Development of an Optimization Framework for Landing Gear Design

P. van Ginneken

Technische Universiteit Delft

TU Delft Delft University of Technology

Challenge the future

Development of an Optimization Framework for Landing Gear Design

by

P. van Ginneken

in partial fulfillment of the requirements for the degree of

Master of Science
in Aerospace Engineering

at the Delft University of Technology,
to be defended publicly on Thursday February 18, 2016 at 10:00 AM.

Supervisor:	Dr. ir. M. Voskuijl,	
Thesis committee:	Prof. dr. ir. L. L. M. Veldhuis,	TU Delft
	Dr. ir. R. Breuker de,	TU Delft
	Ir. P. Vergouwen,	Fokker Landing Gear B.V.

An electronic version of this thesis is available at http://repository.tudelft.nl/.
Thesis registration number: 073#16#MT#FPP
Equivalent number of words = 18,800 + 43 x 200 = 27,400

Abstract

An opportunity was identified to improve the traditional landing gear design process. Especially in the conceptual design phase, too many man-hours are consumed by making the same calculations over and over again, for different concepts. An existing landing gear layout is therefore often used as starting point, to simplify the design process. This results in little technical progress. Additionally, integration between the different disciplines involved is sub-optimal, which can lead to inconsistent results. In this thesis, an optimization framework is described that can perform the preliminary design of a landing gear fully automated. It ensures that communication between disciplines is respected by adding a top-level optimizer which is in charge of changing the design variables. This top-level optimizer will find the optimal landing gear solution, considering the interaction of the different landing gear disciplines and also the interaction between the landing gear and the aircraft. For that purpose, different parametric landing gear topologies will be defined. These different gears are all evaluated to find the optimal design solution. Two main research goals are formulated. The first goal is to define such an optimization framework and the second goal is to create a proof of concept tool that incorporates its principles.

The set-up of the framework can be compared to the different layers of an onion. In four steps, the best preliminary landing gear solution is obtained with an increasing level of detail. In the first step, top-level parameters such as gear layout, strut geometry and strut position are considered based on static load cases and the use of rigid body simulations. Top-level aircraft parameters such as the maximum take-off weight, the cg-range, the wingspan and the touchdown velocity are needed for this step. In the second step, secondary gear components such as lugs and pins are added to the simulation model; the gear geometry is investigated in more detail and the cost of the gear is computed. The third step is to compute dynamic load cases instead of the static ones used before, and to restart from the beginning. As a last step, flexible members are added to the simulation model and more elaborate stress calculations are performed. Constraints to this framework include that foldability of the gear inside the landing gear bay, the fact that no part of the aircraft may touch the ground except for the landing gear, and minimum wall thicknesses for all members.

A proof of concept was developed that is able to perform all calculations defined for the first step of the framework, and some calculations of the second step. A multi-body model is developed to investigate the gear geometry in more detail, but the landing gear cost and the secondary components are not yet added.

Verification of the proof of concept is done by comparing the results against a previous concept study, of which the preliminary design information was available. With similar geometries for the proof-of-concept framework and the previous concept study, the weight computed by the framework has an error margin of around 4 percent. If the gear geometry is allowed to change, the proof of concept tool is able to reduce the gear weight with 7 percent compared to the original concept study.

Ultimately, the realization of this framework greatly reduces the repetitive tasks in the design phase of a landing gear. This shortens the timespan of the design phase, opening the possibility to also evaluate non-standard solutions that may be lighter, safer and/or cheaper.

Acknowledgments

This research would not have been possible without the excellent help and working environment offered by Fokker Landing Gear B.V. I am very thankful to have been given the opportunity of doing a final thesis at this inspiring place.
There were quite a few people involved in helping and motivating me, but five need to be mentioned by name. First, Peet Vergouwen of FLG and Mark Voskuijl of Delft Univerity of Technology for being my tutors. Then, Bert Verbeek of FLG for creating the VLM model and encouraging me to try new programming methods. I also want to thank Mike Smeets for his analytic work on the initial concept gear and helping me to understand it. Finally, thanks to Rob Bergers for proof-reading most of my documents and for keeping spirits up.

Contents

List of Figures

List of Tables

Nomenclature

List of symbols

Symbol	Unit	Description
A	m^2	cross-sectional area
a	m	horizontal distance between nose landing gear and centre of gravity
B	m	wingspan
b	m	horizontal distance between main landing gear and centre of gravity
cg-range	m	the distance between the most aft and most forward possible cg position
D	N	drag force in global coordinate system
d	m	horizontal distance between main landing gear and nose landing gear
E	Pa	Young's modulus
e	m	vertical distance between ground and centre of gravity
F	N	force acting on a member
f	-	1) dynamic response factor
h	m	height
I	m^4	area moment of inertia
J	m^4	polar moment of inertia
L	m	length
M	N*m	moment
n	-	load factor
p	Pa	pressure
Q	m^3	first moment of area
R	-	Ratio
r	m	radius
S	N	side force in global axis system
T	N*m	Torque
t	m	1) lateral distance between the main gear struts
		2) thickness
u	m	distance between centres of main gear tyres
V	N	vertical force in global axis system
v	m	distance between centres of nose gear tyres
W	N	weight
w	m	width
x	m	x position with respect to reference frame aircraft
y	m	y position with respect to reference frame aircraft
z	m	z position with respect to reference frame aircraft

Greek letters

Symbol	Unit	Description
β	rad	nose-wheel steering angle
Γ	rad	dihedral angle of the wing
Δ	rad	sweep angle of the wing
δ	m	clearance
θ	rad	pitch angle
μ	-	friction coefficient
σ	Pa	normal stress
τ	Pa	shear stress
ϕ	rad	roll angle

ψ	rad	sideways turnover angle

SUBSCRIPTS

Symbol	Description
a	axial
av	average
b	bending
br	bearing
h	hoop stress
i	inner
L	left
M	main gear
max	maximum
min	minimum
N	nose gear
o	outer
R	right
r	radial
s	shear
st	torsional shear
t	tension / tensile
tr	transverse
u	ultimate
x	direction of the local x-axis
y	direction of the local y-axis
z	direction of the local z-axis

Acronyms

AAC Aircraft Approach Category.

AC Advisory Circular.

ADG Aircraft Design Group.

BLISS Bi-Level Integrated System Synthesis.

CAD Computer Aided Design.

CG Centre of Gravity.

CO Collaborative Optimization.

CSSO Concurrent Sub-Space Optimization.

DLR German Aerospace Center.

FBD Free Body Diagram.

FEM Finite Element Method.

GA Genetic Algorithm.

IDF Individual Discipline Feasible.

KBE Knowledge Based Engineering.

LIT Landing gear In Thirty days.

MBS MultiBody Simulation.

MDF Multidisciplinary Feasible.

MDO Multidisciplinary Design Optimization.

MF Main Fitting.

MLG Main Landing Gear.

MLW Maximum Landing Weight.

MoS Margin of Safety.

MRW Maximum Ramp Weight.

MTOW Maximum Take-Off Weight.

NLG Nose landing Gear.

OEM Original Equipment Manufacturer.

OOP Object-Oriented Programming.

POC Proof of Concept.

RA Retract Actuator.

SA Shock Absorber.

SQP Sequential Quadratic Programming.

TA Trailing Arm.

TG Tail Gear.

WA Wheel Axle.

1 INTRODUCTION

How can the time spent in the concept design phase be decreased, while also delivering a product with a higher level of detail? That question triggered this thesis project. When Fokker Landing Gear B.V. (FLG) is asked to do a concept study, it is the start of a hectic period. Different landing gear (LG) solutions need to be evaluated and the customer needs to be convinced that the offered solution is better than the concepts proposed by the competition. Among many other activities, preliminary stress calculations are made to size the landing gear components, a shock absorber is defined, the kinematics are evaluated, the rolling stock (tyres, brakes, wheels) is selected and the virtual product is defined in a CAD program.
Four main problems are identified during this concept design phase. Firstly, engineers are spending most of their time performing repetitive work, not creative. The same calculations are made over and over to answer the question "what if part X is moved to position Y?". Some simple and self-written Mathcad sheets are available for different stages of the conceptual design process, but post-processing still takes more time than wanted. Secondly, the initial concepts for a new aircraft are often based on an existing gear, to simplify and speed up the design process. In the best case, this leads to a non-optimal, but acceptable, solution. In the worst case, there is no existing gear that fulfils all requirements. Thirdly, the coordination between different involved disciplines is not optimal. Even in concept design, there are many parameters involved and usually several concepts are evaluated at the same time. It is critical that everyone involved uses the same parameters for their calculation. Although this may seem trivial, previous experience shows that this is often not the case. Lastly, the LG and the aircraft are often designed as two separate systems, without much integration[1]. This may result in an incomplete set of requirements for the landing gear. If that is discovered in a late stage of the design process, it can in turn lead to unwanted, costly, and time-critical redesign activities.
This thesis investigates if the landing gear design process can benefit from the use of engineering optimization methods, to overcome the above mentioned problems. For that purpose, two main research objectives are set. First, an optimization framework for landing gear design is defined. It describes how MDO can be used to find the best landing gear for any aircraft, within the boundaries of preliminary design. This framework must be able to run with a limited amount of input parameters and help the design team to determine the conceptual landing gear within six weeks. It is therefore named the "Landing gear In Thirty days" tool: LIT.
LIT can ultimately be used to search a target of the best landing gear for *any* aircraft, while eliminating many of the repetitive design tasks done by a designer and freeing time for creative investigations. This results in a broader (more options considered) and more detailed analysis of possible landing gear solutions, within the same time frame that is needed nowadays.
The second objective then is to build a proof of concept (POC) tool. This POC tool incorporates the principles described for LIT and proves that these principles have practical purpose.

Previous research in this field has been conducted by Chai and Mason [1], Heerens [2], and Cumnuantip [3]. These publications are described in section 1.3 This chapter first gives an overview of the two disciplines involved in this thesis: landing gear design and engineering optimization. The research objective can be found in section 1.4, while section 1.5 explains the layout of this thesis.

[1] Interviews with Mike Smeets, Peter de Haan, Tjaard Sijpkes, Fokker Landing Gear B.V.

1.1. PRINCIPLES OF LANDING GEAR DESIGN

It is assumed that most readers of this text are already familiar with the basics of LG design and therefore it is kept short. If that is not the case and if you are intested, *Aircraft landing gear design: principles and practices* [4] is recommended. For landing gear design from the perspective of an aircraft designer, part three of the Roskam series is an excellent reference [5].
The landing gear has three principle functions: to provide a smooth ride for passengers, to introduce the ground reaction loads into the airframe structure and, optionally, to retract into the landing gear bay. This section discusses two top-level design choices: the gear layout and the strut topology. The different components of a gear strut are described in section 5.1.

1.1.1. LANDING GEAR LAYOUT

There are two common options for the LG layout: a tricycle landing gear and a taildragger landing gear. The tricycle configuration is seen on virtually all larger passenger aircraft, but also on many helicopters. The tricycle layout has its Main Landing Gear (MLG) behind the Nose landing Gear (NLG). The main advantages of a tricycle landing gear are that the aircraft floor is level to the ground during taxi, that the pilot has good visability from the cockpit and that the steering characteristics on the ground are good compared to the other configurations [5].
A taildragger gear is most often seen on smaller aircraft but also on helicopters. The taildragger has its MLG in front of the Tail Gear (TG). The main advantage of a taildragger configuration is that it is usually light [5]. Disadvantages are the visibility from the cockpit due to its raised nose and the take-off procedure which is more difficult with a taildragger configuration.
A third type that is rarely used is the bicycle gear layout of the B47. It is only used when the other two options are not feasible, due to very specific aircraft requirements. It is both complex and heavy and will not be considered in this thesis.
The tricycle, taildragger and bicycle gear are visualized in figure 1.1:

(a) Landing gear of the Lynx Mk9: tricycle [6].

(b) Landing gear of the Apache: taildragger [7].

(c) Landing gear of the B47: bicycle [8].

Figure 1.1: Tricycle, taildragger and bicycle gear layouts

1.1.2. LANDING GEAR TOPOLOGY

Figure 1.2, shows some of the standard landing gear strut topologies. This section explains each of them and discuss their benefits and weaknesses.

TELESCOPIC GEAR

Also called cantilevered gear, the telescopic gear is the least complex strut arrangement. It has three main components: the shock absorber , a main fitting and an actuator or side brace. The cantilevered gear is mostly used as NG or TG.
Because of its simplicity, it is the cheapest option, both in terms of aqcuiring cost, as in maintenance [9]. Disadvantages of this topology are that it requires a relatively long shock absorber length and that only simple retraction schemes are possible: forward or aft retraction with optionally a rotation of the wheel(s).

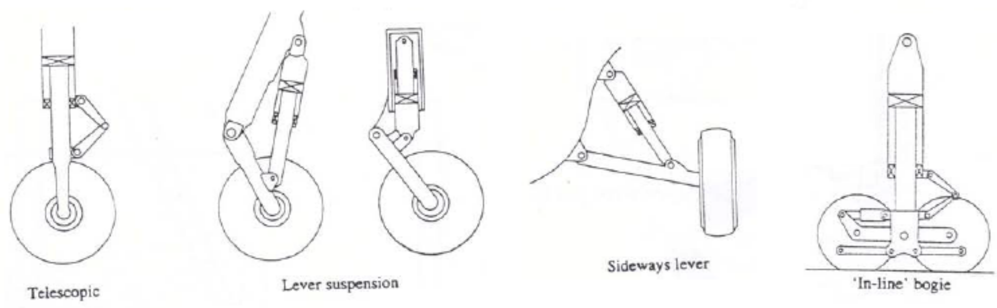


Figure 1.2: Standard strut topologies in use. Figure adapted from [9].

LEVERED SUSPENSION GEAR

More complicated than a telescopic gear, the levered gear offers more design freedom regarding the placement of the shock absorber. This may lead to a shorter gear. Another advantage is its performance on rough airfields, which is considered excellent [9], [4]. The third advantage of the levered gear is that it can fold in a smaller space than the telescopic gear.
Because of its excellent rough field behaviour, the trailing arm LG is often seen on helicopters. Disadvantages are its increased complexity, weight and cost.

SIDEWAYS LEVERED GEAR

The wing is not always suited to mount the landing gear under, although it is often the first option. Reasons can be that the aircraft doesn't have a fixed wing (for helicopters), that the wing is too thin (for a fighter jet) or that the aircraft has a high wing. In these cases, a sideways levered gear becomes an option. Being attached to the body of the aircraft, the sideways levered gear is relatively complex and heavy.
One advantage of the sideways levered gear is that it can fold in a narrow space. The more complex the retraction scheme however, the higher its cost; both in purchase price and maintenance.

BOGIE GEARS

When a gear strut contains three or more wheels, it is called a bogie. These topologies are usually seen as the main gear strut of large passenger aircraft. The design of a landing gear for such an aircraft is however not foreseen in the near future of FLG. Therefore, the bogie design is not considered further.

OTHER LANDING GEAR TOPOLOGIES

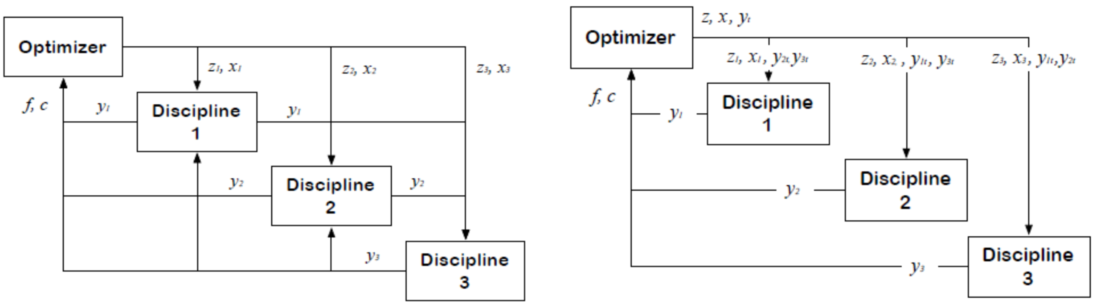
Of course, there are more than four possible landing gear topologies. One might even argue that every landing gear has its own unique topology. Most can be categorized under one of the above mentioned types though.

1.2. PRINCIPLES OF ENGINEERING OPTIMIZATION

This section discusses the advantages of Knowledge Based Engineering (KBE) and Multidisciplinary Design Optimization (MDO) in section 1.2.1. Section 1.2.2 then goes into more detail about different MDO architectures and section 1.2.3 discusses the optimizer. Again, this section is only an introduction into engineering optimization. Two books on this topic can be recommended for those not familiar with it: *Principles of optimal design: modeling and computation* [10] and *Practical optimization* [11].

1.2.1. AVAILABLE ENGINEERING OPTIMIZATION METHODS

KBE is a merger of the fields of Object-Oriented Programming (OOP), artificial intelligence, and Computer Aided Design (CAD) [12]. It is a method that stores information about previously solved problems to solve similar problems faster. This is especially powerful when a problem is iterative and repetitive, as any engineering design problem is. Time reductions of more than 95 percent have been claimed by automating labour intensive, repetitive design tasks with KBE [13], [14], [15].
To optimally use KBE, a design process should be "*highly rule-driven, multidisciplinary, repetitive and demanding geometry manipulation and product (re)configuration*" [16]. It can thus be concluded that KBE is best applied in the detailed design phase, where variations between different configurations are small, geo-



(a) MDF [21]. (b) IDF [21].

Figure 1.3: Principles of IDF and MDF

metric models exist, and extensive amounts of information about the product are available.
During the preliminary design process of a landing gear, which is highly search oriented and where little knowledge about the product is available, KBE is less suited.

MDO solves engineering problems that involve different disciplines by taking into account their interactions. As stated by Schönning *"It should be noted that the multidisciplinary solution might not be the solution for any one discipline analyzed separate from the other disciplines, but is the best solution accounting for the interactions"* [17]. Thus, MDO is best applied to problems that have multiple, coupled, disciplines.
The basis of any MDO technique is to minimize[2] an objective function. Given a set of design variables, the system optimizer searches their optimal value. The set of all design variables is called the design space. Typically, there are also constraints In the case of a landing at high angle of attack for example, the gear struts should be positioned so that the aircraft cannot tip over and crash [18].
The mathematical formulation of a typical MDO problem is:

$$\begin{aligned} &minimize\, f(\mathbf{x},\mathbf{y}) \\ &subject\,to: h(\mathbf{x},\mathbf{y}) = 0 \\ &g(\mathbf{x},\mathbf{y}) \leq 0 \end{aligned} \tag{1.1}$$

where the objective function is f, the design variables are $\mathbf{x}$ and $\mathbf{y}$ and the constraints are given by h and g.
It is concluded that the best optimization technique for LIT, in this stage of development, is MDO.

1.2.2. MDO ARCHITECTURES

Now that it is decided to use MDO, an architecture must be chosen. A description of MDO architectures is available from several sources, including Tedford and Martins [19], Keane and Nair [20], Martins and Carriage [21], and Kroo [22]. The Multidisciplinary Feasible (MDF), the Individual Discipline Feasible (IDF) and the Collaborative Optimization (CO) architecture are most interesting for the scope of this thesis.
The principle of MDF, also called fully integrated optimization or all-in-once optimization, is illustrated in figure 1.3a. Unique for the MDF technique is that variables can be shared by disciplines. A value computed in discipline 1 can be used by discipline 2 and vice versa. This mimics the iterative nature of the design process. The optimizer changes variables x1-x3 and z1-z3, while the disciplines find the optimal value for y1-y3 at each iteration.
This can result in long computational times if the step sizes for y1-y3 are small and/or if the runtime of a certain discipline is long (e.g. if a finite element computation must be performed).
This interdisciplinary sharing of variables is not allowed in the IDF technique, see figure 1.3b. With IDF, the optimizer is in control of changing variables. Analogous to the MDF technique, discipline 1 calculates the y1, based on x1, z1, y2 and y3. y2 and y3 are however dummy variables; a "guess" made by the optimizer. They are computed in disciplines 2 and 3. To allow this decomposition, constraints are added. When the optimization is finished, dummy variable y2 should equal y2 as computed by discipline 2. The same holds for y1 and y3.
The advantage of IDF is that every discipline is only evaluated once per iteration. Another advantage is that the various disciplines can be evaluated in parallel, when the hardware allows so.

[2] or maximize. In that case, give the negative value to the routine.

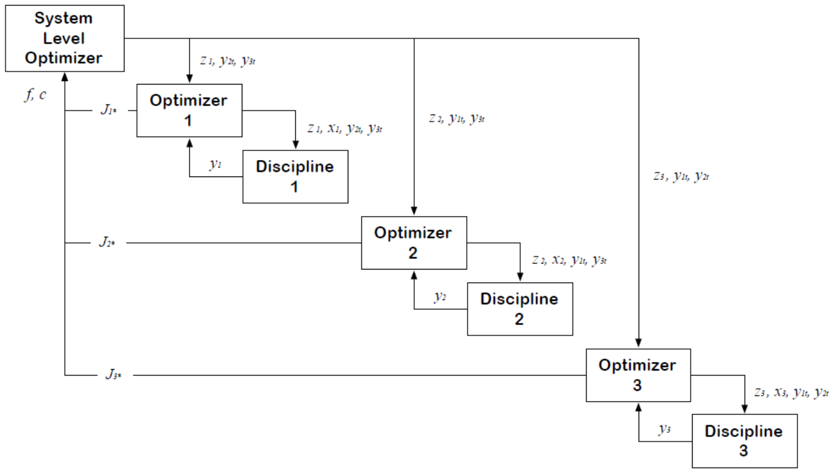


Figure 1.4: Principle of CO [21].

The disadvantage of IDF is that the dummy variables and the computed variables y1-y3 might not converge to the same value. That results in an infeasible outcome.
The IDF architecture is also called Optimizer-Based Decomposition.
Collaborative optimization can best be explained as IDF on two different levels, see figure 1.4. The advantage of using this MDO architecture is that the evaluation of the different disciplines is completely decoupled. Different software packages, different optimization routines and even different servers can be used for each disciplinary analysis. A distinction can be made between the system-level optimizer and the disciplinary optimizer.

1.2.3. THE OPTIMIZER

To find the best optimizer for a certain problem is a specialization in its own and again, this section is only an introduction. Factors that influence the choice of optimization function are the objective function, the type of constraints, the continuity of the design variables, the availability of gradient information, and the existence of local minima.
A distinction can be made between single-objective functions and multi-objective functions. An example of the first is to minimize the weight of a gear strut; an example of the second is to minimize both the weight and the cost of a LG. Single-objective functions are easier to solve, and therefore it is recommended to rephrase multi-objective problems into single-objective problems [11]. For the above example, the problem could be rephrased into minimizing the weight given a maximum cost price. A trend-line can be found by running the program for several maximum cost prices. Alternatively, an objective function can be created based on the combination of cost and weight. A weight reduction of one kilogram is then allowed to cost an X amount of euros. Other solutions are possible [23].
Several types of constraints can be identified, including linear constraints, bounds on the design variables and non-linear constraints. A combination of these constraints is also possible. Every optimization function is limited in the types of constraints it can evaluate. The MATLAB documentation is informative on this.
Design variables can be discrete or continuous. To give an example: one can be completely free to design the tyre, in that case the design space for the tyre variables is continuous. If a standard tyre must be chosen from a catalogue, the design space is discrete and another optimization algorithm needs to be chosen[3].
A fourth factor to consider while selecting an optimizer is the presence of local and global optima, as illustrated by figure 1.5:
Figure 1.5 shows the 3 local minima that are found when evaluating the function y = 2xcos(x) ∈ [-8,8]. This function can be visualised, but when the number of dimensions increases it is in general a challenge to find the global optimum. Optimizers that always find the global optimum exist, but usually need a long runtime. Other strategies to find the global optimum are to start a local optimization algorithm at multiple points spread over the design space.

[3] there are ways to work around this, but to explain that would be too much detail for this introduction. For more information, [24] is a recommended starting source.

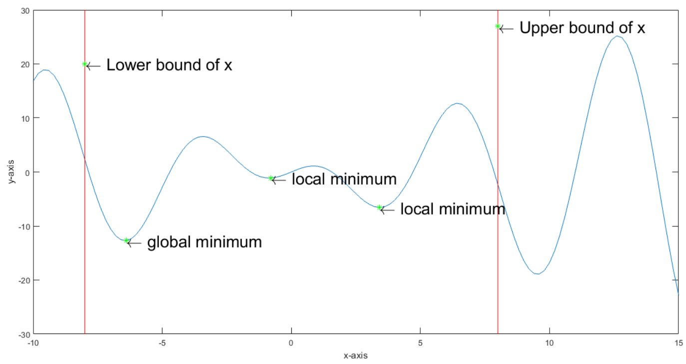


Figure 1.5: Existance of local and global minimum on a constrained interval.

