LINKÖPING STUDIES IN SCIENCE AND TECHNOLOGY.
DISSERTATIONS, NO. 1418

Geometry Based Design Automation

Applied to Aircraft Modelling and Optimization

Kristian Amadori

Linköping University

"Geometry Based Design Automation – Applied to Aircraft Modelling and Optimization"

Linköping Studies in Science and Technology. Dissertations, No. 1418

ISBN 978-91-7519-986-3
ISSN 0345-7524

Printed by: LiU-Tryck, Linköping

Distributed by:

Linköping University
Division of Machine Design
Department of Management and Engineering
SE-581 83 Linköping, Sweden

Tel. +46 13 281000
http://www.liu.se

To Isac

A perfection of means, and confusion of aims, seems to be our main problem.
Albert Einstein

ABSTRACT

Product development processes are continuously challenged by demands for increased efficiency. As engineering products become more and more complex, efficient tools and methods for integrated and automated design are needed throughout the development process. Multidisciplinary Design Optimization (MDO) is one promising technique that has the potential to drastically improve concurrent design. MDO frameworks combine several disciplinary models with the aim of gaining a holistic perspective of a system, while capturing the synergies between different subsystems. Among all disciplines, the geometric model is recognized as playing a key role, because it collects most of the data required to any other disciplinary analysis.

In the present thesis, methodologies to enable multidisciplinary optimization in early aircraft design phases are studied. In particular, the research aims at putting the CAD geometric model in the loop. This requires the ability to automatically generate or update the geometric model, here referred to as geometry-based design automation.

The thesis proposes the use of Knowledge Based Engineering (KBE) techniques to achieve design reuse and automation. In particular, so called High Level CAD templates (HLCts) are suggested to automate geometry generation and updates. HLCts can be compared to parametric LEGO® blocks containing a set of design and analysis parameters. These are produced and stored in libraries, giving engineers or a computer agent the possibility to first topologically select the templates and then modify the shape of each template parametrically.

Since parameterization is central to modelling by means of HLCts, a thorough analysis of the subject is also performed. In most of the literature on MDO and KBE two recurring requirements concerning the geometrical model are expressed: the model should be flexible and robust. However, these requirements have never been properly formulated or defined. Hence, in the thesis a mathematical formulation for geometry model robustness and flexibility are proposed. These formulations ultimately allow the performance of geometric models to be precisely measured and compared.

Finally, a prototyping and validation process is presented. The aim is to quickly and cost-effectively validate analytical results from an MDO process. The proposed process adopts different manufacturing techniques depending on the size and purpose of the intended prototype. In the last part of the thesis, three application examples are presented. The examples are chosen from research projects that have been carried out at Linköping University and show how the proposed theoretical results have been successfully employed in practice.

There's a way to do it better - find it.
Thomas A. Edison

SAMMANFATTNING

Kraven på ökad effektivitet utmanar ständigt produktutvecklingsprocessen. I och med att ingenjörsprodukter blir allt mer komplexa, växer genom hela utvecklingsprocessen behovet av verktyg och metoder för integrerad och automatiserad design. Multidisciplinär Design Optimering (MDO) är en lovande teknik som kan drastiskt förbättra parallell design. I ett MDO ramverk är flera disciplinära modeller sammankopplade för att uppnå ett holistiskt systemperspektiv, men där synergierna mellan olika delsystem också kan fångas upp. Bland alla möjliga discipliner spelar geometrimodellen en central roll, eftersom den innefattar en stor del av all information som är nödvändig för andra disciplinära analyser.

I avhandlingen studeras ett flertal metoder för att möjliggöra multidisciplinär optimering i de tidigaste faserna av flygplansdesign. I synnerlighet är forskningen riktad mot att införa geometriska CAD modeller i designloopen. Det blir därmed nödvändigt att kunna automatiskt generera eller uppdatera geometriska modeller, vilket i avhandlingen kallas för ”geometribaserad design automation”.

Avhandlingen förordar att Knowledge Based Engineering (KBE) tekniker används för att konstruktioner skall kunna automatiseras och återanvändas. Så kallade Hög Nivå CAD mallar (på engelska High Level CAD templates – HLCts) föreslås för att automatiskt generera och uppdatera geometrimodeller. HLCts kan jämföras med parametriska LEGO® klossar som innehåller variabler för design och analys. Mallarna kan samlas i bibliotek; därefter har konstruktörer eller dator agenter möjligheten att först topologiskt välja en mall och sedan ändra på dess utförande genom utvalda parametrar.

Eftersom parameterisering är ett centralt begrepp för HLCt principen, föreslås även en fördjupad analys av ämnet. I stor del av MDO och KBE litteraturen ställs det två återkommande krav på geometrimodellen: modellen bör vara flexibel och robust. Eftersom dessa krav aldrig har getts en formell formulering, förordas i avhandlingen en matematisk beskrivning av modellrobusthet och -flexibilitet. Tack vore formuleringen är det möjligt att noggrant mäta och jämföra till vilken grad geometriska modeller fungerar.

Slutligen presenteras en valideringsprocess baserad på kostnadseffektiva prototyper som används för att snabbt bekräfta analytiska resultat från MDO ramverket. Den föreslagna processen nyttjar olika tillverkningsmetoder, beroende på prototypens tänkta storlek och användning. I sista delen av avhandlingen presenteras även tre applikationsexempel, valda från forskningsprojekt som har bedrivits på Linköpings universitet och som visar hur de teoretiska resultaten har kommit till användning i praktiken.

ACKNOWLEDGEMENTS

The research presented in this thesis was carried out at the Division of Machine Design at Linköping University. First, I would like to thank my supervisor Prof. Petter Krus for his guidance and for giving me the opportunity to embark on this long journey. For all discussions, constructive critiques, suggestions and support, I would also like to express my sincere gratitude to my co-supervisors, Prof. Johan Ölvander and Dr Christopher Jouannet.

I would like to dedicate a special mention to the co-writers of all the papers I have both listed and appended to this thesis. Without your contribution my research would certainly have looked very different. In particular, I owe special thanks to Mehdi Tarkian and David Lundström, with whom I have worked closely during the final period of my carrier as a Ph.D. student.

I would also like to thank all my colleagues at the University: I could not imagine a better work environment than the one we have shared. I have received so many suggestions, ideas, hints, and thoughts and so much good advice from so many of you that I would not even know where to begin to address them all. Thank you all!

The research project has received funding from the National Aviation Engineering Research Programme (NFFP) and the ProViking research program, which I hereby gratefully acknowledge.

Last, but certainly not least, I want to thank my wife Ingrid and my wonderful son Isac for motivating me in anything I do and always being a great source of inspiration.

Linköping, December 2011

Kristian Amadori

APPENDED PAPERS

The following papers are appended and will be referred to by their Roman numerals. The papers are printed in their originally published state, except for changes in formatting and correction of minor errata.

[I] Amadori, K., Lundström, D., Krus, P., "Automated design and fabrication of micro-air vehicles", accepted for publication on Journal of Aerospace Engineering, Proceedings of the Institution of Mechanical Engineers Part G [PIG], DOI: 10.1177/0954410011419612

[II] Amadori, K., Tarkian, M., Ölvander, J., Krus, P., "Flexible and robust CAD models for design automation", invited contribution to the KBE Special Issue of the Journal Advanced Engineering Informatics

[III] Amadori, K., Jouannet, C., Berry, P., "Development of a subscale flight testing platform for a generic future fighter", ICAS2010, 27th International Congress of The Aeronautical Sciences, Nice, France, Sept. 2010

[IV] Lundström, D., Amadori, K., Krus, P., "Evaluation of automatically designed micro air vehicles and flight testing", AIAA-2010-1022, 48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, Orlando, FL, USA, Jan. 2010

[V] Amadori, K., Jouannet, C., Krus, P., "Use of panel code modeling in a framework for aircraft concept optimization", AIAA-2006-7084, 11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Portsmouth, Virginia, USA, Sept. 2006

The following papers are not included in the thesis but constitute an important part of the background.

[VI] Melin, T., Amadori, K., Krus, P., "Parametric wing profile description for conceptual design", CEAS The International Conference of the European Aerospace Societies, Venice, Italy, Oct. 2011

[VII] Jouannet, C., Lundström, D., Amadori, K., Berry, P., "Design and flight testing of an eco-sport aircraft", AIAA-2010-1206, 48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, Orlando, Florida, USA, Jan. 2010

[VIII] Lundström, D., Amadori, K., Krus, P., "Validation of models for small scale electric propulsion systems", AIAA-2010-483, 48th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, Orlando, FL, USA, Jan. 2010

[IX] Lundström, D., Amadori, K., Krus, P., "Automation of design and prototyping of micro aerial vehicle", AIAA-2009-629, 47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition, Orlando, FL, USA, Jan. 2009

[X] Jouannet, C., Lundström, D., Amadori, K., Berry, P., "Morphing wing design, from study to flight test", AIAA-2009-1619, 47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition, Orlando, Florida, USA, Jan. 2009

[XI] Amadori, K., Jouannet, C., Krus, P., "Aircraft conceptual design optimization", ICAS2008, 26th International Congress of The Aeronautical Sciences, Anchorage, Alaska, USA, Sept. 2008

[XII] Lundström, D., Amadori, K., "Raven - a subscale radio controlled business jet demonstrator", ICAS2008, 26th International Congress of The Aeronautical Sciences, Anchorage, Alaska, USA, Sept. 2008

[XIII] Lundström, D., Amadori, K., Krus, P., "Distributed framework for micro aerial vehicle design automation", AIAA-2008-140, 46th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, USA, Jan. 2008

[XIV] Jouannet, C., Lundström, D., Amadori, K., Berry, P., "Design of a very light jet and a dynamically scaled demonstrator", AIAA-2008-137, 46th AIAA Aerospace Sciences Meeting and Exhibit, Reno, Nevada, USA, Jan. 2008

[XV] Amadori, K., Jouannet, C., Krus, P., "A Framework for aerodynamic and structural optimization in conceptual design", AIAA-2007-4061, 25th AIAA Applied Aerodynamics Conference, Miami, FL, USA, June 2007

[XVI] Amadori, K., Johansson, B., Krus, P., "Using CAD-tools and aerodynamic codes in a distributed conceptual design framework", AIAA-2007-964, 45th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, USA, Jan. 2007

ABBREVIATIONS

CAD:	Computer Aided Design
CAE:	Computer Aided Engineering
CFD:	Computational Fluid Dynamics
DA:	Design Automation
DR:	Design Reuse
EKL:	Engineering Knowledge Language
FEM:	Finite Elements Modeling
FML:	Fibre Metal Laminate
GA:	Genetic Algorithm
GDA:	Geometry Based Design Automation
GFF:	Generic Future Fighter
HLCt:	High Level CAD template
HLP:	High Level Primitive
KBE:	Knowledge Based Engineering
KBS:	Knowledge Based System
KP:	Knowledge Pattern
KPI:	Key Performance Indicators
MAV:	Micro Air Vehicle
MDA:	Multidisciplinary Design Analysis
MDO:	Multidisciplinary Design Optimization
NAND:	Nested Analysis and Design
PC:	Power Copy
PIP:	Product Introduction Process
SAND:	Simultaneous Analysis and Design
UML:	Unified Modeling Language
VBA:	Visual Basic for Applications

TABLE OF CONTENTS

Part I - Introduction and Aims **1**

CHAPTER I. Introduction 3

CHAPTER II. Aims 7

Part II - Background and Theory **9**

CHAPTER III. Background 11

CHAPTER IV. Intelligent Systems 13

IV.1 What are Intelligent Systems? 13

IV.1.1 Knowledge-Based Systems 14

IV.1.2 Computational Intelligence 14

IV.1.3 Hybrid Systems 14

IV.1.4 Five Categories of Knowledge-Based Systems 15

IV.2 What is Knowledge? 15

IV.2.1 Definition of Knowledge 15

IV.2.2 Types of Knowledge 16

IV.3 Knowledge-Based Engineering 16

CHAPTER V. Design Automation 19

V.1 Geometry-Based Design Automation 19

V.2 Engineering Design Optimization 21

V.3 Multidisciplinary Design Optimization 23

V.3.1 Generic Framework Structure for Multidisciplinary Design Optimization 24

V.3.2 Geometric Modeling Strategies within Multi-Disciplinary Design Frameworks 25

V.3.3 Decomposition of the Multidisciplinary Optimization Problem 27

Part III – Contributions **29**

CHAPTER VI. High Level CAD Modeling 31

VI.1 Morphological Transformations 32

VI.2 Topological Transformations 33

VI.3 Continuity Requirements on Geometry Models 34
VI.4 Power Copy and Knowledge Pattern 35
VI.4.1 The Aim: an Example 35
VI.4.2 Differences between Power Copy and Knowledge Pattern 35
VI.5 Dynamic Top-Down Modelling 36
CHAPTER VII. Parameterization 39
VII.1 What Are Parameters? 39
VII.2 Parameter Selection 40
VII.2.1 Relative and Absolute Measures 40
VII.2.2 Choice of Parameters 41
VII.3 Creating a CAD Model for Design Automation: An Example 42
CHAPTER VIII. Robustness and Flexibility 45
VIII.1 Design Space Size 45
VIII.2 Model Robustness 47
VIII.3 Model Flexibility 47
VIII.4 Example 48
CHAPTER IX. Fast Evaluation of MDO Models 51
CHAPTER X. Application Examples 55
X.1 MAV Design Automation 55
X.2 Aircraft Wing Conceptual Design 60
X.3 Generic Future Fighter 63
Part IV - Discussion and Conclusion 67
CHAPTER XI. Discussion 69
XI.1 Spread of KBE and DA Techniques 69
XI.2 Cost of Design Automation 71
XI.3 High Level CAD modelling 71
XI.4 Robustness and Flexibility 72
XI.5 Design Automation Opportunities 72
CHAPTER XII. Conclusions & Future Work 75
XII.1 Recommended Future Research Directions 76
CHAPTER XIII. Summary of Papers 79
CHAPTER XIV. References 81

PART I -

INTRODUCTION AND AIMS

Part I of the thesis introduces design process methods and challenges. In particular, the role of geometry models is discussed. At the end of this part, the aims of the thesis are presented and the major research questions formulated.

CHAPTER I.

INTRODUCTION

Product development processes are continuously challenged by demands for increased efficiency, both with respect to development time as well as development cost, without losing product quality. In an analysis of the aeronautical industry at the beginning of the 21st century, McMasters et al. [55] observed for example that the industry is not seeking for "Farther, Faster, Higher" as it did during the Cold War. Increasing performance was the main design driver and much less attention was paid to cost aspects. "*For the present, 'cost uber alles' is a central reality, and […] a new mantra for airplane development might properly be 'Leaner, Meaner, Greener' as a somewhat more optimistic prospect for our future – at least technologically*" [55]. Moreover, at the same time as products become more and more technically advanced, the demand for customized variants increases with new product generations and updates. These increasing challenges can typically not be addressed by adding more development engineers to the project. Instead, they must be addressed with more efficient development tools, methods and processes.

Multidisciplinary Design Optimization (MDO) is one promising technique that has the potential to drastically improve the design process. However, conclusive evidences is still lacking, as noted in a recent review of the state of the art in MDO by Simpson et al. [74], who point out that "*despite* [the remarkable advances of MDO in the last 25 years]*, the design of complex engineered systems remains a challenge, and many large-scale engineering projects are routinely plagued by exorbitant cost overruns and delays*". Computational capacities cannot be held responsible, since in the same 25-year period, the abilities of tools that engineers use daily have made incredible leaps forward. It should be sufficient to think of how much more powerful a simple personal computer is today, compared to what was available 25 years ago. In addition to hardware advances, 3D Computer Aided Engineering (CAE) software has given designers unprecedented possibilities. In a presentation by Kraft et al. [39], it was shown how major aerospace projects had continuously shortened computing turnaround time, despite the increase in size and complexity of the high-fidelity modelling involved (see Figure I.1). Nevertheless, as pointed out by Simpson et al., the same projects have indeed been subject to huge delays.

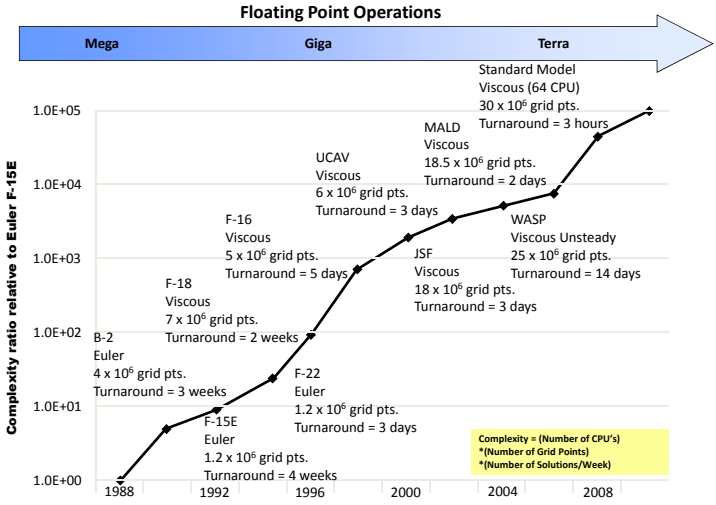

Figure I.1 Trends in constructive, high-fidelity physics modelling, adapted from Kraft et al. [39]

The design process of complex engineering products, such as aircraft, is often divided into three successive steps: conceptual, preliminary, and finally detail design phase. Conceptual design is regarded as the very first stage of the design process. During this phase numerous design concepts are generated and evaluated, to determine whether a particular set of requirements can be met (in terms of performance, cost, safety or any other aspect that may be pertinent) and the associated levels of technology and risk. The key issues of basic configuration layout, size, weight and projected performance are addressed. A large number of widely varying configurations (perhaps ten or more) are evaluated and reduced to one or two basic configurations that will be taken forward to the preliminary design phase.

Following this is the Preliminary Design Phase where the selected concept(s) is(are) looked into in more detail. Detailed analysis and simulations are carried out to fine-tune the geometries, while all sub-systems begin to take shape. The aim of this phase is to completely define the product that is going to be manufactured and to 'freeze' its design.

The final step of the design process is the Detail Design Phase, during which all components and parts are defined in all their details. It is during this phase that all manufacturing documentation (or at least most of it) is produced.

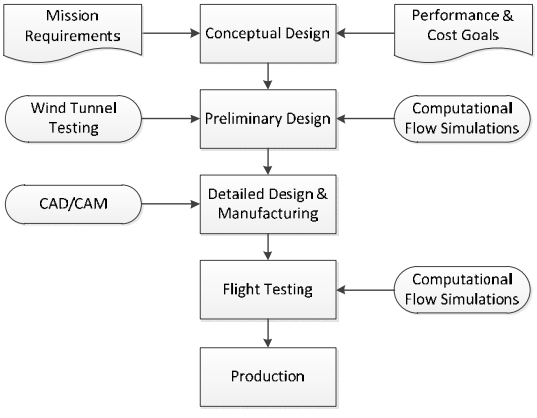

Figure I.2 Aircraft design process as suggested by Brandt et al. [10]

The types of tools that are used are strongly related to the phase of the design process. Typically, the earlier in the design, the simpler the tools are. One of the main reasons for this is that in the beginning the number of people involved and the resources available are very limited while the number of design concepts to evaluate is very large. It is therefore imperative that the tools provide answers as quickly as possible.

Despite computational power having increased drastically, it has not been used to full advantage during the conceptual design. McMasters et al. [54] thought that there should be new ways to use computers to revolutionize the design process, as a complement to, rather than just a copy or an extension of, the process practised by human designers. To increase efficiency in conceptual design, new tools and methods need to be implemented. The current implementation of computational tools during the conceptual design phase has too often been a digitalization of old methods that belong to times gone by or the dawn of the computer age. Even though this has led to an increase in productivity, it has not allowed revolutionary design to be achieved and does not take full advantage of the increase in computational power.

It is not unusual to use fairly simple tools during the conceptual phase. In the aerospace field, semi-empirical or statistical handbook models are among the most common. It should not surprise that Roberts et al. noted that "*the portion of the design process that has been the most lacking in terms of available and implemented computer software is that of conceptual design*" [69].

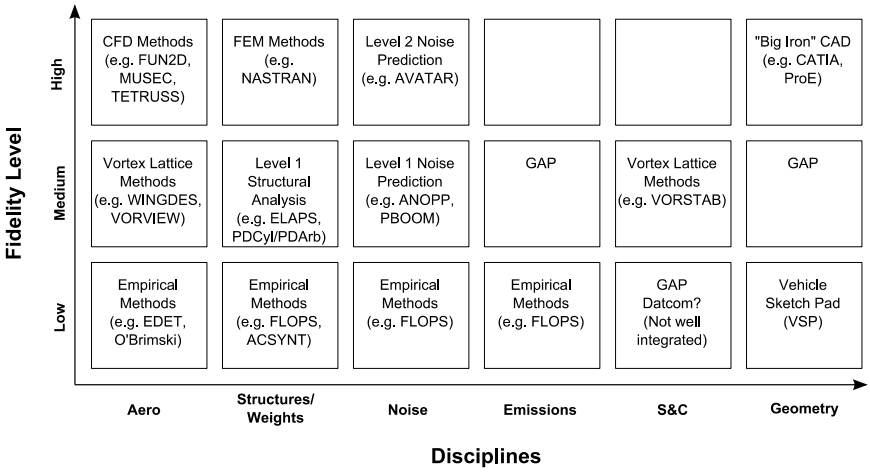

Figure I.3 Design tools classified with reference to disciplines and fidelity level (adapted from Nickol [63])

In the design of new aircraft a large collection of design tools of varying fidelity are used Figure I.3. The process is often disrupted by the need to switch from one tool to another and a more seamless transition would be preferable. In the literature, numerous proposed frameworks can be found that promise to help in that direction (see for instance La Rocca [41], Mawhinney et al. [51], Zweber et al. [90], Manning et al. [49], to name just a few). These frameworks are multidisciplinary, but, interestingly, they all seem to have in common the centrality of the geometric model. It has in fact been pointed out that, in the majority of cases, regardless of discipline, basically all analyses require information that has to be extracted from the geometry model [51]. There are different ways to include a geometry model in an MDO framework (CHAPTER V will briefly present different strategies). Among them, the most complex one, but with the greatest potential, advocates the geometry model to be included in the design optimization loop. Then, the ability to automatically generate or update the geometric model becomes essential, as recognized by several authors (Wakayama et al. [85], Hönlinger et al. [30], Giesing et al. [27], La Rocca [41], to name just a few). This means that it is of necessity to integrate an automated and parametric geometry generation

system into the design framework. On this front, computational advancements are continuously paving the way for new design tools, both providing greater model fidelity and increasing the prospect of design reuse and automation [51]. The last mentioned can also be used to abolish routine-like tasks that reduce human errors while increasing the possibilities to design more customized products [37].

To eliminate non-creative work, methods for creation and automatic generation of High Level CAD templates (HLCts) are suggested [II]. The principle of high HLCts is similar to High Level Primitives (HLP) suggested by La Rocca [43], with the exception that HLCts are created and utilized in a CAD environment. Otherwise, the basics of both HLP and HLCt can, as suggested by La Rocca, be compared to parametric LEGO® blocks containing a set of design and analysis parameters.

Mathematical models are also proposed to verify and measure the quality of geometry models. They are as simple as possible to ensure ease of application and are intended both as an aid during the modelling process and as a way to assess whether a geometry model fulfils the requirements or not.

The importance of well-considered parameterization is also analysed, together with a discussion on proper driving parameters choice and definition.

The thesis presents general methodologies and results that are valid in most engineering fields. However, the aeronautic field is often referred to, since the research projects that have led to this thesis have mostly been related to it.

CHAPTER II.

AIMS

This thesis deals with the adoption of multidisciplinary methods during the conceptual and preliminary design phases, and in particular in the field of aircraft design. A first aim is to develop a generic modelling methodology that enables geometry-based design automation. Such a methodology must provide guidelines for designers to generate highly flexible, yet robust, CAD models. A second aim is to study if and how these geometric models can be included within a multidisciplinary design optimization framework. Finally, the thesis intends to address the use of low-cost prototyping as a cost effective means to support conceptual design.
The principal research questions can be formulated as follows:

[RQ1] How should a geometric model be built to be suitable for multidisciplinary design optimization and geometry-based design automation?

[RQ2] Is it possible to quantify the quality of geometric models with reference to multidisciplinary design optimization and geometry-based design automation?

[RQ3] Which generic geometry modelling methodology could enable design reuse?

