TECHNISCHE UNIVERSITÄT MÜNCHEN
Lehrstuhl für Flugsystemdynamik

Development of a Framework for the Solution of High-Fidelity Trajectory Optimization Problems and Bilevel Optimal Control Problems

Florian Fisch

Vollständiger Abdruck der von der Fakultät für Maschinenwesen der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs

genehmigten Dissertation.

Vorsitzender: Univ.-Prof. Dr.-Ing. habil. Boris Lohmann

Prüfer der Dissertation:
1. Univ.-Prof. Dr.-Ing. Florian Holzapfel
2. Univ.-Prof. Dr. rer. nat. Matthias Gerdts
Universität der Bundeswehr München

Die Dissertation wurde am 21.10.2010 bei der Technischen Universität München eingereicht und durch die Fakultät für Maschinenwesen am 04.02.2011 angenommen.

Table of Contents

List of Figures

List of Tables

Nomenclature

The following tables contain a survey of the formula symbols and the abbreviations utilized most frequently throughout the thesis. Common symbols as well as indices are written in italics, while matrices and vectors are shown in bold. Furthermore, vectors with a physical meaning in the three-dimensional Euclidean space are marked with an arrow on top of the symbol.

For any Euclidean vector $\vec{\mathbf{X}}$, the following declaration scheme is applied:

$$\left(\vec{\mathbf{X}}^{\text{ReferencePoint}}_{\text{Type of Motion/Source of Force/Moment}}\right)^{\text{Reference Frame}}_{\text{Notation Frame}}$$

LATIN CAPITAL LETTERS		
SYMBOL	EXPLANATION	UNIT
A	Aerodynamic force	N
$C_{i,j}$	Aerodynamic derivative of index i due to index j	-
D	Aerodynamic drag force	N
$\vec{\mathbf{F}}$	Force vector	N
G	Gravitational force	N
$\vec{\mathbf{H}}$	Angular momentum	Nms
$\mathbf{I}$	Inertia tensor / identity matrix	[kg·m²] / -
K	Feedback gain	-
L	Aerodynamic lift force	N
L_i	Roll moment derivative due to index i	-
$\mathbf{M}$	Transformation matrix	-
$\vec{\mathbf{M}}$	Moment vector	Nm
M_i	Pitch moment derivative due to index i	-
M_μ	Meridian radius of curvature	m
N_i	Yaw moment derivative due to index i	-
N_μ	Radius of curvature in the prime vertical	m
P	Propulsive force	N
S	Reference area	m²
T	Thrust / time constant / temperature	N / - / °C

$\vec{\mathbf{V}}$	Kinematic velocity vector	m/s
X_i	Force derivative in x-direction due to index i	-
Y / Q	Aerodynamic side force	N
Y_i	Force derivative in y-direction due to index i	-
Z_i	Force derivative in z-direction due to index i	-

LATIN SMALL LETTERS		
SYMBOL	EXPLANATION	UNIT
a	Length of semi-major axis	m
$\vec{\mathbf{a}}$	Acceleration	m/s²
b	Length of semi-minor axis / wing span	m / m
e	First eccentricity	-
f	Flattening	-
g	Gravitational constant	m/s²
m	Aircraft mass	kg
n	Load factor	-
n_C	Number of controls	-
n_S	Number of states	-
n_{var}	Number of variables	-
p	Roll rate	rad/s
$\vec{\mathbf{p}}$	Linear momentum	Ns
q	Pitch rate	rad/s
$\bar{q}$	Dynamic pressure	kg/(m·s²)
q_i	Quaternions (i = 0,1,2,3)	-
r	Yaw rate	rad/s
$\vec{\mathbf{r}}$	Position vector	m
s	Laplace variable / half wing span	- / m
t	Time	s
u	Control variable	-
u	Wind speed component into x-direction	m/s
v	Wind speed component into y-direction	m/s
w	Wind speed component into z-direction	m/s
x	State variable	-

y	Output variable	-

GREEK LETTERS		
SYMBOL	EXPLANATION	UNIT
α	Angle of attack	rad
β	Angle of sideslip	rad
γ	Flight-path climb angle	rad
δ_T	Thrust lever position	-
ζ	Rudder deflection / relative damping ratio	rad / -
η	Elevator deflection	rad
Θ	Aircraft pitch angle	rad
κ	Curvature / engine mounting angle	- / rad
λ	Geodetic longitude	rad
μ	Flight-path bank angle / geodetic latitude	rad / rad
ν	Pseudo-control	-
ξ	Aileron deflection	rad
ρ	Density	kg/m³
σ	Engine mounting angle	rad
τ	Normalized time	-
Φ	Aircraft bank angle	rad
χ	Flight-path course angle	rad
Ψ	Aircraft azimuth angle	rad
$\vec{\boldsymbol{\omega}}$	Angular velocity vector (rotation rate)	rad/s
$\boldsymbol{\Omega}$	Rotation matrix	rad/s
ω_0	Natural frequency	Hz