The last decision factor while choosing an optimizer is the availability of gradient information. If gradient information is available or can be computed, it should be used. This will decrease the runtime [11]. An algorithm that does not use gradient information, for instance Genetic Algorithm (GA), will typically start at many points. For each point, a few design variables are changed and the objective function is evaluated. Based on the changes in objective value, new points are initiated. This is often compared to evolutionary changes, where pieces of DNA are swapped to make organisms perform better. If an optimizer can compute the gradient information, for example Sequential Quadratic Programming (SQP), the evaluation consists of changing the design variables one by one and computing the partial derivatives. The next point is chosen in the direction of the negative gradient. A local optimum has been found if the gradient in a point equals zero. In this thesis, the `fmincon` method of the MATLAB optimization toolbox is used. It is a gradient based, local optimizer that is advised for non-linear single objective functions and non-linear constraints. All design variables will be treated as being continuous.

1.3. Previous research on landing gear design and engineering optimization

With the basic principles of landing gear design and MDO covered, it is now time to look at previous research to merge the two fields. Three subsections will each describe a reference that incorporates the use of MDO in LG design process.

1.3.1. Landing gear integration in aircraft conceptual design

Chai and Mason detail the development of an MDO routine for the sizing of a landing gear [1]. This routine was implemented in an aircraft design tool under development at NASA [25]. The general layout of this routine is repeated in figure 1.6.

The program contains four modules that together find the optimal landing gear. The module CONFIG defines a baseline landing gear model that is used by the other modules. This module calculates internal variables such as the load on the tires and the brake energy based on input parameters such as aircraft weight, number of struts and number of tires, using physical equations [1]. From these calculations, standard parts such as tires, wheels and brakes are selected from a database.The lengths of the MLG components are also estimated by the CONFIG module.

The module LIMIT calculates requirement as the turnover angle, the ground clearance, the maximum allowable pitch and roll angles and the turn radius. Kinematic characteristics are also evaluated by this module: via what scheme is the gear folded and into which space?

The module PAVE determines all airfields the aircraft can operate from, by determining the flotation characteristics - the pressure imposed on the pavement by the tyres, and the behaviour of the runway as a result of that.

If the previous three modules have defined a configuration that fulfils all requirements, the module GEARWEI calculates the gear weight. It first estimates the weights of the different structural components, using their loads. It also estimates the weight of every non-structural component from a database.

One limitation of this program is that a first estimate is needed from the user. The program is not capable

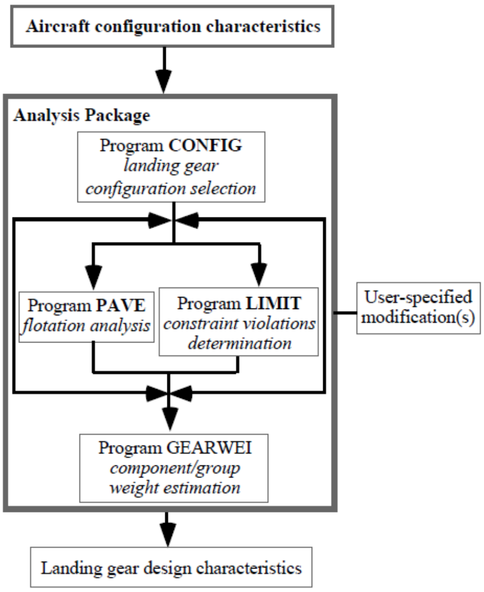


Figure 1.6: General layout of the research by Chai and Mason [1].

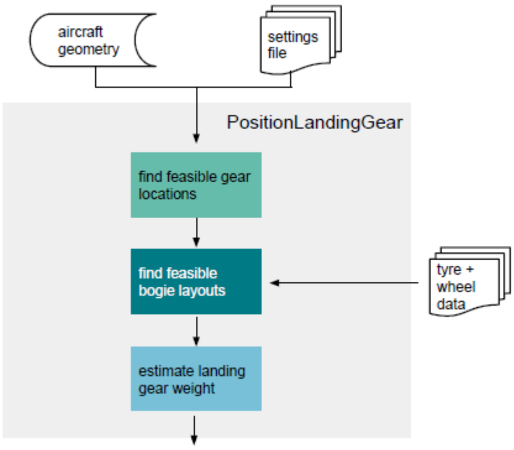


Figure 1.7: Landing gear design module by Heerens [2]

of deriving a feasible starting point. Another limitation is that it was only tested for a specific aircraft design, without considering its applicability to other aircraft concepts. The multidisciplinary architecture used is however a good starting point for this thesis.

1.3.2. LANDING GEAR DESIGN IN AN AUTOMATED DESIGN ENVIRONMENT

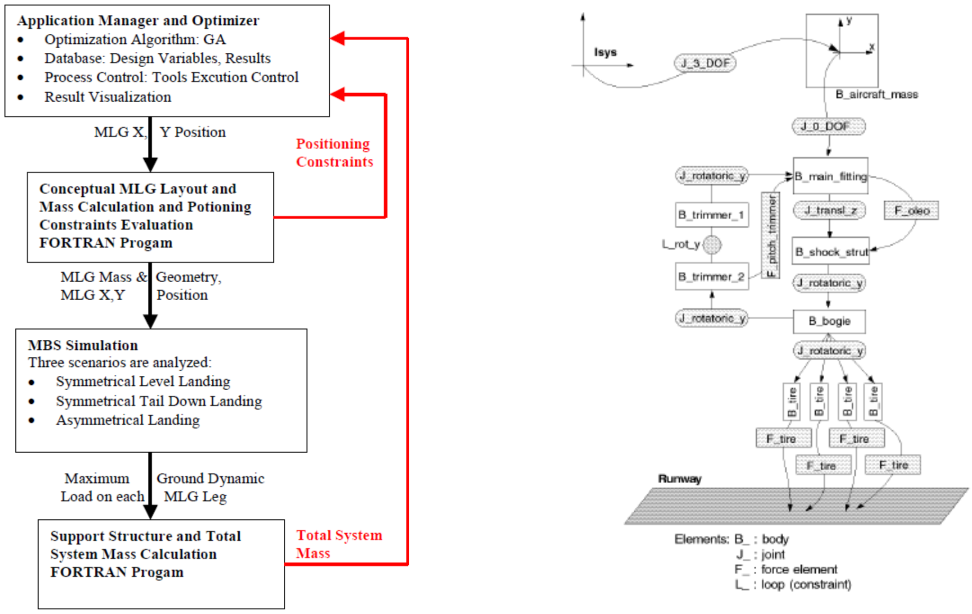
Other research is described by Heerens [2]. His work is also part of an aircraft design tool and fits into the Initiator that is being developed at Delft University of Technology [26]. The Initiator is a software program that can perform a conceptual sizing for existing and new aircraft configurations, such as a blended wing body and the Prandlt plane. It exists out of several modules that cooperate to find the optimal aircraft configuration, using multidisciplinary analysis techniques. Example modules are those for the component sizing, aerodynamic sizing and weight estimation. Heerens added a module to the Initiator that sizes the landing gear. An overview of this module is given in figure 1.7.

The module of Heerens has three routines, that are similar to those used by Chai and Mason [1]. The order workflow is however different. In this program, the limitations of the design space are investigated first. Then, the loads on the landing gear components are determined for several pre-defined bogie layouts. This includes the analysis of flotation requirements. Finally, the landing gear weight is computed using a similar method as the program by Chai and Mason [1].

A limitation to this program is that there is limited optimization: the process flow doesn't have feedback loops. It is more a search, based on existing configurations, for a landing gear layout that is acceptable. Another limitation is that, although this landing gear design module has been verified to give feasible results for different aircraft. This program is only applicable to CS-25 aircraft; test cases included the A380-800, B777-300ER, A320-200 and B737-200.

1.3.3. AN APPROACH FOR SIZING AND TOPOLOGY OPTIMIZATION INTEGRATING MULTIBODY SIMULATION

The research by Cumnuantip [3], [27] goes in more detail than the previous two methods. The aim of this work is namely to find the best landing gear and landing gear bay combination, see figure 1.8a. The case study for this work is that of a blended wing body, but the method is more generally applicable.



(a) General layout.

(b) Implementation in the multibody software

Figure 1.8: Research by Cumnuantip [3]

In the first two blocks of the program an initial position, length and shock strut characteristics of each MLG are determined, based on analytical equations and positioning constraints. This first set-up is the start point for the MultiBody Simulation (MBS) model in block three. The landing gear parts of this MBS model are assumed to be rigid; the tyres are modelled as linear springs. Dynamic loads are evaluated for three load cases. From these dynamic loads, the components are sized. The landing gear model and its degrees of freedom can be seen in figure 1.8b. In the last step, each component is sized based on the acting loads.
This process can be ran independent of the number of gears. It is concluded that 8 main landing gears struts yield the lowest total aircraft weight [27]. Other investigated options were 4, 6 and 12 MLG struts.
The results of this research look very promising. To look at the landing gear in this much detail however, detailed information from the aircraft such as a structural layout of the wing is needed. The framework developed in this thesis is aimed at finding the optimal landing gear for an aircraft in the preliminary design phase, when this information is not available. In that respect, the method presented by Cumnuantip is not a good starting point.
Another distinction between this research and the one by Chai and Mason, is that Cumnuantip uses a GA to find a solution and not a gradient-based method. The given argumentation is that the design space is discrete, and not continuous.

1.4. RESEARCH QUESTIONS AND PROJECT SCOPE

The above sections show the previous research into integration of MDO and landing gear design. The three studies all have clear limitations, however. The program by Chai and Mason is especially developed for a

specific aircraft and cannot be used for other aircraft. The program by Heerens is more generic than that of Chai and Mason, but it does not use true optimization methods. The work by Cumnuantip has more detail than the other studies and uses optimization methods, but it uses discrete variables and requires detailed information about the aircraft. That information is typically not available in the concept design phase.
The objective of this thesis is to redefine the landing gear design process and to make it suitable for engineering optimization techniques; without making any assumption on the aircraft type. As such, this thesis fits within a larger project at FLG. The ultimate goal of that project is to have a software tool that can help in the design a landing gear concept for any kind of aircraft between 3,000 and 50,000 kg, fixed-wing or rotary-wing. The time-frame for this phase of the design is similar to the current turn-around time of a concept design: six weeks. The primary goal is thus not to shorten this design phase, but to do a better job in the given time. A trade-off should be performed between different landing gears, that are all evaluated in more detail and with more consistency than what is currently done for a single gear solution.
Work was already started by generalizing specific software, such that is can be used for other projects. That work follows a bottom-up approach: create all the blocks that are needed and consider their integration afterwards. This research takes a different approach and looks at the landing gear design process from a top-level perspective to define the needed framework.
A proof of concept will be created to prove that these principles have practical purpose. The following research questions were defined:

- Which inputs are needed for the landing gear design process?
    1. Which regulations need to be followed?
    2. Which customer requirements are needed?
    3. Are there other requirements that can drive the design?
    4. What are typical landing gear performance requirements?
- What does the current landing gear design process look like?
    1. Which disciplines are involved?
    2. What are the standard landing gear architectures?
    3. Which software programs are used during the design process?
- How can the landing gear design process benefit from engineering optimization principles?
    1. Which optimization toolset is best suitable for the problem of landing gear design?
    2. What is the most efficient order to evaluate the various disciplines?
    3. How can all these disciplines and tools be integrated in a single software tool?
    4. What is the minimum set of variables to derive a feasible landing gear layout?
    5. What is the output from this process?

1.5. THESIS STRUCTURE

This thesis has 11 chapters. Chapter 2 gives a top-level description of LIT. The rest of this thesis is dedicated to the development of a proof of concept for LIT. It follows a similar approach as described in chapter 2, but it is less general and aimed at a single gear: the POC gear. The specifics of that landing gear are the subject of chapter 5.
Before that, chapters 3 and 4 describe the system-level positioning constraints for a tricycle landing gear and the most important load cases. Chapters 6 describes how the components are sized, based on these ground reaction loads.
The resulting POC framework is the subject of chapter 7; the results that were obtained with it follow in chapter 9. A verification of these results is given in chapter 8.
Finally, the conclusions and recommendations are given in two separate chapters: 10 and 11.

2

"Landing gear in thirty days"

This chapter explains the final outline for the Landing gear In Thirty days (LIT) tool, see also figure 2.1. The final objective for this software tool is to help in the conceptual design phase of a landing gear. This tool should not be of the "push a button and get your answer" type, but a tool-set that assists the LG designer in the evaluation of different configurations. Both fixed-wing and rotary-wing aircraft are included. Not only the landing gear should be optimized, also the interaction with the aircraft structure is of special importance. The best LG solution takes this integration into account. Excluded from the scope are gears of large passenger aircraft that carry more than 100 passengers; these are not in the market niche of FLG.
For the purpose of conceptual design, the preliminary sizing of all load-carrying components is included, as well as the preliminary sizing of the shock absorber, the evaluation of kinematic characteristics, the determination of flotation properties and the preliminary design of the rolling stock. Excluded are hydraulics, electrical harnesses, seals, sensors and other non-loaded components, although space reservations must be made for these parts.
The final outcome of LIT is a breakdown of weight and cost for every landing gear component, as well as a basic CAD model of the gear struts. Per gear component, there is additional information in a text file such as cross-sectional information, the critical load case and the results of a Finite Element Method (FEM) calculation on that part.
LIT can be explained by making an analogy with an onion. Layer by layer, the design is analysed in more detail. This is sketched in figure 2.1. Each step is explained in more detail below.
Note that the description in this chapter does not mention any specific software programs. For each dedicated task, multiple software packages are suitable. As optimization routine, the use of for example iSight, MATLAB, and Python can be considered. For multi-body calculations, there is the choice between for example SimMechanics, VLM and ADAMS. The construction of 3D drawings can be done with CATIA, but Pro-Engineer and solidworks are alternatives. Also for FEM calculations, there are multiple options, such as Abaqus or NASTRAN. To not exclude any options in advance, it will just be stated that, for example, a multi-body package is needed.

2.1. Block A: basic parameters

The first top-level parameters are the overall gear layout (either tricycle or taildragger), and the topology of the gear struts (trailing arm, telescopic, sideways levered, bogie, other). The objective of the first block is to investigate all different possibilities and to determine the most promising combinations. This computation is possible with around 20 design variables and is rather quick: below 5 minutes per possibility. Block A contains three different routines. These are described below.

The first routine in block A, *Find load per gear strut*, calculates the ground reaction forces on each strut (either main, nose or tail), based on its position in the global reference frame and the gear layout chosen. The considered load cases are static and come from the CS requirements; the user can select whether CS-23, 25, 27, 29, or a combination of the four is applicable. On a project-by-project basis, additional customer load cases are added.
The required inputs for this routine are specified in table 2.1. They are treated as constants and must be

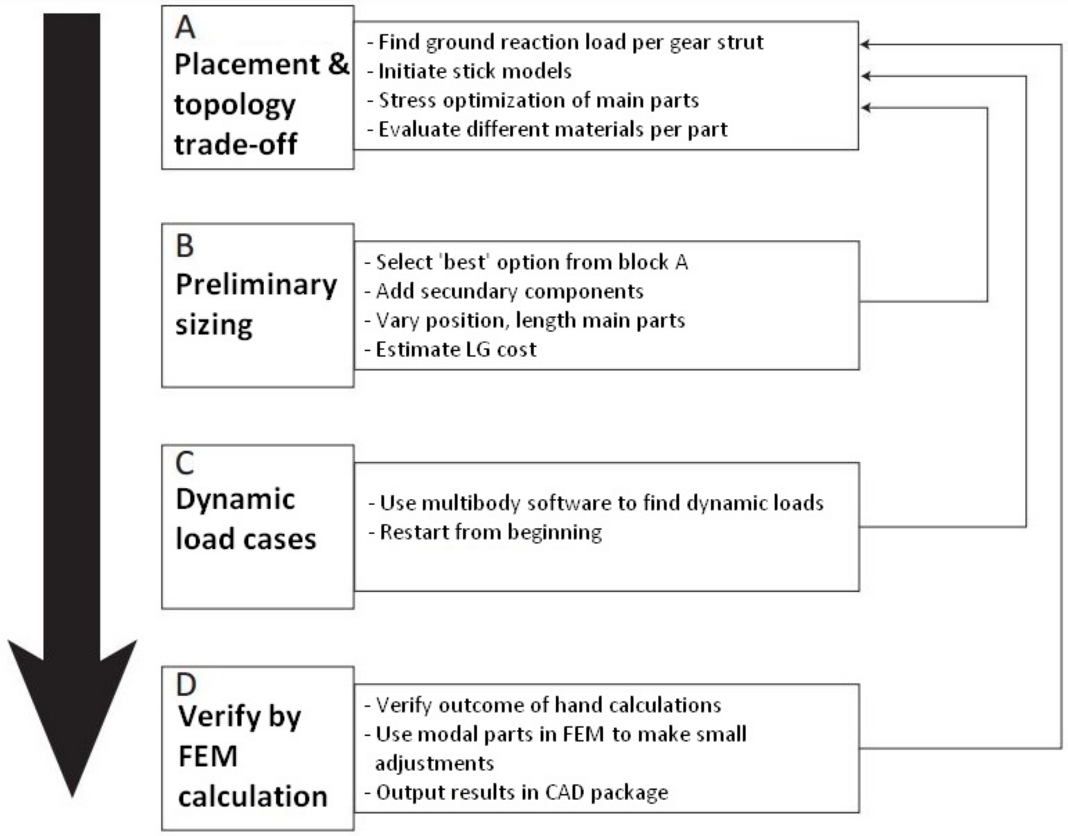


Figure 2.1: Proposed set-up of the landing gear in thirty days tool.

obtained from the aircraft manufacturer.
The output of this routine is a list of all load cases, with corresponding forces on the tyres and at the CG, the location of the tyres and the gear layout considered.
There are 12 positioning constraints. These are treated in a separate chapter: 3

Table 2.1: Inputs for routine 1 in block A of LIT.

Parameter	Unit	Parameter	Unit
Aircraft weight(s)	N	Turnover angle[4]	deg
CG location fwd (x,y,z)	m	wing span	m
CG location aft (x,y,z)	m	tail height	m
Regulatory set	-	max pitch angle	deg
Max approach speed	m/s	steering angle nose wheel	deg
Max roll angle @ TD	deg	wing sweep angle	deg
Optional; to be added after the first iteration:			
distance tyres secondary gear	m	radius sec tyres	m
distance between main gear tyres	m	radius MG tyres	m

The second routine in block A is *Initiate stick models*. The objective of this routine is to get feasible initial designs, for all different combinations of gear layout and topology. Not every combination will be possible for any aircraft, however. A sideways levered gear is normally only seen on helicopters and fighters - and there are good reasons for that. The user of LIT should eliminate infeasible combinations in advance.
The initial design is based on the preliminary sizing of the shock absorber, and a flotation analysis. Based on these calculations, the preliminary size of the other components can be estimated. There are two shock absorber types: a single acting shock strut and a double acting strut. Both options must be considered. The

[4] see explanation in section 3.2

Table 2.2: Inputs for routine 2 in block A of LIT.

Parameter	unit	Parameter	Unit
Efficiency SA	%	Efficiency tyre	%
Load factor at TD	-	vertical TD velocity	m/s
Static SA pressure	Pa	ratio static over fully extended pressure	-
ratio fully compressed over static pressure	-	Tyre stroke	m

flotation analysis determines the number of wheel and their separation.
Inputs of this routine are listed in table 2.2; these are again treated as constants. In some cases, tyre information is provided by the Original Equipment Manufacturer (OEM), in other cases is not. Thus, it must be possible to design the tyre characteristics, as well as to use a pre-determined (and fixed) set.

The output of this step is a stick diagram of the initial gear design, for every combination of gear topology, gear layout and Shock Absorber (SA) type. The total number of possibilities is substantial and does not stop with the distinction between telescopic gear, trailing arm gear, sideways levered gear and bogie gear. If only the trailing arm gear is considered, several sub-types can be identified that all fall under the category trailing arm gear. For example; a gear that retracts forward is different than a gear that retracts aft or to the side. And even then, retraction is possible by folding a member, or by sliding it in like an actuator. This has been illustrated in figure 2.2. Not every branch is fully worked out, however. The reader of this report should be able to fill in the blanks for him/herself.
Constraints to this routine are that the minimum pressure in the shock absorber cannot be too low and that the maximum pressure cannot be too high. Often quoted values for these pressures are 60 and 6000 psi respectively [4]. Furthermore, folding of the gear must be possible within the LG bay and without the components touching each other.

The third routine, *preliminary component sizing*, calculates the cross-sectional parameters for the main LG components. This calculation is based on the loads determined in the first routine, the landing gear layouts determined in the second routine, and simple stress calculations. These calculations are performed for every load case, as it is not known in advance which load case is critical. Typically, there will not be one critical load case for the whole gear strut, but a different one for every component. For that reason, it is highly relevant output for the designer. Every part is modelled as a cylinder or as an I-beam and has a constant cross section. Only the main load-carrying components are evaluated; lugs, pins, bushes and other smaller parts are excluded for this step.
The input for this routine is created in the previous two routines; outputs include a list of component and system weights for all different gear combinations considered, and the optimal gear locations. Additional outputs are the margin of safety[5] and the critical load case for each part.
One constraint is again the kinematic properties of the gear: it must fit inside the LG bay. A second constraint is related to the forces introduced in the airframe structure. They are limited by the strength of this structure. After all, minimizing the weight of the landing gear plus the weight of the surrounding aircraft structure is the final objective. The last, obvious, constraint is that the parts are not allowed to fail due to the forces acting on them.

2.2. BLOCK B: PRELIMINARY SIZING

In the second block, the more promising design solutions from block A are further analysed. More detail is added by allowing the position of the main components to vary, by adding secondary components such as lugs, pins and bushes, and by evaluating the cost of the LG system. Before continuing, there is one manual step. The designer must evaluate the different results of block A and determine which concepts are worth pursuing in more detail. In the first block, a typical evaluation takes between one and five minutes. In the second block, one evaluation may take anywhere between an hour and several hours. This step should thus be done for a handful of concepts only, not for dozens of them.

Block B follows a similar method as block A, but it is more elaborate and shifts from a single-objective

[5] the concept of margin of safety will be explained in chapter 6

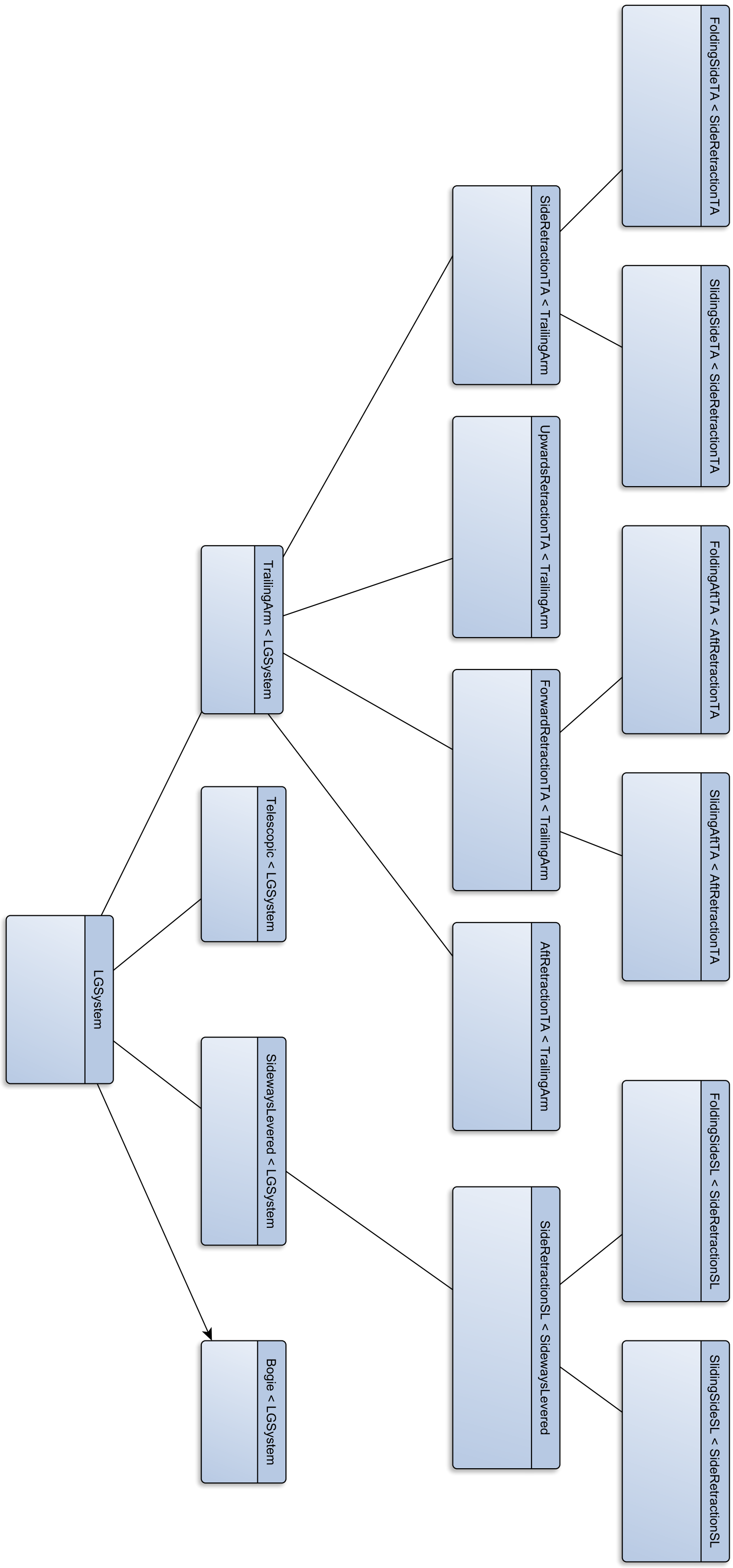


Figure 2.2: An overview of different standard landing gear configurations.

optimization routine to a multiple-objective optimization: the predicted cost of the landing gear is also computed in this block. These cost can be estimated from the knowledge of the manufacturing process. The number of design variables increases; besides the contact points of the gear struts with the ground, the positions of the so-called hardpoints[6] can be varied and the secondary components must be sized. To allow this analysis, a parametric multi-body package is used that derives the loads at the specified hardpoints.
The inputs for the multi-body models are the loads per landing gear strut and the hardpoint locations. For every load case, the loads in the hardpoints are calculated. From these loads; the strut weight is calculated, using the same routines as in block A.
The principle output of this routine, for every concept, is a stick diagram in the multi-body package and a text file with cross-sectional parameters of the different components. This is not very intuitive to interpret for the designer and should thus be converted into a CAD drawing. The designer operating LIT must inspect these CAD drawings to evaluate if a physically achievable result is obtained. Additional outputs are information about the component sizing: the margin of safety and the critical load case.
Besides the constraints of block A, there are additional constraints in this block. Parts cannot overlap in the physical space, all thicknesses must be positive, pins cannot be larger than their corresponding lugs, etc. These constraints are rather trivial, but they must be specified as the optimization routine is purely mathematical and does not know them.