[RQ4] How could the use of low-cost prototypes support the development process?

PART II -

BACKGROUND AND THEORY

Part II of the thesis consists of a review of existing and relevant literature, highlighting the gaps that the present work is intended to fill. Important theoretical concepts are briefly introduced as a base for the contributions that are presented.

CHAPTER III.

BACKGROUND

In discussions of parameterization and flexible geometry models in the available literature, it can be observed that it is not uncommon to focus entirely on morphological changes, while topological ones are discussed only marginally, if not even banned as not allowed. A detailed discussion about different kinds of geometrical transformations is left to CHAPTER VI, while in the present section some literature examples are presented. For instance, Chang and Silva [14] acknowledged that flexible geometry models are necessary since the design process is by nature iterative and "*design changes are frequently encountered in the product development process*". They proposed a set of guidelines to parameterize a geometric model in the most effective way to allow morphological transformations, but almost ignored topological ones. Bowcutt [9] went even further in a paper on future aircraft design methods and tools, in which the author listed requirements that CAD systems should meet to be suitable as MDO enablers from the earliest design phases, such as feed all analysis codes, support multiple engineering views or satisfy shape constraints, etc. Some of the requirements necessitated the model to be continuous and differentiable over the whole input parameters' range, which is not compatible with the topological transformations that involve discrete and therefore non-continuous variable ranges.

Parameterization of geometrical models is frequently discussed in connection with MDO applications. Thanks to greatly improved computing resources, researchers have been trying to include the geometric model inside MDO frameworks. This, in turn, needs what can be referred to as automated design and reuse, i.e. automatic generation, modification and transformation of geometrical features in the models, to allow design space exploration not limited to portions close to a baseline configuration. To accomplish automated design and reuse, knowledge of the design process has to be captured, stored, and deployed as presented by Hopgood [31]. This process can be referred to as Knowledge-Based Engineering (KBE).

In the aircraft design domain, great strides have been made in describing methods for creation of automatic generated geometries. Mawhinney et al. [51] proposed dividing the aircraft geometry model into three sub-models, viz. skeleton, surface and solid models, in order to utilize each model in different analysis disciplines. They claimed that by generating displacement, rotation and profile components, aircraft geometries such as fuselage and wing can be generated for a wide range of concepts. The methods described are sufficient since these are used for conceptual design work and none of the more complex geometrical features that are required in later design phases have to be generated in this early stage. Consequently, the methods cannot be used in a more general product design outline and are impracticable if fidelity enhancement is required.

Rodriguez et al. [70] propose RAGE (RApid Geometry Engine) in order to produce the geometry of the aircraft. By implementing a lofting algorithm, different kinds of cross-sections are used to create surfaces for the various parts of the aircraft. Similar to Mawhinney, Rodriguez's methods are not practicable when a more detailed model is being sought.

La Rocca et al. [42] go one step further and suggest using high-level geometry primitives to support designers on producing automated aircraft geometries. This means that the user is able to pick parameterized geometric elements from a database by using a dedicated parameter in order to achieve concept realization instead of constructing the elements with CAD-based functions such as points, lines and curves, etc. This modelling approach has proven to be effective and time-saving when performing MDO, partly because the geometry is produced much faster than with a CAD tool. However, because of the code-based nature of this application, expanding fidelity is limited and dependent on the code writer. The fact that engineers will have to wait to get new primitives of higher fidelity will ultimately slow down the design pace.

Bérard et al. [6] present methods similar to those of La Rocca where the idea is to eliminate CAD-based functions and allow designers to work with intuitive engineering parameters by storing various parameterized CAD parts of an aircraft as components. However, instead of generating the components in a code-based environment, these are stored in component libraries and automatically generated in commercial CAD tools. The generated components will then undergo a Boolean operation where interfaces between the components are made when eliminating the additional material, resulting in a closed solid model. The idea of using templates to generate the aircraft product is well motivated. However, the methodology adapted to generate the components makes it very hard to firstly define smooth interfaces between the templates and secondly further increase the fidelity level of the aircraft.

In the work presented by Böhnke et al. [8] an aircraft design language is proposed, which is based on the Unified Modelling Language (UML) and allows the creation of a variety of conventional and unconventional aircraft designs. Similar to Bérard's work, the geometry constructor is able to create the geometry in commercial CAD tools. Here, however, the geometries are connected by defining intersection elements which generate higher fidelity models. However, it can be argued that Böhnke has succeeded in creating a tool for a specific application, viz. for geometry generation of aircraft, rather than a general modelling methodology.

Ledermann et al. [45] and [46] make a genuine effort to try to categorize CAD modelling and introduce the benefits of template modelling in CAD tools to develop associative and parametric methods for aircraft design. The methodology proposed in this thesis is partly based on the findings of Ledermann et al.

Quite recently, a series of doctoral theses in the field of design automation has been published by Chalmers University of Technology. The first one, by Cederfeldt [13], investigates how to identify and plan design automation within a given industrial process. This is a highly relevant subject that other researchers have also identified (see for instance Blount et al. [7]). It will, however, be touched upon only marginally in this thesis, that instead focuses more on enabling methods for automatic geometry generation.

Closer to the present thesis is the work by Stolt [78], who wrote his thesis on methods and techniques to reuse geometrical models and engineering knowledge to speed up the design process. The envisioned solution consists of converting existing data in a specialized format, from which geometries can later be re-generated in a selected target CAD system. Finally, Johansson's work [36] describes a knowledge-based framework to allow analysis of manufacturability aspects as well as tooling design. While impressive, it can be argued that the work by Stolt and Johansson is less general from a mere geometry modelling perspective, than the hereby proposed techniques.

CHAPTER IV.

INTELLIGENT SYSTEMS

IV.1 WHAT ARE INTELLIGENT SYSTEMS?

Since computers have become available, their use and applications have been growing at an impressive speed. Today, there are not many fields where computers are not involved. It is nevertheless worth looking at how they are most often employed. Hopgood observed that "*computers are able to perform the same sort of mathematical and symbolic manipulations that an ordinary person can, but faster and more reliably. They have therefore been able to remove the tedium from many tasks that were previously performed manually*" [31]. In other words, computers are used to repeatedly and very rapidly perform tasks that a human user has predefined. However, no intelligence is required on the part of the system, since it is simply doing what it is told. The idea of creating intelligent machines is on the other hand the aim of a branch of computer science that is usually referred to as Artificial Intelligence [53]. The ultimate goal is to be able to build a machine that can think and that can outperform the human brain's intelligence [40]. Impressive results have been achieved in the field (see for instance [33]), but there is still a very long way to go. Of interest for the present thesis are the computing tools that have been developed in artificial intelligence science. Hopgood proposed classifying them into three main groups: knowledge-based systems, computational intelligence and hybrid systems [31].

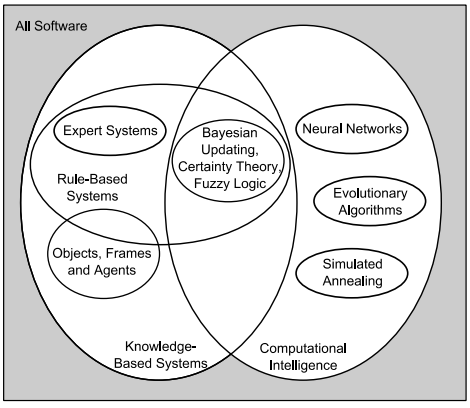

Figure IV.1: Classification of Intelligent Systems into Computational Intelligence, Knowledge-Based Systems and Hybrid Systems (adapted from Hopgood [31])

IV.1.1 Knowledge-Based Systems

Knowledge-Based Systems (KBS) are characterized by a distinct separation between the stored knowledge (called a knowledge base) and the part that controls its application (called an inference engine). In the knowledge base, the knowledge is explicitly recorded so that its management is simplified and easy to understand. The knowledge is stored in a declarative form, so that adding, modifying or removing it does not require any particular programming skills. Moreover, in the knowledge base, no information has to be provided regarding the use of the stored knowledge. On the other hand, it can contain facts as well as rules. Facts can be expressed with varying degrees of certainty, requiring different techniques to be handled (Bayesian updating, certainty theory or fuzzy logic). Moreover, facts and rules can be static or transient, depending on how frequently they need to be updated. Concerning knowledge-based systems, the following terms are of special interest for the present work:

- *Object*: collection of properties (object attributes) and behaviours (object methods) that can be performed on the object itself upon request. Objects are components that have been developed in computer science to be able to create simplified and structured descriptions of complex systems.
- *Attribute*: property of an object instance or class.
- *Relationship* (or *relation*): logical connection between instances or classes.
- *Instance*: one particular object belonging to a class or created using a class as blueprint.
- *Class*: "objects that represent the same idea or concept are grouped together as a class" [31]; a class can be thought of as a general description of a set or family of objects and can contain not only a description of the attributes that characterize the class, but also instructions as to which operations or actions can be performed. A class is a sort of template or blueprint that can be used to create new objects.

IV.1.2 Computational Intelligence

Computational intelligence refers to methodologies that mimic nature in order to solve complex computational problems. As shown in Figure IV.1, computational intelligence methods include evolutionary algorithms, neural networks and simulated annealing. In the first, the problem is solved using natural evolution principles, through mathematical processes that simulate sexual reproduction of individuals within a population [64]. Artificial neural networks are computational schemes that are designed to mimic the human brain. They consist of a set of non-linear software elements (neurons) connected to each other through weighted links. Clearly the size and complexity of artificial networks do not compare to a real brain, but they can still be very useful in certain applications, particularly classification [31]. Compared to knowledge-based systems, computational intelligence differs mainly with regard to where and how knowledge is stored in the system. As explained in section IV.1.1, in a knowledge-based system, the knowledge is explicit and kept distinct and clearly separated from the inference engine. On the other hand, in computational intelligent systems, the knowledge is not explicit but intertwined with the system itself.

IV.1.3 Hybrid Systems

Very often the problems to be solved present varying characteristics that are best handled with combinations of different computational schemes, in order to take advantage of the strengths of them all. There are no actual limitations on how knowledge-based and computational intelligence systems can be combined into so-called hybrid systems.

In the present work, examples of hybrid systems will be discussed in CHAPTER X. In the Micro Air Vehicle design automation framework, computational intelligences (a mix of genetic algorithms) have been used to carry out design optimization of an aircraft and its propulsion system. The framework, however, also contains a combination of knowledge-based systems. The geometry model captures design rules and heuristic knowledge, but also takes advantage of object-oriented strategies.

IV.1.4 Five Categories of Knowledge-Based Systems

It is relevant to identify to what extent knowledge-based systems are used in industry today. Starting from a detailed "*classification system for KBS according to the capabilities they possess*", Roberts et al. tried to map what systems are available and how common their use is [69]. According to their findings, a KBS can be divided into five categories (see Figure IV.2).

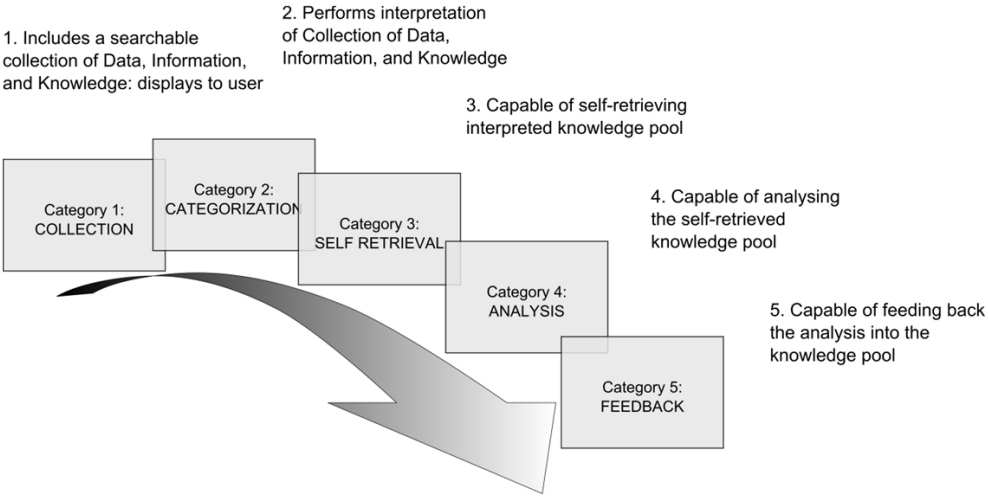

Figure IV.2 Knowledge-Based Systems can be divided into five categories, based on their capabilities (modified from Roberts et al. [69])

The Knowledgeware workbench [12] offered in CATIA V5 can be mentioned as one example of self-retrieval systems. The tools in the workbench have been widely used in the application examples discussed in CHAPTER X in this thesis. Moreover, CHAPTER VI will propose a general modelling methodology which takes advantage of the capabilities that are available in CATIA's workbench.

IV.2 What is Knowledge?

In all intelligent systems the knowledge that is added to or acquired by the system plays a central role. It is therefore relevant to ask what knowledge is. An exhaustive answer to the question is far too demanding and not possible within the scope of this thesis. However, this section will present a very brief introduction to some of the discussions concerning the concept of knowledge that are relevant to the topics in this thesis.

IV.2.1 Definition of Knowledge

Trying to provide a satisfactory definition of knowledge is not easy. Epistemology is a whole philosophical field that is dedicated to the study of what knowledge is and how it is acquired. Tentative definitions can be traced back to Plato or even further back in history. Nevertheless, it is important to explain what will be referred to as knowledge in this work. The Oxford Dictionaries [65] define knowledge as following:

> *"1. Facts, information, and skills acquired thorough experience or education; the theoretical or practical understanding of a subject*
> *2. Awareness or familiarity gained by experience of a fact or situation."*

Other definitions also exist, for instance in the Merriam-Webster's Collegiate Dictionary [56], where it is stated that knowledge is:

"1. The fact or condition of knowing something with familiarity gained through experience or association
2. The fact or condition of being aware of something."

Both definitions assign a central role to experience and understanding of facts. To further clarify the topic, Mullins proposes an interesting description of the path from data to knowledge and wisdom [61]. The lowest level is represented by the raw data that is a simple collection of mere facts. Adding a context and creating relationships between data transform data into information. Furthermore, when we acquire an understanding of the information we can say that we have obtained knowledge. Finally, the highest level is represented by wisdom that requires the knowledge to be applied. From an engineering point of view, the definition of knowledge provided by Milton [58] may seem attractive: "*Knowledge is like a machine in a person's head*" that is able to transform raw data into actions and decisions.

IV.2.2 Types of Knowledge

In addition to the discussion about what knowledge really is, an analysis of the different types of knowledge can also be found in the literature. Awad [3] divides knowledge into three types: *tacit*, *implicit* and *explicit*. Tacit knowledge is the type of knowledge a person has but does not necessarily know that he or she possesses. It is very hard to explain and divulge and is often linked to practical skills. Polanyi [67] wrote about tacit knowledge that *"we know more than we can tell"*. Explicit knowledge, on the other hand, is well-documented and organized. This is the only type of knowledge that can be transferred to other people or to computer programs. Finally, implicit knowledge is the type of knowledge that is half-way between tacit and explicit. Frappaolo [24] describes it as "*knowledge believed to be tacit*" but that "*can be transformed into explicit knowledge*" through "*some sort of mining and translation process*".

IV.3 KNOWLEDGE-BASED ENGINEERING

Knowledge Based Engineering (KBE) has been identified as a promising method to enable design automation. KBE defines a wide range of methods and processes and can be described in several ways, depending on the application focus. In the literature, various definitions can be found that try to highlight the multiple sides of KBE. As recently noted by Verhagen et al. [83] in a comprehensive review of the field, there is no largely agreed definition of KBE. As Amnar-Khodja et al. [1] put it, "*there is no unambiguous definition of KBE. However, most of them are similar*".
Chapman et al. [16] define KBE as "*an engineering method that represents a merging of object oriented programming (OOP), artificial intelligence (AI) techniques and computer-aided design technologies, giving benefit to customized or variant design automation solutions*". According to Blount et al. [7], KBE is a "*true integrator throughout the Product Introduction Process (PIP) supporting the ideas of concurrent engineering*". Furthermore, Verhagen et al. [83] state that "*one of the hallmarks of the KBE approach is to automate repetitive, non-creative design tasks*", which can lead to "*significant cost savings*" and "*free up time for creativity*".
Through KBE, geometric models are given the knowledge (i.e. rules and instructions) which control the geometrical transformations required for automating portions of the design process.
Automated geometry generation with high-fidelity Computer Aided Design (CAD) tools is one area where modelling methods require substantial development efforts. In this thesis, novel methods to eliminate non-creative geometric modelling by performing reuse on specific design features are presented and discussed in CHAPTER VI. The goal is to allow engineers to work on a higher abstraction level where the use of low level CAD functions (i.e. points, lines, sweeps and extrusions) during the modelling and simulation phase is minimized if not fully eradicated. The automated geometry generation is a key enabler for so-called geometry-in-the-loop [44] multidisciplinary design frameworks, where the CAD geometries can serve as framework integrators for other engineering tools. Section V.3 will describe how MDO frameworks can be structured depending on the relation to

the geometric model. A brief discussion of the consequences of the chosen approach will also be provided.

CHAPTER V.

DESIGN AUTOMATION

During the design process, numerous variants, alternatives and possible solutions are explored and compared in an iterative manner, until the product is found that best suits the requirements. Moreover, changes to the design can come after the product has hit the market, as further developments, derivatives and updates are requested. Complying with and implementing all the changes is a very time-consuming and costly activity [55]. Enabling automation of as many as possible of the repetitive processes during product development is a key to increasing productivity, reducing time-to-market and cutting costs. Additionally, freeing designers from tedious work allows them to concentrate on activities that require creativity, which is hardly possible to integrate in a computer system. In this context, this chapter will discuss three enabling technologies: multidisciplinary design optimization (MDO), design automation and design optimization.

V.1 GEOMETRY-BASED DESIGN AUTOMATION

The notion of *design automation* has been mentioned in the previous sections, but not yet properly addressed. Talking about design automation without further explaining its meaning can lead to misconceptions and ambiguities. There are multiple degrees of automation and different meanings of the term *design* that produce a large variety of possible interpretations. There are also several examples in the literature that document the successful application of design automation to a variety of design processes [13][78][36][47][48][14][62].

Generally speaking, design automation refers to a system that is able to perform a design task in an automatic fashion. The system includes a model that is able to transform inputs (in the form of design requirements) into desired outputs. A design automation system can thus be as simple as an Excel spreadsheet to design cantilever beams. Given material and geometry data, outputs such as deflection and maximum stress can be obtained through simple equations.

It is important to make some observations:

- The above description does not say anything about the type and nature of the system, or about the type and number of models used.
- The description does not require or exclude the use of optimization algorithms to search for suitable inputs that yield desired outputs. The system can just as well be used to perform trial-and-error types of search.
- The description does not specify the nature of the generated outputs. To give just a few examples, they might be dimensions (i.e. thickness and size of a beam's cross section), or complete CAD models of a product, or parameters to be used in a manufacturing process.

Conversely, in the present thesis, design automation refers to a process that must include in-the-loop a geometric CAD model of the product at hand. Hence, Geometry-Based Design Automation (GDA) is proposed as a more precise designation. Referring to the modelling methods presented in CHAPTER VI, it can be stated that a GDA framework operates as follows (Figure V.1):

- Required inputs are received from the user/designer
- Inputs are processed according to predefined strategies
- Necessary HLCts are retrieved from a database following the instructions stored in the inference engine
- HLCts are used to generate the geometry model of the product
- The generated geometry model is used to carry out analyses of the product's performance.

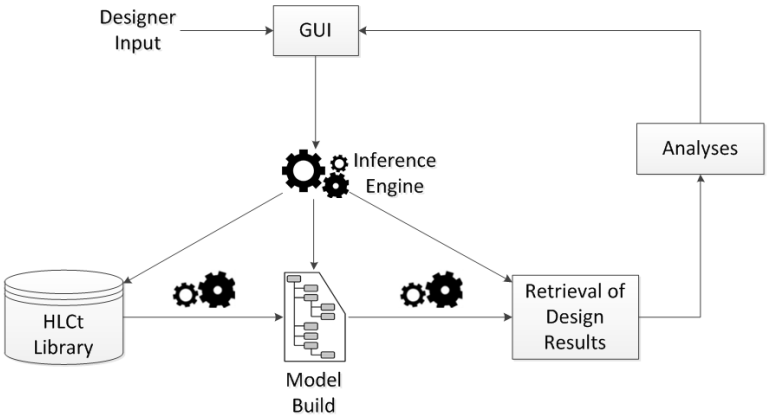

Figure V.1 Geometry-Based Design Automation process

The data processing and analyses necessary to produce the output can be of various kinds. The process can utilize handbook equations just as well as high-fidelity analyses (i.e. FEM or CFD) to generate all required data. For example, in the MAV application described in section X.1, starting from a mission description, a design framework uses several coupled computer programs to generate a CAD model of the optimum geometric design, as well as a list of existing off-the-shelf components for propulsion.

It is interesting to observe that the use of CAD modelling in GDA is not unlike what Skarka defines as "*generative models*" [75] which are "*models of a designed class of products*". What Skarka means is that a generative model does not represent only an instance of a class (i.e. a specific product) but rather the entire class of products. Generative models incorporate design rules and engineering knowledge that are required to determine how a given instance should be.

The proposed definition of GDA does not imply that the process needs to involve a complete system, but can be applied to the automatic design of a sub-system. For instance, to comply with the definition, a design automation process does not necessarily need to involve a complete aircraft, but can just as well be limited to the nose landing gear sub-system. The selection of a suitable design case to which automation strategies can be applied is important but difficult. As long ago as 1995 Blount et al. [7] noted that the failure of the introduction of KBE and automation techniques "*can frequently be attributed to the inappropriate selection of suitable applications*". Similarly, Verhagen et al. [83] have remarked that "*currently, no solid framework or method is available to determine whether a design task, product or process is suitable to develop a KBE application for*".

Another important notion that is frequently recurring in this thesis is *Design Reuse* (DR). DR is related to design automation, and even though it also refers to the production of geometric CAD models, there are some differences. DR implies the reutilization of saved geometric feature templates (HLCts) to aid the geometric modelling process. But DR does not require the automatic generation of

geometries. An HLCt can be done be deployed manually or automatically though scripts. Similarly, DR does not require the morphological transformation of an HLCt instance to fit its context to be carried out automatically. These operations could also be performed manually.
GDA therefore requires DR, but DR does not necessarily imply GDA.

V.2 ENGINEERING DESIGN OPTIMIZATION

A brief reflection on the term optimization is necessary, since it is so often used in the present thesis. In general, optimization can be thought of as the search for the "best" element(s) from a given set. From this simple definition, three cornerstones can be pointed out:

- Search process: there are a large number of different methods, strategies and algorithms available, the choice of which is highly dependent on the nature of the problem at hand. In this section a short overview of the most commonly adopted algorithms is presented.
- Best element(s): depending on the type of problem, the goal can be to find one or several elements that fulfil the requirement of being the best. It is essential to note that this requirement is relative to how the problem has been defined. There is no such thing as an absolute best.
- Given set: the set from which to select the best element(s) confines the search. In the present thesis it is referred to as design space.

Moving from a general discussion to one closer to the engineering domain, some other elements of optimization can be pointed out. The aim of optimization is to find a well-working solution to a problem. This requires translating the real problem into a model that can be studied. The optimization process will search for a solution by modifying the model's behaviour through so-called design variables. Design variables are "*quantities whose numerical values will be determined in the course of obtaining the optimal solution*" [64] and that completely specify the model. The design space is then the domain containing all the allowed values of the design variables. If the design space is unrestricted, the optimization problem is said to be unconstrained. This occurs rarely, since most problems are constrained.
As previously stated, being the best element is a relative condition that refers to what is generally called the objective function, which is a mathematical formulation of the criterion that somehow measures the goodness of a solution. With respect to the objective function, optimization problems can be single-objective or multi-objective, depending on the number of problem objectives. It is obvious that even small changes in the formulation of the objective function can have substantial effects on the optimal solution to be selected. Objective functions are sometimes described as well- or not well-behaving, which indicates qualitatively how difficult it is to search for optimal solutions. A constrained optimization problem is thus generally defined as following:

$$\begin{aligned} &\min_{x \in R^n} f(x) \\ &s.t. \quad c_i(x) = 0, \qquad i \in \mathrm{E} \\ &\qquad\; c_i(x) \geq 0, \qquad i \in \mathrm{I} \end{aligned} \tag{1}$$

In equation (1), x is the n-dimensional design parameter vector; $f(x)$ is the objective function to be minimized while $c_i(x)$ are functions describing the constraints of the problem at hand; E and I are finite index sets. Both $f(x)$ and $c_i(x)$ are real-valued and smooth. As the equation shows, constraints can be expressed both as equalities or inequalities.
Depending on the problem at hand, different solution approaches can be chosen. For instance, the choice can be based on type of problem, number of design variables, nature of the objective function, or number of objectives. Figure V.2 shows an overview of optimization strategies commonly used.

