artificially add steering feedback information through for instance a haptic device.

Two different potential design concepts were evaluated:
1. A planetary gear based servo system consisting of a PMDC-motor, a planetary gearing, a shaft, and an inertia representing the steering wheel.

2. A harmonic drive based servo system consisting of a PMDC-motor, a harmonic drive, a shaft, and an inertia representing the steering wheel.

Two design variables were used; the gear ratio since it has been shown in earlier design cases that it is one of the parameters of a mechatronic servo that affects the total design the most and the shaft radius since it has a significant effect on the stiffness. A total of around 20 component parameters were needed in addition to the design variables. The additional design parameters were chosen according to best guesses or design constraints, e.g. a high quality/precision motor type was chosen and the length over radius ratio of the motor was not allowed to be more than 20.

The steering wheel feedback torque model used directly affects the steering feel and usually depends on vehicle variables such as tire moments, damping, and friction [67]. For the sake of this design case, the steering feedback is approximated with a spring model, i.e. the feedback torque is proportional to the angular displacement of the steering wheel instead of being dependent on vehicle states. The spring model mimics the self-centering behavior of a conventional car’s steering wheel. In addition to the feedback spring model a compensator is used to compensate for inertial effects and losses. The angular position sensor is placed on the motor, and hence the shaft stiffness is not taken into account, causing an inconsistency between the real steering wheel angle and the measured one. To handle this, the maximum difference between measured angle and real angle is constrained during optimization. Figure 4.1 gives an overview of the dynamics.

Small size and high stiffness are generally competing goals for most kinesthetic haptic devices. In line with this, the aim of the optimization in this design case is to minimize the size of the resulting device, while keeping the difference between the measured and real angle below a given threshold.

During repeated optimization runs the optimizer was able to consistently find the same individual minimum for each of the two system concepts. Each optimization run took less than 30 s on a modern PC workstation with a less time efficient, older version of the software tool.
The harmonic drive based system optimization result showed a significantly smaller size than the planetary gear based one, which ended up needing two gear stages in order to achieve the same gear ratio. The harmonic drive concept was hence chosen for further development into a physical prototype.

The physical prototype was designed to use components as close as possible to those resulting from the optimization. Table 4.1 compares parameters of the physical system to the design resulting from the optimization. The harmonic drive parameters are shown with one significant digit whereas the PMDC motor parameters are shown with two. This is due to that repeated optimization runs resulted in a slight variation in harmonic drive length and diameter. However, the total size is the same, i.e. if the diameter was larger the length got smaller and vice versa. The comparison shows that it was possible to implement a real system which is close in sizing to the system resulting from optimization.

Table 4.1. Motor and gear parameters comparison of optimization result and prototype.

<table>
<thead>
<tr>
<th>Property</th>
<th>PMDC Motor</th>
<th>Harmonic Drive</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Opt. result</td>
<td>Prototype</td>
</tr>
<tr>
<td>Length [m]</td>
<td>0,051</td>
<td>0,064</td>
</tr>
<tr>
<td>Diameter [m]</td>
<td>0,034</td>
<td>0,030</td>
</tr>
<tr>
<td>Volume [m³]</td>
<td>46 ·10⁻⁶</td>
<td>45 ·10⁻⁶</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A test-rig, as seen in Figure 4.2 with the steering wheel prototype mounted, was built in order to further investigate the fidelity of the models and method used. Figure 4.3 shows the actual current going into the motor as well as the, by the methodology, estimated current while
following the angular position profile given. Test results show that the approximation method and models for dynamic behavior used during optimization can closely estimate the behavior of the real system. In addition, the design case shows that the methodology and tool prototype are useful in the process of designing mechatronic systems. However, comparing the physical prototype to the optimization results shows the need of extending the component library by adding models for bearings and additional structural components.

Figure 4.2. Steering wheel prototype mounted in test rig.