2.3. BLOCK C: DYNAMIC LOAD CASES

In block C, the dynamic capabilities of the multi-body program are used to analyse the gear in more detail than in block B. At this point, the number of possible gear concepts should be further reduced to a maximum of 4-5 per gear group (main and secondary) as the runtime now becomes a major factor to consider. Using multiple processors this step can still be performed overnight, but the design freedom must be restricted at this point.
Some load cases have been neglected so far due to a lack of information. These are computed in this block. Included are the wheel spin-up load case, the drop test, and the spring back load case. Furthermore, there are several load cases that were approximated with a static method in block A, but which are in fact dynamic. Examples are turning while taxiing on the ground, the sudden application of the brakes by the pilot, and all touchdown load cases. These must be studied in more detail as well. Dynamic load cases defined by the customer should also be evaluated at this point.

To allow this dynamic analysis, the rigid multi-body model must be expanded with a shock absorber, tyre models, and for heaver aircraft also with pavement models. When the load peaks of these dynamic load cases are found, the tool can be restarted with this additional set of loads. These load peaks of the dynamic analysis are then added as static load cases. To speed up the process, static load cases that were not critical can be deleted from the old set.
The number of design variables is not changed when proceeding from block B to block C. The system-level design variables are still the connection points of the LG parts and the position of the tyres, while the component-level design variables remain those that define the cross sections of the different parts. The constraints remain the same as well.
The output of block C is similar to the output of block B, only updated. There is again a stick diagram in the multi-body package, and a collection of text files with information about the cross sections of the components and their critical load cases. This is then used to create a 3D product in a CAD package.

2.4. BLOCK D: VERIFY RESULTS BY FEM CALCULATION

The final step in block D is to verify the simple computations by means of a (static) FEM calculation. This is not a true optimization; to have a FEM analysis incorporated in an optimization routine would simply take too long. To allow this step, modal parts must be used for the LG components, instead of the strongly simplified I-beams and cylinders. This allows for a more detailed analysis of the LG components. A modal part is a flexible 3D CAD part, on which linearised stress calculations can be performed via FEM software. They take local bending, deformation and stress peaks into account. During the optimization, the modal parts are modified until they do not fail due to the acting loads. As the modal parts are defined in the CAD environment, they can be implemented in the multi-body model. They can also be used to visualize the results. That

[6] A point where the two LG parts connect with each other

is another advantage of using modal parts.
In block D, the focus shifts from finding the best solution to verifying that the found solution is indeed feasible. There is no additional output, and the constraints are also similar to the previous blocks. The output of this step is again a 3D CAD product, such that pictures of the design can be shared with the design team and the customer. A detailed weight breakdown per component must also be available, as well as dimensions of all parts and the critical load cases per component.

3

POSITIONING CONSTRAINTS TRICYCLE LANDING GEAR

The previous chapter explained the overall set-up of LIT. The first step in the development of the Proof of Concept (POC) tool, is to identify the allowed gear location. That is done in this chapter. The constraints of this chapter are applicable to all aircraft with a tricycle gear. Specific constraints for the POC tool are postponed until section 7.4.
Torenbeek states the different geometrical limitations for tricycle landing gears [28]. This section describes their implementation. In total, 12 positioning constraints are identified; divided over four topics. Each topic has its own dedicated section.
This chapter needs the lengths between the MLG struts and the CG, the NLG strut and the CG, and the MLG and the NLG. The used variables are introduced in figure 3.1.

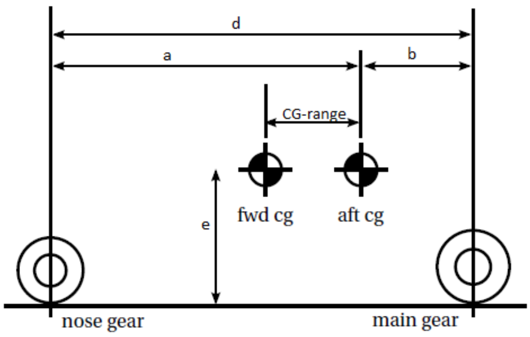


Figure 3.1: Definition of A, B and cg-range [2].

3.1. NO PARTS OF THE AIRCRAFT TOUCH THE GROUND, EXCEPT THE LANDING GEAR

During landing and take-off, the tail, wingtip, or engines are not allowed to hit the ground. There are 3 constraints that prevent this: the tip clearance constraint, the nacelle clearance constraint and the tail clearance constraint.

TIP CLEARANCE CONSTRAINT

The wing tip of an aircraft may never touch the ground. There are two relevant performance parameters for this condition: the maximum pitch angle θ and the maximum roll angle ϕ that will both be provided by the OEM. When the aircraft is both pitching and rolling, for example during a landing with cross-winds at high angle of attack, there is a risk that the wing tips touch the ground. With equation 3.1, the roll angle at which the wing tip touches the ground can be calculated [28, p. 350]:

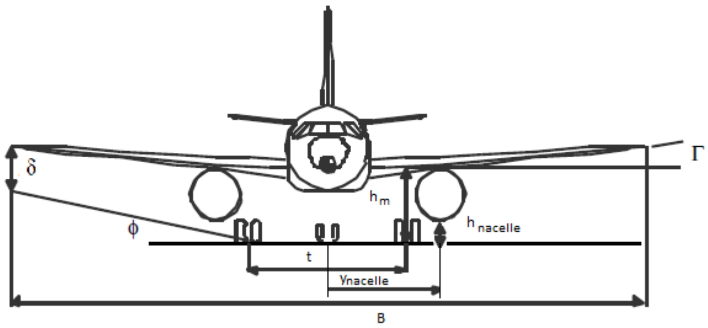


Figure 3.2: Geometrical constraint relating to wing tip clearance [1]

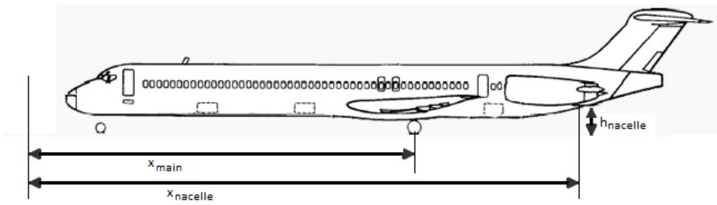


Figure 3.3: Geometrical constraint relating to engine clearance [29]

$$\tan\phi = \tan\Gamma + \frac{2h_m}{B-t} - \tan\theta\tan\Lambda + \partial \tag{3.1}$$

where Γ is the dihedral angle of the wing, h_m is the main gear height, B is the wingspan, t is the gear track, Λ is the wing sweep angle and δ is the tip clearance. See also figure 3.2.
It is noted that some references on landing gear design (e.g. [1] and [2]) try to calculate the pitch angle required for take-off, based on preliminary aerodynamic parameters. Then, they assume that the pitch angle during touchdown equals the pitch angle at take-off. In this thesis, the maximum allowable pitch angle is assumed to be an input, provided by the aircraft manufacturer.

NACELLE CLEARANCE CONSTRAINT

The second constraint is that the engine is not allowed to touch the ground during take-off or landing. It is not relevant whether the engine is a propeller, a jet engine, or an open rotor. The minimum height above the ground and its location are important, not the engine type.
Still, two cases can be distinguished: an engine that is positioned below the wing or an engine that is positioned on the tail, as is seen on the Fokker 100[7].
When the engines are positioned under the wing, the roll angle at which the nacelle touches the ground is given by equation 3.2. The pitch angle is assumed to be 0.

$$\tan\phi = \frac{h_{nacelle}}{y_{nacelle} - \frac{t}{2}} \tag{3.2}$$

where $h_{nacelle}$ is the lowest nacelle height and $y_{nacelle}$ is the y position where the nacelle is closest to the ground. Note that a pitch angle of 0 degrees actually represents the worst-case scenario as the engines will move upwards when the aircraft is under a positive pitch angle.
If the engine is positioned on the tail, figure 3.3 should be considered. For this scenario, the maximum pitch angle can be expressed as equation 3.3:

$$\tan\theta = \frac{h_{nacelle}}{x_{nacelle} - x_{main}} \tag{3.3}$$

[7]Yet other positions are imaginable, for example the Honda HA-420 business aircraft that has the engines on top of the wing. In that case, this constraint will be inactive.

with $x_{nacelle}$ the x location where the nacelle height is minimal and x_{main} the x distance between the MLG and the origin. The effect of a roll angle is not considered as a survey of aircraft with a tail engine shows that they all have a main gear track wider than the engine track. This suggests that the roll angle has no effect on when the engine touches the ground.
This constraint is not relevant for helicopters.

TAIL CLEARANCE CONSTRAINT

A third constraint is that the tail cannot touch the ground. This can be checked with a method similar to the engine-on-tail constraint. The difficulty here is to determine which point of the tail is the most likely candidate to touch the ground.
It is assumed that this point is fixed by the OEM. It is called the location of the tail bumper, with $x_{tailbumper}$ and $h_{tailbumper}$ as its x coordinate and height above the ground respectively. This point will be treated as a constant value. The constraint for this criterion is given by equation 3.4

$$\theta_{\max} - \tan^{-1}\left(\frac{h_{tailbumper}}{x_{tailbumper} - x_{main}}\right) \leq 0 \tag{3.4}$$

3.2. ALLOWED LOCATIONS FOR THE CENTRE OF GRAVITY

As indicated in figure 3.4, a triangle can be drawn that connects the main and nose gear struts. It should be prevented that the CG moves outside that triangle during ground operations. In total, 4 constraints need to be checked in this section: the ground stability constraint, two sideways turnover constraints, and the touchdown constraint.

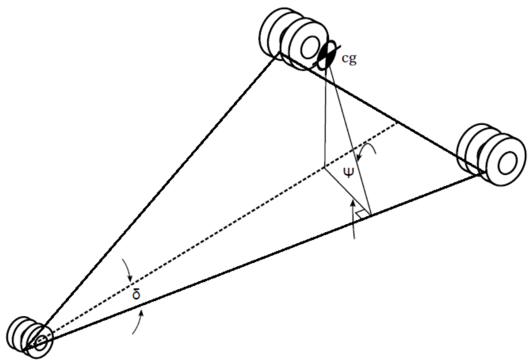


Figure 3.4: Visualization of the sideways turnover angle ψ and the triangle spanned by the NG and MLG

TOUCHDOWN CONSTRAINT

The landing gear must be positioned behind the Centre of Gravity (CG), or the aircraft would rotate and fall on its tail. The worst-case scenario for this constraint is a landing at the highest and most aft CG position. The distance between the MLG and the aft position of the main landing gear is given by equation 3.5 [28, p. 352].

$$b \geq (h_m + e_s)\tan\theta \tag{3.5}$$

Where b is indicated in figure 3.1 and e_s is the total static wheel travel[8].

GROUND STABILITY CONSTRAINT

The so-called sideways turnover angle ψ prevents the CG from falling outside the triangle during a landing on one wheel. It is visualised in figure 3.4 and expressed in equations 3.6 and 3.7 [28, p. 354]

$$\tan\psi = \frac{e}{a\sin\delta} \tag{3.6}$$

with δ:

[8]The deflection of the tyre plus shock absorber.

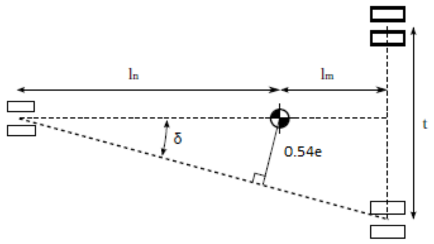


Figure 3.5: Visualization of the sideways turnover constraint [28]

$$\tan\delta = \frac{t}{2(e+a)} \tag{3.7}$$

with e the height of the CG above the ground and a and b as in figure 3.1. The typical maximum value for the turnover angle varies between aircraft types [4].

SIDEWAYS TURNOVER CONSTRAINT 1

The allowed position of the main gears is dependent on the *forward* CG location and the position of the nose gear. A positioning recommendation is to draw a circle with a radius of 0.54 times the CG height around the forward CG in the top view of the aircraft[28]. That circle should fit within the triangle defined by the gear struts, see figure 3.5 and equation 3.8. The factor 0.54 is based on dynamic instability considerations and is statistically supported [28].

$$\frac{1}{2}t_{MIN} = \tan(\sin^{-1}(\frac{0.54e}{a-cgrange}))(a+b) \tag{3.8}$$

SIDEWAYS TURNOVER CONSTRAINT 2

The second sideways turnover constraint limits the most aft position of the nose gear, based on the *aft* CG position and the location of the main gears. Again, a circle of 0.54 times the height of the CG should be drawn around the location of the CG. As before, this circle should be completely inside the triangle defined by the landing gear struts. Mathematically, this can be expressed in equation 3.9 [2].

$$a-cgrange = \tan(\sin^{-1}(\frac{0.54e}{(b+cgrange)\sin\Delta})+\Delta)\frac{t}{2}-(b+cgrange) \tag{3.9}$$

In this equation, Δ is given by $atan(2b/t)$

3.3. LOAD ON THE NOSE GEAR

If the load on the NLG is too low, the steerability of the aircraft will become problematic. If that load is too high, there will not be enough load on the main gears to have efficient braking.

MINIMUM NOSE GEAR LOAD CONSTRAINT

During ground operations, the aircraft is usually steered by turning the nose wheels. Therefore, a minimum load needs to be present on these wheels. This minimum load is defined by the OEM; a conservative value is 6 percent of the aircraft weight [4]. The load on the nose gear is given by:

$$F_N = \frac{b}{d}W \tag{3.10}$$

MAXIMUM NOSE GEAR LOAD CONSTRAINT

There is also a maximum to the static load that can be applied on the nose gear. Usually the NLG wheels are not equipped with brakes. As the load on the nose gear increases, the load on the main gear decreases and thus braking efficiency is affected. The maximum static load is found with the CG is the most forward position, see equation 3.11

$$F_N = \frac{b+cgrange}{d}W \tag{3.11}$$

Table 3.1: Aircraft design group classification [30]

	Group I	Group II	Group III	Group IV	Group V	Group VI
Tail height [m]	<6	6 - 9	9 - 13.5	13.5 - 18.5	18.5 - 20	20 - 24.5
Wingspan [m]	<15	15 - 24	24 - 36	36 - 52	52 - 65	65 - 80

Table 3.2: Aircraft approach category classification [30]

	A	B	C	D	E
Approach speed [m/s]	<46.8	46.8 - 62.2	62.2 - 72.5	72.5 - 85.4	> 85.4

A conservative value for the maximum NLG load is 20 percent of the aircraft weight.
Some references, such as [4] and [2] also prescribe a maximum load for the MLG. The maximum load that can be applied to the main gear strut is however half the aircraft weight. More demanding load cases will be found in chapter 4, so this constraint is omitted.

3.4. Operation on the airfield

A set of rules for ground operations is described in AC150/5300-13a [30] for fixed-wing, conventional take-off and landing aircraft. These rules specify constraints related to the required runway width, manoeuvring on the taxiway and the castor angle.
Reference [30] subdivides aircraft in Aircraft Design Group (ADG)s based on their physical dimensions and in Aircraft Approach Category (AAC)s based on their approach speed. The subdivision in these two categories is given in tables 3.1 and 3.2, where the original values were converted to the metric system.
For the ADG classification, the highest applicable classification should be taken. An aircraft with a tail height of 14 meter and wing span of 35 meter, would thus be ADG IV.

Turning on runway constraint

The minimum required runway width for an aircraft, $r_{180\,\mathrm{deg}\,turn}$, can be found from the AAC and ADG classification. A 180°turn must be possible on this runway, see equation 3.12 and figure 3.6

$$r_{180\,\mathrm{deg}\,turn} = d\tan(90-\beta) + \frac{t}{2} \tag{3.12}$$

β is the nose steering angle and is typically limited to ± 60°[4].

Manouvering on the taxiway

Manoeuvres on taxiways impose another constraint, see figure 3.7. The aircraft must be able to ride on the taxiway while its wheels stay on the centreline of the track. The radius of this centreline, the inner radius of the taxiway and the safety margin between the outer wheels and the edge of the taxiway are specified for every aircraft type in table 3.3.

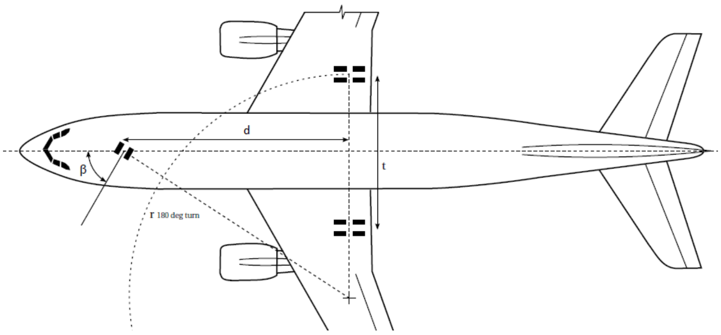


Figure 3.6: Visualization of the turning on runway requirement [2]

Table 3.3: Operation on taxiway [30]

	Group I	Group II	Group III	Group IV	Group V	Group VI
$r_{centerline}$ [ft] /[m]	75 / 23	75 / 23	100 / 30	150 / 46	150 / 46	170 / 52
r_{fillet} [ft] / [m]	60 / 18	55 / 17	55 / 17	85 / 26	85 / 26	85 / 26
Safety margin S[ft] / [m]	2.5 / 0.76	2.5 / 0.76	13 / 4.0	20 / 6.1	20 / 6.1	25 / 7.6

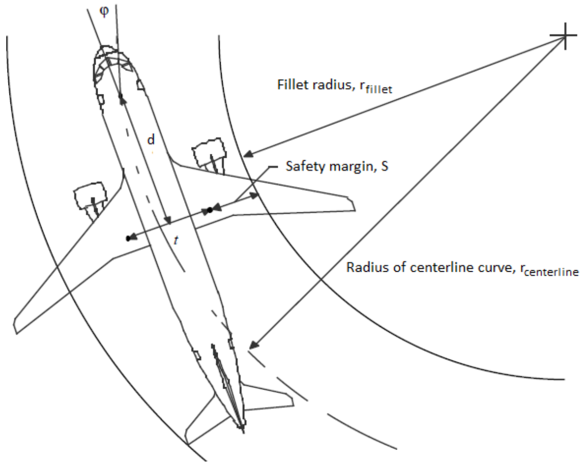


Figure 3.7: Visualization of the turning on taxiway requirement [1]

The limit to the combination of track width and wheel base is given in equation 3.13:

$$r_{fillet} = -\sqrt{r^2{}_{centerline} - d^2} + \frac{t}{2} + S \tag{3.13}$$

CASTOR ANGLE CONSTRAINT

The so-called castor angle is defined as the angle between the nosewheel orientation and the tangent of the centerline curvature [30]. It should be smaller than the nose wheel steering angle β, leading to equation 3.14:

$$\sin^{-1}\left(\frac{d}{r_{centerline}}\right) - \beta \leq 0 \tag{3.14}$$

4

Ground reaction loads on the wheels

The first step of LIT is to determine the loads per gear strut. These loads come from two sources: the aircraft regulations - described in FAR/CS25[9] [31] and the OEM. Only the CS-25 load cases are considered in this thesis.

There are different regulations for the various aircraft categories. The most important ones for the scope of LIT are CS-23 [32], CS-25 [31], CS-27 [33] and CS-29 [34]. A comparison between these regulating bodies has been made. It was concluded that CS-25 has the most elaborate rules for the landing gear. Therefore, only those rules are described in this thesis. The differences between CS-25 and the other regulations can be found in appendix A. Note that the rules in this chapter are only relevant for tricycle gear layouts.

In section 4.2, it is assumed that the load is evenly divided between both MLG struts. There are more cases to be considered however; see section 4.3. Before the load cases can be described however,a reference frame is needed: the global (or aircraft) reference frame. This is introduced in section 4.1.

4.1. Reference frame used

The reference frame of this thesis is visualized in figure 4.1. The origin of this frame is placed below and in front of the nose. This results in positive x and z coordinates. The y-axis is positive through the right wing, to ensure a right-handed axis orientation.

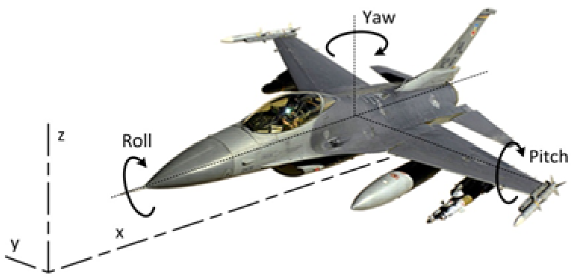


Figure 4.1: Aircraft global reference frame as used throughout this thesis

4.2. Static loads

The static load cases can be divided into two scenarios: ground operation loads and touchdown loads. In total, 41 unique load cases are defined. It is generally unknown which load case will be critical, so all must be investigated. Only those load cases that turn out to be critical in chapter 8 have been detailed in the current chapter, however. The other load cases are described in appendix B.

All static load cases are in fact simplifications of a dynamic event. This section is therefore in agreement with CS-25. In the design process, these dynamic event are all considered. That is however only done in the detailed design phase, not in the preliminary design phase.

[9] For loads on the LG, the CS regulations from America are equivalent to the European FAR regulations

4.2.1. BRAKING LOADS

Section 25.493 of the regulations describes 3 different braking scenarios:

1. braking without any load on the nose wheel (2-point braking);
2. braking with load on the nose wheel and zero pitching acceleration (3-point braking);
3. a sudden braking motion with dynamic pitching behaviour as a result (dynamic braking).

Points 1 and 2 must be evaluated at the MRW with a load factor of 1 and at the Maximum Landing Weight (MLW) with a load factor of 1.2. The third point only needs evaluation at the Maximum Take-Off Weight (MTOW).
When there is no load on the NG, all weight is carried by the MLG struts, as in equation 4.1

$$V_M = \frac{nW}{2} \tag{4.1}$$

Where V_M is the vertical force on the main gears, n the load factor and W the aircraft weight. The maximum drag force per strut must be taken as 0.8 times the vertical load per strut; the side load as 0.
The force on the main gear can be found by taking moments around the attachment point of the nose gear normal force, see equation 4.2:

$$V_M = \frac{nWa}{2(d+0.8e)} \tag{4.2}$$

With a the distance between the nose gear and the CG, d the distance between the MLG and the NLG, e the height of the CG and the other symbols as before.
The load on the nose gear strut V_N can be calculated by taking moments around the CG and substituting equation 4.2. The result is equation 4.3:

$$V_N = \frac{nW(b+0.8e)}{(d+0.8e)} \tag{4.3}$$

where b is the distance between the MLG and the CG. There is an additional drag load on the main gear of 0.8 times the vertical load. The side load is 0.
For the dynamic braking case, an equation is provided by the regulations, see equation 4.4:

$$V_N = \frac{W}{a+b}[b+\frac{f \cdot \mu \cdot a \cdot e}{a+b+\mu \cdot e}] \tag{4.4}$$

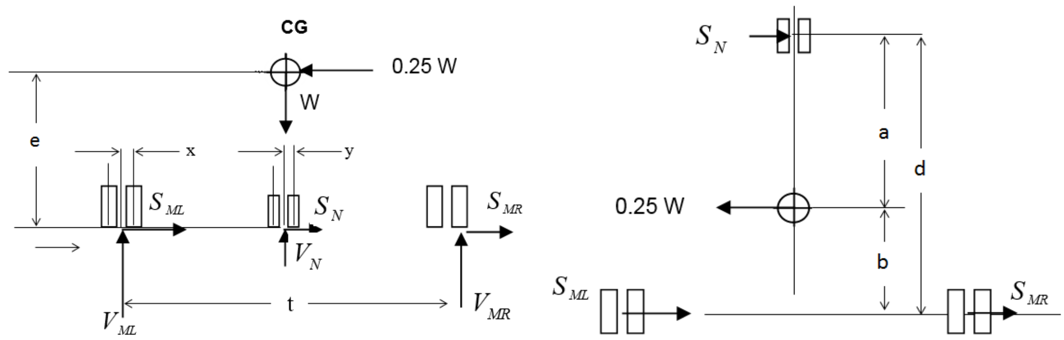
In this equation, f is the dynamic response factor that is 2.0 unless a lower factor can be proven and μ is the friction coefficient which is 0.8.
The load that is not carried by the nose gear is divided over the main gear struts. These main gear struts experience an additional drag load of 0.8 times the vertical load. The side load is 0, for both the MLG and the NG.