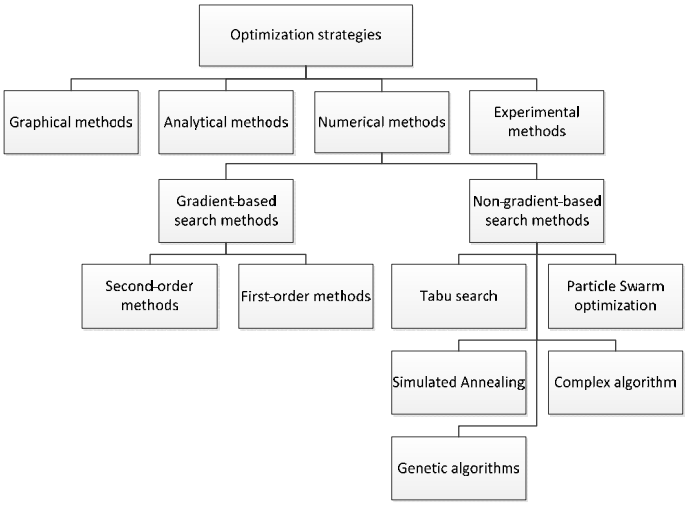

Figure V.2 Overview of some common optimization strategies available

In the case of MDO and design optimization, numerical methods are the most frequently adopted strategies. As can be seen from Figure V.2, they can be further divided into sub-categories. A description of them all would be outside the scope of this thesis; the focus in this section will therefore be on Genetic Algorithms (GA), which are a particular type of non-gradient-based search methods that have been largely employed in the examples reported in CHAPTER X.

GAs are mathematical schemes that try to replicate the natural selection process to search the extremum of the objective function. Each possible solution or point in the *n*-dimensional design space is represented by an individual, whose *n* genes correspond to the *n* design variables. Genes can be coded as numbers, strings or bits, depending on the nature of the problem. To be able to carry out the optimization the algorithm requires a population of individuals to be initially defined. The idea is then to mate the best individual in the population, generating new individuals that are hopefully better than their parents. There are several mechanisms through which the mating can be performed, and different ways to eliminate the overflowing individuals from the population, whose size is kept constant through the optimization. It is even possible to define mutation factors that randomly change some genes during the creation of new individuals, which increases the algorithm's robustness and capability to explore the whole design space.

The strength of GAs is their ability to locate the optimal solution even in problems that do not possess well-behaved objective functions. On the other hand, they are usually computationally heavy, requiring the objective function to be evaluated for each individual at every generation. Onwubiko [64] suggests population sizes between 20 and 100 individuals, depending on the number of optimization variables involved. Clearly, a large population implies a longer time to get to convergence, due to the larger number of objective function evaluations.

One particular type of GA has been used in the MAV Design Automation framework, reported in section X.1. The optimization problem demanded two conflicting objectives, thus requiring a GA able to perform multi-objective optimization. One essential peculiarity of this kind of problem is that there cannot be a single solution, but a set of them, usually called a Pareto front. In the 2-dimensional case, the Pareto front is a curve where all non-dominated designs are located. A design is said to be non-dominated if there is no other design that is "*better or equal in all attributes, and strictly better in at least one attribute*" [2].

V.3 MULTIDISCIPLINARY DESIGN OPTIMIZATION

In the design of complex and tightly integrated engineering products it is essential to be able to handle cross-couplings and synergies between different subsystems [9]. Typical examples of such products are transportation vehicles like automobiles and aircraft, or mechatronic machines like industrial robots. It is necessary to combine models from several disciplines in order to fully understand the system. The subsystems in general have conflicting optimal solutions; hence a holistic perspective over the complete system is necessary to reach a balanced global optimal design, rather than developing the different subsystems independently [15]. Optimizing the subsystems separately would most likely lead to a sub-optimized system, since "*the optimal system-level design is not simply a collection of individually optimized subsystems*" [19].

To effectively design and develop such products, efficient tools and methods for integrated and automated design (see section V.1) are needed throughout the development process. Multidisciplinary Design Optimization (MDO) is one promising technique that has the potential to drastically improve concurrent design. Giesing et al. [27] have defined MDO as "*a methodology for the design of complex engineering systems and subsystems that coherently exploits the synergism of mutually interacting phenomena*". MDO is a "*systematic approach to design space exploration*" [82], the implementation of which allows the designer to map the interdisciplinary relations that exist in a system. Relations and dependences between different subsystems may also be captured by an MDO process. Hence, enabling the introduction of MDO in the early design phases can increase the level of knowledge that is captured. In an early study on blended-wing-body aircraft design, Wakayama et al. [85] established that MDO offers the ability "*to handle many more degrees of freedom and track many more interactions across disciplines than conventional advanced-design processes*". At the same time they acknowledged that it is important to guarantee that enough disciplinary requirements and variables are accounted for ("*breadth*") while being able to use models accurate enough to capture all critical phenomena and behaviours ("*depth*").

Although promising, MDO techniques are not yet and commonly adopted and widely used in industry, the main reasons being the following:

- MDO frameworks should be based as much as possible on higher fidelity models, but are often limited to lower fidelity ones in order to reduce processing time within reasonable limits;
- Reliability and trustworthiness of results: "*an MDO process should be able to accommodate the detailed design processes normally used by engineers within the company to access and validate their products*" [5];
- Limited educational resources in the industry, since "*successful application of MDO* […] *requires a broad range of engineers, specifically discipline engineers, who understand MDO concepts and methods*" [19];
- Strategies and capability to handle very large numbers of design parameters and constraints that are required to ensure that all critical disciplines are included.

However, looking at historical data, at least in the aircraft industry, a positive trend towards broader and wider adoption of MDO can be observed, see Figure V.3. Work is still needed to raise MDO from being a technique with great potential to assume a central role in industrial applications as a mature and trusted design tool.

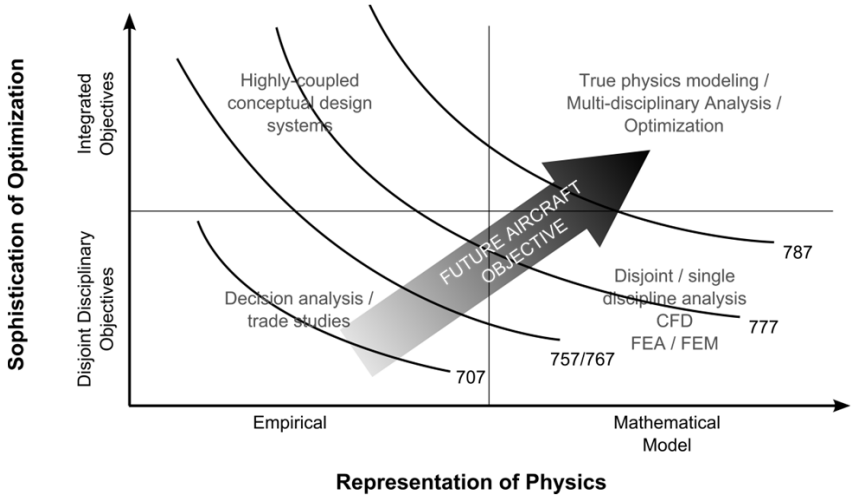

Figure V.3 Trend in MDO application in the aircraft industry (adapted from De Weck et al. [19])

V.3.1 Generic Framework Structure for Multidisciplinary Design Optimization

The traditional legacy aircraft design process, as described by Jenkinson et al. [35] or Raymer [68], is centred on the designer, who takes advantage of his/her experience to generate a baseline initial configuration. Thereafter, through successive refinements, the concept is optimized, small steps at the time, iterating single-disciplinary analysis and evaluating the results. The task is very difficult, at least for two main reasons. Firstly, each adjustment that introduces an improvement in one aspect may affect others negatively, due to the highly integrated nature of aircraft system performance. Secondly, because of the large number of disciplines and requirements that must be taken into account. The complexity of the design space is such that it is practically impossible to efficiently and accurately explore it all, also because grasping its boundaries may be a challenge as difficult as the design of the new aircraft itself. This can be one of the reasons why since their appearance in the middle of last century, jet transport aircraft have looked almost the same, with no real conceptual difference. Over a period of sixty years we have seen a steady evolution of a concept born with Boeing's 707, that still lives on in the latest Boeing 787 and Airbus A380.

An MDO approach differs deeply from the process described earlier. As explained in detail by Vandenbrande et al. [82], it is generally based on three elements (see Figure V.4):

- a design explorer that can select appropriate design points $\{x_1, x_2, \ldots, x_n\}$ to be fed to the system model
- an optimizer that controls the design points' selection, with the purpose of efficiently exploring the design space to ultimately find the "best" concept
- a multidisciplinary design analysis response model (MDA) that is able to capture and simulate all critical aspects of multiple disciplines; it can be modified and updated to fit the chosen design points and it provides the simulated system responses $\{f_1, f_2, \ldots, f_m\}$

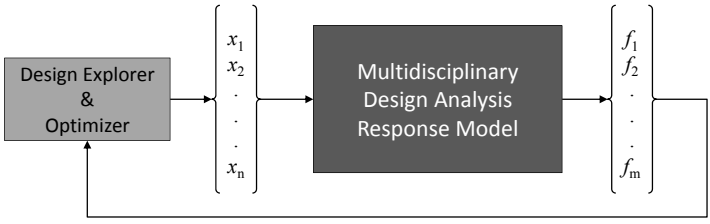

Figure V.4 General representation of an MDO framework (adapted from Vandenbrande et al. [82])

One fundamental benefit of the MDO approach over the previously mentioned legacy method is that it enables "*a systematic search through design space*" [82]. The quality of the results is directly dependent on the ability of the optimizer to control and direct the search, and on the breadth and width of the MDO, as previously reported in section V.3.
The next sections will briefly discuss how the framework can be designed with respect to implementation strategies and decomposition of the multidisciplinary optimization problem.

V.3.2 Geometric Modeling Strategies within Multi-Disciplinary Design Frameworks

Multi-disciplinary frameworks can be arranged in different ways. Since it represents a core topic of the thesis, in this section the main focus will be on the geometric model and three different implementations will be analysed.
In the simplest arrangement the geometric model is not included in the MDO and is kept separate from the optimization loop (see Figure V.5). This type of approach is very common, especially during the initial design phases, the so-called conceptual design phase. In this case the MDO uses simple mathematical models that are able to compute the system properties required. There are many books in the literature that collect a variety of "handbook" formulas that allow physical properties and performances of an aircraft concept to be estimated (see for instance Raymer [68] or Torenbeek [81]). The geometric model is generated separately as required, often at the end of the optimization process. The advantages of this type of approach can be summarized as follows:

- The equations are simple and require only a limited amount of input to generate a fairly good prediction
- The equations can be solved very quickly, allowing a complete system to be analysed or optimized with very limited computational expenses
- The mathematical models are trusted because they are widely known and have been used for a long time
- Since the MDO composed of mathematical equations that simulate the different disciplines, the framework connections are simple to implement and results can be obtained quickly

Although this type of implementation can be very practical, it is strongly limited by the availability and validity of the mathematical models. Their semi-empirical or statistical nature bounds them to be used only within the range of "known" aircraft concepts, while "*the applicability […] is extremely limited – if not useless – to novel aircraft configurations*" [41].
The study presented in appended paper [V] is based on a framework of this kind and confirms the properties just presented.

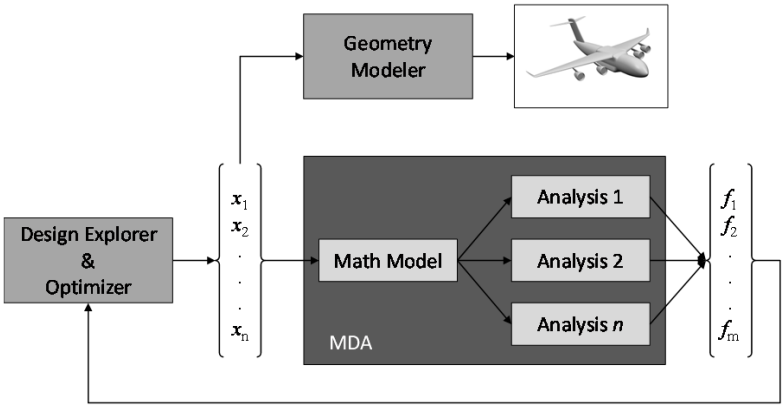

Figure V.5 MDO framework without geometry model (adapted from Vandenbrande et al. [82])

A different approach is shown in Figure V.6. In this case the geometric model is still not included in the optimization loop, but a CAD model of a baseline configuration is generated beforehand. A suitable grid is constructed and inserted in the MDA, and used to perform analyses and estimations. To be able to carry out the design space exploration, the optimizer adopts grid perturbations that modify the grid, transforming the baseline concept. Compared with the previous solution, this approach is clearly more cumbersome to implement and certainly much more computationally expensive. On the other hand, it is far more accurate and not limited to known aircraft configuration, since the analyses are carried out on the actual geometry rather than on a mathematical model of its properties. Nevertheless, this approach presents some limits as well. The grid perturbations cannot be too large, or the quality of the solution will be greatly affected. The perturbations cannot move too many points either, or the resulting transformed geometry smoothness can be at risk. The only transformations that the baseline configuration can be subjected to are morphological, thus limiting the design space. Finally, since the grid is constructed with respect to the main analysis' requirements, the other analyses may have to work with sub-optimum data, hence affecting fidelity.

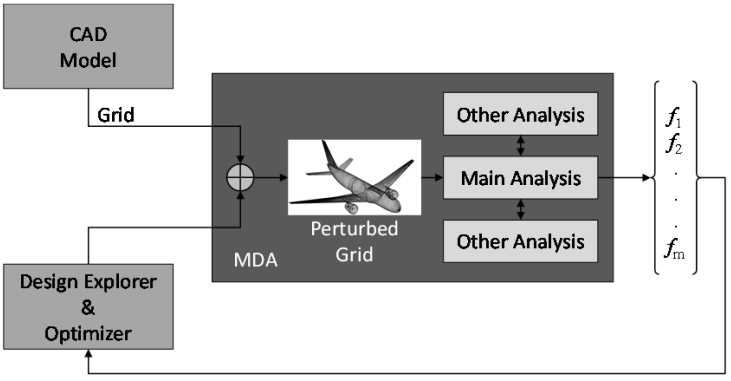

Figure V.6 MDO framework with geometry model based on grid perturbation (adapted from Vandenbrande et al. [82])

The final approach is characterized by the inclusion of the geometric model in the MDA. This means that for each design point, a new geometry must be created before the analyses can be performed. The geometries do not need to be perturbations of a baseline concept, but can be drastically different, as shown in Figure V.7. This strategy is enabled by the capability of parametrically producing the required geometry at each optimization step. Clearly, this implementation strategy is the most complex and computationally heavy. But it is extremely powerful because it eliminates "*any representation related restrictions on allowable geometry changes*" [82], both with respect to modelling and analysis. Moreover, the same framework architecture can be used for different fidelity analyses. The inputs required for each analysis included in the MDA are extracted directly from the geometry model in the most appropriate format.

As anticipated, including the geometry model inside the design optimization loop is challenging from a computational point of view. On the one hand, it becomes necessary for the geometry model's reliability over the defined design space to be guaranteed. Hence, the robustness and flexibility metrics proposed in CHAPTER VIII can be helpful to quantitatively ensure that the geometry model is acceptable. On the other hand, the optimization loop's cycle time will be greatly affected by the geometry model's performances. The methodology described in CHAPTER VI provides effective means to build efficient models, but only through experience and a trial-and-error process is it possible to eventually achieve sufficient performances.

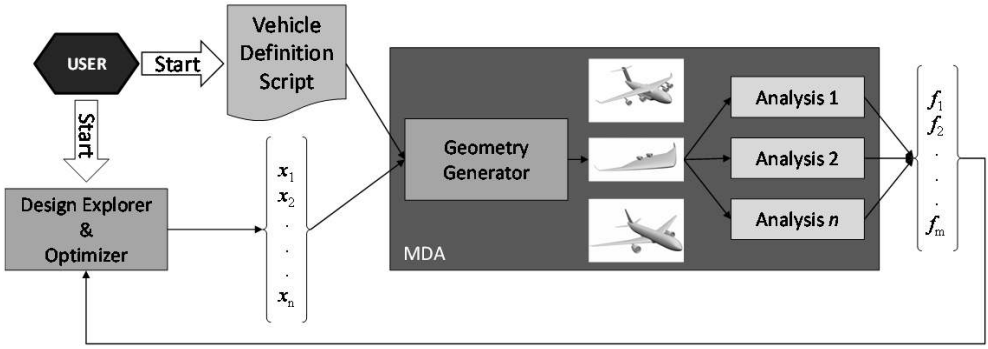

Figure V.7 MDO framework with geometry model in-the-loop (adapted from Vandenbrande et al. [82])

V.3.3 Decomposition of the Multidisciplinary Optimization Problem

In the previous section, MDO frameworks have been categorized with respect to the geometric model implementation. Nevertheless, other categorizations can be found in the literature. Balling et al. [4] presented a very comprehensive overview of optimization approaches for coupled systems. They suggested two aspects to be taken into consideration. First, MDO framework approaches are divided into *single-level optimization* and *multi-level optimization*. Balling et al. [4] stated that "*in multi-level disciplinary design variables are determined by disciplinary optimizers and the system design variables are determined by the system optimizer. In single-level optimization approaches, both disciplinary and system design variables are determined by the system optimizer*". The second distinction is between Simultaneous Analysis and Design (SAND) and Nested Analysis and Design (NAND). For a given MDO framework, the difference can be made both at system and disciplinary level. At system level, a SAND approach implies that the system optimizer determine the value of both system design variables and coupling variables, while with a NAND approach the coupling variable values are obtained from a system analyser. At disciplinary level, with a SAND configuration the disciplinary optimizer simultaneously determines the value of both design and state variables, while in the case of a NAND framework the optimizer determines only the design variables, requiring state variables to be calculated at each iteration. Figure V.8 shows a conceptual scheme of a three-discipline framework, to which the classification can be applied.

Although well known, the categorization proposed by Balling et al. [4] do not apply very well to the application examples presented in CHAPTER X. The main reason is that the disciplines of the systems studied in the examples are less tightly coupled. Imaging to draw schemes similar to the one of Figure V.8 for the application example systems, the coupling functions y_{21}, y_{32} and y_{31} could be ignored, if present at all. The solution approach there adopted can thus consider analysing one discipline at the time, successively propagating data from one discipline to the other. The remaining couplings between disciplines are therefore relatively easy to compute, since they consist of state variables of one discipline that are used as disciplinary design variables in successive disciplines.

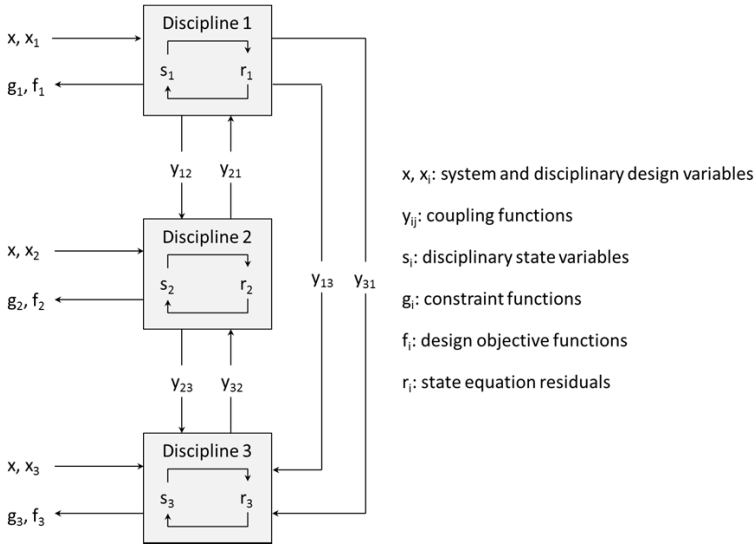

Figure V.8 Conceptual scheme of a three-discipline coupled system (adapted from Balling et al. [4]). In the application examples presented in CHAPTER X the coupling functions y_{21}, y_{32} and y_{31} are negligible.

PART III –

CONTRIBUTIONS

Part III of the thesis presents the theoretical contributions of the research that has been carried out. First a generic CAD modelling methodology is described; then models to quantify model robustness and flexibility are proposed. Prototyping and testing activities are also tackled. Thereafter, three different application cases describe how the proposed principles have been applied to different research projects.

CHAPTER VI.

HIGH LEVEL CAD MODELING

The previous chapter described how GDA can enable MDO featuring geometry models inside the optimization loop, although it is necessary to discuss how GDA capabilities can be achieved. Modern CAD systems offer a variety of tools to increase the flexibility of geometric models. However, to utilize them profitably, there is a need for tool-independent, generic modeling techniques [80]. Structuring these techniques in a clear categorization eases the way towards design automation, since they represent:

- A backbone for discussions and descriptions of tasks involving geometry design automation
- A helpful guide to consult when searching for solutions
- A map to locate the required geometric design automation level more easily.

To achieve design automation, KBE methods can be employed to effectively capture knowledge by storing rules, relations and facts. In the presented work, KBE is used to support High Level CAD modeling by creating HLCts. These are higher abstraction objects that can be automatically instantiated in the design of new products. HLCts are produced and stored in libraries, giving engineers or a computer agent the possibility to firstly topologically select the templates and then modify the shape of each template parametrically.
Besides the elimination of routine-like operations, the following results are expected to be achieved when applying the proposed geometric modelling methodologies:

- Enabling of MDO of the entire product, including the geometry
- Use of high fidelity models in early design phases
- Increase in model fidelity throughout the design process
- Reduced need for empiric-based models enables the search for truly novel concepts.

Moreover, since the HLCts are created within the CAD system, they do not require specialists for their creation or modification. HLCts are thus more easily modified and kept updated throughout the design process.
In this chapter geometry transformations have been divided into two categories. The first will describe the *morphological* levels of geometric modelling while the second will reflect on how to effectively increase, reuse or replace the number of geometric objects, hence its *topology*. Morphological changes are transformations that occur within the same instance of a given class, i.e. it is enough to re-evaluate the instance. Topological changes are ones that require instances of classes to be added, replaced or removed. Instantiating an HLCt therefore requires a topological

transformation, while adapting its geometry to the surrounding context requires morphological transformations.

Myung et al. [62] reworked an arrangement into classes of design problems initially proposed by Huang et al. [32] (see Figure VI.1), in which morphological and topological transformations are included. The classification that overlaps the classic division into design phases tries to describe from a macroscopic point of view the nature of the problems to be solved during the design of a product. During the initial *Class 1* that starts from the conceptual design, very little is known about the product. The task is to try to decide on a promising product structure and layout by changing the configuration and trying to evaluate performances. As Myung et al. [62] noted, this is a very challenging problem since "*parts selection, generic structures of the product, and topology (relative position between parts) have to be designed according to the design requirements*". Once a product schema is decided, with *Class 2* problems, the objective is to study how components should be arranged or placed within the product. Finally, *Class 3* problems aim at finalizing all attributes and characteristics. It is interesting to note that the problem classification both overlaps and is not confined within the design phases' boundaries, as described in CHAPTER I. It should also be noted that class 2 and class 3 problems can be included in a class 1 problem and a class 2 problem can comprise class 3 issues. This means that a parametric design problem can be included in all classes, but the main characteristic of the previous classes is to be respectively configurational or topological.

The methods that will be discussed in this chapter are well suited for tackling problems of any class, since they provide efficient means to create models that allow both topological and morphological (i.e. parametric) changes. It should be noted that topological transformations, as they will be defined in section VI.2, are valid for both configurational and topological types of problem.