Figure 4.3. Comparison of current required for load profile measured in the rig and calculated during optimization.
4.2. Two-axis gantry system

Two-axis gantry systems have a wide range of applications in industry such as pick-and-place-machines, 3D-printers, and CNC-machines. These gantry robots are capable of translational movement in two orthogonal axes, i.e. in a Cartesian coordinate space. The purpose of this design case, as described in appended Paper A, was to apply the design methodology to a system with multiple degrees of freedom, more components, and hence more design variables. The linear belt drive models used are not as detailed as the ones used for other components. This however, is not a limitation with respect to the purpose of the design case since the intention is not to build a physical prototype but to investigate how the methodology and tool handles more complex cases.

Each axis of the gantry system concept consists of a PMDC-motor, a harmonic drive, and a linear belt drive. The difference between the two axes is that the Y-axis servo needs to also move the X-axis servo in addition to the load. Figure 4.4 shows an illustration of the gantry system. A total of eight design variables were used; namely the radius and length of the motors, the gear ratios, and the width of the belts. The optimization objective used was minimizing the total system mass while being able to move a 1.1 kg load according to the motion profile shown in Figure 4.5. In addition, the figure also shows the velocity and force profiles as calculated by the tool from the equivalent mass and by integrating the sinusoidal terms resulting from the FFT transformation applied to the position profile.

In order to investigate the dynamics capabilities of the method, the two servo controllers are designed with only the gear and motor of their axis in mind (second order system). This reflects a system with un-modeled
plant dynamics and having the sensor placed at the motor. The un-modeled dynamics include belt stiffness as well as belt and load inertias. Figure 4.6 shows a block diagram of the dynamic system. The resulting closed loop system is of sixth order. The maximum integrated square error (a commonly used measure of control performance) of the closed loop system is constrained in order to fulfil dynamic performance requirements.

In order to gain a better understanding of the system, a single axis servo was first investigated. A belt with low stiffness was used in order to increase the effects of the un-modeled dynamics for the purpose of this analysis. Figure 4.7 and Figure 4.8 show visualizations of parts of the design space. The figures show mass over gear ratio and motor radius while, and while not fulfilling dynamic constraints. In order to make these visualizations, the best possible values for the other two design variables are chosen for each data point. As can be seen in the figures, the
added dynamic constraint significantly limits the allowed solution space, which can make it difficult for the optimizer to even find a single possible solution.

In order to evaluate the fidelity of the dynamics approximation used in the methodology the closed loop single axis system was simulated in Simulink and compared to the approximated response using 5, 15, and 30 harmonics, see Figure 4.9 and Figure 4.10. As can be seen in the figures, using 30 harmonics results in a reasonably good fidelity. The un-damped
The oscillations seen in the figures are due to losses being modeled as efficiency and not damping as well as the use of a very flexible belt.

The results from optimizing the stiffer two-axis system show, as expected, a significantly larger motor, higher gear ratio, and wider belt for the Y-axis servo, i.e. the servo which has to move the other servo in addition to the load, as compared to the X-axis servo. See Figure 4.11 for an overview of the resulting mass distribution among components. The
results also show that the methodology is able to handle un-modeled dynamics well; the ISE constraint applied to the design gave control results very close to what was achieved with full-state-feedback. The optimizer will automatically avoid design variable combinations which result in bad dynamic performance, e.g. if the closed loop poles are too slow compared to the dominant frequencies of the input, since these would violate the ISE constraint.

On a modern desktop PC the optimizer was able to consistently find about the same solution within 60 s while not considering dynamic constraints and within 120 s while considering them. The design case shows that it is feasible to apply the method to larger systems. However, adding additional design variables can significantly affect the optimizer’s ability to find a solution in a short time frame.
Chapter 5
Discussion and conclusions

Patience is a virtue, and I'm learning patience. It's a tough lesson.
— Elon Musk, 2008

This chapter presents the author’s conclusions and thoughts on future work combined with a discussion.