4.2.2. TURNING LOADS

There are 6 turning scenarios defined in CS25.495 and CS25.511:

- Turning left with all tyres inflated
- Turning with one of the nose wheels deflated
- Turning left with the inner tyre of the right strut deflated
- Turning left with the outer tyre of the right strut deflated
- Turning left with the inner tyre of the left strut deflated
- Turning left with the outer tyre of the left strut deflated

Another 6 cases can be thought of, symmetric to the cases described above. Figures 4.2a and 4.2b give a free body diagram for the case where the inner tyre of the right strut is deflated.
Without any deflated tyres, equations 4.5 to 4.7 can be derived for the vertical load on the gear struts by evaluating the sum of moments on the nose gear and at the CG:



(a) Back view (b) Top view

Figure 4.2: Free body diagram for turning conditions

$$V_{MR} = \frac{W}{2}(-\frac{e}{t} + \frac{a}{d}) \tag{4.5}$$

$$V_{ML} = \frac{W}{2}(\frac{e}{t} + \frac{a}{d}) \tag{4.6}$$

$$V_N = \frac{Wb}{d} \tag{4.7}$$

Where V_{MR} and V_{ML} are the load on the right and left main strut, and t is the main gear track width. There is a side load of half the vertical load on all three struts; the drag load is 0.

The equations for the other 5 load cases are given in table 4.1. In these equations, u is the distance between the main gear wheels and v is the distance between the nose gear wheels.

Table 4.1: Loads on left and right main gear strut for different turning load cases.

Scenario	Left strut	Right strut
Flat nose wheel	$\left(\frac{W}{t}\right)\left(\frac{t}{2d}a + 0.25e + v\frac{b}{d}\right)$	$\left(\frac{W}{t}\right)\left(\frac{t}{2d}a - 0.25e - v\frac{b}{d}\right)$
Flat inner right wheel	$\left(\frac{W}{t+u}\right)\left(\frac{t}{2d}a + 0.25e\right)$	$\left(\frac{W}{t+u}\right)\left(\left(\frac{t}{2d} + \frac{u}{d}\right)a - 0.25e\right)$
Flat outer right wheel	$\left(\frac{W}{t-u}\right)\left(\left(\frac{t}{2d} - \frac{u}{d}\right)a + 0.25e\right)$	$\left(\frac{W}{t-u}\right)\left(\frac{t}{2d}a - 0.25e\right)$
Flat inner left tyre	$\left(\frac{W}{t+u}\right)\left(\left(\frac{t}{2d} + \frac{u}{d}\right)a + 0.25e\right)$	$\left(\frac{W}{t+u}\right)\left(\frac{t}{2d}a - 0.25e\right)$
Flat outer left tyre	$\left(\frac{W}{t-u}\right)\left(\frac{t}{2d}a + 0.25e\right)$	$\left(\frac{W}{t-u}\right)\left(\left(\frac{t}{2d} - \frac{u}{d}\right)a - 0.25e\right)$

4.2.3. LANDING WITH SIDE LOAD

The side load condition is described in CS25.485. For this load case, it is assumed that the weight of the aircraft is divided over both main gear struts. There is an additional side load of 0.8 times the vertical load on one gear strut, and -0.6 on the other. The drag load is zero.

4.3. SUB LOAD CASES DEPENDENT ON TYRE INFLATION

The load per strut will not always be perfectly divided between both wheels. The regulations describe that in CS25.511. If a gear strut has two wheels, below five different scenarios must be investigated:

- The load per strut is equally divided; both wheels carrying 50 percent (the 50-50 scenario).
- The left tyre is flat and carries no loads; the right tyre carries 60 percent of the nominal load (the 0-60 scenario).
- The same scenario as above, but now the right tyre is flat (the 60-0 scenario)

- A difference in tyre pressure resulting in an unequally balanced load: 40 percent for the left tyre; 60 percent for the right tyre. This will be referred to as the 40-60 scenario.
- The same as above, but now the left tyre carries 60 percent of the load: the 60-40 scenario.

For the flat load cases, the side and drag loads must be halved with respect to the 50-50 case.
Combining the above rules with the fact that each load case must be considered at the most forward and the most aft CG position, there are 10 sub load cases per main load cases. This increases the total number of load cases to 410.

5

INITIAL GEAR SET-UP

Chapter 2 introduced a four-step approach for the determination of the best landing gear concept. The first step contains 3 routines: to find the loads on the gear strut, to initiate a stick diagram and to do a stress optimization of the main parts. The derivation of the loads per gear strut was explained in chapter 4. The next step is to initiate stick diagrams for all possible gear topologies. The POC tool takes a different approach: a gear design from an existing concept study is used. The aim is to optimize that gear further, where only the weight will be considered.
This chapter describes that gear in section 5.1. Section 5.2 explains how the loads per component are found.

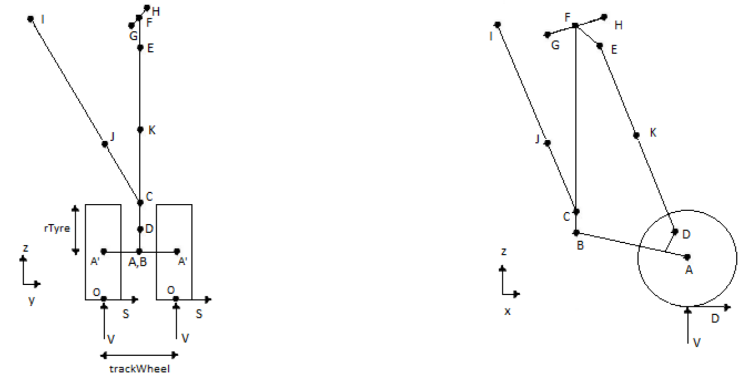
5.1. INITIAL STARTING POINT FOR THE OPTIMIZATION

The POC gear was designed as the main gear of a longe-range business aircraft. It is of the Trailing Arm (TA) type and has a tricycle layout. In the concept study, only the MLG struts were sized. Figures 5.1 and 5.2 give an isometric view of the gear in extended position, and stick diagrams of the aft and side view. A short description of the gear follows in section 5.1.1, while some limitations of this concept study are introduced in section 5.1.2.

Figure 5.1: Isometric view of original concept

5.1.1. FEATURES OF THE ORIGINAL CONCEPT STUDY

This gear has 5 main components. The Wheel Axle (WA), A'AA' in figure 5.2, connects both wheels. The Trailing Arm (TA), AB in figure 5.2, connects the WA, SA, and Main Fitting (MF). The TA can rotate around the global y direction, based on the compression of the SA. This shock absorber, DE in figure 5.2, is connected to the trailing arm and the main fitting. The function of the shock absorber, together with the tyres, is to absorb



(a) Aft view

(b) Side view

Figure 5.2: Stick diagrams of the initial concept

impact loads during taxi and landing. The absorbed energy is then gradually released, resulting in a smooth ride for the passengers.
The folding side brace consists of two parts: a lower FSB, member CJ, and an upper FSB, member JI. When the gear retracts, point J moves up and point F is pulled inboard. The rectract actuator that connects point F to the aircraft, is not shown in figure 5.2.
The main fitting, finally, is member BGH. It connects the trailing arm and the folding side brace to the aircraft interface points. It has a fixed lug in point E, where the shock absorber is mounted. The main fitting is subdivided in two parts: the trunnion GFH and the main fitting BF.
Interface points with the airframe are points I, G and H. The points where the tyres touch the ground is given the point O. The points A to J are called hardpoints. The loads at these hardpoints are calculated with a MBS package, as a function of the ground loads in the points O. That is explained in section 5.2

5.1.2. LIMITATIONS OF THE ORIGINAL CONCEPT STUDY

There are three sources that describe the original concept study, all internal FLG documents. These sources are contradicting, however. There is a document that describes the different features of the gear, on a more general level[10]. From that document, the interface points with the aircraft are found. Then, there is a specific load derivation report[11]. That report introduces the hardpoints of the previous section and also specifies the orientation of the different parts with respect to each other. Finally, there is a CATIA product that visualizes the gear. The three sources unfortunately disagree on the physical dimensions of the gear.
The first problem is with the interface points to the aircraft: points G, H and I. Point I is not mentioned in the loads report, and its location is different in the CATIA product and the general description document. Points G and H are mentioned in all three sources, but now the CATIA product and the general description document agree, while the loads description report uses other coordinates. It can thus not be verified which set of points is correct. For points G and H, the locations from the loads report are used; while the coordinates of point I as mentioned in the general description document are taken.
The second problem is with the computation of the loads on the folding side brace CJI. In the loads description document, these loads are a function of the ground reaction forces only. In reality, the orientation of the shock absorber will also have an influence on these loads. This has been corrected for, resulting in differences in chapter 8.
The third problem is also related to the folding side brace. Besides the incorrect load calculation, the orientation of this member is not properly documented in the loads calculation report. The location of point C is clearly stated with respect to point B, but the orientation of the member CJI is undocumented. As seen before, the location of point I can also not be verified. So, the orientation of the folding side brace in the global reference frame is unknown for the original concept. For the analytical calculation explained in section 5.2.1, this is not a problem as the method from the load description report is used. For the general solution of section 5.2.2 this is a problem, as will be seen in chapter 8.

[10]report number 2010/00060 Section 1 Appendix A
[11]report number 2009/00144JTAP003

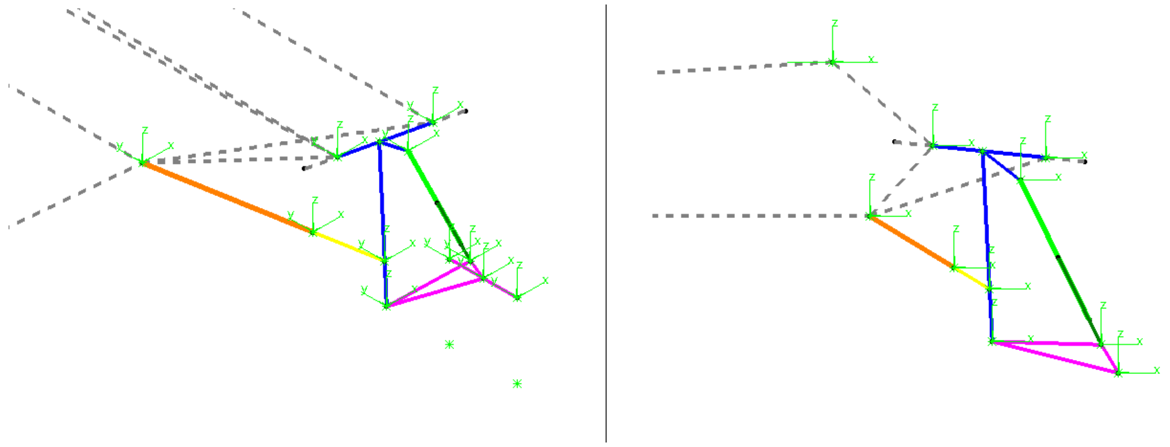


Figure 5.3: The virtual VLM model: isometric and side view

5.2. DETERMINING THE LOADS IN THE HARDPOINTS

From the ground reaction loads on the wheel axle, the loads in the hardpoints can be found. This is done in two ways: via an analytical derivation for the original geometry, and via dedicated software for a parametric gear model.

5.2.1. ANALYTIC SOLUTION

As a first step, the load in every hardpoint is derived analytically. This is done for a single gear geometry only: the one visualised in Fig. 5.2. To find the loads for a parametric gear model is of more interest, so this section is kept short.
As all load cases in this thesis are static, each member must be in equilibrium. Using that knowledge, the loads at every hardpoint can be derived from the load at the wheels. By first evaluating the loads on the wheel axle, then the trailing arm, the shock absorber, the main fitting, the folding side brace and finally the trunnion, the load in every hardpoint is found [12]. The analytic model is an exact copy of the original concept. It is used to study the optimal position of this gear under the aircraft, but not to study the optimal position of the different members. For that purpose, a multi-body model is created with dedicated software.

5.2.2. USING DEDICATED SOFTWARE

A fully parametric virtual model was created in the Siemens program LMS Virtual.Lab Motion (VLM). The required inputs for the model are the location of each hardpoint, the position of the CG and the position of the nose gear. The lengths of the members and their orientation are thus not directly settable, but indirectly via the definition of the hardpoints. This model evaluates the static equilibrium when the location of a hardpoint is moved. The initial set-up of this VLM model is shown in Fig. 5.3. It is equal to the analytic model, with the exception that the location of point I must be specified to set up the VLM model. This was not needed for the analytical model.
Where the analytical model describes the gear in terms of part-lengths and orientation angles, the VLM model is set up with the coordinates of the hardpoints. For the initial point, these definitions are exactly the same. A global axis system is located at every hardpoint. This is also visible in Fig. 5.3. It is assumed that the left gear is symmetric to the right gear. It is possible to evaluate the extension and retraction of the shock absorber with the VLM model.
The type of joint at each point is given in table 5.1. The revolute (REV) allows one rotational degree of freedom (DOF); the spherical (SPH) allows three rotational DOFs; the translational joint (TRANS) allows a member to extend or retract; and the bracket joint (BRA) is a fixed connection, like a weld.
To perform an analysis, loads are introduced at the nose gear and at the center of the main gear wheel axles. By fixing the location of the CG, the forces at the other hardpoints of the main gear are found, in a global axis system. This can be done in sequence for any number of load cases, directly from MATLAB.
As a post-processing step, the forces in the global axis system are rotated such that each member has a purely axial force and two shear forces. This rotated axis system will be called the part-local axis system. The orien-

[12] for more information, see Fokker report 2009/00144JTAP003

Table 5.1: Types of joint at hardpoints.

Hardpoint	type of joint
A	BRA
B	REV
C	SPH
D	SPH
E	SPH
F	BRA
G	SPH
H	SPH
I	SPH
J	BRA
K	TRANS

tation of the part-local axis system with respect to the global axis system is found from the set of hardpoints used to set up the gear model. Spherical coordinates are used. With these local forces per member, the needed cross-sectional parameters can be determined.

6

COMPONENT WEIGHT ESTIMATION

This chapter explains how each component is sized, based on the acting forces. How to obtain these was described in the previous chapter. Every LG component is modelled as an H-beam, a cylinder, or a socket. The forces found in chapter 5 should be multiplied with a factor of 1.5 before they are used to the size the components, to obtain the ultimate loads. Thin-walled theory is *not* valid in this chapter. A generally accepted assumption for thin-walled structures is that the diameter of the member must be at least 20 times larger than its thickness. This is not true for landing gear components. Although this chapter is entitled component weight estimation, the focus of the chapter is on determining the cross-section of the LG components. Once a cross-section is defined, the weight is easily found.

6.1. STRESSES IN AN H-BEAM

The FSB is modelled as an H-beam. It has a purely axial load, as it is mounted between two spherical joints. The maximum load F_{max} that a beam can support before it buckles is thus given by equation 6.1 [35, p.257]:

$$F_{\max} = \frac{\pi^2 EI}{L^2} \tag{6.1}$$

where E is the Young's modulus of the material, L is the beam length and I is the moment of inertia. An H-beam is modelled as an I-beam with cut-outs, see figure 6.1. That cut-out is used to place sensors, wires, locks, or actuators. This is done to limit the space taken up by the component when the gear is folded.

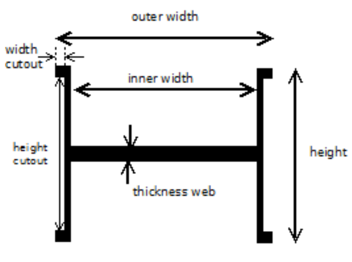


Figure 6.1: Model of an H-beam

The relevant moment of inertia, in terms of the design variables introduced in figure 6.1, is given by equation 6.2.

$$I = \frac{h^3 \cdot (w_o - w_i)}{12} + \frac{w_i \cdot t_{web}^3}{12} - \frac{2 \cdot w_{cutout} \cdot h_{cutout}^3}{12} \tag{6.2}$$

The maximum force that can be supported by the I-beam should be higher than the compressive force acting on the member.

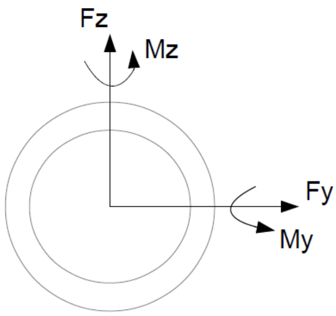


Figure 6.2: Forces in x, y, and z direction acting on a cylinder

6.2. STRESSES IN CYLINDERS 1: EXTERNAL FORCES AND MOMENTS

This section uses the local axis system per component; see figure 6.2.
The method explained in Bruhn [36] is used to find the optimum beam cross section. That method can be summarized as follows:

1. Find the tensile, bending, shear and torsional stress, based on standard equations
2. For every stress type, compare it to the allowable stress (a material property) and express it as a fraction
3. Find the Margin of Safety (MOS), based on these fractions.

To obtain the tensile stress σ_t, equation 6.3 is used [36]:

$$\sigma_t = \frac{F_x}{A} \tag{6.3}$$

where F_x is the force along the local x axis and A is the cross-sectional area of the cylinder.
The bending stress is given by equation 6.4 [36]:

$$\sigma_b = \frac{M_z \cdot r_o}{I_{yy}} - \frac{M_y \cdot r_o}{I_{zz}} \tag{6.4}$$

with the moment of inertia around the y and z axis I_{yy} and I_{zz} and r_o the distance from the center to the outer section of the beam.
The shear stress due to an axial torque T is given by equation 6.5 and the (average) shear stress due to shear forces F_y and F_z by equation 6.6.

$$\sigma_{st} = \frac{T \cdot r_o}{J} = \frac{M_x}{\frac{\pi}{2}(r_o^3 - r_i^3)} \tag{6.5}$$

$$\sigma_s = \frac{\sqrt{F_y^2 + F_z^2}}{A} \tag{6.6}$$

Knowing the stress in the member, it can be compared to the allowable stress F to find a stress ratio. Equation 6.7 gives the equation for tensile stress; the other ratios are found in the same way:

$$R_t = \frac{\sigma_t}{F_{tu}} \tag{6.7}$$

For stress due to shear forces and stress due to torque, the allowable shear stress of the material must be taken. For bending stress, the allowable bending stress of the beam is selected. This allowable bending stress is depending on the ratio between the diameter of the cylinder and its thickness, as described in the metallic materials properties development and standardization handbook [37]. For 300M steel, figure 6.3 is presented. This non-linear line is approximated by 3 different linear lines as indicated in the same figure.
Equations 6.3 to 6.6 have all assumed that there is not other load acting than those specified. In reality, all 6 forces act at the same time and their combined effect needs to be considered. For that reason, the MOS

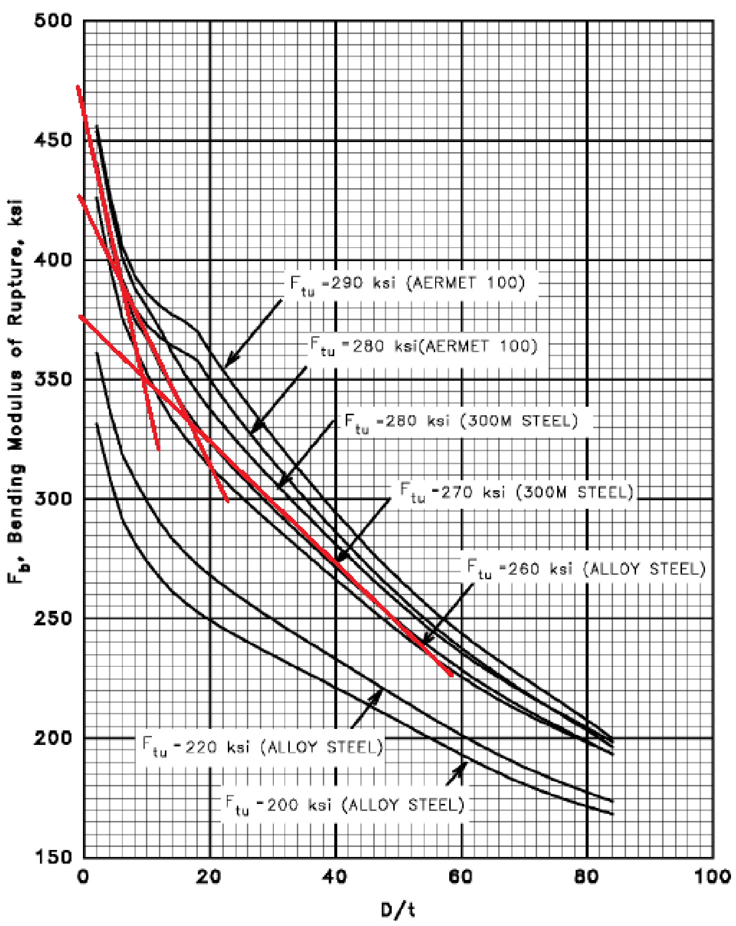


Figure 6.3: Allowable bending stress of cylinders made from 300M steel [37]

should be found from the stress ratios. This MoS can be determined with equation 6.8. It should always be higher than 0, but it can be set to a higher value by the user of LIT if this is required.

$$MoS = \frac{1}{\sqrt{(R_b + R_a)^2 + (R_s + R_{st})^2}} - 1 \tag{6.8}$$

6.3. STRESSES IN CIRCULAR TUBES 2: INTERNAL PRESSURES

The SA and Retract Actuator (RA) housing contain compressed hydraulic fluids. Therefore, the radial and tangential hoop stress in these components must be checked. They are not allowed to exceed the yield stress of the used material. For thick-walled cylinders, the radial and tangential hoop stress are given by equation 6.9 and 6.10 respectively [36]:

$$\sigma_{h,1} = [\frac{p_i r_i^2 - p_o r_o^2}{r_o^2 - r_i^2}] + [\frac{r_i^2 r_o^2 (p_0 - p_i)}{r^2 (r_o^2 - r_i^2)}] \tag{6.9}$$

$$\sigma_{h,2} = [\frac{p_i r_i^2 - p_o r_o^2}{r_o^2 - r_i^2}] - [\frac{r_i^2 r_o^2 (p_0 - p_i)}{r^2 (r_o^2 - r_i^2)}] \tag{6.10}$$

The variables in the above two equations are best explained by looking at figure 6.4:
The tangential stress is most critical for a cylinder where the outer pressure is negligible. Setting the outside pressure to 0 and changing the inner pressure to the overpressure, equation 6.10 can be simplified into equation 6.11:

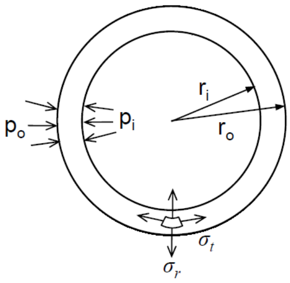


Figure 6.4: Cylinder with hoop stress

$$\sigma_h = p\left(\frac{r_i^2 + r_o^2}{r_o^2 - r_i^2}\right) \tag{6.11}$$

This hoop stress in the material should be lower that the ultimate tensile stress of the material.

6.4. STRESSES ON A SOCKET

The trunnion is modelled as a socket. A socket is a structural member that is designed to hold a pin; in this case the pins that connect the airframe structure and the landing gear. The main load for a socket is in the transverse direction; see also figure 6.5.