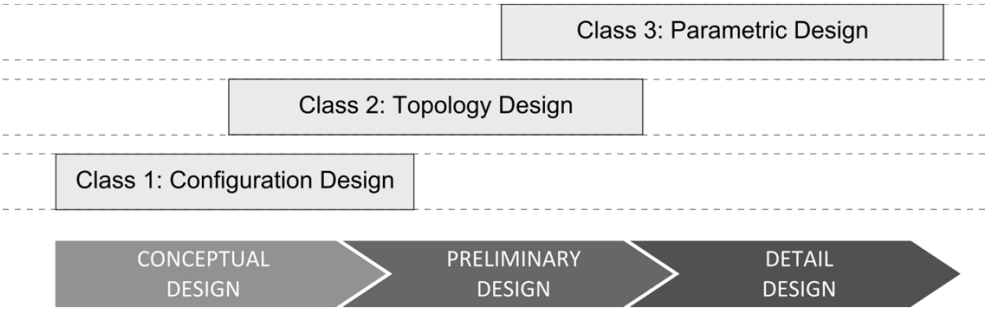

Figure VI.1 The design process can be divided into classes of problems to be solved alongside with the classic design phases (modified from [32])

VI.1 Morphological Transformations

The attribute "morphological" indicates transformations that involve the form or shape of objects. The name derives from the Greek word *morphe* that signifies shape or form. During a morphological transformation, the object will change continuously like stretching rubber.

Different morphological levels which will be discussed in this section are shown in Figure VI.2, with increasing modelling complexity for each step in the pyramid. The higher the level in the pyramid, the higher the knowledge content in the method described.

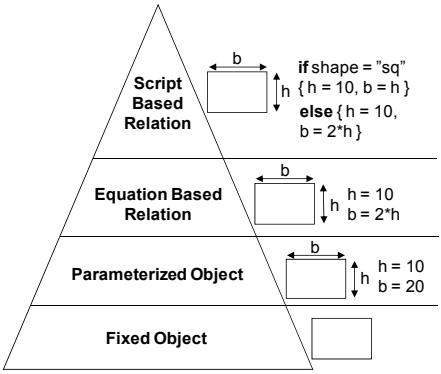

Figure VI.2 The various stages of morphological transformation

Morphological geometric objects can be divided into four stages:

1. *Fixed Objects* are essentially objects that cannot change shape. These objects are intentionally or non-intentionally static and therefore have a fixed set of geometric output. This stage has zero morphological value.
2. *Parameterization* is made on geometric objects of which the values can change and hence do not have a static set of output. Because of the lack of relations between the various geometric objects, it is not realistic to use models based only on parameterization other than cases of non-complex geometries.
3. An effective way to decrease the number of input parameters is to set up relations between a model's geometric objects. This can be done strictly mathematically, referred to as *Equation Based Relations* in Figure VI.2.
4. *Script Based Relations* are models created by writing the relations using the various programming languages provided by the CAD system, described in Figure VI.2. Both equation- and script-based relations are of course rule-based, but the use of the latter allows logic reasoning to be included in a more user-friendly manner, not necessarily mathematical.

VI.2 Topological Transformations

The attribute "topological" describes transformations that involve the location of features or objects in a geometrical model. The term derives from the Greek word *topos*, which signifies spot or location. As stated previously, there are three types of events that can occur during topological transformations:

- Adding an instance; an object is placed in a wanted position
- Removing an instance; an object is removed from a chosen position
- Replacing an instance; an object is removed and one from a different class is added.

The main shortcoming of only using morphologically based models in the automatic design process is that the number of objects cannot be changed. For instance, in a structural optimization that uses only morphological transformations, the number of structure elements needs to be known in advance or alternatively the optimization must be repeated several times with different configurations. To describe the topological process the following terms are important: template, constraint and context. A template refers to an initial model to be re-instantiated; constraints are conditions which have to be satisfied by the instantiated instances; context is the geometric surroundings of an instance to which the constraints refer.
The various levels of topological instantiation are shown in the pyramid in Figure VI.3.

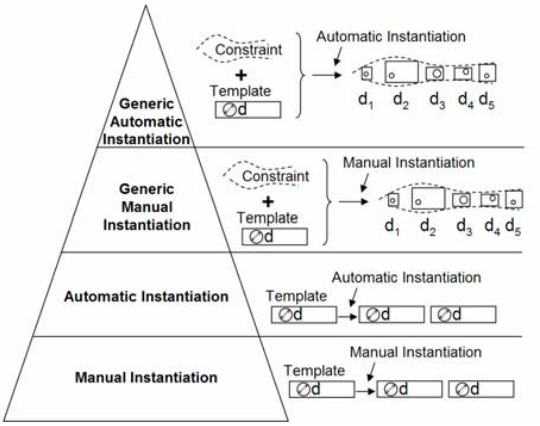

Figure VI.3 The various stages of topological instantiation

The topological pyramid consists of the following stages:

1. *Manual Instantiation*: Manual instantiation is accomplished by performing copy and paste functions on various objects. However, the instances made are not context-dependent upon creation.
2. *Automatic Instantiation*: Though only the template is defined, the instances created will follow the template model slavishly and cannot be context-dependent due to missing constraint definitions. The number of instances is parametrically modified.
3. *Generic Manual Instantiation*: Context dependency for initiated instances is achieved by producing template and constraint manuals. These contain complete construction procedures for objects within the template and to which geometric features they are constrained. These definitions enable the template to be manually instantiated into different contexts, as seen in stage three. This increases the level of reusability in a geometric model.
4. *Generic Automatic Instantiation*: This stage is achieved when pre-defined functions can automatically generate or delete the instances depending on user input. The instances are both context-dependent and able to vary parametrically, hence full reuse and automation are achieved.

VI.3 Continuity Requirements on Geometry Models

The description of novel design optimization frameworks is usually accompanied by considerations concerning what requirements a geometric model should fulfil. A common requirement that is asked to the CAD model and to the corresponding system properties is to be continuous over the design space defined by the input parameters [82] [9]. However, it can be argued that this would be the case only if model transformations were restricted to morphological ones. During morphological changes the shape changes in a continuous manner, hence it is justified to expect the modelled system properties to also be continuous. But as soon as topological transformations are considered and allowed, the very same requirement decays. Topological changes imply discrete – hence discontinuous – variations in the model and its properties. Another case in which continuity cannot be required is when components are selected from a database instead of having “rubber models”. Here again, the model and its properties will vary discontinuously over the design space. For instance, let us assume an aircraft model that can vary the number of engines and that allows the engine model to be selected from a finite list of existing engines. Changing the number of engines would introduce discontinuities in properties such as available thrust, weight or drag. Similarly, keeping the number of engines constant but selecting a different model would also cause discontinuous behaviours. Hence, it is believed that it is not only unnecessary but also not correct to

require the geometric model to "*form a continuous function of the input parameters*" [82]. What must be kept in mind is that including discrete variables, components' lists, topological transformations or any other characteristic that brings along discontinuous behaviours will aggravate the design optimization problem, affecting computing and analysis performances, thus increasing the cycle time. Also, not all optimization algorithms are able to handle discontinuous design spaces, requiring a careful selection and tuning of the algorithm (see chapter V.2).

VI.4 Power Copy and Knowledge Pattern

The topological and morphological transformation pyramids are valid independently of the CAD system adopted. They represent a general way of categorizing all the different types of transformation that a geometric model can be subjected to. Morphological transformations are very common today and most CAD system offer tools to implement them in an automatic fashion. Topological transformations, on the other hand, are much less common, and definitely less well-documented and supported in modern CAD. This section will therefore give a brief introduction to the two tools that have been adopted in CATIA V5 to implement transformations that belong to the top of the topological pyramid: Power Copy (PC) and Knowledge Pattern (KP). CATIA V5 is the CAD system that has been used during the research projects described in appended papers [I] – [IV] to create the application examples that will be presented and discussed in CHAPTER X. Since both PCs and KPs played a fundamental role and have been used extensively, it is worth describing their functionalities, strengths and weaknesses.

VI.4.1 The Aim: an Example

Both PCs and KPs allow a predefined geometry (HLCt) to be instantiated as many times as the user desires. A simple example is given here to clearly explain the aim of both methods. Let us suppose that an HLCt has been created so that each instance contains a hollow circular area of given thickness (individual parameter *t*) that is assumed to be tangential to two lines (see Figure VI.4 for reference). To proceed with the instantiation, the system requires two limiting lines (constraints 1 and 2) and the position of the centre of the circle to be specified. The template geometry is then modified so that the circle is enlarged or shrunk in order to fit within the borders. The thickness can be specified freely so that each instance can be modified independently of all the others.

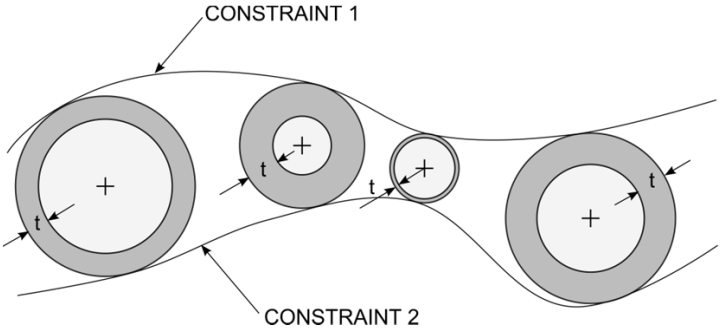

Figure VI.4 Using Power Copies or Knowledge Patterns, upon instantiation the template is modified to fit the constraints. Individual parameters can be defined to customize each instance.

The two constraints together with the position of the centre of the circle define what will from now on be referred to as the *context* of an instance. The dependency on the context is the fundamental characteristic of PC and KP, which makes them extremely powerful and versatile tools for achieving automation or reuse of the design features (GDA).

VI.4.2 Differences between Power Copy and Knowledge Pattern

There are important differences between the methods. To automate the instantiation by PCs or KPs, a script has to be added. However, the scripting language is different: PCs use Visual Basic (VB) code

while KP scripts are written in a CATIA proprietary language. Compared to VB, the latter offers limited objects and methods, but is more concise and synthetic. Not compiled, VB scripts also perform less well, so given the same geometry template, creating several instances through a KP can be quicker that using a PC by an order of magnitude [76].
Moreover, once a new instance is created with a PC, it does not have any links left with the template it originated from. The template is used as a blueprint to re-generate a set of geometrical features, but a later modification of the template will not have any effect on the instance. Moreover, the new instance will show and allow editing all its geometrical features. A PC can be thought of as a "cleaver" copy-paste operation. KPs work differently. All instances keep a link to the template, so that when the latter is modified, it is enough to initiate an update operation to have all instances reflect the changes in the template. Each instance will show the user only the geometric features or parameters that were defined as input or output. The remainder is hidden to the user and can therefore not be edited or changed without modifying the template, but the instances can be used for further modelling. For example, if an extruded solid (called Pad in CATIA) is the output of a KP, its extruding length cannot be modified unless the parameter has been explicitly defined as an input. However, a hole through the Pad can be defined after instantiation.
Finally, there are also differences concerning the flexibility of the two methods. KPs are certainly more restrictive by nature, mainly due to the limited scripting language they employ. They can therefore not control the same number of operations as PCs. Additionally, all the instances created with a KP can only be created within the same part of the CAD model structure, whereas the scripts associated with PCs can include a very large variety of operations as well as allowing the destination of each instance and the product model structure to be fully customized.

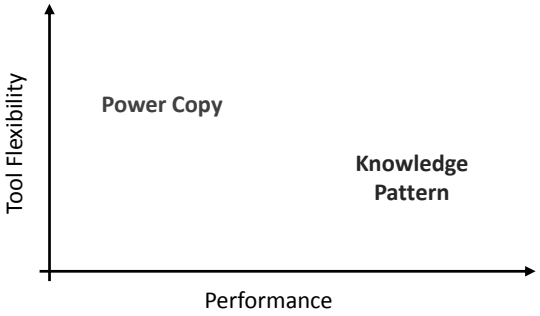

Figure VI.5 Power Copies and Knowledge Patterns can be used to achieve similar functionalities, but the two methods differ in both flexibility and performance

In section X.2 the differences between PC and KP will be clearly highlighted when comparing two different solution approaches for the same geometric model.

VI.5 Dynamic Top-Down Modelling

Traditionally, CAD design is divided into top-down and bottom-up approaches (Mäntylä [50]). These design strategies have their roots in software development and can be associated to analysis and synthesis respectively. Mills [57] and Wirth [88] are among the first to suggest the adoption of top-down design, that requires "*thinking and problem solving before integration, rather than afterwards*" [57]. Hence, in the top-down approach, the critical information is placed on a hierarchal top level and branches down to all lower component levels in the product. The holistic representation of the product is thus in focus, and as a result, the complexity is managed. The possibility to revise the product structure and parametrically modify the morphology of the geometry is also improved. Conversely, in the bottom-up approach, all base elements are modelled in detail separately and finally assembled into larger (sub-)assemblies. Since there is no context dependency between the

parts, the final geometry may be difficult to modify. This approach is therefore less well-suited for GDA purposes.

Although the top-down approach is a well-proven method, it needs to be modified to enable topological automation of the geometry. In true design optimization of the geometry, the shape, placement and number of the CAD components should be modifiable. This in turn generates a new means of CAD modelling, referred to here as dynamic top-down modelling. When applying a dynamic top-down development process, the actual CAD models can be generated from pre-described High Level CAD templates, see Figure VI.6. The critical information on how the HLCt should be instantiated is stored in the inference engine [31]. The geometry model is divided into sub-models that are linked to each other in a hierarchic relational structure [45]. From an initial model, the user starts to identify the number of instances needed from each HLCt library through a user interface. Various components can be attached to the model dynamically and their shape altered by the inherited design variables.

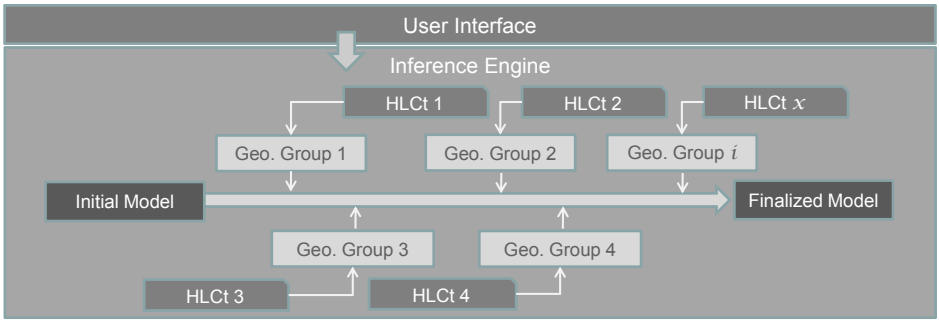

Figure VI.6 The geometric models are constructed by instantiating HLCts in the context defined in the inference engine

Clearly, the needed HLCts can be designed already in the early modelling processes. This approach may appear to resemble the bottom-up strategy, but with the fundamental difference that the information flow and the product structure must be carefully studied and finalized in the inference engine long before the HLCts' design can be initiated.

Topological parameterization requires reference geometries of both the HLCt and the surrounding context. The inference engine is then able to instantiate the selected HLCt into the CAD model and constrain it according to the predefined logic. In the simple example in Figure VI.7, the references stored in the knowledge base (as seen in Appendix 10.4 of appended paper II), are sought after by the inference engine following a logic resembling the pseudo code in Appendix 10.1 [II]. The references in this case are:

- Point $\mathbf{p}_c$, and splines $\mathbf{s}_1$ and $\mathbf{s}_2$ from the context geometries, are defined as *Context_ReferenceList* in Appendix 10.4 [II]
- Points $\mathbf{p}_1$ and $\mathbf{p}_2$ within the HLCt, are defined as *HLCt_ReferenceList* in Appendix 10.4 [II].

After instantiating the HLCt, the inference engine will constrain $\mathbf{p}_1$ on $\mathbf{s}_1$ and $\mathbf{p}_2$ on $\mathbf{s}_2$ following a logic similar to the one presented in Appendix 10.2 [II]. The length parameters l_x and t_x are then freely defined by the user, as seen in Appendix 10.3 [II].

As shown in this example the instantiated HLCt will adapt to the defined context. Each instance, however, is unique due to the characteristics of its contextual placement and the user-defined parameters which are set independently for each instance.

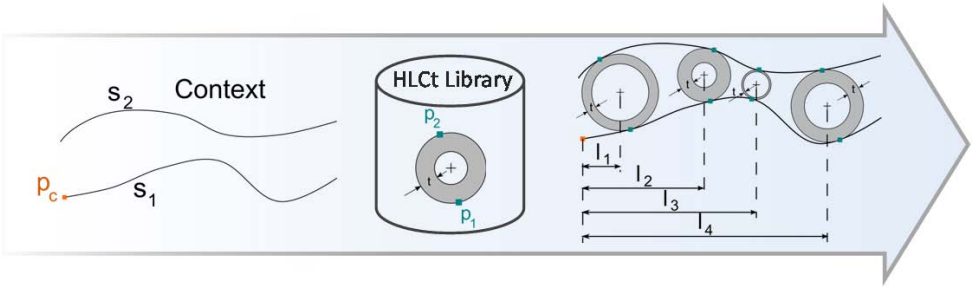

Figure VI.7 The topological instantiation process of an HLCt

The given example is deliberately simplified to highlight the basic principles of the methodology. Naturally, when applying the method in a real case scenario, all the features involved in the topological transformation are more complex. The principles of HLCt and inference engine, however, remain the same.

An example of a more complex HLCt that has been used to generate aircraft fuselage frames, is shown in Figure VI.8. In the figure, the principal contextual references are also pointed out. The main direction line indicates where the frame needs to be placed, while the start and end reference lines provide length limitations. The remaining contextual features are surfaces that the HLCt requires to assume the correct shape to fit inside the fuselage.

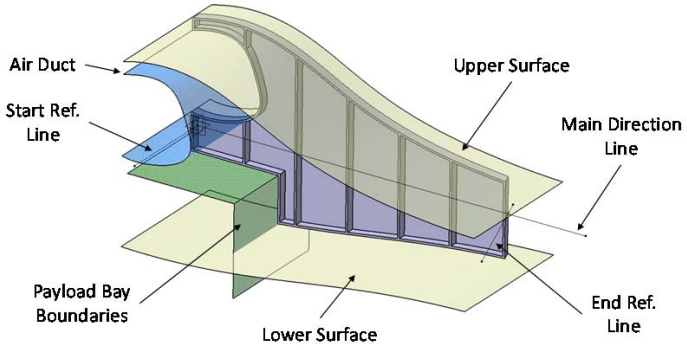

Figure VI.8 An example of an HLCt used to generate fuselage frames. In the pictures the main contextual references are indicated.

In appended paper II further details on the HLCts and a comprehensive explanation of the underlying mechanisms on geometry modelling and knowledge based integration can be found. The paper also presents more application examples were the High Level CAD modelling methodology has been successfully employed.

CHAPTER VII.

PARAMETERIZATION

The previous chapter suggested dividing the transformations that can occur in a geometric model into two main types (morphological and topological transformations). This chapter will discuss the subject further but from a slightly different perspective, i.e. how a geometric model should be created to ensure the best performances with respect to morphological transformations. It is very interesting to observe that this question is equally relevant for topological changes. It has previously been explained that topological transformations are achieved by creating new instances of a given class (or HLCt), which can include individual parameters (see Figure VI.7). This means that a class is a general model that will be subjected to morphological transformations both during and after the instantiation process.
When creating a parametric geometric model, the designer is required to decide upon a set of parameters that will be used as input and output. What the model will be used for and which functions are required will greatly influence the choice of parameters. This calls for careful, detailed planning of the model structure through an iterative process that closely resembles product development processes. The starting point is the requirement list that documents and answers questions such as:

- What will the model be used for?
- How will it be used?
- Who will use the model?
- What other software will be involved/connected?
- Which functions are required?
- Are there model variants to cover?
- …

When designing and creating a geometry model, a multitude of possible strategies is available to the designer. This chapter proposes a collection of aspects connected to parameterization issues that can be relevant to take into consideration when comparing different modelling solution strategies.

VII.1 WHAT ARE PARAMETERS?

The term *parameter* needs to be clearly defined since it is central to large portions of the present thesis. With the term parameter it will be referred to:

- An independent input that can be used by the user to control the geometric model.
- An output from the model.

In the definition, the type of input is not mentioned, because parameters can be of very diverse nature, ranging from the more usual numbers and text strings, to geometric elements such as curves or surfaces. Parameters should not be confused with variables, which are all the dependent objects, inside the model itself, that are controlled by the parameters through some sort of relation. Variables do not need to depend directly upon parameters, but may be functions of other variables as well. In the vast majority of cases, the number of variables is much larger than the number of parameters. The user mentioned in the definition above does not need to be human, but may also be a computer agent if the geometric model is part of a design optimization loop. Parameters are also the outputs read from the model that can be fed to other software or presented to the user. They represent the response of the geometric model to a given set of input parameters. Clearly, the number of output parameters does not need to be the same as the number of inputs or internal variables.

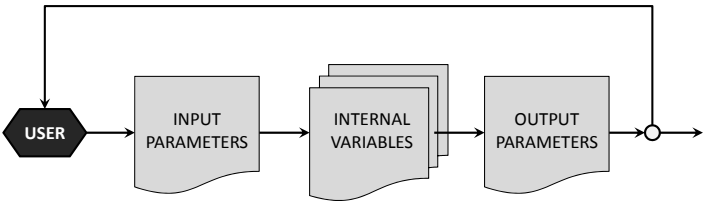

Figure VII.1 Input parameters entered by the user control the model's internal variables, from which output parameters can be retrieved

VII.2 PARAMETER SELECTION

The same geometry with the same number of degrees of freedom can be controlled by different combinations of independent parameters. Some combinations are better and others not so good, and choosing among them is not always easy, especially when the model's complexity increases and the number of parameters controlling it is large. There are nevertheless best-practice guidelines; these are presented below.

VII.2.1 Relative and Absolute Measures

It is usual for parameters to control the length or position of items. In such cases, one of the first choices is whether to use relative or absolute measures. If an absolute measure is used, the parameter controlling a certain dimension contains its actual value. So, for instance, an edge will be 150 mm, a plane will be offset 0,45 m, an angle will measure 32 degrees. Relative measures imply instead that a given dimension is written as a fraction of a reference size. So, reconsidering the previous examples, the edge would become, for instance, 0,75 times the length of a reference length. Both strategies have strengths and weaknesses. Absolute measures are easy to understand and clear to the user, but can produce errors when modifying the geometry. Let us consider the example in Figure VII.2, where the position of a point P_X between two extremes P_1 and P_2 can be expressed as absolute (i.e. as the distance from P_1) or as relative (i.e. as the ratio $\overline{P_1P_X}/\overline{P_1P_2}$). The absolute measure is easier to read, but if the distance between P_1 and P_2 is changed, an absolute measure could position P_X outside the two extreme points. If this is an unacceptable combination, then a control law is necessary to limit the position of P_X. Using a relative measure instead, P_X would always be between P_1 and P_2, but its location would be less easy to read. Moreover, if the distance between P_1 and P_2 is changed, the position of P_X relative to a fixed coordinate system would change if a relative measure is used, but not if an absolute one is used.

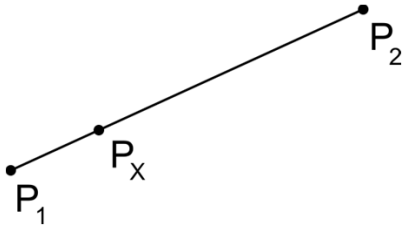

Figure VII.2 The position of a point P_X between two extremes P_1 and P_2 can be expressed as absolute or relative

It is not possible to state a priori which method is best, since the choice depends entirely on the specific case considered, although it is important to be aware of the consequences of the choice made, in order to avoid faulty behaviours and errors.

VII.2.2 Choice of Parameters

Parameterization is used in order to more easily modify a model. Starting from the type of changes that the model should be able to accommodate, it is possible to identify potential parameters to choose. As previously mentioned, there is more than one possible combination. In the literature, several studies can be found that discuss the requirements concerning parametric geometry models [82] [9]. According to Vandenbrande et al. [82], it is fundamental that "*any combination of parameter values within their specified ranges produces a sensible and realizable configuration*". This means that as long as parameter values are chosen within the predefined ranges:

1. No error should occur, i.e. the model should be robust. This requirement will be discussed in depth in CHAPTER VIII
2. The resulting geometry should make sense. It is important that the geometry model represent configurations that are realistic.

Unrealistic geometric outcomes should be avoided, especially if the model is fed into analysis frameworks, since otherwise the computations linked to the model might either crash or produce totally misleading predictions. Only in some fields, such as aeronautics, are there standardized and largely accepted parameters that can be adopted to describe a geometry. However, the way they are combined has huge relevance, as already exemplified by the aircraft wing example proposed by Vandenbrande et al. [82] (see Figure VII.3).