5.1. Conclusions

The overall goals of this research, G1 and G2, as presented in Chapter 1, are to devise and evaluate a new mechatronic systems design methodology that improves product development efficiency through better concept analysis. In order to achieve this, the research questions, Q1-Q3, presented in Chapter 1 have been answered; a summary follows below.

**Q1:** How can mechanical, electrical, and control engineering domains be treated concurrently in order to evaluate the potential of a mechatronic system without increasing the design effort to an unmanageable level?

The presented methodology along with the supporting tool is capable of treating the abovementioned engineering domains concurrently through model based design and optimization. A conceptual system is described by combining a set of components from a library, allowing easy reuse of component models between different design processes and concepts. The goals and requirements of the system are defined and its potential compared to other conceptual candidates is determined by means of optimization. The optimizer determines each component’s
design variables in parallel, i.e. non-sequentially and over domain borders, in order to treat them concurrently. In this way, otherwise unresolved cross-domain dependencies between design variables will not hinder the holistic optimization process. By using efficient models, linear dynamics, and a straightforward system modeling approach the computational complexity of the methodology can be kept at a reasonable level.

Q2: How can a software tool be designed and how can an optimizer be used to support efficient evaluation?

A software toolbox has been developed in order to support and implement the design methodology. A graphical user interface is used to allow easy configuration of conceptual systems, requirements and goals as well as to present optimization results. The toolbox uses symbolic manipulation to simplify and solve algebraic equation systems in order to allow efficient model evaluation.

A flexible and straightforward component model interface allows users to add new types of components and models in a straightforward manner, hence encouraging creation of reusable component models.

A genetic algorithm based optimizer is used for the software toolbox. This non-gradient-based optimizer is able to solve a wide set of complex optimization problems, including the one formulated in this research. The optimizer applies an objective function to each of its proposed solutions in order to evaluate their potential. To do so, the objective function is able to interpret the optimization goals specified by the user, determine which models need to be evaluated, and how to do it.

The method fits well with computation parallelism since all the individual solutions of the population can be evaluated concurrently. This allows time efficient optimization; e.g. the presented design cases required just a couple of minutes for optimization.

Q3: How can the static properties and the dynamics of a mechatronic system be concurrently evaluated sufficiently fast to allow for hundreds or thousands of evaluations per second on a normal computer?

The nature of the software tool facilitates the use of different kinds of models describing static system properties as well as efficient evaluation of these. The component models used in the design cases performed
during this research are based on making use of documented design rules, physics based modeling, and scaling of existing devices, such as motors and gears. This allows sufficient model fidelity for concept evaluation, while still keeping the required computational effort low in comparison to detailed modeling methods such as FEM based ones.

Dynamic behavior is described with linear time invariant differential equations. This allows the software toolbox to symbolically determine controller structures and transfer functions outside the optimization loop as well as to then numerically evaluate these for each potential solution generated by the optimizer. By conducting Fast Fourier Transforms on the input motion profile and other potential exogenous inputs such as disturbances the dominant frequencies are identified together with their individual phase shifts and amplitudes. This allows replacing simulation by approximating the time-domain response of the system to a given input by superimposing the responses of each individual input frequency within the selected set of frequencies. Evaluating the frequency domain response for e.g. 30 frequencies instead of simulating the system response using numerical differential equation solving methods requires significantly less computational effort.

5.2. Discussion and future work

This dissertation can be seen as a major step towards better evaluation of conceptual mechatronic systems, allowing more efficient product development and better products. As with all research, a large number of further research and development issues can be identified. Some of the author’s ideas are discussed below.

5.2.1. Methodology

The presented approach is based on some component parameters, such as gear ratio, being more salient than others, in terms of affecting the specified fitness value, and hence treating these as design variables. However in order for the method to be truly holistic all component parameters would need to be free design variables, something which would be unrealistic even for a small number of components due to the curse of dimensionality mentioned in Chapter 2. A potential addition to the design methodology would therefore be smarter optimization heuristics, e.g. being able to automatically identify which parameters are
more salient and initially put more focus on these during the optimization process. This might allow for an increased number of design variables or shorter optimization time.