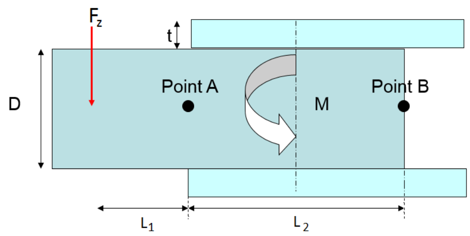


Figure 6.5: Socket parameters

In figure 6.5, the pin diameter is given by d and the socket thickness by t. The moment is highest in point B and is given by equation 6.12:

$$M = W(L_1 + L_2) \tag{6.12}$$

The load in point A can then be derived as equation 6.13 [13]:

$$F = \frac{48M}{11L_2^2} + \frac{W}{L_2} \tag{6.13}$$

Based on this force, the burst stress σ_b and the bearing stress σ_{br} can be calculated with equations 6.14 and 6.15:

$$\sigma_b = \frac{F}{2t} \tag{6.14}$$

$$\sigma_{br} = \frac{F}{d} \tag{6.15}$$

the stress ratio's are then found with the same method as in equation 6.7, after which the MoS is obtained with equation 6.16 [38]:

$$MoS = \frac{1}{\sqrt{R_b^2 + R_{br}^2}} - 1 \tag{6.16}$$

[13] Internal FLG method provided by Mike Smeets. No scientific source available

7

DESCRIPTION OF THE PROOF OF CONCEPT

With all theory described, this chapter explains the set-up of the POC tool, in terms of software. The chosen optimization architecture will first be explained. The set-up of the tool is the topic of section 7.2. The needed input to start the tool is explained in section 7.3. Section 7.4 discusses some specific constraints that were omitted in chapter 3. After that, the output from the tool can be found in section 7.5. The results will be given in a new chapter: 9.

7.1. MDO ARCHITECTURE CHOSEN

In section 1.2.1, three different MDO architectures were introduced. One of the most important choices that had to be made is the selection of an architecture.

Studying the works by Chai and Mason [1] and Heerens [2], the first of these uses a methodology closely resembling the IDF architecture and the latter uses a brute-force approach to find a result. Cumnuantip [27] uses an MDF-like approach in his study. Based on these studies, it is thus inconclusive which MDO architecture is best for the landing gear optimization problem.

Therefore, more research was performed. A usefull article in that respect is the one by Alexandrov and Kodiyalam [39]. This study took 10 different optimization problems from different engineering disciplines that have been published. Each of these problems used either MDF, IDF or CO. Alexandrov and Kodiyalam used all three methods to solve these problems again, to study if one architecture is superior for certain problems. The results were inconclusive, but they did show that not every architecture can solve every problem. Another conclusion was that certain architectures are much more efficient than others for the same problem. So, the choice of MDO architecture is indeed not trivial.

It is concluded that all three discussed MDO architectures will result in a working POC tool. With that in mind, the choice of architecture is mainly based on code efficiency. It is explained in the next section that object-oriented programming principles are used during the development of the POC tool. The MDF scheme aligns best with the principles of OOP, so that optimization architecture is chose. The resulting scheme is visualized in figure 7.1

The optimization occurs on two levels. Only the top-level optimization is shown in Fig. 7.1, along with the system-level design variables and the internal variables. The system-level variables are those controlled by the optimizer (location of the hardpoints); the internal variables are those computed by the different disciplines (the other variables). The sizing of each structural part is however an optimization of its own, with component-level design variables. This optimization module determines the needed cross section for each part. For the sizing of the trailing arm, for example, there is an optimization routine that uses the force on the trailing arm and the location of points A and B to determine the inner and outer radius of the trailing arm. These two radii are the component-level design variables.

This breakdown into a component-level and a system-level optimization was a deliberate choice, for which two arguments are given. First, it reduces the dimensionality of the problem. Including the component-level design variables there are over 30, weakly coupled, design variables. It they had all been declared as system-level design variables, it would have resulted in a 30-D problem. Such a large problem in general has convergence problems and needs a long computational time [11]. As the design variables are weakly coupled, the problem is split into several lower-dimension problems that converge much faster.

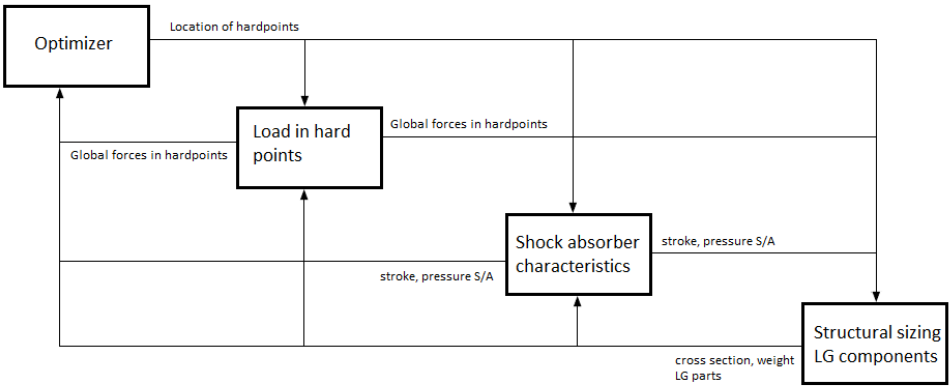


Figure 7.1: Optimization structure of the POC framework.

The second reason for including a component-level optimization is the foreseen expansion of this framework. As described in section 2.4, the parts that have now been modelled as rigid, should be replaced by modal parts in a later stage. These modal parts are analysed with specialised software. By sizing the rigid parts in a component-level optimization, they can be replaced by modal parts without making any changes to the overall framework.

It should be noted that there are other optimization architectures that use a multi-level approach. Examples are Concurrent Sub-Space Optimization (CSSO) [40] and Bi-Level Integrated System Synthesis (BLISS) [41]. A literature study concluded that these are not the best fit for the LG design problem.

7.2. SOFTWARE IMPLEMENTATION

With the expansion and maintainability of the POC tool in mind, the software was written with OOP principles in mind. For those unfamiliar with OOP, appendix C gives an introduction into the concept. To summarize that introduction here, OOP gives structure to software by using classes. A class can be seen as a blueprint for some physical part (e.g. a cylinder). It defines the variables (properties) needed to create (initiate) an object, and some specialized functions (methods). These methods only work on that specific class. For instance, the WA, TA and MF are all cylinders. They are three different instances of the same class.

The inheritance principle is important in OOP. Ideally, no single line of code should be copied. If blocks of code are needed twice, a parent class can be constructed together with some child classes. The child inherits all properties and methods of its parent. Other, new, properties and methods can be defined per child; properties and methods that its 'brother' doesn't have. See for example the classes `Cylinder` and `HBeam` in figure 7.2. These two classes share some properties that they inherited from their parent: `lengthMember`, `geomInitial`, etc. They also have their own properties and methods, for example `weightPresHydraulic` for the class `Cylinder` and `HBeam` for the class `HBeam`.

Two top-level parent classes are used for the POC tool: `StructuralMember` and `LGSystem`, see also figures 7.2 and 7.3.The construction of a trailing arm gear in figure 7.3 does not benefit from the OOP principles. However, with LIT as described in chapter 2 in mind, this set-up is useful. It places some of the blocks already in place.

The benefit of using OOP can be explained with figure 7.2, which shows all classes derived from `StructuralMember`. The parent class `StructuralMember` defines the properties that all derived classes have: material properties, a length, initial design parameters, and boundaries for these parameters. It is not important that these design parameters are different for the classes `Hbeam` and `Cylinder`. As long as they are a 1xN row, `StructuralMember` will accept them. It is checked in the child classes whether the given input is correct. For example: an object of class `Cylinder` needs two design variables: the inner radius and the outer radius. If an object of class `Cylinder` is initiated with three design variables, the program will throw an error. This is just one example of the structure given to the program.

The link between the two top-level classes `StruturalMember` and `LGSystem` is visible in the class `TAGearFSB`. Its properties are specified to be of a certain class. For example, the main fitting must always be of type `Cylinder`. This is another example of a structured software set-up.

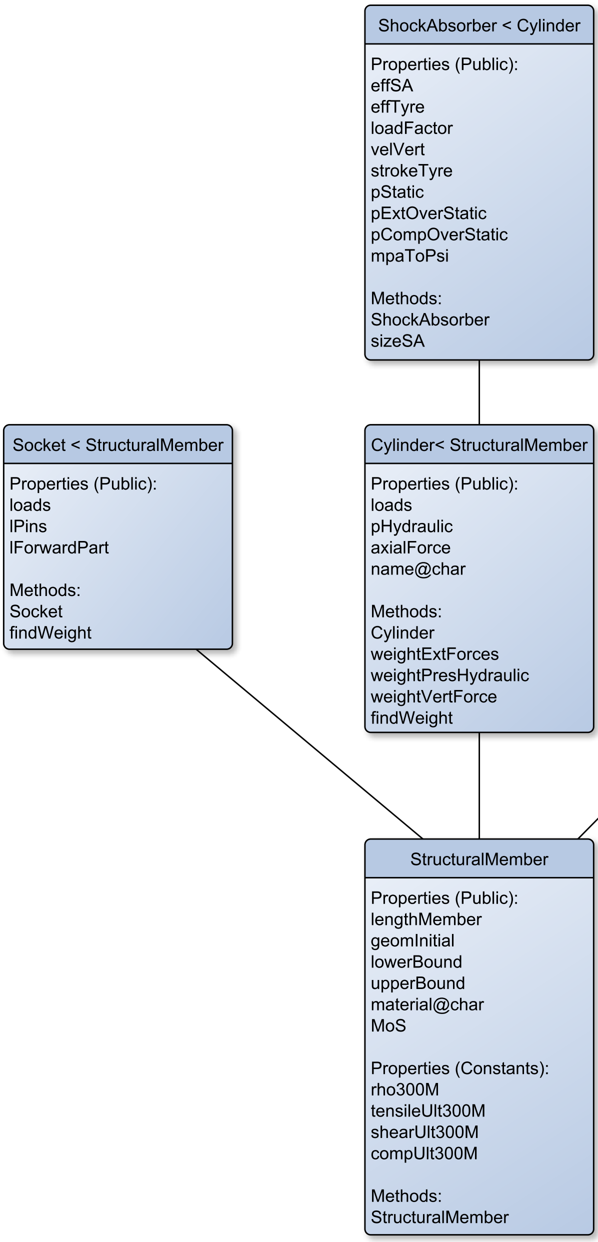


Figure 7.2: Classes derived from parent StructuralMember.

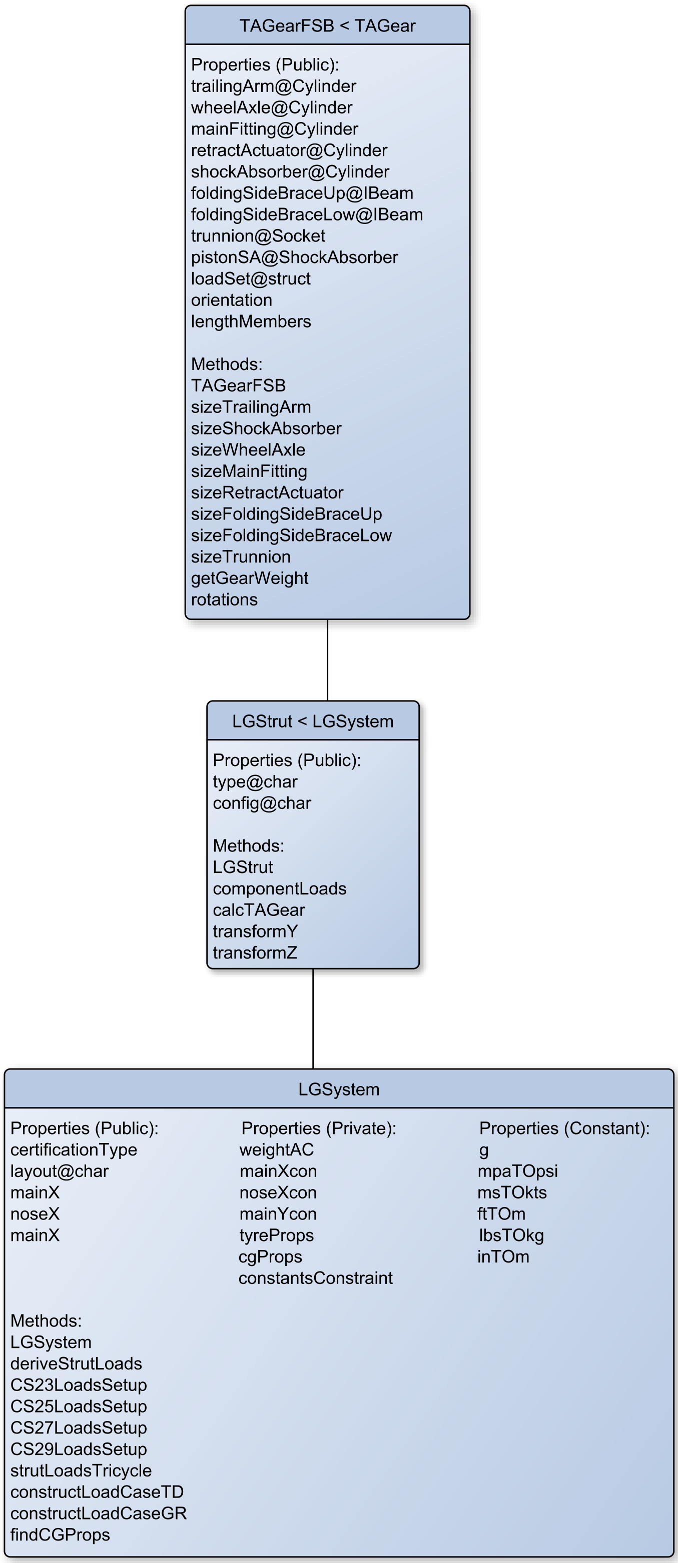


Figure 7.3: Classes derived from parent LGSystem.

Table 7.1: Inputs for POC tool: aircraft parameters.

Parameter	Unit	Parameter	Unit
Aircraft weight(s)	N	Wing span	m
CG location fwd (x,y,z)	m	Initial position NLG (x,y,z)	m
CG location aft (x,y,z)	m	Initial position MLG (x,y,z)	m
Minimum MLG position (x,y,z)	m	Maximum MLG position (x,y,z)	m
Minimum NLG position (x,y,z)	m	Maximum NLG position (x,y,z)	m
Turnover angle	deg	max pitch angle	deg
Max approach speed	m/s	Tail height	m
Max roll angle @ TD	deg	wing sweep angle	deg
distance between secundary tyres	m	radius sec tyres	m
distance between main gear tyres	m	radius MG tyres	m
Minimum load on NLG strut	% W	Maximum load on NLG strut	% W
steering angle nosewheel	deg		

Table 7.2: Inputs for POC tool: shock absorber parameters.

Parameter	unit	Parameter	Unit
Energy absorption efficiency SA	%	Energy absorption efficiency tyre	%
Load factor at TD	-	vertical TD velocity	m/s
Static SA pressure	Pa	ratio static over fully extended pressure	-
ratio fully compressed over static pressure	-	Tyre stroke	m

The component-level optimizers introduced in section 7.1 are contained in the classes derived from `StructuralMember`. The `fmincon` routine is used, see appendix C for more information. The `fmincon` function is an optimization routine from the MATLAB optimization toolbox. It is a gradient-based method, that finds local optima only. To find a local optimum is fine for the POC tool, as its main objective is to identify improvements to the original concept. For a more generic problem however, the use of global optimizers must be considered.
The system-level optimizer is also the `fmincon` routine. The objective function, the constraints and the design variables are however different.
This system-level optimizer is defined in the run-file. This file initiates an instance of the class `TAgearFSB`, defines the objective, finds its minimum given some constraints, and gives the results. How to start this process, which constraints act, and what kind of results to expectis explained in the next sections.

7.3. INPUT HANDLING

The needed input for the POC tool can be divided into two categories: aircraft data and parameters for the sizing of the shock absorber. The aircraft data is given in table 7.1, while the SA parameters are given in table 7.2. For this aircraft, type of tyre is fixed by the OEM.
Other inputs are the initial and maximum/minimum allowable cross-sectional parameters of all parts. As initial parameters, those of the initial concept are used. How the maximum is determined, can be found in section 7.4. The minimum allowed dimension is in general zero, except for cut-outs. The lengths of the members are obtained from the MBS model.

7.4. CONSTRAINTS

Besides the system-level positioning constraints described in chapter 3, there are also component-level constraints. Additionally, there are some constraints specific to the POC gear. The component-level constraints are usually straightforward: the MoS from chapter 6 must be above 0; the inner radius of a part must be smaller than its outer radius, all dimensions must be larger than zero, the cut-out of the H-beam cannot be larger/deeper than its height/thickness and every thickness must be larger than 2 mm to make manufacturing possible. Although straightforward, these constraints must be specified: the optimization is purely mathematical and does not take any physics into account.
Then, there is maximum on the cross-sectional parameters. As explained in chapter 2, these parameters are normally constrained by the gear kinematics: the gear still must fit inside the gear bay. An introduction into

this field of kinematics is provided in chapter 8 of Currey [4]. There was however insufficient time to implement it in the POC tool. Thus, it was decided to restrict the outer dimensions of the parts to the values used in the original concept. With these dimensions, the gear fits in the bay. With a larger cross-section, that cannot be guaranteed. It is allowed that components become thinner or thicker, but it is not allowed to increase the outer dimensions.

7.5. OUTPUT HANDLING

The resulting gear weight, the position of the gear struts, the final stick diagram from the MBS model, and the cross sections of each component are obvious output parameters. Additionally, the critical load case for each part is given as an output, as well as the weight of every individual part.

The most important output parameter is however a message from the component-level optimizers. One problem encountered with the multi-level optimization is namely that the component-level optimizers can be erroneous. It can happen that the component-level optimizer does not converge to a correct solution. This doesn't have an effect on the system-level optimizer. During development, results were obtained where parts had an MoS below zero (and thus violated constraints in the component-level optimizer), without the system-level optimizer giving a warning. The results were presented without any warnings. So, an additional output is the maximum value of the constraints from the component-level optimizers.

It is the designers task to study the generated output, as there can be several errors in a component-level optimizer. Two common errors are described here. The first option is a negative margin of safety. This error can come from the incorrect calculation of stresses, an incorrect computation of the MoS, or too tight boundaries on the design variables[14]. The first two option occurred during development, but should now be fixed. So when this error occurs, a relaxation of the boundaries on the design variables should be the first option.

The second possibility is that the optimizer algorithm has convergence problems. This can be tested by changing the algorithm and running the code again.

[14] the optimizer will never use invalid values for the design variables to satisfy the constraints. The boundaries on design variables have a higher priority than the constraint value

8

VERIFICATION OF THE PROOF OF CONCEPT

This chapter compares the results of the analytic and the multi-body model to the results of the original concept study. This is done for the initial configuration only: the one described in section 5.1. To ensure the same set of requirements and constraints, the following factors are equal for all three models:

- The maximum allowed outer dimensions of each part
- The global position of the gear struts
- The margin of safety of each gear component
- The evaluated load cases

As an extra verification step, also the critical load case per gear component will be computed. It was described before that there are 41 different load cases, that the critical case differs per part, and that it is generally unknown in advance. Therefore, it is relevant output.
The analytic model is modelled in the exact same way as the original concept study. It uses a (fixed) length to define parts and angles to describe the orientation of the parts with respect to each other. The exact interface points with the aircraft are not known, as was described in section 5.1. They are not needed to define the model.
The multi-body model is defined by a coordinate set for all the hardpoints. These coordinates follow from the description of the original study. The interface points with the aircraft are needed for this model. As these points are not properly defined, they have been assumed to correspond to the values of the gear description document.
Table 8.1 breaks down the gear weight in the different component weight and compares them for the original concept study, the analytical model and the multi-body model. No reliable data is available for the wheel axle, so that part is not included. The critical load case is also given for each model, as an additional verification step. As the used data is confidential, the numbers in this chapter are all normalized.
The results are discussed separately for the analytical model and the multi-body model.

8.1. ANALYTICAL MODEL

The analytical model is defined in the exact same way as the original concept study. The following conclusions can be made from table 8.1

- The system weight is predicted within a 2 percent accuracy comparted to the original study
- Except for the folding side brace, all critical load cases are correctly computed by the framework
- The results for the cylindrical components are very similar: within 4 percent
- The results for the H-beams have differences of up to 26 percent

Table 8.1: Verification of analytical and multi-body model with respect to original model.

	Original results		Framework Analytical			Framework multibody		
Component	Weight [kg]	Critical load case	Weight [kg]	Critical load case	difference [%]	Weight [kg]	Critical load case	difference [%]
Trailing arm	1.337	Turning1	1.335	Turning1	-0.15	1.335	Turning1	-0.15
Main fitting	3.258	2-point braked roll	3.271	2-point braked roll	+0.39	1.976	2-point braked roll	-39.35 [15]
Folding Side Brace up	0.522	Side load	0.386	Turning1	-26.14 [16]	0.472	Turning1	-9.58
Folding side brace low	0.236	Side load	0.191	Turning1	-18.64 [17]	0.232	Turning1	-1.69
Shock absorber	1	Burst pressure	0.998	Burst pressure	-0.20	0.998	Burst pressure	-0.20
Retract actuator	0.166	Burst pressure	0.159	Burst pressure	-4.00	0.159	Burst pressure	-4.00
Trunnion	3.175	2-point braked roll	3.168	2-point braked roll	-0.23	4.105	2-point braked roll	+29.29 [18]
Total System	9.693	-	9.507	-	-1.91	9.228	-	-4.18

[15] Due to unknown position of interface point with aircraft. See section 5.1.2
[16] Due to different computation of force on member, see section 5.1.2
[17] Due to different computation of force on member, see section 5.1.2
[18] Due to unknown position of interface point with aircraft. See section 5.1.2

The difference in the results for the folding side braces traces back to an error in the original concept study. The loads on the side brace are incorrectly computed, leading to approximately 20 % higher loads. In the original concept study, the loads on the folding side brace are independent from the position of and the loads on the shock absorber. The effect of the shock absorber should however be taken into account.
With that knowledge, it can be concluded that the results from the analytical approach are sufficiently close to the results from the concept study and the value of the framework is shown. The computational time for one iteration is reduced from a day to five minutes.

8.2. MULTI-BODY MODEL

The results for the VLM model are not as expected, although the total gear weight is within 4.2%. The main problem is that it cannot be verified that the used geometry for the VLM model is equal to the used geometry for the original model. The location of the interface points with the aircraft is not documented, so they had to be taken from a different source than the source that describes the gear layout. These two sources do not correspond to each other, as was seen in section 5.1.2.
For the two components of which it is known that they are correctly modelled, the weight correspond almost exactly to the original model. These two components are the trailing arm and the shock absorber.
The orientation of the folding side brace is uncertain. Indeed, this component shows a weight difference of up to 10 percent compared to the original model. This difference on the folding side brace also has an impact on the main fitting and the trunnion, as these parts still need to be in equilibrium. These two components also show large differences between the original concepts and the VLM model.
Additional checks were performed to verify that the uncertainty in the model geometry is indeed the cause for these differences. They were all ruled out, leaving only the geometry uncertainty left as an error possibility. These checks include:

- Input forces on the wheel axle are the same for the concept study and the VLM model
- Orientation of the gear parts in 3-D space is correctly computed
- The rotation matrices used to change forces from the global to the part-local axis system are correct
- The global forces computed with the VLM software package are correct: all parts are in equilibrium.

As the main objective of this thesis was to develop a working optimization framework and not to optimize an old concept study, no more effort is spend to further align the results of the concept study and the VLM model. The differences are accepted as they are. Even more so, the value of the developed framework is again shown by this verification step. Integration errors such as those made in the original concept are no longer possible.

9 RESULTS

It can be argued that the developed framework, as described in the chapter 7, is the main result of this thesis. It was decided to keep it out of this chapter however, such that the focus of this chapter can lie on the trade studies that are possible with the framework.
This chapter then shows some of these results. Section 9.1 shows two case-studies, while section 9.2 has a more commercial perspective and lists some of the other studies that could be done with the developed framework.

9.1. RESULTS OF CASE STUDIES

This section gives the result of two performed trade studies. For the first trade study, the large business jet trailing arm gear of section 5.1 is analysed with the analytical model. The gear thus has a fixed geometry and is placed at different locations under the aircraft, to find the optimum location. As this computation is relatively easy and only a few system-level design parameters are involved, the complete design space can be visualized.
For the second case study, the gear location is fixed and the relative orientation of the different landing gear components is varied. This involves more variables and thus visualization becomes harder. As there are now 7 system-level design variables, the complete design space cannot be visualized. The second case uses the multi-body gear model.