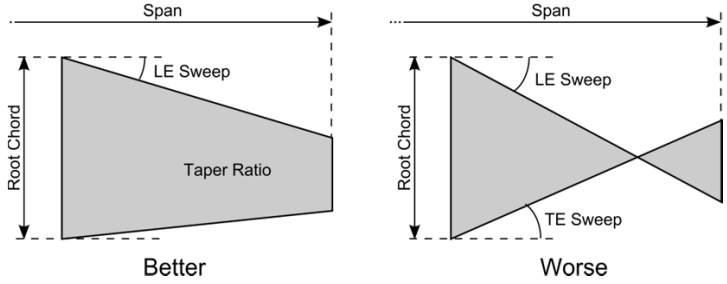

Figure VII.3 Two different ways to parameterize an aircraft wing: the one to the right can clearly produce unrealistic configurations (adapted from Vandenbrande et al. [82])

Both parameterizations in the example have equal degrees of freedom, but the one to the left in Figure VII.3 will always result in realistic configurations, while the other one can end up producing irrelevant wing planforms with intersecting leading and trailing edges. To avoid such risks, the right-hand case requires additional checks or rules that oversee the configuration. In the parameterization to the left the rules are instead implicit in the parameters' definition. In real life, deciding between them would also be dictated by the relationships with all surrounding geometries and parameters.

Hence, in the figure, it is stated that the two cases are *better* or *worse* rather than *good* or *bad*, to highlight the relativity of the decision to many other aspects not shown.

VII.3 CREATING A CAD MODEL FOR DESIGN AUTOMATION: AN EXAMPLE

In this section, a very simple example will be used to show how the scalable nature of the methodology presented in this thesis does not necessarily require a significant increase in modelling time. If the modelling "groundwork" is done correctly, parameterization and adaption to possible automation needs can be done at any time simply by adding functionalities as necessary.
Imagine that a linkage between three axes needs to be designed (Figure VII.4). For simplicity, in the example, hierarchical relations will not be included. Hence the location and diameter of the three axes will not be used as input, and all geometries will be controlled by dimensions or parameters.

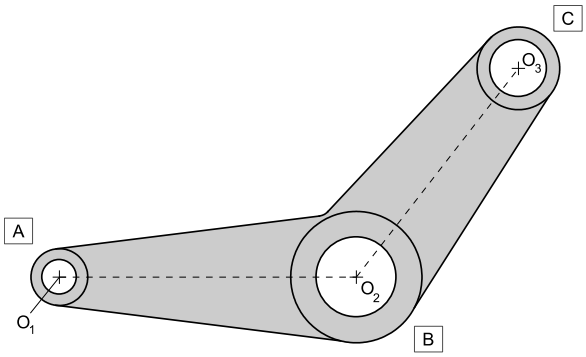

Figure VII.4 A hypothetical example of a linkage between three axes

The first fundamental action is to identify the driving features of the model. In the example, the main driving features are the positions of the three axes O_1, O_2 and O_3. For instance, they could easily be identified putting O_2 in the coordinate system origin, then locating O_1 and O_3 through the length of segments $\overline{O_2O_1}$ and $\overline{O_2O_3}$, together with the angle between the segments. At this point, from a functional point of view, the linkage is already completely defined. The next step would be to assign diameters to the three axes' sleeves. Recalling from section VII.2 the discussion about the choice of parameters, it might be wiser in this case to assign for each sleeve an inner diameter and a thickness rather than two diameters. Finally, the actual arms of the linkage can be added, using tangency constraints referred to the outer surface of the three sleeves. The model obtained would require no internal constraints or any kind of controls to ensure that the model is feasible at all times. More importantly, modelling it in the described way does not require any more time than modelling it in less proper ways, except for the initial step that requires the designer to think ahead before starting the actual modelling, to identify the features driving the design.
The resulting model can be placed at the second level of the morphological pyramid shown in Figure VI.2. Changes are possible, but they require the designer to manually edit the definition of the geometric features to modify. If one would like to raise the model to higher levels, the groundwork would not need any modification and would guarantee good flexibility and robustness (see CHAPTER VIII for a detailed discussion of these terms). For instance, explicitly designing parameters and adding relations between them to enable some sort of optimization would be easily and quickly done. The model could also be used as an HLCt of a linkage, using the three axes as contextual inputs. Then the shape of the part could be morphed according to different bushes' thicknesses.
The adoption of alternative modelling choices would most probably lead to significantly worse performance. Some examples are presented in Table VII-1. It should be noted that the potential problems described in Table VII-1 can be avoided, for instance adding rules or control functions that keep the model from assuming invalid shapes. However, the more complex a model is, the more difficult it is to identify the potential problems and to create efficient control functions. Even though

it may not be as simple as in this example, it is certainly preferable to model the part correctly rather than try to fix bad choices later.
The observant reader will notice that the proposed linkage model can also incur an invalid geometry, for instance if the diameters of the sleeves become so large that they intersect with each other. This raises a question: Is there a perfect model? It is the author's opinion that, for complex engineering products, there is not. By widening the design space, any model will eventually reach an invalid configuration. When that happens, the designer has two options:

- If the design space was enlarged beyond what can be reasonably expected from the model, then the problem discovered can be disregarded
- If the design space is reasonably large, then the model may need to be modified or discarded in favour of a new version.

Table VII-1 Some modelling alternatives that can lead to potential problems

Modelling Alternative	Potential Problem
An inner and an outer diameter are used to define the sleeves	If the inner diameter is set to a value greater than the outer diameter, the model will generate an invalid geometry
The contour of the linkage is chosen as driving feature	1. The contour requires several internal constraints to be uniquely defined 2. If the sleeves' diameters are larger than the contour, the geometry becomes invalid
The sleeves are defined with an outer diameter and a thickness	If the thickness value is larger than the radius of the sleeves, the geometry becomes invalid

There is rarely only one way to create a model, but many solutions are available. Product development processes often include an initial phase during which a large number of concept solutions are generated and at the end screened and ranked. Several concepts can be generated concurrently before being sorted. Conversely, when creating a geometric model, it is usual to progress in a more iterative and sequential manner (see Figure VII.5). A concept or test model is created, evaluated and discarded or saved before the next one is started. Clearly, if the model's complexity or size requires more people to work on its development, concurrent activities will take place anyway. It can be imagined that a model is verified against increasingly tough objectives until a crash requires a new version to be produced.

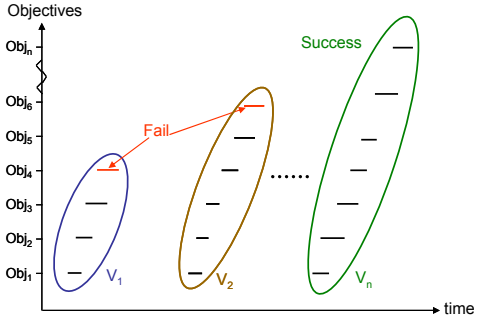

Figure VII.5 Eventually, a model will incur an invalid configurations, at which time it may be necessary to create a new and improved version

CHAPTER VIII.

ROBUSTNESS AND FLEXIBILITY

Creating CAD models of complex products is a “trial and error” process. It is impossible to provide a recipe for building a perfect model. It would therefore be very useful to be able to measure quantitatively whether the previous trial is better or worse than a new one. In this scenario, models to measure the robustness, flexibility and design space size are proposed.

In most papers and articles on MDO and KBE two recurring requirements concerning the geometrical model are expressed: the model should be flexible and robust. This sounds very reasonable, but what does it actually mean?

Qualitatively, it can be explained as follows: The flexibility of the geometrical model refers to the ability to represent a wide range of different product configurations, arrangements and sizes. The wider the range of products the model can cover, the more flexible it is. Robustness refers instead to the errors or instability issues that changes to the geometrical model may provoke. Or, more correctly, the robustness of the model indicates the lack of these. The fewer errors, the more robust the model is.

Even though this reasoning works on a qualitative level, a very important question that has so far been ignored in the literature is how to translate the concepts of flexibility and robustness to a quantitative level or in other words: how can flexibility and robustness be measured?

This is actually a key issue. One of the first lessons learned when creating geometrical models for MDO applications is that it is an iterative process where the designer keeps on improving the result by “trials and errors”, making it an inductive process. It is therefore useful to be able to know when the model is good enough to meet the requirements. Moreover, the requirements themselves are very difficult to specify if there is no way to measure how flexible and robust the model should be. Another case is when comparing two models for the same application that are created employing different techniques or strategies. Which one is the best?

Robustness and flexibility depend on the size of the design space which the model should cover. Model errors and instability issues are usually less frequent for a geometry model whose input variables have narrow allowed ranges compared to larger ones.

To try to solve these issues, a model to quantify design space size, robustness and flexibility is proposed.

VIII.1 DESIGN SPACE SIZE

A geometrical model is usually controlled though a set of n independent input variables. These are not all the available parameters in the model, but only a sub-set that is selected by the designer.

Each input variable may directly or indirectly control a number of other parameters by means of the mechanisms described in the previous chapters.

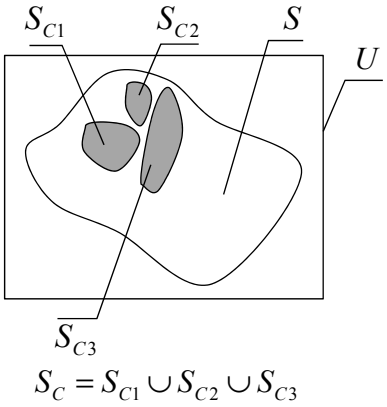

Figure VIII.1 The design space of a geometric model can be thought of as the sum of disjoint sub-spaces that represent all the product variants covered by the model

With reference to Fig. 3, let U represent the n-dimensional space describing all possible configurations that could theoretically be obtained by changing the input variables over unlimited ranges. S represents instead the portion of U that is obtained by limiting the parameters' range between minimum and maximum values. Finally, S_C represents the union space that describes all the product variants that can be represented by the model. Within each S_{Cj} space only morphological transformations can take place. S_C is composed of disjoint sub-spaces since topological transformations do not guarantee continuity. Taking into consideration one of the S_{Cj} sub-spaces at the time, i.e. considering only morphological transformations, and assuming that each variable x_i can be varied between a minimum and a maximum value (x_i^{MIN} and x_i^{MAX} respectively), a dimensionless variable range Δ_i can be defined for each design parameter as:

$$\Delta_i = \frac{x_i^{MAX} - x_i^{MIN}}{x_i^{REF}} \tag{2}$$

In the equation above, x_i^{REF} represents a reference value for the variable x_i and must not be zero. Note that the choice of x_i^{REF} will affect the end result. It must, however, be remembered that the aim is to have a quantity to compare different model variants of the same product. Hence, for the same variable x_i, the variants will have the same reference value x_i^{REF}.

It is important to be able to measure the size of the design space that the model is intended to cover. The design spaces of different models may vary significantly in size, since the scale of different products may differ by several orders of magnitude. In order to be able to compare sizes from different models a normalized mean design space size is defined as follows:

$$\overline{V}_{S_{Cj}} = \prod_{i=1}^{n} \Delta_i \tag{3}$$

S_{Cj} is the sub-space in which the model's flexibility and robustness will be measured. The effects on the design space size of discrete variables controlling topological transformations is very hard to predict and would require a much more complex model to be captured properly. For instance, in the case of discrete variables, it may be difficult to define a reference value, which is required in

equation (1). Another issue is that topological transformations allow infinite instantiations of the same class of objects. In this scenario it would be of no help to add all the design space sizes of each instance. Moreover, the ability to provide a simple enough formulation that can be easily computed to guarantee ease of use is a fundamental requirement.

VIII.2 MODEL ROBUSTNESS

The second keyword is robustness and can be translated into a quantitative expression that reflects the qualitative meaning described above. It might be the case that some combinations of design parameters within the defined design space do not produce valid design. Ideally, all designs within the design space should be valid. However, there could be combinations within the design space that yield invalid designs, hence the fraction of such designs represents a measure of the quality of the model. With a good parameterization the fraction of invalid designs within the design space can be kept at a minimum. Consider a test where the model is fed with a random set of input variables represented by the vector $\mathbf{X} = (x_1\ x_2 \cdots x_n)$ so that $\forall x_i \in \mathbf{X}, x_i^{MIN} \le x_i \le x_i^{MAX}$. The values to assign to each variable are selected simultaneously using a uniform probability distribution over the allowed intervals. Let $N_{Failures}$ represent the number of trials that results in a crash (i.e. the CAD system is unable to calculate a valid solution) in the update procedure and $N_{Updates}$ the total number of trials. An approximation of the robustness R_{SC} can then be formulated as:

$$R_{S_C} = 1 - \frac{N_{Failures}}{N_{Updates}} \tag{3}$$

The index S_C indicates that the robustness is relative to the sub-space S_C considered. Clearly, changing S_C will alter the value of R_{SC}. It is worth noting that the value and correctness of R_{SC} will greatly depend on the number of trials that are carried out. If the number of trials is small, there is a risk that the results will be distorted. With an infinite number of trials, if $N_{Success}$ represents the number of successful updates, then the robustness R_{SC} can be written as:

$$R_{S_C} = 1 - \frac{N_{Failures}}{N_{Updates}} = \frac{N_{Updates} - N_{Failures}}{N_{Updates}} = \frac{N_{Success}}{N_{Updates}} = \frac{S_C}{S} \tag{4}$$

In practice, a sufficiently large $N_{Updates}$ is required to give the result statistical relevance. Since the outcome has only two possible values (1: *failure*, 2: *success*), a relatively small number of trials can be enough. Typically, 100 trials can give a good indication of R_{Sc}. Note that this number is independent of the number of parameters since the robustness is a scalar value. To increase the confidence of R_{Sc}, independent portions of the model can be tested separately, for instance the wing and the fuselage of an aircraft model. In such a case the total robustness would be the product of the robustness of the parts.

The space S_C is not necessarily continuous but can be made up of several disjoint spaces or sets. The reason is that most CAD systems cannot guarantee that for any combination of the input variables, each one having a continuous range of values, the geometry will be feasible [9]. The situation becomes even more complicated if topological transformations are considered, since discrete variables are introduced.

VIII.3 MODEL FLEXIBILITY

Finally, in order to evaluate the flexibility F_S the following formulation is proposed:

$$F_{S_C} = R_{S_C} \cdot \overline{V}_{S_C} \tag{5}$$

This means that the flexibility depends on two factors: the size of the design space the model is intended to cover and the robustness over that space. If a model is very robust, but only represents a very small portion of the design space, the flexibility will not be particularly high. Similarly, the

flexibility will suffer if a very large design space is chosen, exposing the model to frequent crashes due to poor robustness.

It is important to observe that it is not possible to perform the robustness test by changing one parameter at a time, since in that case the cross-couplings between different variables would be disregarded. Moreover, the choice of values for the variables to keep constant would be significant for the end result. On the other hand, it would be possible to define another *parameter flexibility* and *parameter robustness* for the case when only one parameter is varied at a time.

Considering once more the design space represented in Figure VIII.1, it can be observed that S_C does not necessarily include the space of all feasible or desirable designs (D). In general, there may be designs that the geometry model at hand cannot represent (i.e. the portions of D that are outside S_C), but there may also be designs that the model can cover, but that are not of interest or feasible (i.e. the portions of S_C that are outside D).

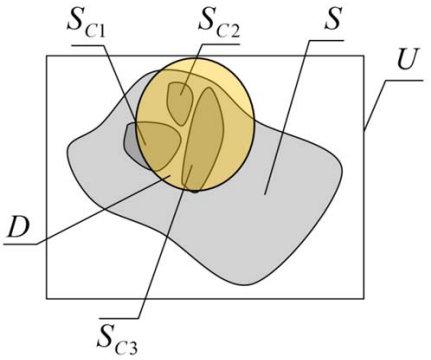

Figure VIII.2 The feasible design space intersects the space that can be represented by the geometric model at hand

The aim of the geometric model generation is to minimize the difference between D and S_C. On the one hand, it is desirable that the geometry model is able to represent as many of the feasible and desired designs as possible, so that the usefulness of the model is maximized. Conversely, the designs that the model can represent but that fall outside *D* are unnecessary and do not add any value to the model. They can be regarded as a waste of modelling resources and should therefore also be minimized.

It can be noted that when going from space U to S, the designer adds information to the model, by limiting the range of the design variables. Similarly, when going from S to S_C, the amount of information required to reduce one space to another is proportional to the ratio between the spaces' size. In practice, it may be impractical to calculate the exact number of bits of information, since measuring the size of the spaces can be difficult, as already discussed in the previous section.

VIII.4 EXAMPLE

As an example, measurements of robustness and flexibility have been carried out on the simple wing models in Figure VII.3. The tests were performed without using a CAD tool, since the simplicity of the geometry allowed it to be modelled in Excel. As expected, the model parameterized using span, root chord, leading edge sweep and taper ratio never degenerates into unfeasible configurations. Conversely, the other one, which has root chord, span, leading and trailing edges as inputs, gives an erroneous layout in 66% of cases. Table VIII-2 shows the range of the parameters used during the test, while Table VIII-2 summarizes the results from the test.

Table VIII-1 Parameters' range for the simple wing model test

Param. Name		min value	max value	ref. value
Root Chord	[mm]	150	250	200
Semispan	[mm]	1000	1500	1250
LE Sweep	[deg]	20	60	40
TE Sweep	[deg]	0	45	20
Taper Ratio		0.1	1	0.5

Table VIII-2 Results of robustness and flexibility measurements on simple wing model

Application Type	Trials	Variables	Design Space	Robustness	Flexibility
Wing model 1	100	4	0.675	1	0.675
Wing model 2	100	4	0.675	0.23	0.155

CHAPTER IX.

FAST EVALUATION OF MDO MODELS

One of the goals of the methodologies reported in the present thesis is to enable design processes that do not preclude revolutionary designs. Figure IX.1 shows how a system's performance might be qualitatively plotted as a function of time. A system evolves continuously until a break-through occurs, at which time a discontinuity appears as a result of completely new possibilities becoming available. In the case of an engineering system, there may be many reasons behind a revolution in performance: new materials, new engine technology, aerodynamic break-through, new control systems' generation, new system concept. However, it must not be forgotten that the availability of a new technology is not sufficient to cause a leap forward in the system's performance. To take advantage of it, it is also necessary to be able to include it into the models used during system design [72].

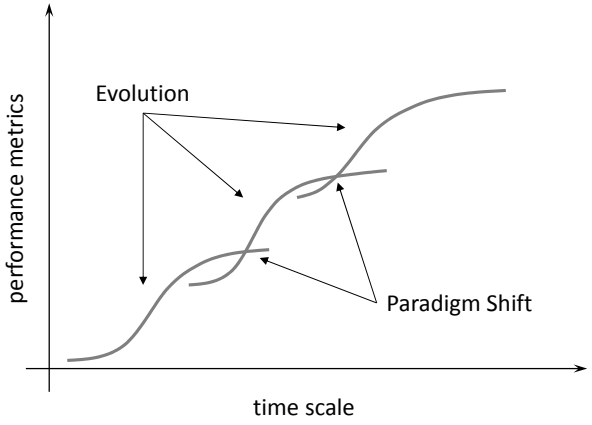

Figure IX.1 The development of system performance over time is composed of alternating evolutions and revolutionary paradigm shifts

To increase system performance in an evolutionary fashion, simpler models may be sufficient. Statistical, semi-empirical or heuristic models can often guarantee fairly good accuracy, which can

also increase the more the models are used if their statistical base is continuously updated. However, such models are not the best suited for enabling paradigm shifts or revolutionary designs, because such designs would lie outside the validity boundaries of the modelling tools themselves. Conversely, the MDO techniques described in the previous chapters allow a direct analysis of the system at hand and can thus be used to predict unconventional designs.
When a paradigm shift occurs and a novel solution is embraced, designers must also face the possibility that unpredicted and unexpected phenomena can occur. With reference to Figure IX.2, as long as designs evolve along the same S-curve, most of the physics related to the system at hand can be assumed to be known or known to be unknown. But when radically new solutions are introduced, the eventuality of "unknown unknowns" becomes significant. Therefore, being able to include practical and cost-effective tests allows designers to cast light on such unknowns and hopefully move them to the much less problematic "known unknowns".

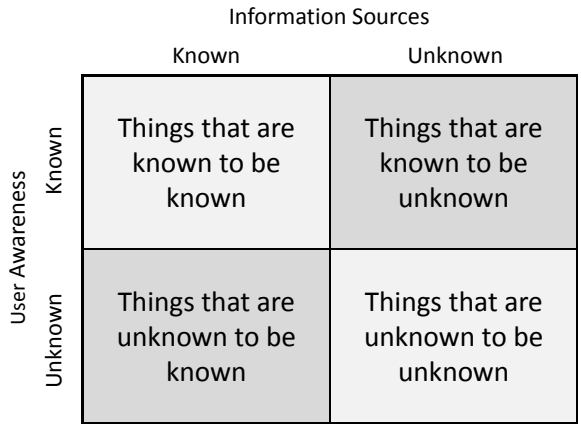

Figure IX.2 The known-unknown matrix (adapted from Frappaolo [25])

In general, there be many reasons for conducting practical testing. It may be necessary to test how possible solution works, or an analytical model may need to be validated to secure its outcomes. During early design stages, the details and knowledge of a system are generally sparse and its complexity demands even quite drastic simplifications to be able to shorten analysis turnaround time to acceptable levels. If the system is known well enough, previous experience can be of great help during the process. But if one is working on something completely new and revolutionary, the unknowns will be many. Then it is indispensable to validate analytical results through experiments. Being able to produce prototypes capturing the salient system characteristics quickly and cheaply then becomes a valuable aid also during initial design space exploration or the conceptual design phase.
Figure IX.3 shows a simplified schematic of the low-cost prototyping and testing activity. An MDO process serves in most cases as the starting point, providing at its end the geometric data of the aircraft to be evaluated. Depending on the required type of testing and on the size of the test vehicle, one of the two manufacturing paths presented above can be chosen. It is worth noting that the process always concludes with a post-processing activity. Generally speaking, this step involves the following:

- Water tunnel testing: the test vehicle's surface must be smoothed and painted. The 3D printer produces finely rugged surfaces; hence manual work is required to smooth them out. A final coat of paint is recommended to help visualization of the flow once in the water. Attachments for the water tunnel support struts and eventual die channel need to be installed.

- Smaller size aircraft (i.e. MAVs): in these cases the superficial roughness does not represent an issue; however control and propulsion system components need to be installed manually. Moreover, to allow the printer to manufacture vehicles larger than its available volume, the aircraft can be broken down into major blocks, in which case manual assembly will be required.

Larger size aircraft: moulds are designed based on the geometric data from the MDO process. They have been purchased from a third-party supplier, because so far it has not been possible to manufacture them at the University. Prior to being used, the moulds need some manual work for preparation (mainly with wax and release agents). Also, once all parts are manufactured, construction and assembly of structural elements is needed before final control and propulsion system components are installed.

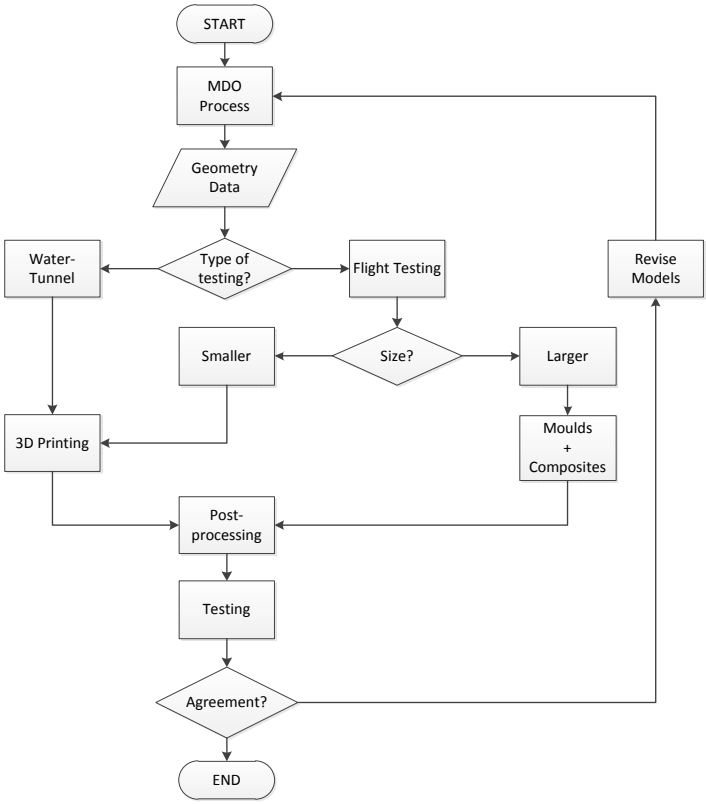

Figure IX.3 Low-Cost prototyping and testing process

The aim of the tests is to validate the MDO framework's models to ensure the correctness of optimization outcomes. Test results must therefore be compared with predictions and it must be verified that there is an acceptable degree of agreement. Otherwise, the selected models need to be revised and the whole process repeated.