Another approach could be investigating if it is possible to further simplify the models used and approximate them with convex functions. This would allow the use of convex optimization methods, which in general are able to optimize problems with larger numbers of design variables. However, since so many of the models are non-convex and even the convex ones, when combined with other component models, are likely to result in non-convex expressions, it is probably difficult to find points where it is possible to approximate all of these with convex polynomials at a high enough fidelity. One possible approach might be a two-step hybrid process, in which a convex optimization step is used to determine rough values for a large number of parameters. The first step would be followed by the currently used global optimizer, which uses higher fidelity models but is more limited when it comes to the amount of design variables it can handle. A pre-processor would in this case be designed to find the best convex approximations for the component models for a given design case before the convex optimizer could be applied.

In addition to improving the optimizer heuristics, other optimization methods, such as simulated annealing, should be investigated in order to determine if they are better suited for the design problem (e.g. by being faster or able to find better solutions). All the design cases studied so far are based on the use of genetic algorithms, but most non-gradient based methods should be applicable to the methodology. In addition, it could be worthwhile to investigate adding a second optimization step (or third if a first convex optimization step is used) in which a local optimizer is used to narrow in further on the optima found by the global optimizer.

Implementing the software as a MATLAB toolbox has many advantages, not least the cross-platform support and familiarity among engineers with MATLAB based tools. MATLAB, however, is not well suited for graphical user interfaces, hence severely limiting the possibilities with it. It would be worthwhile to investigate a standalone implementation of the software tool, making use of open source software libraries, or a hybrid solution making use of the MATLAB backend for calculations. A useful improvement could be SysML integration in order to allow integration into existing design processes. Another suggestion for an improvement would be implementing a 3D visualizer for the designed systems as well as tying this to different post-processing features in order
to better interact with the user as well as to present the optimization results.

An important future step would be a proper usage study of the tool in industry or in a project oriented class of graduate students, in order to identify potential problems and investigate the design process improvements that can be achieved as well as where further development focus should be put.

5.2.2. Component models

The design cases have shown the potential of the models used. The cases have, however, also shown the need for models of more component types, including for instance structural ones. In addition to adding new components, models for more types of properties, such as cost, efficiency, and aspects related to sustainability, should be added. Sustainability could for instance include the efficiency, life span and the environmental impact due to material or production method choices.

Current model implementations assume the mechanical components to be grounded at the motor. This pose an unnecessary limitation since the methodology itself is capable of handling a more flexible ground placement. It would be reasonably simple to change the current models and implement a ground component which could be used to specify where the system is grounded.

In order to improve the dynamics capabilities of the method additional dynamic component models such as controller types, sensors and disturbance models should be added.

5.2.3. Design cases

New mechatronic system design methodologies are seldom evaluated by actually developing physical prototypes using them, but rather through theoretical design cases. Both of these case study types are important in order to verify the usefulness of a methodology and should therefore be performed. The research presented in this dissertation includes several different theoretical design cases, where no physical prototypes are built, as well as one where a physical prototype was developed and built.

The multiple degrees of freedom gantry case presented has coupled static properties between axes but not coupled dynamics; i.e. the second axis mainly affects the first through the addition of its mass. A next step would be to apply the methodology to a dynamically more complex
system, such as a two joint robot arm working in a plane to achieve the same motion as the presented gantry case. This would result in coupled non-linear dynamics which would have to be linearized around a number of working points in order to be evaluated. In addition it would require the use of inverse-kinematics in order to determine joint references resulting in a specified end effector position profile.

An important further step would be studying the use of the methodology in a more realistic design case together with industry partners. This would provide very useful data about the methodology and could be combined with the tool usage study mentioned earlier.
References


Appended Papers