9.1.1. CHANGING THE GEAR LOCATION

For this trade study, the positions of the hardpoints were fixed, using the analytical model. The coordinates of the NLG and MLG struts are the system-level design variables and their optimal location is designed for. Normally, only one answer is relevant: which gear position gives the lowest weight? This section takes a different approach and computes the gear weight at over 700 different, unique positions. This results in figure 9.1, which shows the weight of the MLG as a function of the x position of the NLG, the x position of the MLG and the y position of the MLG. The gear weight is represented as a fourth dimension: the color. Only a few slices of the full design space are visualised.
As figure 9.1 may be difficult to interpret, figures 9.2 to 9.4 visualize the same data set in another way. In each figure, one of the design variables (x position MLG, x position NLG or y position MLG) is fixed at a certain value, giving the gear weight as a function of the other two design variables. A contour plot of the system weight is given.
Figure 9.2 gives the gear weight as a function of the y position of the MLG and the x position of the NLG. In each sub-figure, the x position of the MLG is frozen. It is at its most forward position in the top left figure and moves aft in clock-wise direction.
Similarly, the x position of the nose gear is frozen in figure 9.3; giving a contour plot of the gear weight as a function of the MLG position. Again, the NLG is at its most forward location in the top left figure and moves aft in the clock-wise direction. Figure 9.4 finally gives a contour plot of the gear weight as a function of the x position of both NLG and MLG. The y position of the MLG is frozen. It is most inboard in the top left figure and moves outboard in the clockwise direction.
In line with the previous chapter, the values in these figures are again normalised.

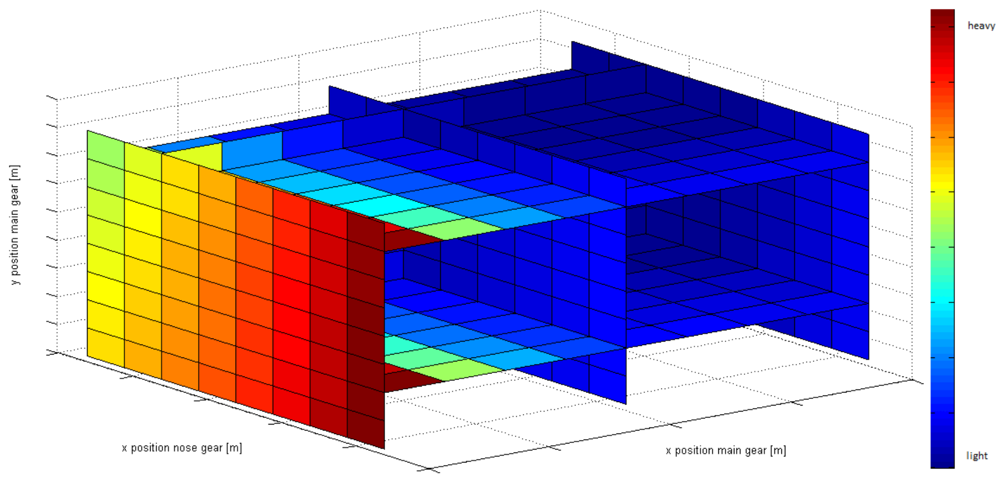


Figure 9.1: 4-D representation of the design space. System weight represented by color as a fourth dimension.

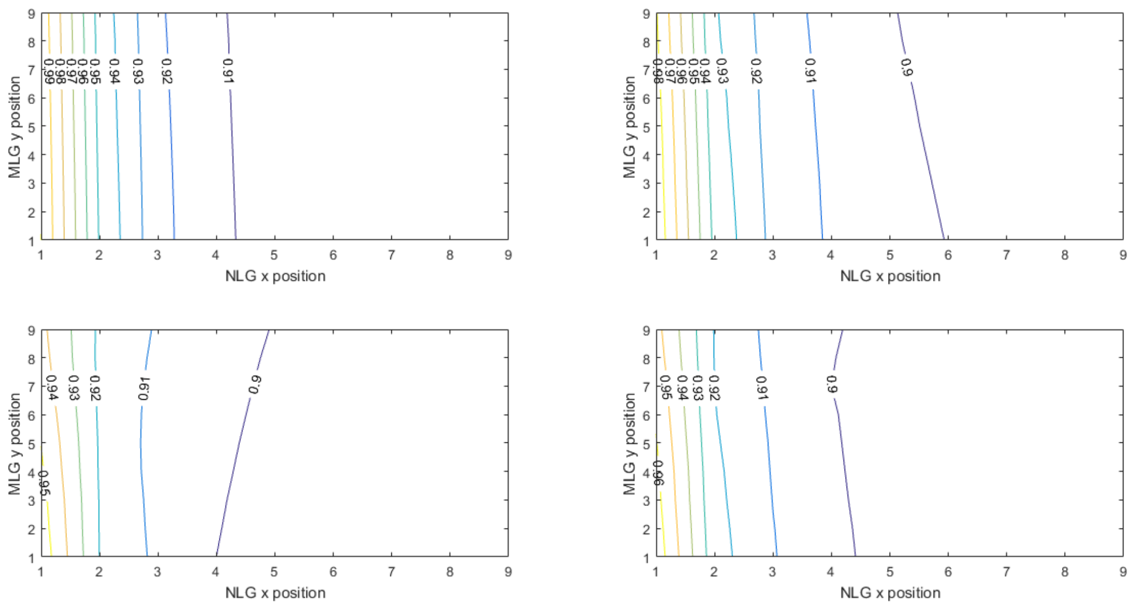


Figure 9.2: X position MLG fixed in each sub-figure. Moving aft from top left and in clockwise direction

The following conclusions are made:

- If the main gear is moved aft, the system weight decreases. This is as expected. If the MLG moves aft, it moves further away from the CG and it has a lower static load. The load on the NLG becomes higher. Since only the MLG is sized in this study, it is as expected that a lower load on it leads to a lower system weight[19]. This is clear in figure 9.2, where the system weight decreases in a clockwise direction if the value for the x position of the MLG increases. It can also be seen in all sub-figures of figures 9.3 and 9.4, where the system weight decreases as the value for the x position of the MLG increases.
- If the nose gear is moved aft, the system weight decreases. This is also as expected. Again, if the NLG moves aft, it moves closer to the CG and takes a higher load. This decreases the load on the MLG, leading to a lower system weight. This can also be seen in all sub-figures of 9.2 to 9.4.
- The influence of the y position of the MLG is marginal. Figure 9.4 shows a contour plot of the system weight for 4 different track widths. There are slight changes between the different sub-figures, but they

[19] the NLG, in contrast, will experience a higher load and become heavier, but that effect is not taken into account yet

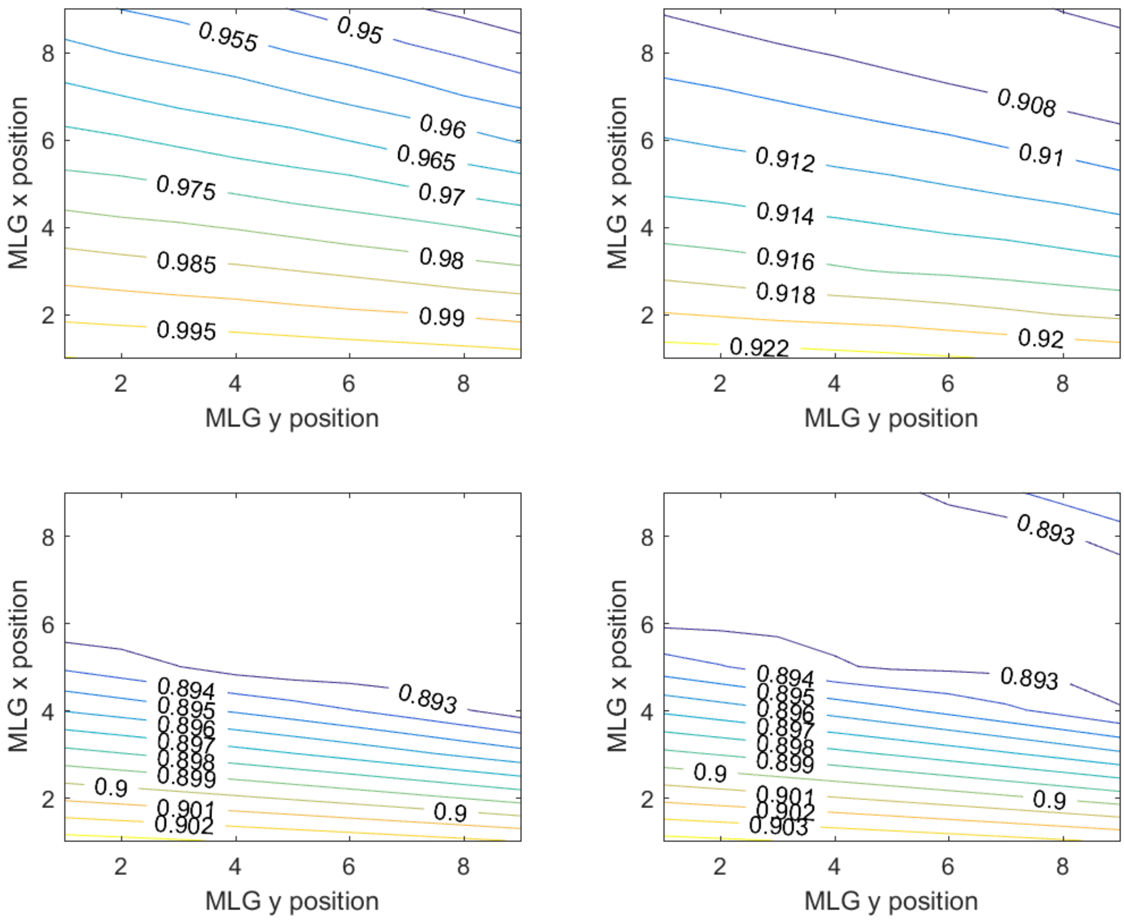


Figure 9.3: X position NLG fixed in each sub-figure. Moving aft from top left and in clockwise direction

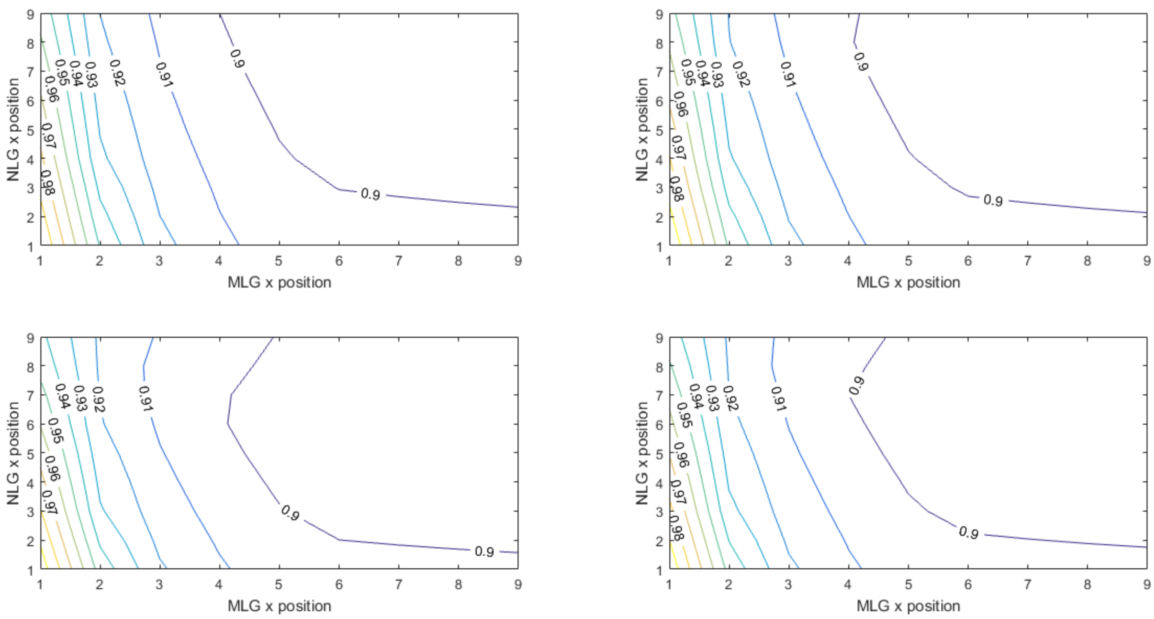


Figure 9.4: Y position MLG fixed in each sub-figure. Moving outboard from top left and in clockwise direction

are much weaker than the effect of the other two parameters. A similar behaviour is seen in figure 9.2, where the lines are almost vertical and the MLG y position has thus little influence. In figure 9.3 the lines are nearly horizontal, again supporting the conclusion that the MLG y position has little influence on the system weight.

It should however be noted that figures 9.1 to 9.4 do not look like the results of a typical optimization problem, where one would expect a single optimum. For the results of this study, there is an area (valley) of optimal gear locations that all result in the same unique gear weight: 0.893.
This can be explained by studying the results file and by looking back at the load cases defined in section 4.2. For this particular case study, it is observed that the critical load case for every component is either the side load or the braked load case at the optimal point. For these two load cases, the gear position does not have an influence on the strut load. So, as long as these two load cases are critical, the gear weight is always the same. Moving away from that valley, other load cases become critical, for example the turning load case. For these points, the gear however is heavier. This is graphically indicated with figure 9.5.

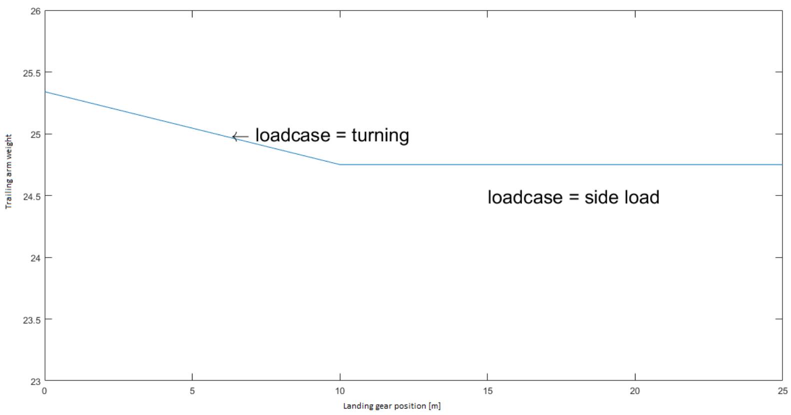


Figure 9.5: Weight of the trailing arm as a function of the main gear location

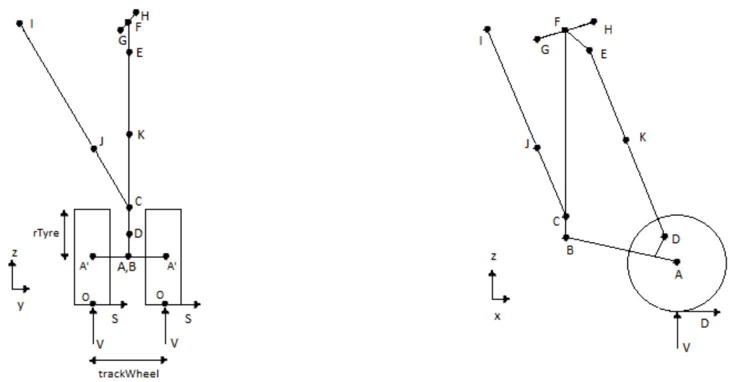
This figure shows the critical load case and the component weight as a function of the gear position. As long as the side load case is critical, the gear position has no influence on the component weight. A similar figure can be drawn for all other components.
A typical evaluation with this level op detail takes around five minutes. To generate figures 9.1 to 9.4 over 700 evaluations were however needed, increasing the runtime to several hours.

9.1.2. CHANGING THE GEAR GEOMETRY

In the figure 9.7, the location of hardpoints B, C, D and E is changed while the location of the gear under the aircraft is fixed. Figure 5.2 is repeated in this chapter as figure 9.6 to define the hardpoints once more.
Only points B to E can be moved, as I, G, and H are the interface points to the aircraft, and points J and F are dependent on the location of B and C. Point A was varied in the previous case study and is fixed for this computation. The top figure of 9.7 shows how much these seven design variables are changed with respect to their initial point. The second figure shows the gear weight at each iteration. After a relatively large first step, the solution slowly converges to an optimum in 11 additional iterations. The figure is a perfect example of what one would expect from an optimization [11]. A reduction in gear weight of almost seven percent is achieved, from the initial configuration of the multi-body model to the computed optimum. The runtime is becoming much longer in this evaluation, as the VLM model must be evaluated at each iteration and the number of system-level design variables is increased from three to seven. The model can still be run overnight however; it has a typical runtime of seven to eight hours.

9.2. OTHER POSSIBILITIES OFFERED BY THE FRAMEWORK

The possibilities offered by the current framework are more elaborate than the two trade studies described in the previous section. Some example application are mentioned below:



(a) Aft view (b) Side view

Figure 9.6: Stick diagrams of the initial concept

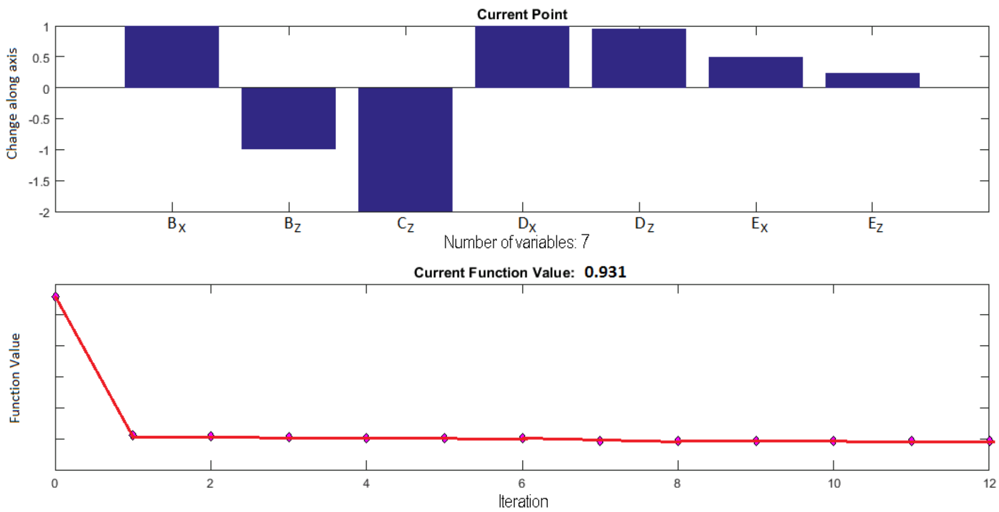


Figure 9.7: Optimizing the gear by changing the locations of the hardpoints.

- With the current optimization framework, a landing gear geometry can be found that limits the forces on the airframe to a pre-defined value. One common problem in landing gear design is namely that the airframe needs local strengthening at the attachment point of the landing gear, because the introduced loads are too high. With the presented framework, it is possible to predict in an early stage of the design process how large the introduced forces are and thus how much reinforcement will be needed.
- If an aircraft manufacturer has not fixed the location of the landing gear yet, a very quick study to find the optimal location can be performed. This is especially useful if the aircraft configuration itself is not fixed yet. One example could be that the wing shape and placement is undefined. For all possible wing configurations, the optimal gear and its weight can be computed. This analysis can be done within 15 minutes.
- It can be investigated what the effect of an increase in aircraft MTOW will be on the weight of the landing gear system, by simply changing one input parameter.
- Application of the framework is not limited to the main landing gear of a long-range business aircraft. The framework can also be used to study the MLG or NLG of a helicopter, or the NLG of a short- to medium-range passenger aircraft.

10
CONCLUSION

Four weaknesses were identified for the traditional preliminary landing gear design process. First of all, it is relying too much on existing topologies. Secondly, it is not very efficient. Many man-hours are wasted by performing repetitive design tasks instead of performing creative work. Communication between disciplines is also a problem. In preliminary design, many concepts are considered which all have their own set of parameters. Informal communication between designers leads to mistakes and inconsistent results. Lastly, integration between the design of the landing gear and the design of the aircraft is often sub-optimal.

This thesis had two main objectives to improve this preliminary landing gear design process. First, a framework was described that can perform the concept design of a landing gear automatically: LIT. The analogy to an onion was made to explain its different layers: step by step, more design variables are added to the problem until the optimal landing gear solution is found. As a first step, the locations of the gear struts are determined based on static load cases and some aircraft parameters. Then, the gear geometry is studied in more detail. As a third step, dynamic load cases are added and the problem is restarted. As a final step, the rigid parts are studied in more detail by interchanging them with modal parts.

A proof-of-concept (POC) tool is developed to show the working principles of LIT. This POC studies one particular fictitious gear: the main landing gear of a long-range business class aircraft. A trade-study to find the optimal location of this gear is shown in the results chapter. The runtime for this trade study is shorter than five minutes. Also, the position of the hardpoints was varied to analyse if this yields a lower weight. With a runtime of eight hours this analysis takes longer, but it can still be done overnight. Compared to the initial configuration, a weight decrease of almost seven percent was realized.

These two trade studies are only some of the options offered by the framework. Another possibility would be to tailor the loads introduced at the airframe, in the earliest stages of the design process. It is also possible to find a trend in the weight of the landing gear versus the MTOW of the aircraft. Furthermore, this framework is not limited in application to the main gears of a long-range business aircraft. It is equally suited to size the NLG or MLG of a helicopter or the NLG of a short- to medium-range passenger aircraft.

The analytic model used for the first trade study does not exactly correspond to the multi-body model used for the second trade study, as there is an insecurity in the position of the interface points with the aircraft. The difference in system weight is however below 4 percent. As the main objective of this thesis was to develop a working framework for landing gear design, and not to study one particular gear in full detail, it can be concluded that the research objective is met. The further development still needs a lot of work, but the proof-of-concept version is there.

11

Recommendations

This thesis explained the final lay-out for LIT and the development of a POC tool. This chapter describes the road towards completion of LIT. The tool must be expanded both in width (adding more capabilities) as in depth (adding more detail). That expansion is treated in sections 11.1 and 11.2.

11.1. Expanding the tool: adding more capabilities

At this point, the tool is only capable to analyse aircraft with a tricycle gear layout. Another limitation is that only the TA gear design is implemented. To add more functionality, the following expansions are recommended:

- Derive the strut loads for a taildragger gear layout, such that a comparison between both layouts can be made. This is mainly relevant for helicopters.
- Also military aircraft requirements should be implemented. This allows LIT to be used for the LG of fighter jets. The articles by Kempf [42] and Thorby et all [43] might serve as a starting point.
- A module should be added that estimates the LG cost. This has several implications, as the problem now transforms into a multi-objective optimization.
- Throughout this thesis, it was assumed that the loads are perfectly alligned with the gear struts. This is not true in real life. Significantly higher loads of up to 40 percent more may be the result, according to NACA technical note 2596 [44]. Although the aircraft regulations do not mention such eccentric loads, this deserves further attention.
- Flotation characteristics[20] are not considered in the current method. This becomes important when a landing on unpaved runways is considered. A good starting point in this field would be chapter 7 of [1] and the references there.
- The result from LIT should be visualized in CATIA to obtain graphic results instead of numerical tables. This requires the definition of parametric landing gear parts in CATIA, as well as a method to initiate them. Sending data from MATLAB to CATIA can be done via iSight or data sheets.
- At this moment, only the trailing arm gear is modelled. The other gear topologies from chapter 2 should be added.
- In this thesis, an existing concept gear was further optimized. Sizing rules should be defined to initiate a first concept for any type of aircraft.

[20] where the load per tyre is compared to the runway strength

11.2. EXPANDING THE TOOL: ADDING MORE DETAIL

Besides adding more capabilities to the tool, a more detailed analysis of the landing gear should be possible. The following topics are proposed:

- Retraction characteristics of the gear are not considered. It should be checked if the gear can fold in the LG bay.
- Energy absorption of the tyres is not properly modelled in this thesis. Tyre models are already created at FLG, and these should be incorporated with the work of this thesis.
- During the creation of this tool, the `fmincon` function from the MATLAB optimization toolbox is used. This function has many benefits, but also some weaknesses. In general, it only finds the local minimum of an objective function and not the global minimum. When there is no initial design to start from, the use of global optimizers might be needed. These are available in the global optimization toolbox of MATLAB.
- It was also assumed that the design space is continuous. That might not always be true. If the design space is discontinuous, a gradient-based optimizer such as `fmincon` is not the best solution and a direct search method such as genetic algorithms or particle swarm optimization should be considered. For some problems, the use of gradient based methods is still possible for a discontinuous design space, however.
- The current toolset minimizes the landing gear weight. A related problem is to minimize the weight of the landing gear + aircraft structure. This might become a feasible research direction. A good starting point in that direction is the research at the German Aerospace Center (DLR) [3], [27], [45], [46].
- The shock absorber is sized using simplified hand calculations. With VLM it is possible to add 1D SA models. This improves the solution, and allows the implementation of dynamic load cases.
- Sizing of the components is done with simple hand calculations. Modal parts should be added to include the effect of displacements and to study critical points better.
- This thesis only considers the main landing gear components such as the wheel axle, the shock absorber and the main fitting. Smaller components such as lugs and pins should be added.