It should be noted that the procedure described in Figure IX.3 is not separate and independent from the overall design process. On the contrary, it can be included both in the conceptual or preliminary design phase, when the predicted characteristics of the design at hand do not appear certain enough. The design process suggested by Brandt et al. [10] (see Figure I.2) is therefore modified as shown in Figure IX.4. The MDO process in Figure IX.3 can either represent the conceptual or

preliminary design phase. In-between the design phases, a prototyping and testing phase is introduced, to validate the analytical models adopted.

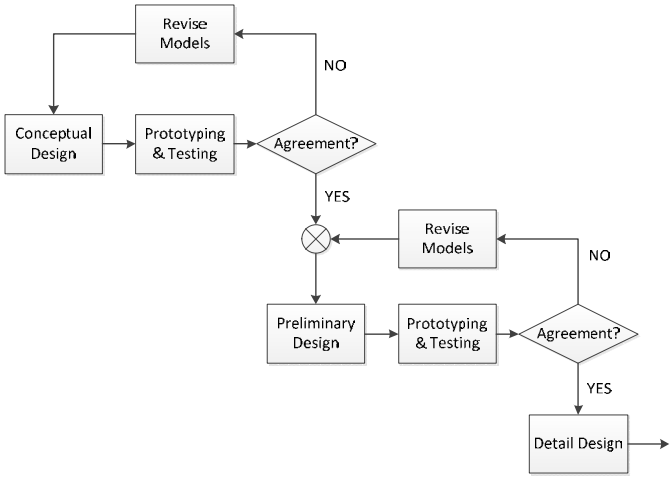

Figure IX.4 Revised design process with the introduction of prototyping and testing phases to validate analytical results

CHAPTER X.

APPLICATION EXAMPLES

In this section three application examples are presented. The first one concerns a framework that enables automated design and fabrication of Micro Air Vehicles (MAV). Since MAVs are relatively simple products, the example can show how a fully automated MDO framework can be realized, and connected to automatic fabrication by means of a 3D printing machine. The focus of MAV application is on the multidisciplinary nature of the design task, on the framework's structure, and on the design optimization issues. The MAV example is largely based on appended papers [I] and [IV].
The second application case presented describes a CAD model intended for aircraft wing conceptual design automation. The design task in general, and the geometric modelling problems in particular, are in this case much more complex than in the MAV case. The application example will focus on geometry modelling strategies and comparison between two automation methods available in CATIA V5 and V6. The wing design example is based on several references and on findings reported in appended paper [II].
Finally, the third and last application example describes how a subscale flying prototype of a generic future fighter aircraft has been manufactured and test-flown at Linköping University. The aim of the example is to show how the process presented in CHAPTER IX has been employed in a successful project. The example is largely based on appended paper [III].

X.1 MAV DESIGN AUTOMATION

MAVs can generally be defined as small unmanned aircraft that are easily carried and operated by one person. They are in most cases used to gather information using some sort of sensor and can be either remotely piloted or autonomous. Sensors carried by MAVs are most commonly optical imagers, such as visual or infrared video cameras, but could just as well be chemical, biological, acoustic or electromagnetic sensors [60] [18]. It is expected that with technological advances, miniaturisation and a reduction in cost of modern electronics, MAVs will come to be commonly used in the modern society.
In this example, fixed wing MAVs are optimized using a framework of coupled computer software to realize a fully automated design process. MAVs are small, relatively simple to build, and require few components. This is therefore an application where fully automated design can be implemented. It can be regarded as a stepping stone from which design automation of more advanced vehicular systems can be developed.
A principle view of the ideal MAV design automation procedure is shown in Figure X.1. Required performance numbers are extracted from an objective or mission specification and a decision is

made as to what on-board systems are needed, such as autopilot and sensors. A design tool, or design framework, then uses several coupled computer programs to generate a CAD model of the optimum geometric design, as well as a list of existing off-the-shelf components for propulsion, and computer code for autonomous flight, ready to upload in the intended autopilot. The presented work focuses on the airframe design and components selection, while the autopilot code generation is a topic left for future developments.

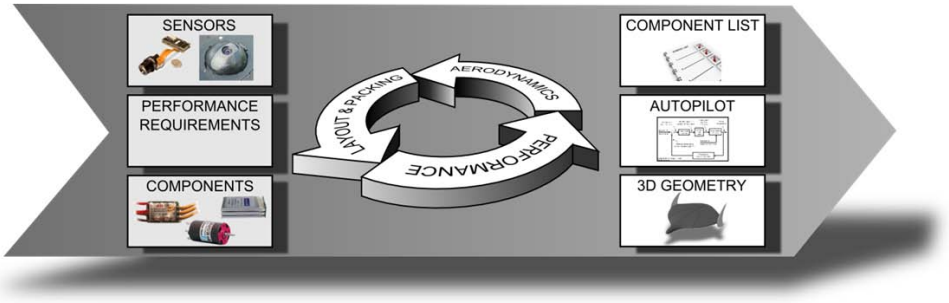

Figure X.1 MAV design automation [I]

A detailed description of the framework and its modules can be found in appended paper [I]. For the aims of this section, it will be sufficient to observe the main features of the process adopted in the design framework (see Figure X.2). The system includes three main modules dedicated to geometry modeling (CAD), aerodynamic analyses and performance evaluations. A large database of propulsion system components is accessed at each optimization step for retrieval of components' specifications. The optimization problem is very complex, since it is multi-objective and it includes a mix of continuous and discrete variables. The optimization objectives are to minimize aircraft weight (*W*) and to maximize endurance (*E*).

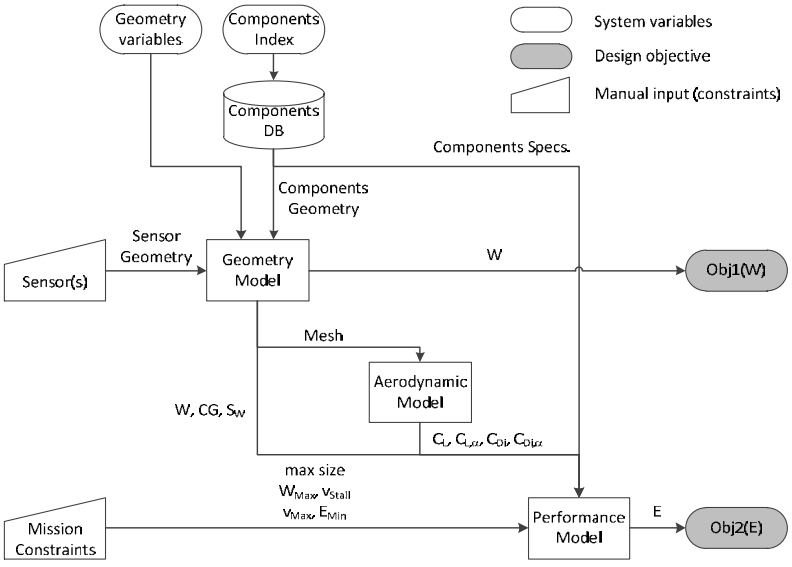

Figure X.2 The MAV design process and framework structure [I]

The discrete variables are used to point to components stored in the database and are greatly responsible for the complexity of the optimization problem. The database is large and it is very

difficult to order it in such a way as to assign a logical relationship between the discrete index value and the specifications extracted. For example, ordering motors by ascending weight does not ensure that the motors' torque, speed, power or current are ordered accordingly. It therefore becomes very difficult for the optimization algorithm to orient itself and a very large number of iterations are necessary. The problem has been tackled in two ways: breaking the overall design task into two successive steps (see Figure X.3) that investigate the optimization algorithm's way of solving problems. During the first step, the system uses low-fidelity models to explore the design space as quickly as possible. In preparation for the second phase, to reduce the number of variables, among all the pareto-optimal designs from the first optimization step, all the unique propulsion systems are individuated and saved in a shortlist. During the second optimization step, the algorithm is allowed to choose only from the propulsion systems stored in the shortlist, instead of having access to all the components in the database. Thus, even though high-fidelity models are adopted, the optimization performances are acceptable.

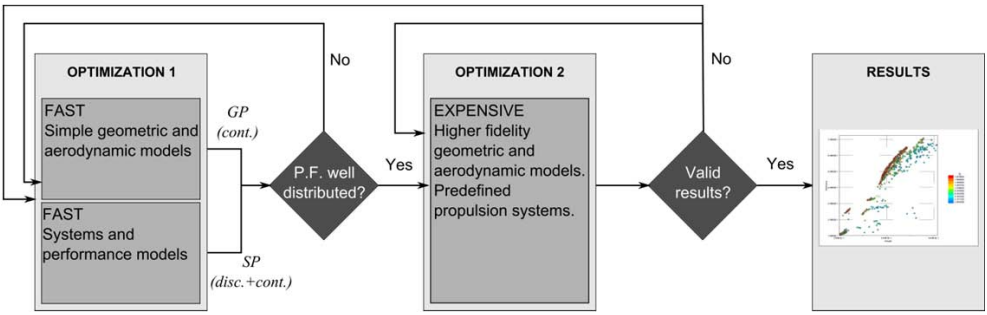

Figure X.3 The optimization procedure has been divided into two successive steps to enhance performances [I]

MOGA-II is the optimization algorithm used and is available from the process integration and design optimization tool (modeFRONTIER [59]) that was used to set the framework up. The algorithm showed a clear tendency to lock itself into the lower-left region of the design space, where low-weight and low-endurance designs are located. Noticeably, the optimizer struggled to find feasible high-endurance and higher-weight solutions, leaving the upper-right region of the design space barely investigated.

To study how MOGA-II operates to satisfy the different objectives, a test was carried out. The two goals (minimizing weight and maximizing endurance) were weighted together using the following expression:

$$obj = \max\left(K_1 \cdot \frac{E}{E_{REF}} - K_2 \cdot \frac{W_{REF}}{W} \right) \tag{6}$$

The weights K_1 and K_2 were then varied from 1 to 0 and from 0 to 1 respectively; for each combination the optimization was then tested, evaluating several settings. The interested reader can find the full details and test results in appended paper [IV]. Figure X.4 summarises the results and evidences quite well the algorithm's struggles to disclose the full pareto front. The tests helped a good parameter setup to be found for the algorithm, but that alone is not sufficient. To force a thorough search of the higher region of the allowed space, the constraint on the total flight endurance was changed dynamically during the optimization itself, according to the following relationship:

$$E_{MIN}^{Actual} = E_{MAX}^{Absolute} - \frac{E_{MAX}^{Absolute} - E_{MIN}^{Absolute}}{ID_{MAX} - ID_{REF}} \cdot (ID - ID_{REF}) \quad (7)$$

where:

- E_{MIN}^{Actual} is the minimum allowed endurance value at a given time during the optimization
- $E_{MAX}^{Absolute}$ is the maximum value considered for the minimum allowed endurance
- $E_{MIN}^{Absolute}$ is the minimum value considered for the minimum allowed endurance
- ID_{MAX} is the final iteration number planned for the optimization
- ID_{REF} is the reference iteration number from which to start relaxing the endurance constraint
- ID is the iteration number at a given time during the optimization

The reason for introducing such a formulation is to keep the endurance constraint at a maximum ($E_{MAX}^{Absolute}$) at the beginning, forcing the optimization algorithm to concentrate on solutions lying in the upper-right region. Then, from a predefined point in the optimization (ID_{REF}) the constraint begins to relax and the algorithm starts to move towards the lower-left region. The real constraint is that the endurance value should not be less than $E_{MIN}^{Absolute}$, so that the solutions that are discarded at the beginning are in fact feasible.

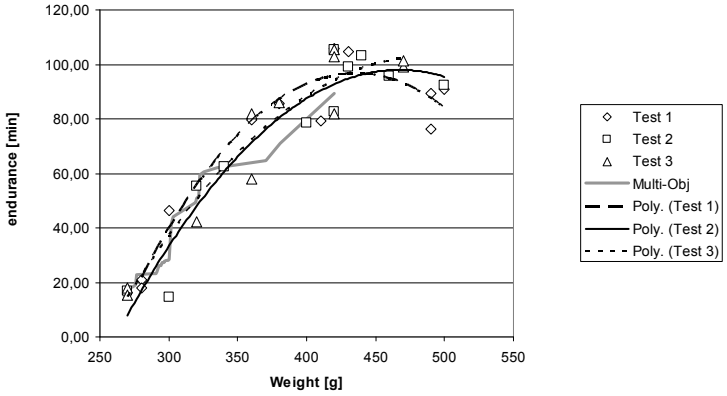

Figure X.4 The pareto front resulting from the tests carried out to evaluate the MOGA-II algorithm, compared with a multi-objective optimization front (see appended paper [IV])

Typical results from the design automation framework are diagrams like those in Figure X.5. Each dot in the plots represents a complete MAV design; the colours indicate when during the optimization the design was generated. All plotted MAVs fulfil the initial requirements. It is interesting to observe that the final Pareto front is not a monotone curve, but presents successive “bumps” or portions. Each of these sections is characterized by having the same propulsion system. Within one of the “bumps” the different performances are determined by changes in the geometry and size of the MAV. This particular shape of the Pareto front emphasizes the importance of optimizing using a database of real propulsion system components rather than theoretical models. Theoretical models would have resulted in a continuous Pareto front representative of the theoretical maximum endurance, but in reality there are no components available that fulfil the theoretical maximum along the entire design envelope.

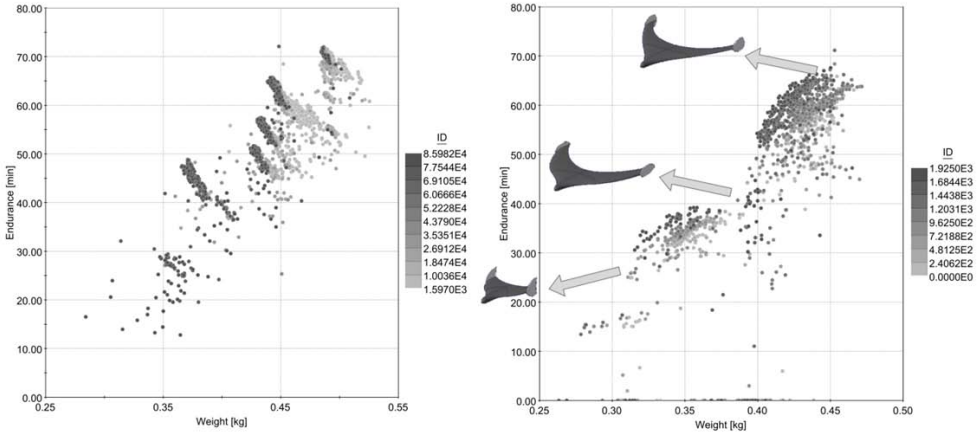

Figure X.5 (*left*) The results of the first optimization step (population size 100; number of generations 500). (*right*) The results of the second optimization step (population size 24; number of generations 84). The ID numbers of the designs are shown in the legends; the number increases as the optimization progresses [IV].

Finally Figure X.6 shows one of the MAVs automatically manufactured using the 3D printing machine available at the University. The particular aircraft in the picture is small enough to be printed in one piece and only required propulsion and control system components to be installed.

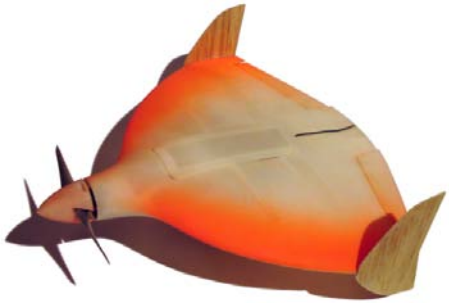

Figure X.6 Automatically manufactured MAV [I]

Measurements of robustness, flexibility and design space size have been carried out on two different MAV model and are reported in Table XI-1. To explain the differences between the models to the level where they could be appreciated and would be understandable, would require a great deal of space in the chapter, which would take the focus from the main contributions. The changes in the models are multiple and consist of very in-depth modifications (i.e. how points are defined) and can sadly not be explained generically. In appended paper [II], additional measurements carried out on other CAD models can be found.

Table X-1 Results from robustness, flexibility and design space size measurements carried out on two versions of the MAV geometry model, indicated in the table by the indexes "A" and "B" (adapted from [I])

Application Type	Trials	Variables	Design Space	Robustness	Flexibility
Micro Air Vehicle^{A}	50	12	2,332	0,64	1,492
Micro Air Vehicle^{B}	50	12	2,332	0,86	2,006

X.2 Aircraft Wing Conceptual Design

Within an on-going research project at Linköping University, an aircraft geometry model for conceptual design studies and design space exploration is being developed. The project makes use of the modelling methodology presented in this thesis to produce a parameterized generic aircraft model for the structural design and analysis. From the 3D representation of an aircraft, suitable input data for aerodynamic and structural studies must be generated automatically. These essentially consist of a mesh of the outer surfaces of the vehicles to evaluate their aerodynamics and a structure mesh for structural investigations.

So far most attention has been paid to the aircraft wing design. The CAD model creation proceeds in three steps. First the aircraft must be efficiently parameterized, and selected HLCts created to represent the largest possible design space that must include both military and civil transport aircraft. Among these HLCts, two types of templates are stored:

- An HLCt, called wing partition, needed to create the outer surface of a wing. A wing partition is a wing section defined by its root and tip airfoils, connected by straight leading and trailing edges. Wing partitions can be used to create both main wings and tails, canards, vertical fins, or any wing-like element.
- HLCts required to create the simplified structural layout, that includes ribs, spars and skin.

By instantiating a number of wing partitions, the desired wing shape can be obtained. Structure HLCts are then instantiated to automatically generate an initial layout of the main load carrying structures. Finally, two mesh models are built to enable aerodynamic and structural analyses. The whole model can be used either within an MDO framework for aircraft conceptual design or as an aid for the designer to quickly carry out studies of particular configuration. In this sense, it is desired to reduce, or possibly remove, tedious and repetitive tasks from the designer's agenda, freeing him/her for more creative ones.

Upon instantiation the inference model is required to manage the interactions between the elements, viz. cutting ribs at the intersection with spars. This is necessary to ensure proper connections between all parts and to avoid errors in mass and inertial measurements due to material overlaps. Figure X.7 shows the wing model generation process, including the mesh models' creation.

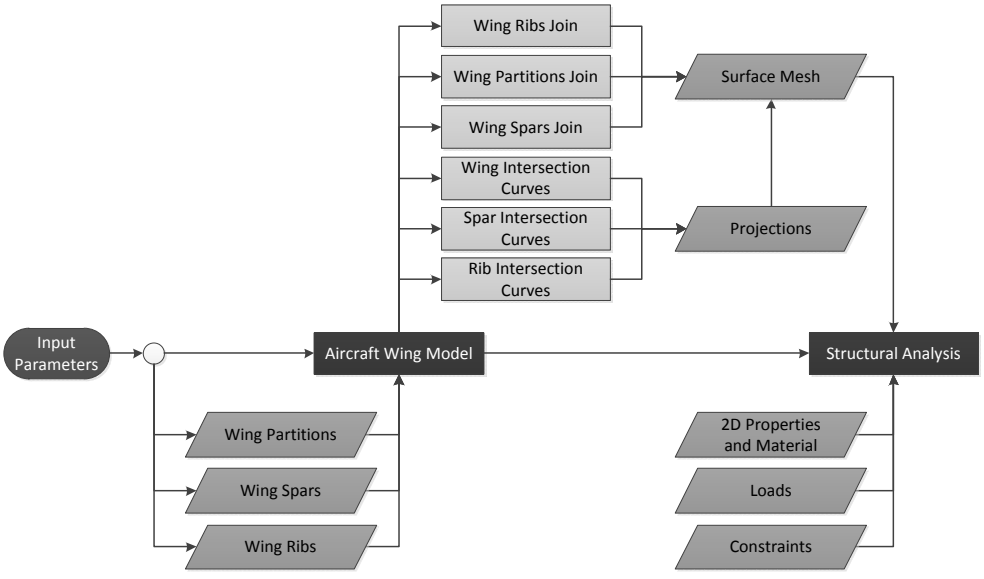

Figure X.7 Parameterized wing modelling process (adapted from Sohaib [76])

It is essential that the system automatically guarantees the mesh models' quality. A problem that is frequently faced when automatically generating meshes of multiple-body assemblies is that at the boundaries between two or more bodies, the mesh nodes are not properly aligned. To correctly transfer loads and dislocations from one mesh element to the adjacent ones, nodes should preferably not be connected to an edge (see Figure X.8).

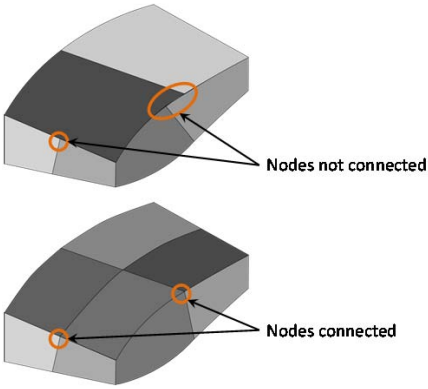

Figure X.8 The automatic mesh generation must guarantee the quality of the mesh. A typical problem is represented by nodes not properly connected (adapted from La Rocca [41]).

Mesh connectivity is a rather complicated issue in an aircraft wing model since there are several parts that intersect and are connected to each other. During model generation, the inference engine must keep track of how many times and where each structure element intersects others. An ordered process has therefore been developed. First, spars are instantiated. A proper parameterization prevents these elements from intersecting each other, since that would generate an invalid solution. Once all spars are in place, ribs can be instantiated. Each rib must be divided into at least two parts, since at least one spar will always be required. Hence, three different rib HLCts are needed to create the following:

- Front part of the rib, placed in front of the first spar. For each rib there is only one front part.
- Mid-part of the rib, placed between of two spars. If N_s is the number of spars, there will be N_s - 1 mid-parts.
- Aft part of the rib, placed behind the last spar. Only one of these parts is needed for each rib.

Figure X.9 shows the process that the inference engine repeats to instantiate all ribs inside the wing. If the number of spars is changed, the system executes a similar procedure. But instead of instantiating all ribs from scratch, it will modify the division of each rib according to the new number of spars.

To create a finite element analysis, the geometry can be discretized with elements of different types. An aircraft wing is made of components that have two very predominant dimensions. For instance, the skin panels have thicknesses that are usually much smaller than the breadth and width. It is therefore convenient to employ so-called 2D finite elements, which can be triangular or quadrangular, depending on the number of nodes. The elements used in the application described here are of the first order, meaning that finite elements only have nodes at the extremes of the elements' sides. The chosen setup guarantees a good compromise between short mesh generation time and quality of results that can be obtained. In Figure X.10 a typical transport aircraft wing is shown, together with a detailed view of the structure mesh. In the particular case pictured, triangular elements have been used.

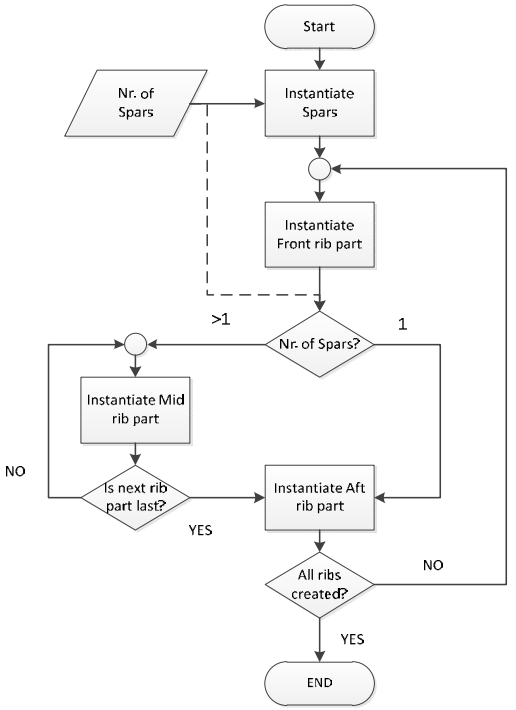

Figure X.9 Wing ribs instantiation process

In the project, CATIA V5R18 has been adopted, and geometry generation automation has been pursued by means of both Power Copies and Knowledge Templates (see section VI.4). This has allowed a detailed performance comparison. On the one hand, update time has been in focus, since the time required to generate or update the geometry and mesh models is of fundamental importance in MDO applications. On the other hand, programming difficulties, maintainability and flexibility have also been compared, since system setup and modification time are also of great interest.

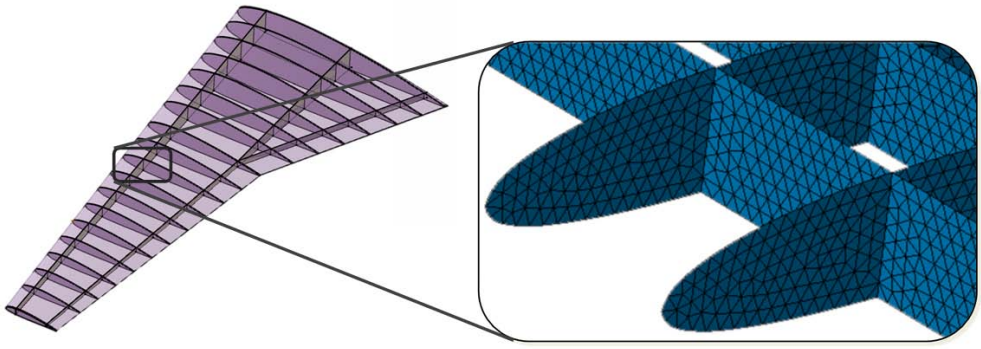

Figure X.10 Typical result that can be obtained with the design automation system. Note how mesh nodes are properly connected at the intersections between ribs and spars.

Following the discussion in section VI.4 about the different automation tools available in CATIA V5, both knowledge patterns and power copies have been implemented and tested with the conceptual design process with the described parameterized wing model. A performance comparison has then been possible, the results of which are summarized in Table X-2. It is evident that knowledge patterns are indisputably faster. The time required to instantiate HLCts in a wing or to modify a given layout ranges from a factor 5 to 10. However, there are other important aspects that Table X-2 cannot show which yield a harder choice between the two strategies than the mere numbers could foresee: mainly programming flexibility and simplicity and maintainability of the code. The interested reader can find full details of the comparison in the work by Sohaib [76]. Briefly, power copies allow complete freedom to choose the preferred CAD model structure in which to place the HLCt instances. Conversely, knowledge patterns require all instances to be created within one single part. Hence, if CAD models must be structured in a very specific way, when using knowledge patterns a final script is needed to construct the required product structure and to relocate the HLCt instances in their final locations. Moreover, scripts controlling power copies are written in Visual Basic for Application (VBA) [84], which is a rather simple programming language, with often self-explanatory function and method names. Knowledge pattern scripts, on the other hand, are written in the so-called Engineering Knowledge Language (EKL), developed by Dassault Systems and used in CATIA V5 and V6 for scripting in the different tools of the Knowledgeware workbench. EKL is not as simple to begin working with as VBA and there are fewer examples to take inspiration from but which can be found on the Internet. However, EKL scripts are much more concise and synthetic than VBA ones, so the total amount of code is drastically reduced. In conclusion, from a pure execution performance point of view, knowledge pattern have a clear edge on power copies, but there are other aspects, mostly related to setup time and maintainability of the system, that could advocate the adoption of power copies.

Table X-2 Performance comparison between knowledge pattern and power copy (adapted from Sohaib [76])

	Nr. of HLCts added	Knowledge Pattern [sec]	Power Copy [sec]
Wing Partitions	3	11.9	45.2
	5	16.4	105
	10	37.8	410
Spars	3	3.1	12
	5	3.5	22.3
	10	6.1	40.9
Ribs	3	2.5	11
	5	3.5	20
	10	6.8	44

X.3 GENERIC FUTURE FIGHTER

With the MAV example it has been shown how 3D printing capabilities, included in the low-cost prototyping process in Figure IX.3, are used to automatically manufacture small flying aircraft. This third and final application example will describe how the same process has been used to manufacture a much larger aircraft: a prototype of a Generic Future Fighter (GFF).

In 2006 a research study was commissioned from the Swedish Material Board (FMV). The study concerned the aeronautical design and integration of a GFF with stealth capabilities, super-cruise and long range. The study involved the following parties: Saab AB, the Swedish Defence Research Agency (FOI), Volvo Aero, Linköping University and the Royal Institute of Technology (KTH). The specification of the GFF asked for:

- Multirole
- Stealth
- Internal payload bays
- Super-cruise
- Integration of future sensors and system architecture
- Studies of a new engine
- Scaled demonstrator

The concept for the GFF has three internal payload bays in the fuselage: two centrally placed for heavier payload, located close to the centre of gravity, and a forward bay for lighter payload ().

Figure X.11 General view of the Generic Future Fighter (GFF) [III]

The basic configuration is a canard, i.e. a stealthy development of the Gripen system. The aircraft's layout and general configuration presents several technical challenges that need to be addressed during the conceptual design. One major concern derived from the location of the fins relative to the vortices created by the sharp edges of the forebody and/or canard at high angles of attack. Many modern fighter aircraft are subject to the same phenomenon and it has been a difficult problem for many of them. If left unsolved, it might lead to flutter and/or fatigue problems, requiring structural modifications and hence a heavier structure than anticipated. Although theoretical studies were carried out (see Figure X.12), it was decided to manufacture and test fly a scaled prototype.

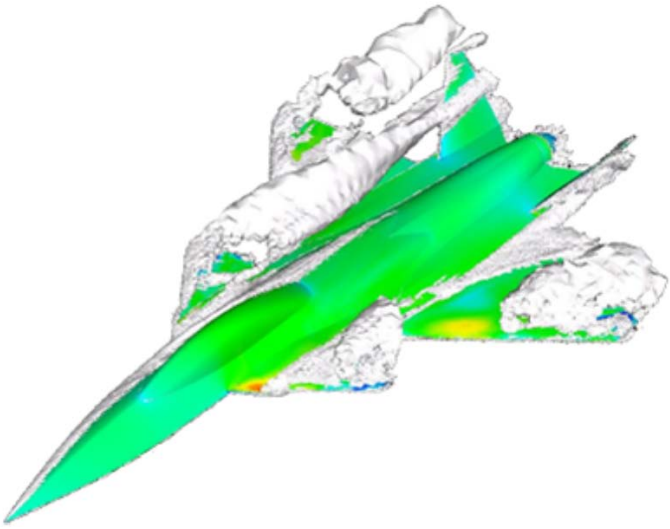

Figure X.12 FOI CFD studies that clearly show vortices investing the fins [III]

Since the aircraft is too large to be manufactured automatically with 3D printing technologies, the GFF's CAD data was used to design moulds that were milled out of RenShape™ 5460. The subscale model aircraft is then realized in composite materials with the internal structural elements of the fuselage made of plywood and carbon-fibre. The composite was realized as a sandwich of two glass-fibre layers and one Herex™ sheet, cured in vacuum bags. All control and propulsion system components come from the RC hobby market, which ensures relatively low costs. The scaling factor was set to 13%, due to both handling and manufacturing considerations, as well as availability of suitable RC components. Several scaling methods are described in the literature, but were discarded mostly because of weight and overall geometrical considerations.
Since the full-scale aircraft is an "unstable" configuration, but due to the fact that the demonstrator is remotely controlled, the prototype has to be a stable aircraft, thus reducing the possibilities to use other scaling methods even more.

Table X-3 The GFF's main dimensions as function of scaling factor [III]

Scale	Size [mm]	Wing Span [mm]	Weight [kg]	Design Weight [kg]
1.00	17000	10500	23500	15400
0.17	2890	1785	115.456	75.660
0.16	2720	1680	96.256	63.078
0.15	2550	1575	79.313	51.975
0.14	2380	1470	64.484	42.258
0.13	2210	1365	51.630	33.834
0.12	2040	1260	40.608	26.611
0.11	1870	1155	31.279	20.497
0.10	1700	1050	23.500	15.400

A low-cost data logging system is also being developed to enable important flight parameters to be monitored. These will serve to validate theoretical results and to evaluate the flight characteristics prior to building a larger scale prototype. It will also permit extreme, high-risk portions of the flight envelope to be investigated without risking expensive prototype air vehicles. Figure X.13 shows a picture of the finished prototype.

Figure X.13 The subscale RC prototype of the GFF ready for its maiden flight [III]

To further study the vortices and their interaction with the fins, a much smaller test aircraft was manufactured for the water tunnel available at the University. In accordance with the process in Figure IX.3, the same geometric data used to design the moulds for the flying aircraft was used to manufacture a water tunnel sized prototype with the 3D printer. The experiments carried out in the tunnel visualized the vortices and, in particular, the breakdown position as a function of the angle of attack was measured. The water tunnel investigation indicates that for some angles of attack the wake behind the vortex breakdown due to the forebody/canard impinges on the tail surfaces as suspected. Qualitative observations were also possible. For instance, in Figure X.14 a strong interaction between the forebody vortex and the vortex emanating from the canard can be observed. No clear couplings between the wing vortex and the other vortices were observed.

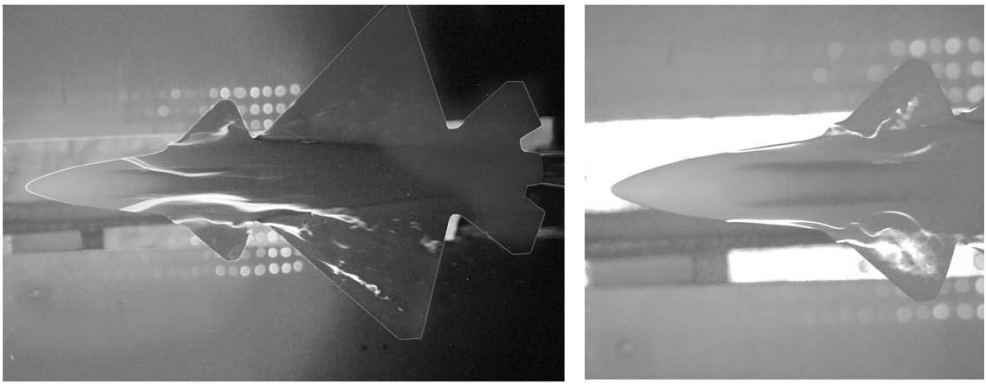

Figure X.14 Flow visualization in the water tunnel using a GFF scaled prototype manufactured with the 3D printer [III]

PART IV -

DISCUSSION AND CONCLUSION

Part IV concludes the thesis. Discussion topics related to the contributions of the previous part are presented. Conclusions are drawn both in general and in relation to the research questions that were formulated. Directions for future researches are also described.

CHAPTER XI.

DISCUSSION

In this final chapter reflections and discussion topics related to the contributions of this thesis are presented. In order to present them in a clear, structured way, the discussion topics are divided into thematic sub-sections that to some degree mirror the thesis's chapter structure.

XI.1 Spread of KBE and DA Techniques

Knowledge-based engineering and design automation techniques are largely recognized as promising tools to achieve reductions in design lead time and costs. However, the same techniques have not been as widely adopted by the industry as one would have hoped. One of the main reasons is that it is believed that the time needed to set up a working KBE system will counter any gains. Sadly, in the literature there are only few examples of the application of KBE or DA that also report hard data to describe the benefits achieved compared to traditional processes. A paper by Emberey et al. [23] represents a welcome exception, presenting the diagram shown in Figure XI.1. The application case concerned designing Fibre Metal Laminate (FML) panels for aircraft application. Not only does the chart clearly show how many more designs can be evaluated in the same time, it also illustrates the time that the KBE system has taken to set up.

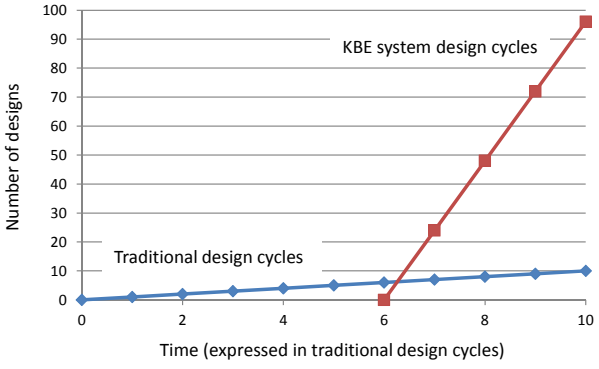

Figure XI.1 Comparison of time and number of design cycles between traditional and KBE approaches (adapted from Emberey et al. [23])

Through the adoption of the KBE system, a substantially shorter lead-time was achieved, which according to the Emberey et al. [23] could be measured at about 75%. In their report, the authors frequently refer to repetitive design tasks that can be automated by KBE systems. Repetitive or routine tasks are not those that will characterize and define a product, but are nevertheless necessary. They often account for the majority of the design activities and in most cases do not represent serious difficulties that can put a project in jeopardy. This of course does not mean that they cannot be complex. According to Stokes, in a given design project, as many as 80% of all design activities can be routine tasks (see Figure XI.2), but, at the same time, the "*creation of a good project depends largely on creativity of the designer himself*" [75]. It should be noted that Stokes based the results upon a somewhat limited number of studies, so the percentages shown should be taken cautiously. However, independently from the exact numbers, the studies certainly highlight that there are possible gains deriving from the minimization of the time spent on repetitive tasks, which are necessary but do not add value to the product. In this perspective, KBE and DA methodologies, such as the ones presented and discussed in the previous sections, become very attractive. They aim at freeing up time for the designer to focus on what he or she does best and that computers cannot yet achieve: use their imagination and creativity. It should be noted that the quantities in Figure XI.2 largely depend on the task at hand. In some cases the time profit could be larger and in some other it could be zero or even become a loss. The difference can be traced to the nature of the project, in particular to how well it is defined, how complex the routine tasks and the project are overall, and whether the processes involved are regulated by rules. In CHAPTER XII these issues will be discussed in more detail.

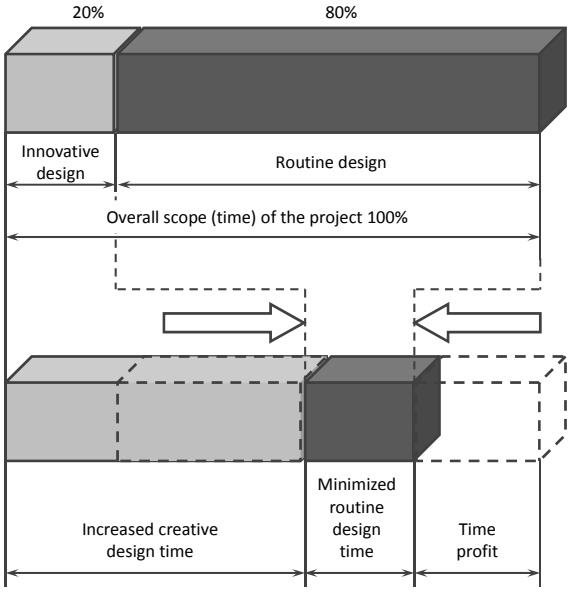

Figure XI.2 Influence of KBE usage on time of main design tasks (adapted from Stokes [77])

Another interesting example of results obtained with KBE and DA is reported by Pinard [66], who used a selection of so-called Key Performance Indicators (KPI) to measure the success of the system produced. In this case, the application was to automate the tooling design process for an aircraft windshield. Table XI-1 summarizes the results. It can be seen directly how lead-time has been drastically reduced and, coherently, how the number of designs per time unit has increased. Other interesting outcomes are the minimization of human errors and administration requirements, while completely removing process errors. The latter is entirely dependent on the goodness of the

knowledge-base and its implementation, and can be verified for example by testing the system before implementing the KBE system in the design process. The former two KPIs' results are instead both only qualitatively expressed and would require a more detailed definition to be fully appreciated. In other words: what does it really mean that they have been minimized? One could argue that even the smallest (and maybe insignificant) reduction is a minimization.

Table XI-1 Efficiency examples achieved through adoption of KE techniques (adapted from Pinard [66])

KPI	Original	After KE Implementation
Cycle time from request to delivery	40 hours/design	½ hour/design
Volume of designs per staff	1 design/week	80 designs/week
Number of staff involved	1 FTE	¼ FTE
Number of design process errors		Eliminated
Number of human errors		Minimized
Time allocated for administration, management, training		Minimized

However, the three reported examples are significant, since they quantify gains from KBE and DA adoption, which in so many other cases can only be presumed or expected.

XI.2 Cost of Design Automation

Much of the criticism of KBE systems' design and implementation is often aimed at the geometric modelling process required to create CAD models suitable for DA. One of the most frequent concerns is that it may not be worth spending the extra modelling time required to "parameterize" a model, and instead the modelling will simply be redone in case of design changes.

As an example, according to Rodriguez et al. [70], geometric model created using CAD tools are unfit to be used in early design stages because "*generating the initial geometry model in CAD can be tedious and time-consuming*" and "*converting the CAD model into something useful to a design analysis method can also be laborious and often problematic*".

However, others have a different view. For instance, Doherty et al. [21] explain that, in order to be able to assess realistic predictions of aerodynamic performances of an aircraft, it is necessary to generate a "*comprehensively detailed*" geometric model. The pre-processing time required to prepare a geometry is "*a significant factor in the time taken to perform CFD analyses*" [21]. Hence it is justifiable to invest resources in the creation of a geometry model that can be included in the design loop.

As a general rule, it can be observed that the adoption of DA can be argued if design changes are not expected. Furthermore, it can be difficult to justify investments in automation if the model is simple enough to be quickly redone from scratch every time a modification is required. However, the modelling example from CHAPTER VII showed that if a CAD model is created properly from the start, it can evolve up to a HLCt without necessarily requiring much additional time.

XI.3 High Level CAD modelling

The high level CAD modelling methodology described in CHAPTER VI provides the technical bases for creating a single central product model that includes the information required by all the engineering disciplines involved in the design, thus obviating the need for standalone disciplinary models. It is therefore reasonable to imagine new relationships between companies or company departments. Usually, each department is responsible for analysing the product within a particular discipline. Adopting high level CAD modelling, each department could instead be responsible for managing and maintaining the part of the central product model required to run the analysis, thus eliminating the burden of verifying that all separate disciplinary models are up to date and represent the same version. The Internet as it is today could be used as a backbone for a common data repository, but it

can be argued that such a scenario would probably require significant reorganizations of the collaboration balances between and within companies.
It is also worth noting that the proposed high level CAD modelling methodology is generic and independent of the CAD tool employed. In the methodology, no mention is made of the functions specific to a CAD system. The presented application examples were, however, all developed using CATIA V5. Nevertheless, the same methodology has been used successfully in both SolidWorks and ProEngineer, thus corroborating its generality.

XI.4 Robustness and Flexibility

As stated in section VIII.1, the presented design space size formulation is only valid for a sub-space within which only morphological transformations are allowed. This is clearly a strong limitation, deriving from the fact that the mathematical models are still under development. However, it has been identified that a formulation that allows robustness, flexibility and design space size to be quantified is lacking in the literature. The models proposed in CHAPTER VIII are a first attempt to fill this gap, but more refinement and continued development are still required.
According to the mathematical models proposed, when working with large geometric models the number of input parameters can easily become quite large, in the order of hundreds or more. The number of tests needed to have a reasonable statistical background for the flexibility and robustness calculation thus grows very rapidly. If, for instance, the model has n input parameters and it can be assumed that for each parameter it may be sufficient to test q values, the total number of trials to execute to compute the total flexibility is q^n. It may therefore be more practical to test smaller parts separately, preferably during the development of the model.
Moreover, if the designer does not build the model correctly (i.e. he or she defines variables as absolute instead of relative as discussed in section VII.2.1) or does not feed the model parameters in the correct way or order, the flexibility analysis will be negatively affected and may not provide any useful information. Once again, experience and know-how are required to prevent or resolve such issues.
Having previously presented models to describe the model's flexibility and robustness, it is felt that the concept of model complexity is also important. However, it is extremely difficult to provide a reasonable definition that would enable the complexity of a CAD model to be measured and there is as yet no universally adopted definition of complexity. In the software development field, several definitions are available. McCabe [52], for instance, proposed relating the complexity of a program to the number of paths that are possible through a program. The same reasoning is not really possible when relating to a geometry model. However, one reasonable measure of complexity could be the width and depth of the parameters' dependency tree structure. It can be argued that robustness and flexibility values tend to worsen when the models' complexity increases. Hence, when defining or describing a model, it would be of help to be able to quantify its complexity to allow other figures to be put in perspective.

XI.5 Design Automation Opportunities

As already pointed out in previous chapters in the thesis, it is of vital importance to carefully select suitable design cases to which to apply automation strategies. The success or failure of the automation effort is highly dependent on that choice [83]. Nevertheless, some guidelines can be found in the literature. For instance, La Rocca [41] and Emberey et al. [23] indicate that only processes that are well understood can profitably be formalized in rules by means of KBE methodologies. This means that a process needs to fulfil a set of requirements:

- Sufficiently stable: since computer systems lack intuition and creativity, processes that require improvisation or original ad hoc solutions are dangerous candidates
- Complete documentation: a design automation system are causal, thus requiring a complete documentation to correctly construct the logic architecture of the inference engine

- Maturity: in general, a design automation effort is rather expensive in terms of time and human resources. It is therefore wiser to choose processes that are mature enough so that significant changes are unlikely.

Despite the availability of these qualitative recommendations, a clear metric by which to identify and classify proper application fields is still missing.

CHAPTER XII.

CONCLUSIONS & FUTURE WORK

Complex products generally have an intricate dependency between geometry, dynamic performance, functionality and cost. Flexible, reusable geometry models are therefore key framework enablers to achieve automated design. The geometry models provide various analysis tools with geometric input, i.e. exterior surface for CFD analysis, solid geometry for FE-analysis, and mass properties for multi-body dynamics.

Since the terminology design automation is rather common and does not precisely describe and limit the field explored in the thesis, the concept of Geometry Based Design Automation (GDA) has been coined to indicate high fidelity geometry models that can be automatically generated within design optimization loops to enable physics-based analyses of the product at hand.

The methodologies for geometric modelling proposed in this thesis are based on top-down modelling, with the fundamental difference that the model hierarchy does not remain static when evaluating particular concepts. KBE is used to manage the framework which creates the geometric CAD models. The geometries are stored as HLCts and instantiated into the CAD models according to the logic described in the inference engine. By using high-fidelity tools already in the conceptual design phase, the geometry models can be seamlessly further developed in later phases. Hence, the adoption of the discussed methods eliminates the need to re-generate models between conceptual and preliminary design phases: after a certain number of design instantiations it is therefore expected that the total design process lead time could be shortened. Moreover, the geometry models are suitable for inclusion in multidisciplinary design and optimization (MDO) environments.

Geometric models can be included in different ways in an MDO framework. In the thesis a brief description of the most common is presented in CHAPTER V. Moreover, the appended papers elaborate the subject by presenting how the different solutions have been employed in research projects. In appended paper [V] an MDO framework for unmanned aircraft is discussed. The framework only included a simplified mathematical model that allowed geometrical properties to be evaluated. Conversely, appended papers [I], [II] and [IV] describe applications where a high-fidelity CAD model has been included in the optimization loop by means of GDA techniques.