A

LOAD CASES FROM REGULATIONS

Subject	CS25 [31]	CS23 [32]	CS27 [33]	CS29 [34]	STANAG4761 [47]
471: Ground loads - General	Ground loads are external forces, and the complete CG range should be considered	No mention made of CG range	Same as CS-25	Same as CS-25	Same as CS-23
473: Landing load conditions and assumptions	The vertical velocity is 10 fps at MLW and 6 fps at MTOW, and the aircraft lift equals its weight	The vertical velocity should be 10 fps as maximum and 7 fps as minimum, MTOW should always be taken, Wing lift equals 2/3 of the weight, and a load factor of 2.0 must be used	The vertical velocity must be determined from a free-fall from a height of 0.33m, Always use MTOW, Rotor lift is 2/3 of the weight	Same as CS-27, but drop height should be 0.20m	Same as CS-23
475: tyres and shock absorbers	-	-	Tyres in static position, and shock absorber in critical position	Same as CS-27	-
479: Level landing	see page 61	No drift landing	No drift landing	No drift landing	No drift landing
481: Tail-down landing	See page 62	Same as CS-25	Same as CS-25	Same as CS-25	Same as CS-25
483: One-wheel landing	See page 62	Same as CS-25	Same as CS-25	Same as CS-25	Same as CS-25
485: Side load	See page 25	Vertical force is 1.33 times V_{static}, and side load is 0.5 and 0.33 times vertical force	Same as CS-25, but extra condition 3-point landing must be considered as well: side load on nose is 0.8 times vertical load	same as CS-27	Same as CS-23
491: Taxi, take-off and landing roll	See page 58	Not mentioned	Not mentioned	Not mentioned	Not mentioned
493: Braked roll	See page 24	Load factor is always 1.33 and dynamic braking not mentioned	Load factor is 1.33 for 3-point landing, Load factor is 1.0 for 2-point landing, no dynamic braking is mentioned	Same as CS-27	Same as CS-23
495: Turning	See page 24	Not mentioned	Not mentioned	Not mentioned	Not mentioned
497: supplementary conditions for tail wheels. Not considered during this thesis					
499: supplementary conditions for nose wheels	See page 60	The nose wheel should be able to withstand: V = 2.25*V_{static} and D = 0.8*V, and V = 2.25*V_{static} and D = -0.4*V, or V = 2.25*V_{static} and S = 0.7*V but no combination of the above	Not mentioned	Not mentioned	Same as CS-23
501: Ground loading conditions: landing gear with skids. Not relevant for LIT					
503: pivoting	See page 60	Not mentioned	Not mentioned	Not mentioned	Not mentioned
505: Supplementary conditions for ski-planes. Not relevant for LIT					
507/519: Jacking	See page 61	Load factor vertical: 1.35, Any combination of side and drag load of 0.4 times static vertical load	Not mentioned	Not mentioned	Same as CS-23
507: Reversed braking	See page 60	Not mentioned	Not mentioned	Not mentioned	Not mentioned
509: Towing loads	See page 60	Same conditions as CS-25, but different table	Not mentioned	Not mentioned	Same as CS-23
511: unsymmetric loads on multiple-wheel units	The 50-50, 40-60, 60-40, 0-60 and 60-0 scenarios of section 4.3	Essentially the same as CS-25	Not mentioned	Essentially the same as CS-25	Essentially the same as CS-25

B

LOAD CASES DESCRIPTION

This appendix has been written as a stand-alone chapter and details all 41 static load cases from CS-25. They can be divided into 2 scenarios: ground load cases and touchdown load cases.

B.1. GROUND LOADS

This section treats the derivation of the ground loads, as specified in paragraphs 491 until 519 of CS-25. These include:

- taxi loads
- braking loads
- turning loads
- jawing loads
- pivoting loads
- towing loads
- jacking loads

STATIC LOADS

The static load condition is used as a basis for several load cases. Therefore, it is included in this chapter. Figure B.1 shows an Free Body Diagram (FBD) of this condition, where n equals 1. The load carried by the main gear struts, V_M, is shared by the left and right strut and must thus be divided by 2 to obtain the load per strut.

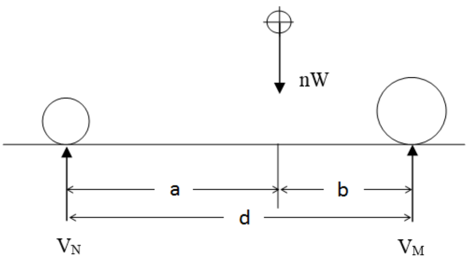


Figure B.1: Free body diagram for static load condition. Side view

Taking moments about the point where the load on the main gear struts acts, equation B.1 can be derived to calculate the static load on the nose gear:

$$V_N = \frac{Wb}{d} \tag{B.1}$$

Where V_N is the load on the nose gear, W the aircraft weight, b the distance between the main gear and the centre of gravity and d the distance between the nose and main landing gear.
Taking moments about the point where the normal force of the NLG acts, the static load per MLG strut can be calculated according to equation B.2

$$V_M = \frac{Wa}{d} \tag{B.2}$$

With V_M the load on the main gear and a the distance between the nose gear and the CG. For the static load condition, the drag and side loads on the struts are 0.

TAXI LOADS

AC25-491 states that for taxi load, a load factor of 1.7 times the static loads can be assumed to give the maximum loading during taxi[21] [48]. A more elaborate dynamic analysis should be made later in the design process, for example as described by Freund et all[49].
For now, the factor of 1.7 will be used. When doing so, figure B.1 and equations B.1 and B.2 can be applied, where the load factor should be added to the numerator.
A second requirement is the combined taxi case. In this case, the vertical force must be taken as 90 percent of the above vertical load, and the drag and side loads must be 20 percent of the vertical load.

BRAKING LOADS

Section 25.493 of the regulations describes 3 different braking scenarios:

1. braking without any load on the nose wheel and with the pitching moment of the aircraft resisted by its angular acceleration (2-point braking);
2. braking with load on the nose wheel and zero pitching acceleration (3-point braking);
3. a sudden braking motion with dynamic pitching behaviour as a result (dynamic braking).

Points 1 and 2 must be evaluated at the MRW with a load factor of 1 and at the MLW with a load factor of 1.2. The third point only needs evaluation at the MTOW.
When there is no load on the NG, all weight is carried by the MLG struts, as in equation B.3

$$V_M = \frac{nW}{2} \tag{B.3}$$

Where n is the load factor. The maximum drag force per strut must be taken as 0.8 times the vertical load per strut; the side load as 0.
The force on the main gear can be found by taking moments around the attachment point of the nose gear normal force, see equation B.4:

$$V_M = \frac{nWa}{2(d+0.8e)} \tag{B.4}$$

With e the height of the CG and the other symbols as before.
The load on the nose gear strut V_N can be calculated by taking moments around the CG and substituting equation B.4. The result is equation B.5:

$$V_N = \frac{nW(b+0.8e)}{(d+0.8e)} \tag{B.5}$$

There is an additional drag load on the main gear of 0.8 times the vertical load. The side load is 0.
For the dynamic braking case, an equation is provided by the regulations, see equation B.6:

$$V_N = \frac{W}{a+b}[b+\frac{f\cdot\mu\cdot a\cdot e}{a+b+\mu\cdot e}] \tag{B.6}$$

In this equation, f is the dynamic response factor that is 2.0 unless a lower factor can be proven and μ is the friction coefficient which is 0.8.
The load that is not carried by the nose gear is divided over the main gear struts. These main gear struts experience an additional drag load of 0.8 times the vertical load. The side load is 0, for both the MLG and the NG.

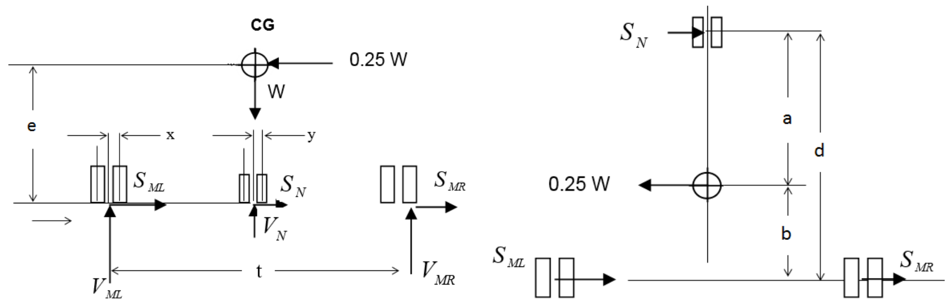
[21] This is true for multi axle gears. For single axle gear, a load factor of 2.0 should be applied according to Advisory Circular (AC) 25-491

TURNING LOADS

There are 6 turning scenarios defined in CS25.495 and CS25.511:

- Turning left with all tyres inflated
- Turning with one of the nose wheels deflated
- Turning left with the inner tyre of the right strut deflated
- Turning left with the outer tyre of the right strut deflated
- Turning left with the inner tyre of the left strut deflated
- Turning left with the outer tyre of the left strut deflated

Another 6 cases can be thought of, symmetric t the cases described above. Figures B.2a and B.2b give a free body diagram for the case where the inner tyre of the right strut is deflated.



(a) Top view

(b) Top view

Figure B.2: Free body diagram for turning conditions

Without any deflated tyres, equations B.7 to B.9 can be derived for the vertical load on the gear struts by evaluating the sum of moments on the nose gear and at the CG:

$$V_{MR} = \frac{W}{2}\left(-\frac{e}{t} + \frac{a}{d}\right) \tag{B.7}$$

$$V_{ML} = \frac{W}{2}\left(\frac{e}{t} + \frac{a}{d}\right) \tag{B.8}$$

$$V_N = \frac{Wb}{d} \tag{B.9}$$

Where V_{MR} and V_{ML} are the load on the right and left main strut, and t is the main gear track width. There is a side load of half the vertical load on all three struts; the drag load is 0.

The equations for the other 5 load cases are given in table B.1. In these equations, u is the distance between the main gear wheels and v is the distance between the nose gear wheels.

Table B.1: Loads on left and right main gear strut for different turning load cases.

Scenario	Left strut	Right strut
Flat nose wheel	$\left(\frac{W}{t}\right)\left(\frac{t}{2d}a + 0.25e + v\frac{b}{d}\right)$	$\left(\frac{W}{t}\right)\left(\frac{t}{2d}a - 0.25e - v\frac{b}{d}\right)$
Flat inner right wheel	$\left(\frac{W}{t+u}\right)\left(\frac{t}{2d}a + 0.25e\right)$	$\left(\frac{W}{t+u}\right)\left(\left(\frac{t}{2d} + \frac{u}{d}\right)a - 0.25e\right)$
Flat outer right wheel	$\left(\frac{W}{t-u}\right)\left(\left(\frac{t}{2d} - \frac{u}{d}\right)a + 0.25e\right)$	$\left(\frac{W}{t-u}\right)\left(\frac{t}{2d}a - 0.25e\right)$
Flat inner left tyre	$\left(\frac{W}{t+u}\right)\left(\left(\frac{t}{2d} + \frac{u}{d}\right)a + 0.25e\right)$	$\left(\frac{W}{t+u}\right)\left(\frac{t}{2d}a - 0.25e\right)$
Flat outer left tyre	$\left(\frac{W}{t-u}\right)\left(\frac{t}{2d}a + 0.25e\right)$	$\left(\frac{W}{t-u}\right)\left(\left(\frac{t}{2d} - \frac{u}{d}\right)a - 0.25e\right)$

NOSE WHEEL JAW LOADS

For nose wheel jaw, there are 2 cases described in CS25.499. In the first, the vertical forces on the struts are equal to the static case. There is an additional side load of 0.8 times the vertical load on the nose gear. This load should be balanced by the side loads on the main gears; this is shared evenly.
In the second case, it is assumed that brakes are used to lock 1 of the MLG struts. The airworthiness specifications for this case can be translated into figure B.3, where a top view and a side view are given:

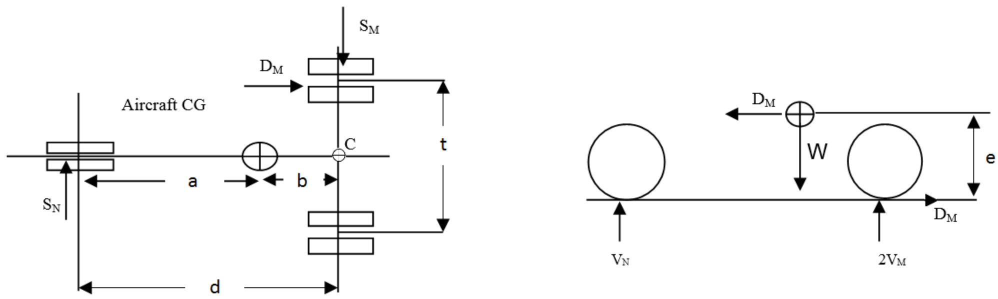


Figure B.3: Free body diagram for the second nose wheel jaw condition. Top view on the left and side view on the right.

The load on the main gear can be derived by taking moments around the nose gear, resulting in equation B.10.

$$V_M = \frac{aW}{2(d + 0.4e)} \tag{B.10}$$

The load on the nose gear is given by equation B.11. It can be found by summing the moments around the CG in the side view of figure B.3, and substituting equation B.10.

$$V_N = \frac{b + 0.4e}{d + 0.4e} W \tag{B.11}$$

The side load on the nose gear is obtained by evaluating the moment around point C in the top view of figure B.3 and is given below in equation B.12. The side load on the main gear strut balances the side load on the nose gear.

$$S_N = \frac{0.2t}{d} \frac{aW}{(d + 0.4e)} \tag{B.12}$$

The drag load on the nose gear can be taken as 0, while the drag load on the main gear should be taken as 0.8 times the vertical load.

PIVOTING LOADS

The vertical, drag and side loads during pivoting are equal to the static loading conditions, following CS25.503. There is however an additional moment that must be taken into account for the NLG, as computed by equation B.13.

$$M_Z = 0.4 V_m v \tag{B.13}$$

REVERSED BRAKING LOADS

The reversed braking load case is described in CS paragraph 25.507. In this case, the vertical load is equal to the static case. There is an additional drag load of 0.55 times that vertical load and no side load.

TOWING LOADS

For towing, the regulations provide a table of load cases that need to be checked in paragraph CS25.509; see table B.2. The first step in computing the towing loads is to calculate the required towing force. That towing force must be taken as 0.3 times the Maximum Ramp Weight (MRW) if the MRW is below 30,000 pounds; 0.15 times the MRW if the MRW is over 100,000 pounds and calculated according to equation B.14 if the MRW is in between these weights.

$$F_{tow} = \frac{6 \cdot MRW + 450,000}{70} [lbs] \tag{B.14}$$

Then, the side and drag loads on the struts need to be calculated according to table B.2. On top of that, the gears should also support the static vertical loads.

JACKING LOADS

For jacking, described in CS25.519, the vertical force is 1.33 times the static value; combined with a horizontal load of 0.33 times the static vertical load. Worst-case scenarios to check here are any horizontal force vectors pointing 45°inboard or outboard, and any multiple of 90°.

B.2. TOUCHDOWN LOADS

The different touchdown load cases are described in paragraphs 479 till 487 of CS-25. This section again treats every load case in a separate section. For all touchdown load cases in CS-25, it can be assumed that the lift provided by the wings equals the weight of the aircraft, unless some system drastically alters the lift generating capability of the aircraft.

LEVEL LANDING

The level landing case is illustrated in figure B.4 and described in CS25.479. For this condition, 4 load cases need to be checked:

1. Normal landing on 3 struts
2. Drift landing on 3 struts
3. Normal landing on main gears only
4. Drift landing on main gears only

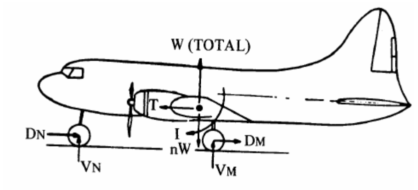


Figure B.4: Level landing condition [31].

For a normal landing, the vertical forces on the struts are the same as those in the static load case. The drag loads must be taken as 25 percent of these vertical loads and the side loads can be assumed to be zero. This

Table B.2: Load cases for aircraft towing [31].

Tow Point	Position	Load		
		Magnitude	No.	Direction
Main Gear		0.75*F_{tow} per main gear unit	1	Forward, parallel to drag axis
			2	Forward, at 30°to drag axis
			3	Aft, parallel to drag axis
			4	Aft, at 30°to drag axis
Nose gear	Swivelled forward	1.0*F_{tow}	5	Forward
			6	Aft
	Swivelled aft		7	Forward
			8	Aft
	Swivelled 45° from forward	0.5*F_{tow}	9	Forward, in plane of wheel
			10	Aft, in plane of wheel
	Swivelled 45° from aft		11	Forward, in plane of wheel
			12	Aft, in plane of wheel

holds for both the NLG and the MLG.
For the drift landing, the vertical load per strut is 75 percent of the load in the normal landing case. The drag load is then 40 percent of the vertical load and the side load 25 percent of the vertical load.
When there is a normal landing on the main gears, the weight of the aircraft is divided over both gear struts; both carrying half the weight. The drag load is again 25 percent of the vertical load and the side load is zero. The nose gear takes no loads.
When the drift landing on the main gears is considered, the vertical load is 75 percent of the vertical load in the normal 2-point load case. The factors for the drag load and side load are again 0.4 and 0.25. The nose gear takes no loads.

Tail down landing

In the tail down landing scenario, CS25.481, the load case is similar to the third load case described above: the main gears each carry half the aircraft weight and the drag load is equal to 25 percent of the vertical load. In this case, however, the aircraft must be assumed to be at its maximum allowable pitch angle θ.

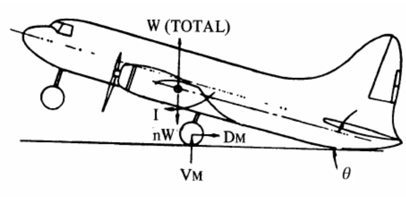


Figure B.5: Tail down landing condition [31].

As the aircraft has a significant angle with respect to the ground, the loads experienced by the struts have to be transformed according to equations B.15 and B.16. The side load in this case is zero. The nose gear is zero.

$$V_{strut} = \sin\theta \cdot D_M + \cos\theta \cdot V_M \tag{B.15}$$

$$D_{strut} = -\sin\theta \cdot V_M + \cos\theta \cdot D_M \tag{B.16}$$

One gear landing

The loads for the one gear landing should be taken equal to the normal landing on two wheels, where it is assumed that the unbalanced external load is reacted by the aircraft inertia. This load case is thus irrelevant.

Landing with side load

The side load condition is described in CS25.485. For this load case, it is assumed that the weight of the aircraft is divided over both main gear struts. There is an additional side load of 0.8 times the vertical load on one gear strut, and -0.6 on the other. The drag load is zero; see Fig. B.6:

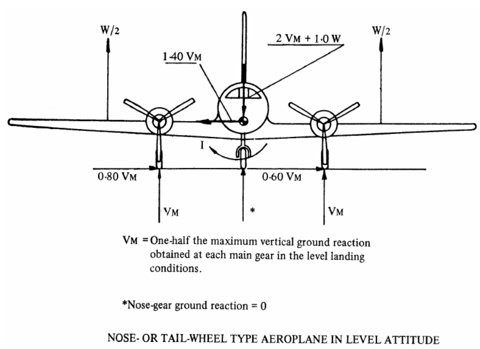


Figure B.6: Side load landing condition [31].

C

DETAILED DESCRIPTION OF LIT SOFTWARE

This appendix gives a more detailed description of the code structure. Before describing that in detail, some important design choices are explained. Why is object oriented programming (OOP) used, and what are its benefits; how does the `fmincon` routine in MATLAB work and why is it chosen?

C.1. NOTES ON OBJECT-ORIENTED PROGRAMMING

This section starts with an often made mistake about OOP. No: it will not save time during development of a program. It will save time when expanding a program, making changes and having several other people involved, however. So, why would you want to use OOP?

In normal procedural programming, everything is about data, variables, and functions. As the program grows in complexity, more lines of code are added, more functions are written and more variables are introduced. Often, pieces of code that are needed in more than 1 place are copied. As a program grows and grows (for this thesis, over 3000 lines of code was the moment my eyes opened), things become very messy and unstructured. Input and output handling to functions is a nightmare, data is sent from function to function via Excel sheets, a typo adds new fields to a structure and the overall structure of your program is just impossible to explain to someone else. You start to look for another way of writing your code and the answer is found: OOP.

So, what is OOP?

Object-Oriented Programming is all about classes. A class can best be explained as an intelligent MATLAB structure. A class has predefined properties (similar to MATLAB fields). If one tries to change the name of these properties, it is not allowed and an error is thrown. Also, if an incorrect number of inputs is given when initiating an instance of the class, this is not allowed. So in that respect, classes are much more robust than structures. They are ordered more rigidly and typos result in errors instead of unexpected outcomes. The properties of a class can be classes itself (for example, referring back to chapter 7, the trailing arm is of type `Cylinder`: a self-defined class), but also structures or any other type. The properties of a class are always available once the class is initiated, much like global variables. The difference is that the use of global variables is generally strongly discouraged, and that classes function properly.

The class is intelligent because it has methods. A method can be explained as a function that only works for that particular class. For example, there are different sizing functions for the different structural members used in this thesis (I-beams, Cylinders, Sockets, etc.). All have the same name: `findWeight`. But if a part is defined to be of the type `Cylinder`, it will never be possible to use another sizing function on it than the one belonging to `Cylinder`.

Helper functions, functions that are only needed for a particular class, can be defined below the class definition in the same file. This keeps the number of functions in the folder structure lower. Because the set-up of a class always follows the same structure, it is easy to understand for another programmer (assuming he/she knows a bit about OOP).

One major advantage of using OOP is that data is always available. For example, when sizing a member, the acting loads are needed. In procedural programming there are 2 options to obtain that data, when there is a separate function that calculates these loads. As a first option, the function calculating these loads could be called, with the relevant in- and output. That input should be gathered somewhere and many outputs of

these functions are probably not needed, but some are. It quickly becomes an administrative nightmare to keep track of the correct calling routines, especially if these functions are called at several places. Additionally, if this function takes a bit of time and needs to be evaluated for *every* member, this option is not feasible. The second option is to write the output to some file from which it can be read. This is not very efficient though, and these writing files quickly becomes a mess. At some point in time, another value is needed, shifting the locations of the other values. These locations are hard-coded and spread over different files. One forgets to edit one of these hard-coded values, and a nasty error is the result.
Using OOP, the load data is a property of the gear. Thus, when sizing the trailing arm, the loads data can be obtained by just typing, for example, `obj.loads.pointB`. No function calls or temporary writing files, but simple and clean coding.
Another advantage of OOP is that some properties can be fixed or hidden, by making the constants or by protecting their access. For example the gravitational constant g is stored as the constant property `gear.g`. So, the field `gear.g` cannot be overwritten by accidence at a later point. This in contrast to ordinary structures, where every field can be overwritten.
The third major advantage of OOP is the DRY principle. Don't Repeat Yourselves. This is where the inheritance principle of OOP should be mentioned. The basic principle is that no piece of code should be occurring twice. If it is needed twice, a parent and child function should be created. This has been done several times, but will be explained based on the class `ShockAbsorber`.
Both the retract actuator and the shock absorber are modelled as a cylinder. The main failure mechanism for both members is the burst pressure case, as a result of the compressed hydraulic fluid inside these members. Where the pressure inside the retract actuator is more-or-less constant[22], the pressure inside the shock absorber varies with the load on the wheels. There is an additional calculation needed to determine this pressure, but the principle strength calculations are similar for both members. The retract actuator is modelled as the class `Cylinder`.The additional calculations for the shock absorber call for a child class `ShockAbsorber`, which is constructed from the class `Cylinder`. It inherits all sizing methods available to `Cylinder`, but has its own method to calculate the needed pressure. The methods of the class `Cylinder` are available to the shock absorber and the retract actuator, while they have been specified just once. The methods of the class `ShockAbsorber` are only available to the shock absorber, and not to the retract actuator.
When using procedural programming, it is also possible to not repeat yourselves. This requires to make functions for every computation that should be done more than once. This typically results in many functions and does not make the understandability of the code any easier.

C.2. NOTES ON FMINCON

Throughout this thesis, the solver has been a black box. It was there to find the minimum; it was needed as the minimum of the objective function could not be found without it; but which solver was used and why was not properly addressed. This section goes into some more detail, but without using lots of mathematics.
Within the scope of this thesis, where finding a local minimum is the goal and the global minimum is not considered yet, the MATLAB optimization toolbox is needed. This toolbox offers several different pre-defined optimization routines. Depending on the problem at hand, multiple solvers may be chosen. A nice article is written by MATLAB to explain the different solvers and how to choose one[23]. It turns out that for a smooth but non-linear objective function, with smooth but non-linear constraints, `fmincon` is the recommended algorithm.
To run the optimization, `fmincon` needs an objective function with a single output, an initial vector containing all design variables, and constraints. These constraints are placed in a dedicated constraints function. Boundaries for the design variables are specified: physical dimensions should be larger than zero and cannot be infinite. This is a trivial example, but as the optimization is pure mathematics, the constraints need some attention. Another example of a trivial but critical constraint is that the outer radius should be larger than the inner radius [24]. If that is not told to MATLAB, very strange results can be the result.
The function `fmincon` can take 4 different types of constraints: linear equalities, non-linear equalities, linear inequalities and non-linear inequalities. All constraints have been specified as being non-linear, even if they are in fact linear. This is less efficient in terms of computation time, but it allows all constraints to be grouped in the same function. This makes the set-up of the program easier to be understood.