The application examples presented in CHAPTER X and those in appended papers [I] and [II] all describe the automation of the design of complex engineering products, in order to verify the hypothesis concerning elimination of repetitive work. The application examples confirmed that various aspects of repetitive and non-creative design can be eliminated by implementing dynamic top-down modelling and storing recurring geometries as HLCts. HLCts then also offer the opportunity to reuse design efforts. The MAV design automation example in particular proved that in some cases

the whole design process can be automated by means of a computer-based system that takes a set of initial requirements to produce a manufactured finished product at the end.
Based on the template instantiation mechanisms, a clear distinction has been made between Design Reuse (DR) and GDA. The first allows both the topological transformations (occurring when generating an instance from an HLCt) and the morphological transformations (necessary to adapt the instance to its surrounding context) to be manual. To achieve GDA instead, all processes need to be automated, typically by means of scripts within the inference engine. If not, geometry models would not be suitable for being in-the-loop in optimization problems.
To address the effectiveness of geometry models, the terms flexibility and robustness are often used in a qualitative fashion. However, a need has been identified for a simple and usable formulation by which to quantify these characteristics. Mathematical models of flexibility and robustness in relation to the geometry model's design space size have therefore been developed. Moreover, in appended paper [II], the flexibility and robustness of the application examples have been computed and thereby used to objectively compare models. It is expected that these formulations can help to ensure quality when developing parametric CAD models. Even though, as highlighted in the discussion chapter, more work is still required, it is believed that these formulations are a necessary first step in the right direction.
The adoption of MDO techniques and physics-based analyses of higher fidelity can lead to paradigm shifts and revolutionary design solutions. This is clearly favourable from a product performance perspective, but represents a challenge from a modelling and design process point of view. Novel design solutions can hide unknown and unexpected characteristics that design models need to capture and replicate correctly. A revised design process is therefore proposed in CHAPTER IX, to incorporate cost-effective prototype testing already from the early design phases. The goal of the tests is to validate the modelling results produced by the MDO design framework and help reveal unanticipated phenomena. The proposed process is based on fast, cheap manufacturing technologies to quickly deliver prototypes that can be used in testing campaigns. CHAPTER X and appended papers [III] and [IV] present examples of how the described methods have been used successfully. The reported examples are based on the experiences from two projects: one aimed at building and test flying a subscale generic future fighter aircraft and one whose goal was to validate the propulsion system components' models used to automatically design MAVs by means of test flights.

XII.1 Recommended Future Research Directions

Trying to give an answer to the research questions listed at the beginning of this thesis has inevitably resulted in suggesting several new research directions. In this final section, a shortlist is presented of the ones that are believed to be most urgent.

- A main part of the High Level CAD modelling methodology described in the thesis is represented by the scripts that control the instantiation and transformation of HLCts. In the case of complex models with several templates to be controlled, these codes can grow to be quite large until their maintenance becomes a burden. Especially in an industrial environment, the scripts would require a standardized and easily understandable interface to make them easier to manage. Of particular interest is the case where the person who wrote the scripts would quit the job, retire or simply move to another position. A tool in which the structure of a complex engineering product could be designed, taking into account all the necessary templates and their mutual relationships would therefore certainly be welcome. To give an example: process integration tools have been developed to easily design MDO frameworks and manage the connections between several tools. Similarly, a tool could be designed to design a (meta)-CAD model and manage all its HLCts.
- If design automation techniques are to be introduced in real industrial applications, it must be possible to assess costs, risks, return on investment and advantages that can be achieved. At the present time, the field lacks documented cases of both successes and failures that can be referred to. It is therefore recommended that a database of applications be built up that

would report at least the same kind of data as shown in the examples presented in section XI.1.

- At the present time, the robustness and flexibility models are bound to continuous design spaces. Only morphological transformations can therefore be accounted for by the models. Further work is necessary to widen the applicability of the formulations, taking particular care to maintain simplicity of use. For the models to gain acceptance in the field it is vital that the measures can be performed quickly and easily. Also, as stated in previous chapters, it would be helpful if the measures could also relate to the CAD model's complexity.
- The choice of suitable cases for KBE and GDA applications is no less than fundamental, but a methodology for search, classification and selection is still lacking. Many researchers have written about the importance of this topic and shown how improper choice of application field has resulted in failures, despite valid tools and automation strategies being used.

CHAPTER XIII.

SUMMARY OF PAPERS

This chapter provides short summaries of the appended papers. The aim is to clarify the role of each paper within the thesis and help the reader put the papers in their correct perspective.

PAPER [I]

AUTOMATED DESIGN AND FABRICATION OF MICRO-AIR VEHICLES

This paper describes a multidisciplinary design framework that allows the automatic design and manufacturing of Micro Air Vehicles (MAVs). From an initial set of requirements, entered through a spreadsheet interface, the computer system is able to optimize the design of the aircraft with respect to geometry, selection of propulsion system components, and performance. At the end of the MDO cycle, a complete 3D CAD geometry is available for automatic fabrication by means of a 3D printer. The paper illustrates how, in some cases, the whole design process can indeed be automated.

The paper's co-author worked mainly on the propulsion system components' modelling, user interface and low-fidelity aerodynamic modelling. The automatic CAD geometry generation, high fidelity modelling, optimization and process integration with modeFRONTIER were the author's main contribution.

PAPER [II]

FLEXIBLE AND ROBUST CAD MODELS FOR DESIGN AUTOMATION

This paper investigates novel methodologies for enabling Multidisciplinary Design Optimization (MDO) of complex engineering products. Instead of repeatedly modelling similar instances of objects, the proposed methodologies allow engineers to create more general models that can represent entire classes of objects. High Level CAD templates (HLCt) are proposed and discussed as the building blocks of flexible and robust CAD models, which in turn enables high-fidelity geometry in the MDO loop. Quantification of the terms flexibility and robustness is also presented, providing a means to measure the quality of the geometry models.

All the authors have worked in close collaboration on the paper, which summarizes many of the theoretical findings that have been gained over several research projects. It is therefore very difficult to explicitly list the contributions of each co-author.

PAPER [III]
DEVELOPMENT OF A SUBSCALE FLIGHT TESTING PLATFORM FOR A GENERIC FUTURE FIGHTER

The paper describes the planning, manufacturing and testing of a generic future fighter subscale demonstrator that was built at Linköping University. The idea behind the project is that, in some cases, a flight test will provide more answers than several computations ever could. Subscale flying concepts must therefore be able to be created quickly and as much reliable information as possible obtained from the tests. The paper covers design and manufacturing aspects, as well as test instrumentation equipment.
The author's contributions to the project were manufacturing planning, design and execution, and development of a process that would include prototype testing to validate MDO results.

PAPER [IV]
EVALUATION OF AUTOMATICALLY DESIGNED MICRO AIR VEHICLES AND FLIGHT TESTING

The paper describes the evaluation of Micro Air Vehicles (MAV) that have been automatically designed and manufactured. A MAV design automation framework uses several coupled computer programs to generate the geometric design in CAD, as well as a list of off-the-shelf components for the propulsion system, and computer code for autonomous flight ready to upload to the intended autopilot. The paper reports results from the design optimization process and contains a comparison between predicted and measured performance.

The author's contribution to the paper was the in-depth analysis of the optimization algorithm and the search for suitable settings to enhance design space search performances.

PAPER [V]
USE OF PANEL CODE MODELLING IN A FRAMEWORK FOR AIRCRAFT CONCEPT OPTIMIZATION

The paper presents a multidisciplinary design optimization framework for unmanned aircraft. On the one hand, the work aimed at developing a module for aerodynamic analysis of concepts as a basis for a direct search optimization of the concept layout. On the other, a comparison between three different optimization algorithms is reported: one gradient-method-based (Fmincon), one non-gradient-based (Complex) and one Genetic Algorithm (GA). The paper makes two main contributions to the thesis. Firstly, to give an example of an MDO framework where the geometrical model is absent and geometrical properties are instead predicted by a simplified mathematical model, and secondly to show how important the choice of optimization algorithm is to the result.

CHAPTER XIV.

REFERENCES

[1] Ammar-Khodja, S., Perry, N., Bernard, A., ”Processing Knowledge To Support Knowledge-Based Engineering System Specification”, Concurrent Engineering Research and Applications, Vol. 16, Issue 1, 2008, pp. 89-101

[2] Andersson, J. and Krus, P., “Metamodel Representations for Robustness Assessment in Multiobjective Optimization”, Aug. 2001, ICED01 International Conference On Engineering Design, Glasgow, UK

[3] Awad E.M., “Building knowledge automation expert systems”, Exsys Inc., 2003

[4] Balling, R.J., Sobieszczanski-Sobieski, J., “Optimization of Coupled Systems: A Critical Overview of Approaches”, ICASE Report No.94-100, NASA Contractor Report 195019, 1994

[5] Bartholomew P., “The Role of MDO within Aerospace Design and Progress Towards an MDO Capability”, Sep. 1998, AIAA 1998-4705, AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, MO, USA

[6] Berard, A., Rizzi, A., Isikveren, A.T., “A New Geometry Construction Tool for Aerospace Vehicle Pre-Design and Conceptual Design”, Sep. 2008, ICAS 2008, Anchorage, AK, USA

[7] Blount, G.N., Kneebone, S., Kingstone, M.R., “Selection of Knowledge-based Engineering Design Applications”, Journal of Engineering Design, Vol. 6, No. 1, 1995, pp. 31-38

[8] Böhnke D., Reichwein A., Rudolph S., “Design Language for Airplane Geometries Using the Unified Modeling Language”, Aug. 2009, DETC2009-87368, Proceedings of the ASME 2009 International Design Engineering Technical Conference & Computers and Information in Engineering Conference, San Diego, CA, USA

[9] Bowcutt K.G., "A Perspective on the Future of Aerospace Vehicle Design", Dec. 2003, AIAA 2003-6957, 12th AIAA International Space Planes and Hypersonic Systems and Technologies, Norfolk, VA, USA

[10] Brandt, S.A., Stiles, R.J., Bertin, J.J., Whitford, R., "Introduction to Aeronautics: A Design Perspective", AIAA Educational Series, Reston, VA, USA, 1997

[11] CATIA on Dassault Systèmes' homepage, www.3ds.com/products/catia/welcome/, last accessed on 24-05-2011

[12] CATIA V5 Knowledgeware Solutions, http://www.3ds.com/products/catia/portfolio/catia-v5/all-products/domain/Product_Synthesis/?no_cache=1&cHash=2390190825, last accessed 16-05-2011

[13] Cederfeldt, M., "Planning Design Automation – A Structured Method and Supporting Tools", Doctoral Thesis, Chalmers University of Technology, Göteborg, Sweden, 2007

[14] Chang K.-H., Silva J., "Design Parameterization for Concurrent Design and Manufacturing of Mechanical Systems", Sep. 2001, DETC2001/DFM-21161, Proceedings of ASME 2001 Design Engineering Technical Conference and Computers and Information in Engineering Conference, Pittsburg, PA, USA

[15] Chapman C.B., Pinfold M. "Design engineering – a need to rethink the solution using knowledge based engineering", Journal of Knowledge Based Systems, Vol. 12, Issue 5-6, Oct. 1999, pp. 257-267

[16] Chapman, C.B., Pinfold, M., "The application of knowledge based engineering approach to the rapid design and analysis of an automotive structure", Journal of Advances in Engineering Software, Vol. 32, Issue 12, Dec. 2001, pp. 903-912

[17] Crawford C.A., Haimes R., "Synthetizing an MDO Architecture in CAD", Jan. 2004, AIAA 2004-281, 42nd AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, USA

[18] Davis, W.R., Kosicko, B.B., Boroson, D. M., Kostishack, D. F., "Micro Air Vehicles for Optical Surveillance", The Lincoln Laboratory Journal, Vol 9. Issue 2, 1996, pp197-213

[19] De Weck O., Agte J., Sobieszczanski-Sobieski J., Arendsen P., Morris A., Spieck M., "State-of-the-Art and Future Trends in Multidisciplinary Design Optimization", Apr. 2007, AIAA 2007-1905, 48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Material Conference, Honolulu, HI, USA

[20] Deets, D.A., DeAngelis, V.M., Lux, D.P., "Himat Flight Program: Test Results And Program Assessment Overview", Technical Report NASA TM-86725, 1986

[21] Doherty J.J., McParlin S.C., "Generic Process for Air Vehicle Concept Design and Assessment", AIAA 2004-895, Jan. 2004, AIAA 2004-693, 42nd AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, USA

[22] Dornheim, M.A., "McDonnell Douglas Rolls Out X-36", Aviation Week & Space Technology, Vol. 144, Issue 13, 1996

[23] Emberey, C.L., Milton, N.R., "Application of Knowledge Engineering Methodologies to Support Engineering Design Application Development in Aerospace", Sep. 2007, AIAA-2007-7708, 7th AIAA Aviation Technology, Integration and Operations Conference (ATIO), Belfast, Northern Ireland

[24] Frappaolo C., “Implicit Knowledge”, Journal of Knowledge Management Research & Practice, No.6, 2008, pp.23-25

[25] Frappaolo, C., “What's Your Knowledge IQ?” Delphi Group Case Studies, Mar. 2005, at http://www.delphigroup.com/consulting/case_studies/knowledge_IQ.htm, last accessed on 06-12-2011

[26] Gibson, C.S., Neidhoefer, J.C., Cooper, S.M., Carlton, L., Cox, C.J., “Development And Flight Test Of The X-43A-LS Hypersonic Configuration UAV”, May 2002, AIAA 2002-3462, 1st UAV Conference, Portsmouth, VA, USA

[27] Giesing J.P., Barthelemy J-F.M., “A summary of industry MDO applications and needs”, Sep. 1998, AIAA 1998-4737, AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, MO, USA

[28] Hahne, D.E., Wendel, T.R., Boland, J.R., “Wind-Tunnel Free-Flight Investigation of a Supersonic Persistence Fighter”, Technical Report NASA TP-3258, 1993

[29] Henriksson, K.E., “Spin Testing the Viggen Aircraft. Technical Report”, Society of Experimental Test Pilots, 1974

[30] Hönlinger H.G., Krammer J., Stettner M., ”MDO Technology Needs in Aeroelastic Structural Design”, Sep. 1998, AIAA 1998-4731, AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, MO, USA

[31] Hopgood A.A., “Intelligent Systems for Engineers and Scientists”, CRC Press, Boca Raton, 2001

[32] Huang G.Q., Brandon J.A., “Cooperating Expert Systems in Mechanical Design”, John Wiley & Sons, Inc., New York, NY, USA, 1193

[33] IBM Watson homepage, http://www-03.ibm.com/innovation/us/watson/index.html, last accessed on 14-10-2011

[34] Internet Encyclopaedia Wikipedia, www.wikipedia.org, last accessed on 06-12-2011

[35] Jenkinson L.R., Simpkin P., Rhodes D., “Civil Aircraft Design”, American Institute of Aeronautics & Astronautics, Reston, USA, 2000

[36] Johansson, J., “Automated Computer Systems for Manufacturability Analyses and Tooling Design”, Doctoral Thesis, Chalmers University of Technology, Göteborg, Sweden, 2011

[37] Johansson, J., “Design Automation Systems for Production Preparation – Applied on the Rotary Draw Bending Process”, Licentiate Thesis, Chalmers University of Technology, Göteborg, Sweden, 2008

[38] Klein, V., Noderer, K.D., “Aerodynamic Parameters of The X-31 Drop Model Estimated From Flight-Data at High Angles Of Attack”, Aug. 1992, AIAA-1992-4357, AIAA Atmospheric Flight Mechanics Conference, Hilton Head Island, SC, USA

[39] Kraft, E. M. and Matty, J. J. “Applications of High Performance Computing at the Arnold Engineering Development Center,” ITEA Journal, Vol 26, No.2, June/July 2005

[40] Kurzweil, R., “The Singularity Is Near”, The Viking Press, 2005

[41] La Rocca G., “Knowledge Based Engineering Techniques to Support Aircraft Design and Optimization”, Doctoral Thesis, Delft University of Technology, Delft, The Netherlands, 2011

[42] La Rocca, G., and van Tooren, M.J.L., “Enabling distributed multi-disciplinary design of complex products: a knowledge based engineering approach”, Journal of Design Research, Vol. 5, No. 3, 2007, pp.333-352

[43] La Rocca, G., van Tooren, M.J.L., “A Knowledge Based Engineering Approach to Support Automatic Generation of FE Models in Aircraft Design”, Jan. 2007, AIAA 2007-967, 45th AIAA Aerospace Science Meeting and Exhibit, Reno, NV, USA

[44] La Rocca, G., van Tooren, M.J.L., “Knowledge-Based Engineering Approach to Support Aircraft Multidisciplinary Design and Optimization”, Journal of Aircraft, Vol. 46, No. 6, Nov.-Dec. 2009, pp. 1875-1885

[45] Ledermann C., Hanske C., Wenzel J., Ermanni P., Kelm R., ”Associative parametric CAE methods in the aircraft pre-design”, Journal of Aerospace Science and Technology, Vol. 9, Issue 7, Oct. 2005, pp. 641-651

[46] Ledermann, C., Ermanni, P., Kelm, R., ”Dynamic CAD objects for structural optimization in preliminary aircraft design”, Journal of Aerospace Science and Technology, Vol. 10, Issue 7, Oct. 2006, pp. 601-610

[47] Lin B.-T., Hsu S.-H., “Automated design system for drawing dies”, Journal of Expert Systems with Applications, Vol.34, Issue 3, Apr. 2008, pp.1586-1598

[48] Liu Q., Xi J., “Case-based parametric design system for test turntable”, Journal of Expert Systems with Applications, Vol.38, Issue 6, June 2011, pp.6508-6516

[49] Manning, T.A., Gage, P.J., Nguyen, J.M., ”ComGeom2: A Geometry Tool for Multidisciplinary Analysis and Data Sharing”, Sep. 2004, AIAA-2004-4303, 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, NY, USA

[50] Mäntylä M., “A Modeling System for Top-Down Design of Assembled Products”, IBM Journal of Research and Development, Vol.34, Issue 5, Sep. 1990, pp.636-659

[51] Mawhinney P, Price M, Curran R, Benard E, Murphy A, Raghunathan S., “Geometry-based approach to analysis integration for aircraft conceptual design”, Sep. 2005, AIAA-2005-7481, 5th Annual Aviation Technology, Integration & Operations (ATIO) Forum, Washington, DC, USA

[52] McCabe, T.J., “A Complexity Measure”, IEEE Transactions on Software Engineering, Vol.SE-2, No. 4, Dec. 1976, pp.308-320

[53] McCarthy J., “What is Artificial Intelligence”, Computer Science Department, Stanford University, Stanford, CA, USA, 2007

[54] McMaster, J., Cummings, R.M., “Airplane Design-Past, Present, and Future”, Journal of Aircraft, Vol. 39 Issue 1, 2002, pp 10-17

[55] McMasters J.H., Cummings R.M., “Rethinking the Airplane Design Process – An Early 21st Century Perspective”, Jan. 2004, AIAA 2004-693, 42nd AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, USA

[56] Merriam-Webster's Collegiate Dictionary, www.merriam-webster.com

[57] Mills H.D., "Software Development", IEEE Transactions on Software Engineering, Vol.SE-2, Issue 4, Dec. 1976, pp.265-273

[58] Milton, N., "Knowledge Acquisition in Practice: A Step-by-Step Guide", Springer-Verlag, London Ltd., London, UK, 2007

[59] modeFRONTIER, Process Integration and Design Optimization Software, v.4.1.0, ESTECO, Trieste, Italy, URL: www.esteco.com (last access 2011-09-19)

[60] Mueller, T.J., "Overview of Micro-Air-Vehicle Development", Introduction to the Design of Fixed-Wing Micro Air Vehicles, American Institute of Aeronautics and Astronautics, Reston, 2007, pp.1-38

[61] Mullins C.S., "What's knowledge and how can it be managed", The Data Administration Newsletter, March 1, 1999

[62] Myung S., Han S., "Knowledge-based parametric design of mechanical products based on configuration design method", Journal of Expert Systems with Applications, Vol.21, Issue 2, Aug. 2001, pp.99-107

[63] Nickol, C., "Conceptual design shop", presentation to Conceptual Aircraft Design Working Group (CADWG21), Sep. 2004

[64] Onwubinko C., "Introduction to Engineering Design Optimization", Prentice-Hall Inc., 2000

[65] Oxford Dictionaries, www.oxforddictionaries.com

[66] Pinard, P., "Maximizing Automation of Intelligent Processes", Apr. 2011, Oral Presentation at COE 2011 Annual Conference & Technifair, Orlando, FL, USA

[67] Polanyi M., "The Tacit Dimension", University of Chicago Press, 2009

[68] Raymer D., "Aircraft Design: A Conceptual Approach", American Institute of Aeronautics & Astronautics, Reston, USA, 2006

[69] Roberts K.N., Chudoba B., "Flight Vehicle Design Heritage: Are We On the Road to the Same Fate as Alexandria?", Jan. 2007, AIAA 2007-657, 45th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, USA

[70] Rodriguez D.L., Sturdza P., "A Rapid Geometry Engine for Preliminary Aircraft Design", Jan. 2006, AIAA 2006-929, 44th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, USA

[71] "Saab Tests Subscale UCAV Concept", Flight International, No. 4842, Vol. 162, 2002, pp. 22-23

[72] Schueler, K., Hale, R., "Object-Oriented Implementation of an Integrated Design and Analysis Tool for Fiber Placed Structures", Apr. 2002, AIAA 2002-1223, 43rd AIAA/ASME/ASCE/AHS/ASC Structural Dynamics, and Material Conference, Denver, CO, USA

[73] Sensmeier M.D., Samareh J.A., "Automatic Aircraft Structural Topology Generation for Multidisciplinary Optimization and Weight Estimation", Apr. 2005, AIAA 2005-1893, 46th

AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Material Conference, Austin, TX, USA

[74] Simpson, T.W., Martins, J.R.R.A., "Multidisciplinary Design Optimization for Complex Engineered System Design: State of The Research and State of the Practice – Report From A National Science Foundation Workshop", Aug. 2011, DETC2011-47237, Proceedings of the ASME 2011 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, Washington, DC, USA

[75] Skarka, W., "Application of MOKA Methodology in Generative Model Creation Using CATIA", Engineering Applications of Artificial Intelligence (20), 2007, pp.677-690

[76] Sohaib, M., "Parameterized Automated Generic Model for Aircraft Wing Structural Design and Mesh Generation for Finite Element Analysis", Master Thesis LIU-IEI-TEK-A--11/01202-SE, Linköping University, Linköping, Sweden, 2011

[77] Stokes, M., "Managing Engineering Knowledge; MOKA: Methodology for Knowledge Based Engineering Applications", Professional Engineering Publishing, London, 2001

[78] Stolt, R., "CAD-Model Parsing for Automated Design and Design Evaluation", Doctoral Thesis, Chalmers University of Technology, Göteborg, Sweden, 2008

[79] Suh N.P., "The Principles of Design – Oxford Series on Advanced Manufacturing", Oxford University Press, New York, NY, USA, 1990

[80] Tomiyama, T., "Intelligent computer-aided design systems: Past 20 years and future 20 years", Artificial Intelligence for Engineering Design, Analysis and Manufacturing, Vol. 21, Issue 1, 2007, pp. 27-29

[81] Torenbeek, E., "Synthesis of Subsonic Airplane Design", Delft University Press, Kluwer Academic Publisher, 1982

[82] Vadenbrande J.H., Grandine T.A., Hogan T., "The search for the perfect body: Shape control for multidisciplinary design optimization", Jan. 2006, AIAA 2006-928, 44th Aerospace Sciences Meeting and Exhibit, Reno, NV, USA

[83] Verhagen, W.J.C., Bermell-Garcia, P., van Dijk, R.E.C., Curran, R., "A Critical Review of Knowledge Based Engineering: An Identification of Research Challenges", Advanced Engineering Informatics, doi:10.1016/j.aei.2011.06.004, 2011

[84] Visual Basic Developing Center, http://msdn.microsoft.com/en-us/vbasic/ms789056, last access 2011-09-19

[85] Wakayama S., Kroo I., "The Challenge and Promise of Blended-Wing-Body Optimization", Sep. 1998, AIAA 1998-4736, AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, St. Louis, MO, USA

[86] Walker, L.A., "Flight Testing the X-36 - The Test Pilot's Perspective", Technical Report NASA CR-198058, 1997

[87] Warwick, G., "Gulfstream Unveils Concepts For Supersonic Business Aircraft", Flight International, No. 4751, Vol. 158, 2000, pp.5, 18-22

[88] Wirth N., “Program Development by Stepwise Refinement”, Communication of the ACM, Vol.14, Issue 4, Apr. 1971, pp.221-227

[89] Yanagihara, M., Miyazawa, Y., Alimoto, T., Sagisaka, M., Cretenet, J.C., Venel, S., “HOPE-X High Speed Flight Demonstration Program Phase II”, Apr. 2001, AIAA Paper 2001-1805, AIAA/NAL-NASDA-ISAS International Space Planes and Hypersonic Systems and Technologies Conference, Kyoto, Japan

[90] Zweber, J.V., Kabis, H., Follett, W.W., Ramabadran, N., ”Towards An Integrated Modelling Environment For Hypersonic Vehicle Design and Synthesis”, Sep. 2002, AIAA-2002-5172, AIAA/AAAF 11th International Space Planes and Hypersonic Systems and Technologies Conference, Orleans, France