[22] it is dependent on the weight of the gear, which doesn't change too much in a single iteration
[23] google for "choosing a solver Matlab"
[24] Alternatively, one could specify the outer radius as the inner radius plus some thickness. In that case, the thickness must be positive

Table C.1: Runtime needed for the different algorithms of `fmincon`

Algorithm	Runtime [sec]	iterations [-]	function calls [-]
active-set	85.34	2	8
sqp	134.13	3	12
interior-point	653.45	6	57

The optimizer, the objective function and the constraint function are placed in a nested function. All calculations are performed in 1 function: `computeall`. This function is called once per iteration and sends all correct values to objective function, constraint function, and top-level optimization function. This reduces the runtime with a factor of two.
There are many options to be set while performing optimizations with `fmincon`. Most importantly, probably, is the choice of algorithm. Four different ones have been defined within MATLAB for `fmincon`: interior-point, trust-region-reflective, sqp and active-set. Not all algorithms work on all problems. For the same problem setup, some algorithms of `fmincon` are able to find an answer, and others are not. It can not be concluded in general which algorithm is best for a certain objective function. For the first problem presented in the results chapter, where a fixed gear configuration is used to find the optimal gear location, the four different algorithms were tried in the system-level optimizer. The results of that investigation are presented in table C.1. Where all algorithms found the same optimum, the time needed for it varied significantly. No explanation can be given for the fact that the active-set algorithm performs best.
The trust-region-reflective algorithm could not be used, as the gradient of the objective function needs to be specified for this algorithm. The gradient is not available for this problem.

C.3. STRUCTURE OF THE SOFTWARE FOLDER

The folder *Software* contains 7 sub-folders: CLI_tools, Data, MATLAB, Parts_Catia, Parts_VLM, READ_NOTES_IN_THIS_FOLDER, Results and SubMechs. When operating the software, this folder must be placed in a location *that has no spaces in its pathname*, or error will occur. The standard local folder My Documents, should thus be avoided. This section describes the files contained in each folder, while sections C.4 and C.5 describe the written MATLAB files in some detail.
The sub-folder CLI_tools contains a windows batch file that allows the VLM model to be ran fully automatic. The batch file opens VLM, initiates the model, runs the load cases that are specified in the solution manager of VLM, and closes VLM again. This batch file can be opened directly from MATLAB, once the correct path is specified.
The folder Data contains two Excel sheets that are needed to set up the VLM model. The file `BASE_geometryV4_Original` contains the initial coordinates of all hardpoints. That file is never edited during an optimization run, to keep track of the initial design. The file `BASE_geometry` is changed during each iteration. The values from `BASE_geometryV4_Original` are obtained, the global optimizer of MATLAB modifies them, and writes them in the file `BASE_geometry`. From that file, the VLM model is initiated.
The folder MATLAB has many files. These are discussed separately in the next sections.
The folder Parts_VLM contains three CATIA parts: two related to the definition of the global coordinate system and one that defines the interface points between the airframe structure and the landing gear.
The folder READ_NOTES_IN_THIS_FOLDER contains two m-files that should be placed in the *installation* folder of MATLAB. The files are self-defined classes which are needed to read the VLM results and convert it to a MATLAB structure. If they are placed in the MATLAB folder however, they are somehow not found; an error is thrown when running the code. Placing them in the installation folder of MATLAB solves this problem. The folder contains the results from the VLM model. It has a separate sub-folder for each loadcase. All sub-folders are investigated by the function `InvestigateVLMResultsfile`, which selects the wanted results and creates a table from it. This table is available for further analysis, and is also written in an Excel file: StaticLoadCase.xlsx. The results table has around 110 columns.
The first few contain the load case number (1 column), the length of the shock absorber (1 column), the x,y and z coordinates of points A and D for the left and the right gear (3x2x2=12 columns) and the input forces on the nose gear, the left gear and the right gear (6x3=18 columns). The VLM model allows the contraction and extension of the shock absorber. With the output in the second column it can be checked if that is done correctly; the next 12 columns allow the calculation of the orientation of the trailing arm in the global reference frame. For other components this is done from the input data, but the trailing arm moves when the shock

absorber extends or retracts. Therefore, its coordinates are output data.
The other columns define the output loads. In the first 6, the forces that are needed to fix the CG are given. After that, the loads in point A to J follow; first for the left gear and then for the right gear.
The folder SubMechs contains the parts definition of the VLM model. The right gear is mirrored from the left gear, by making all y coordinates postive. Each part is simply defined as a line between two hardpoints. The definition of these hardpoints is contained in the Data folder; they are linked to each other.

C.4. INPUT FILES

The MATLAB folder contains two more input files: inputGlobal7000.xslx and inputSizes.xslx. The first contains general aircraft data such as the maximum allowable roll and pitch angle, tyre geometry, tail height and wingspan. It also defines the different aircraft weights and their corresponding forward and aft CG positions. As a third set, this file defines the initial, minimum and maximum global strut positions. The file *inputSizes.xslx* contains initial sizing parameters for the different LG components and specifies upper and lower bounds for these design variables. This file has three sheets. The first contains input data for cylindrical components, the second contains data for H-beams, and the third specifies some parameters for the shock absorber.

C.5. MATLAB FILES

Figures 7.2 and 7.3 list eight MATLAB classes. Each of these classes is defined in one MATLAB file, that has the same name as its class. With eight class files and 11 MATLAB scripts, this leaves three scripts. The first of these is `runThisFile.m`, which should be ran to execute the tool and get an answer. It is explained in more detail in section C.5.3. The function `computeBU` estimates the ultimate bending stress of a cylindrical member according to figure 6.3. It is needed in several files and therefore placed apart. The function `initialGeom` is placed apart for the same reason. This function contains the hardpoint coordinates. For the static shock absorber position this data is contained in the Data folder; for the extended shock absorber position, it is found in the Results folder.
The other files can be divided into two groups: one with `StructuralMember` as its parent and one with `LGSystem` as its parent. The connection between the two is that all components of the POC gear are modelled as some `StructuralMember`.

C.5.1. MATLAB FILES FOR THE LANDING GEAR

This section describes all files that inherit from the class `LGSystem`. A short description of what these files do was already given in section 7.2, while the broader set-up of these files was described in section 2.1 This section is more elaborate than the main body of this thesis.
A sub-folder BlockA is present in the Matlab folder. Its files are of similar set-up as the main files, only the analytical model is used to find the forces in the hardpoints and not the VLM model. These files will therefore not be described in any more detail.

LGSYSTEM.M

The parent MATLAB file is `LGSystem.m`. It has a method to calculate the ground reaction forces acting on a LG strut: deriveStrutLoads. The position of the gears, the gear layout, and the certification type are needed as input. Besides these parameters, the aircraft parameters are gathered and some constants are defined. Regarding certification type, only CS-25 is fully tested at the moment but the file is set up to allow evaluation of CS-23, CS-27 and/or CS-29 as well, however.
The helper function `strutLoadsTricycle` calculates the loads on the gear struts, depending on the position of these struts. As described in chapter 4, all load cases should be considered, as it is not known beforehand which load cases are critical. The outcomes of this function are 4 MATLAB structures: `mleft`, `mright`, `n` and `cg` for the loads acting at the left main gear, the right main gear, the nose gear strut and the CG, respectively. Each structure contains 41 fields: the different load cases described in chapter 4 and Appendix B. The field `m.Taildown` for example contains the load cases for the tail-down load case of section B.2. Each field is a structure itself, containing the 10 sub load cases that were explained in section 4.3. How these 10 sub load cases are obtained, is explained below.
Each field has again 2 sub-fields: aft and fwd. Thus, taking the field `m.Taildown` again as an example, it

contains 2 subfields: `m.Taildown.Aft` and `m.Taildown.Fwd`. These fields contain the loads for the most forward and the most aft CG position. This is initiated with the first loop in the code, in the lines

```
cg = {'Aft' 'Fwd'};
for pos=1:2                   % This loop is present such that every equation is evaluated at the
                              % most forward CG and the most aft CG
    if pos==1
        A = aAft; B = bAft; E = constantValues(8);
    else
        A = aFwd; B = bFwd; E = constantValues(9);
    end
```

Finally, for every load case and for every CG position, 5 tyre conditions should be evaluated: the 50-50, 40-60, 60-40, 0-60 and 60-0 sub load cases as explained in section 4.3. As a field name in MATLAB cannot start with a number, these have been stored in the field `Sym`, `aSym1`, `aSym2`, `Flat1` and `Flat2`, respectively. For the ground handling cases, the evaluation of these 5 tyre condition is done in two steps. Specific requirements for the flat tyre load case are mentioned in CS25.511. These requirements are implemented in the function `strutLoadsTricycle`, in the loop that is initiated with the following piece of code:

```
 for cond=1:2             % This loop is present such that every equation is evaluated for flat tyres
                          % and for normal tyres.
        if cond==1
            mu=factors.mu(1);
            tow=factors.tow(1);
            reversed=factors.reversed(1);
            %disp('Below results are for all inflated tyres');
        else
            mu=factors.mu(2);
            tow=factors.tow(2);
            reversed=factors.reversed(2);
            %disp('Below results are for 1 flat tyre');
        end
```

This results in two sub load cases: the 50-50 and the 0-60 load case. The remaining three load cases are computed in the helper function `constructLoadCasesGR`. An example input of this function is `m.Static.Aft`, containing both the 50-50 and the 0-60 sub load case. Per main load case, there are now ten different fields with loads. For the example of `m.Static`, these are:

- `m.Static.Fwd.Sym`
- `m.Static.Fwd.aSym1`
- `m.Static.Fwd.aSym2`
- `m.Static.Fwd.Flat1`
- `m.Static.Fwd.Flat2`
- `m.Static.Aft.Sym`
- `m.Static.Aft.aSym1`
- `m.Static.Aft.aSym2`
- `m.Static.Aft.Flat1`
- `m.Static.Aft.Flat2`

A similar thing happens for the touchdown load cases, but these load cases do not have a separate computation for the flat tyre conditions in the helper function `strutLoadsTricycle`. This is all done in the function `constructLoadCasesTD`. The sub-fields of the field `m.Taildown` are the same as those of `m.Static`. In total, 410 load cases are now considered.
It is assumed in the function `strutLoadsTricycle` that the loads act in the middle of the wheel axle. For the asymmetric and flat load cases, this results in a moment in that point. This is accounted for in the functions `constructLoadCasesTD` and `constructLoadCasesGR`.

LGStrut.m

`LGStrut` is a child of `LGSystem` and thus inherits its properties and functions. In `LGStrut`, the loads at different hardpoints of the landing gear are computed, based on the loads per strut. This is done via a call to the VLM program.
The VLM program is set up via the file `BASE_geometry`, so this file is overwritten at every iteration. The initial configuration is contained in another excel file, which is opened and read in the function `LGStrut.m`. The values of the design vector are then added to the coordinates and the resulting coordinates are written in the file `BASE_geometry`.

After running the VLM model (this requires browsing through several folders; this is all hardcoded so do not change the folder structure!), the forces need to be rotated to their part-local axis system. That is done with standard transformation matrices in the helper functions transformY and transformZ. The loads in point F are also obtained.
These loads in the part-local axis system are then used to do the stress calculations in TAGear.

TAGEAR.M

In TAGear, all components of the POC gear are modelled. Since several members are of the same type, a class has been created for all main structural components; see section C.5.2. It is specified for every component which class they should be. This is done in the following MATLAB lines. The trailing arm, for example, can only be of type Cylinder.

```
properties
        trailingArm@Cylinder;
        wheelAxle@Cylinder;
        mainFitting@Cylinder;
        retractActuator@Cylinder;
        shockAbsorber@Cylinder;
        foldingSideBraceUp@IBeam;
        foldingSideBraceLow@IBeam;
        trunnion@Socket;
        pistonSA@ShockAbsorber;
        orientation
        lenghtmembers
    end
```

The component sizing is done based on the loads in the hardpoints that can be calculated in the parent class TAGear. As every component is a class of itself, it has an optimization routine. The only needed input for this routine are the loads, the material, the initial geometry, and the boundaries for the geometry. This sizing function will be described when the different classes are explained.

C.5.2. MATLAB FILES FOR STRUCTURAL MEMBERS

To find the optimal properties of each gear component, different structural member classes are defined. The properties of these structural members are the subjects of the following section.
The (component-level) optimization function follows a similar set-up for all classes. Every sizing function has three nested functions: one to obtain the part weight, one to evaluate the constraints, and one to perform calculations. In the sizing functions, first some properties that are needed in the different nested functions are initialized. Then the initial values of the design vector are read, as well as the boundaries. This is enough to set the fmincon routine up.
The main calculations are performed in the nested function computeall [25]. For a particular set of design variables, this function calculates the margin of safety of the part; the critical load case and the weight. It uses the equations of chapter 6 to do so. This calculation is performed once per iteration. The weight is needed in the objective function; the margin of safety is needed in the constraints function. The critical load case is wanted as output.

STRUCTURALMEMBER.M

There are three principle structural members in the landing gear: cylinders, H-beams, and a socket. One could argue that a socket is a special type of a cylinder, but as it requires completely different calculations (see chapter 6), it has been modelled as a different part.
All structural members share some properties, and these are stored in the class StructuralMember. All structural members need material properties such as density, yield strength, etc. Three different materials are modelled: 300M steel that is often used in landing gear parts, a more standard steel and an often-used aluminium alloy. Furthermore, all structural members will have a length, an initial geometry, a final geometry, a margin of safety, and upper and lower bounds for its design variables. This is defined in the file StructuralMember.m

[25] Sometimes this function has a different name, but it always starts with compute

CYLINDER.M
The cylinder is a child of `StructuralMember` and inherits its properties. Additional properties are the loads acting on it, the hydraulic pressure inside it and the name of the member.
Based on the loads acting on a cylinder and its geometry, a sizing can be performed. This is done according to 3 different routines: `weightExtForces`, `weightPresHydraulic` and `weightVertForce`. Depending on the loads acting on the member, the relevant function is called.
If there are forces acting in the local x, y and z direction, the sizing method described in section 6.2 is used in the function `weightExtForces`, following a component-level optimization. Based on the geometry of the cylinder, the margin of safety is computed for all 410 load cases. The critical load case, the one resulting in the lowest margin of safety, is remembered. The geometry resulting in the lowest weight (given the constraints and the upper and lower boundaries) is the output of this function.
If there are hydraulic forces, a similar optimization is performed in the function `weightPresHydraulic`. This optimization is based on the physics described in section 6.3. One difference is that there is no critical load case. The sizing is purely done based on the pressure of the hydraulic fluid inside the component. This pressure is depending on the aircraft weight (for the shock absorber) and the LG weight (for the RA).
For the shock absorber and the retract actuator, it is assumed that there is a compressive axial force acting on them. Thus, the buckling criteria (described in section 6.1 for an I-beam, but similarly applicable to a cylinder) should be checked. That is done in the function `weightVertForce`.
Depending on the acting forces, the relevant functions are computed. The main fitting, for example, does not have any hydraulic fluids inside it, so `weightPresHydraulic` is not evaluated for this part. If the sizing is done for two different sets of forces, the heaviest result is selected.
Specific constraints are that the computed margin of safety must be larger the specified MoS and that the outer radius of the member must be larger than the inner radius, with a minimum wall thickness of 2mm.

HBEAM.M
The set-up of `HBeam` is similar to `Cylinder`. It has only one input: the axial force on the member. Other properties are inherited from `StructuralMember`. As the force on an H-Beam is always axial, there is only one sizing function: `findWeight`. It is based on the physics described in section 6.1.
Specific constraints are that the outer width must be larger than the inner width, that the web thickness must be smaller than the beam height, the that cut-out height cannot be larger than the beam height - with a margin of 4mm on both sides, and that the cut-out thickness cannot be larger than the web thickness - with a margin of 2mm. Again, the computed margin of safety must be larger or equal to the specified MoS.

SOCKET.M
The file `socket.m` again has a similar set-up as the above two classes. Inputs include the acting loads, the length of the pins inside the socket and the geometry of the socket. The other design variables are inherited from `StructuralMember`.
The equations used to size a socket were given in section 6.4 and are implemented in this file. Again, a component-level constraint is that the computed margin of safety must be larger or equal to the computed margin of safety. Additionally, the thickness of the socket must be positive.

SHOCKABSORBER.M
Sizing of the shock absorber is based on the method described in Currey [4]. First, the needed piston diameter is computed. Design variables for this calculation are the shock absorber efficiency, the tyre efficiency, the load factor, the maximum vertical touchdown velocity, the tyre stroke, the static pressure, the pressure when the shock absorber is fully extended and the pressure when the shock absorber is fully compressed.
As the shock absorber is a child of the class `Cylinder`, the thickness of the shock absorber piston can be computed with the function `findWeight` that is specified for this parent class.

C.5.3. RUNFILE
In the runfile, the first step is to find the optimal location of the gears under the aircraft. This is done with the analytical model from the folder BlockA. The coordinates of the optimal location are saved in the file `BASE_geometry.xlsx`.
The next step is to vary the gear layout. To do so, an object of `TAGear` is initiated and a design vector is defined, as well as upper and lower boundaries.
The boundaries are specified in such a way, that no global constraints are needed. There is thus no constraint

function.
An additional function `outfun` ensures that there is some real-time information while the optimizer runs. A figure is created that plots the current values of the design vector, the current value of the objective function and the maximum constraint violation.

Bibliography

[1] S. Chai and W. Mason, *Landing Gear Integration in Aircraft Conceptual Design*, Tech. Rep. MAD 96-09-01 (NASA Ames Research Centre, 1997).

[2] N. Heerens, *Landing gear design in an automated design environment*, Master's thesis, Delft University of Technology (2014).

[3] S. Cumnuantip, M. Spieck, and W. Krueger, eds., *An approach for sizing and topology optimization integrating multibody simulation* (46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference, 2005).

[4] N. Curry, *Aircraft Landing Gear Design: Principles and Practices*, 1st ed. (American Institute of Aeronautics and Astronautics, 1988).

[5] J. Roskam, *Airplane Design part IV: Layout design of landing gear and systems*, 2nd ed. (DARcorporation, 1989).

[6] Kable Intelligence Limited, http://www.army-technology.com/projects/lynx-mk7/lynx-mk72.htmL (2015), accessed 21-apr-2015.

[7] Fokker B.V, http://www.fokker.com/Landing_Gear_Programs (2015), accessed 21-apr-2015.

[8] The Aviation History Online Museum, http://www.aviation-history.com/boeing/b47.html (2015), accessed 21-apr-2015.

[9] J. Heeren, *Landing gear design handbook for engineering* (Fokker Landing Gear B.V., 1997).

[10] P. Papalambros and D. Wilde, *Principles of optimal design: modeling and computation* (Cambridge university press, 2000).

[11] P. E. Gill, W. Murray, and M. H. Wright, *Practical optimization*, Vol. 5 (Academic press London, 1981).

[12] C. B. Chapman and M. Pinfold, *Design engineering—a need to rethink the solution using knowledge based engineering,* Knowledge-based systems **12**, 257 (1999).

[13] T. van den Berg, *Harnessing the potential of Knowledge Based Engineering in manufacturing design*, Ph.D. thesis, Delft University of Technology (2013).

[14] S. Cooper, I.-s. Fan, and G. Li, *Achieving competitive advantage through knowledge-based engineering: a best practice guide* (Prepared for the Dept. of Trade and Industry by Dept. of Enterprise Integration, Cranfield University, 1999).

[15] M. Stokes, M. Consortium, *et al.*, *Managing engineering knowledge: MOKA: methodology for knowledge based engineering applications* (Professional Engineering Publ., 2001).

[16] G. la Rocca, *Knowledge based engineering: Between ai and cad. review of a language based technology to support engineering design,* Advanced Engineering Informatics **26(2)**, 159–179 (2012).

[17] S. A., J. Nayfeh, and R. Zarda, *An integrated design and optimization environment for industrial large scaled systems,* Research in Engineering design **16**, 86 (2005).

[18] S. Yi, J. Shin, and G. Park, *Comparison of mdo methods with mathematical examples,* Structural and Multidisciplinary Optimization **35**, 391 (2008).

[19] N. Tedford and M. J.R.R.A., eds., *On the common structure of MDO problems: a comparison of architectures* (11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, 2006).

[20] A. Keane and P. Nair, *Computational approaches for aerospace design: the pursuit of excellence* (John Wiley & Sons, 2005).

[21] J. Martins and C. Marriage, eds., *An object-oriented framework for multidisciplinary design optimization* (3rd AIAA Multidisciplinary Design Optimization Specialist Conferernce, 2007).

[22] I. Kroo, *Mdo for large-scale design,* in *Multidisciplinary design optimization: state-of-the-art*, ICASE/-NASA Lanley Workshop on Multidisciplinary Design Optimization (SIAM, 1997) pp. 22–44.

[23] K. Deb, *Multi-objective optimization,* in *Search methodologies* (Springer, 2014) pp. 403–449.

[24] M. v. Tooren, G. L. Rocca, and T. Chiciudean, *Advance Design Methodologies Course Book* (TU Delft, 2013).

[25] A. Myklebust, S. Jayaram, and P. Gelhausen, *Acsynt-a standards-based system for parametric computer aided conceptual design of aircraft,* AIAA Paper , 92.

[26] G. La Rocca, T. Langen, and B. Y.H.A., eds., *The design and engineering engine. Towards a modular system for collaborative aircraft design* (28th international congress of the aeronautical sciences, 2012).

[27] S. Cumnuantip, ed., *Landing gear conceptual design and structural optimization of a large blended wing body civil transport aircraft* (5th Challenges in European Aerospace Conference, 2015).

[28] E. Torenbeek, *Synthesis of subsonic airplane design: an introduction to the preliminary design of subsonic general aviation and transport aircraft, with emphasis on layout, aerodynamic design, propulsion and performance* (Springer Science & Business Media, 1982).

[29] Leibowitz, Bruce, http://http://flightlineaviationmedia.com/planespotting/2-engines-tail/ (2015), accessed 13-jul-2015.

[30] F. A. Circular, *150/5300-13,* Airport Design (1994).

[31] E. CS25, *Certification specifications and acceptable means of compliance for large aeroplanes,* Amendment **15** (2014).

[32] E. CS23, *Certification specifications for normal, utility, aerobatic, and commuter category aeroplanes,* Amendment **3** (2012).

[33] E. CS27, *"certification specifications for small rotorcraft,* Amendment **3** (2012).

[34] E. CS29, *"certification specifications for large rotorcraft,* Amendment **3** (2012).

[35] T. H. G. Megson, *Aircraft structures for engineering students* (Elsevier, 2012).

[36] E. F. Bruhn, R. Bollard, *et al.*, *Analysis and design of flight vehicle structures* (Jacobs, 1973).

[37] Federal Aviation Authority, *Metallic Materials Properties Development and Standardization* (Batelle Memorial Institute, 2013).

[38] C. Niu, *Airframe structural design: practical design information and data on aircraft structures* (Conmilit Press, 1988).

[39] N. M. Alexandrov and S. Kodiyalam, *Initial results of an mdo method evaluation study,* AIAA paper **4884**, 1998 (1998).

[40] J. Sobieszczanski-Sobieski, *Optimization by decomposition: a step from hierarchic to non-hierarchic systems,* Recent Advances in Multidisciplinary Analysis and Optimization , 51 (1988).

[41] J. Sobieszczanski-Sobieski, J. S. Agte, and R. R. Sandusky, *Bilevel integrated system synthesis,* AIAA journal **38**, 164 (2000).

[42] G. Kempf, ed., *Development of undercarriage design loads* (Advisory Group for Aerospace Research and Development, 1991).

[43] D. Thorby, J. Johnson, A. Auld, H. Newman, and M. Brooker, eds., *The Special Requirements of a VSTOL Aircraft* (Advisory Group for Aerospace Research and Development, 1991).

[44] R. Yntema and B. Milwitzky, *An impulse-momentum method for calculating landing-gear contact conditions in eccentric landings,* (1952).

[45] W. Krueger, *Integrated Design Process for the Development of Semi-Active Landing Gear for Transport Aircraft,* Ph.D. thesis, University of Stuttgart (2000).

[46] W. Krueger and M. Spieck, *Interdisciplinary landing gear layout for large transport aircraft,* (1998).

[47] N. standardization Agency, *"standardization agreement 4674 - unmanned aerial vehicles systems airworthiness requirements (usar),* Amendment **1** (2009).

[48] Federal Aviation Authority, *Taxi, takeoff and landing roll design loads.*

[49] D. Freund, D. R. McKissack, L. C. Hanson, and H. Brodman, *Dynamic taxi, take-off and landing roll analyses for large business jet aircraft,* AIAA paper **1526** (2000).