INDICES	
SYMBOL	EXPLANATION
*	Sub-optimal solution
~	Scaled quantity / dimensionless quantity
+	Pseudo-inverse
0	Initial
A	Aerodynamic Frame / aerodynamic motion

a	Set of equality and active inequality constraints
AUX	Auxiliary
B	Body-Fixed Frame
CMD	Commanded value
D	Aerodynamic drag
DR	Dutch-roll
E	Earth-Centered Earth-Fixed Frame (ECEF)
EF	Error Feedback
eq	Equality
f	Final
FP	Footpoint
G	Centre of Gravity / gravitational force
$Gyro$	Gyroscopic
hor	Horizontal
I	Earth-Centered Inertial Frame (ECI)
I	Inlet
$ineq$	Inequality
ini	initial
INV	Simulation model with inversion controller
K	Kinematic Flight Path Frame / kinematic motion
$\bar{K}$	Intermediate Kinematic Flight Path Frame
kin	Kinetic
L	Aerodynamic lift / rolling moment
l	Rolling moment when used in a derivative
LB	Lower bound
loc	Local
LTF	Simulation model with linear transfer functions
M	Pitching moment
m	Pitching moment when used in a derivative
MS	Multiple shooting
N	Navigation Frame / yawing moment
n	Yawing moment when used in a derivative
NLI	Full non-linear 6-DoF simulation model with non-linear inner loop

O	North-East-Down Frame (NED) / outlet
opt	Optimal
OPT	Optimization result
P	Propulsive force
p	Roll rate
PC	Path constraint
PM	Point-mass simulation model
pot	Potential
q	Pitch rate
r	Yaw rate
REF	Reference value
Rot	Rotor Reference Frame
SIM	Simulation result
sol	Solar
SP	Short-period
SSM	Simulation model with linear state-space models
T	Total force / total moment / transpose
UB	Upper bound
W	Wind motion
WP	Waypoint
Y / Q	Aerodynamic side force

ABBREVIATIONS	
SYMBOL	EXPLANATION
A/C	Aircraft
abs	Absolute value
DoF	Degrees of Freedom
DR	Dutch-roll
ECEF	Earth-Centered Earth-Fixed Frame
ECI	Earth-Centered Inertial Frame
EOT	Equation of time
eps	Machine precision
ISA	International standard atmosphere

max	Maximum value
MIMO	Multi-input multi-output
min	Minimum value
N	Navigation Reference Frame
NED	North-East-Down Reference Frame
Rot	Rotor Reference Frame
SAT	Saturated
SP	Short-period
T	Trajectory Frame
vec	Matrix vectorization
WGS84	World Geodetic System 1984

1

Introduction

1.1 Optimal Control and Trajectory Optimization

From a historical perspective, the theory of optimal control evolved from the theory of calculus of variations. The origins of calculus of variations date back to the 17th century when *Johann Bernoulli* posed the famous Brachistochrone problem to the mathematicians at that time. In the 18th century, it was up to *Leonhard Euler* and *Joseph-Louis Lagrange* to develop the principle framework of the theory of calculus of variations, resulting in the first order necessary optimality conditions called the Euler-Lagrange equations. Only if the Euler-Lagrange equations are fulfilled, the corresponding function is an *extremal* function and the respective functional reaches a maximum value, a minimum value or a saddle point. A comprehensive treatment of the theory of calculus of variations can be found e.g. in Refs. [Bolza, 1909] and [Bryson, 1998].

The theory of optimal control distinguishes from the calculus of variations in that it separates the control variables from the state variables. An introduction to the theory of optimal control is given e.g. in Ref. [Kirk, 2004]. With regard to continuous time optimal control problems, the necessary optimality conditions are provided by the maximum principle that was derived by the Russian mathematician *Lev S. Pontryagin* and his co-workers in the middle of the 20th century (see Ref. [Pontryagin, 1962]). The maximum principle results in a two-point boundary value problem involving the original states of the dynamic system as well as additional adjoint states and constitutes the basis for the solution of optimal control problems either by analytical or indirect methods.

Optimal control is nowadays applied in many different disciplines, like e.g. economics, medicine, chemistry, robotics, etc. Many optimal control applications can also be found in the field of aerospace engineering. Here, the applications range from atmospheric flight problems to space flight problems as well as from civilian to military applications. In the majority of cases the task is to find a flight trajectory and the corresponding control inputs for a flight mission such that a certain objective value becomes minimized or maximized. Therefore, optimal control in the field of aerospace engineering is often referred to as *trajectory optimization*. Trajectory optimization has the goal…

…to find an optimal control history and the corresponding optimal trajectory for a ***dynamic system*** *that are fully* ***compatible*** *with the regarded dynamic system itself, minimize a certain* ***objective function*** *(cost functional), meet* ***initial boundary conditions***, ***interior point conditions*** *and* ***final boundary conditions*** *and fully satisfy given* ***equality and inequality constraints***.

At this, the dynamic system is usually represented by a set of ordinary differential equations for the computational solution of the optimal control problem. The set of ordinary differential equation is referred to as simulation model, since a simulation of the flight path can be done by integration of the ordinary differential equations given the controls and the initial state values.

Besides the analytical and the indirect methods that are mainly based on *Pontryagin's* maximum principle, a vast number of direct methods exists for the solution of trajectory optimization problems. With direct methods, the *infinite dimensional* continuous time optimal control problem is first transcribed into a *finite dimensional* parameter optimization problem by approximating the states and / or the controls by discrete functions. Then, non-gradient based algorithms like e.g. genetic algorithms or gradient based algorithms are applied for the solution of the parameter optimization problem. For a comprehensive overview on direct methods for optimal control see e.g. Refs. [Betts, 1998] and [Betts, 2001].

Especially for gradient based algorithms, the optimization procedures suffer from small convergence areas and the convergence properties of the applied algorithm depend heavily on the quality of the initial guess that is used for the control variables and the state variables to start the optimization. If the initial guess is too far away from the optimal solution, the optimization algorithm might fail to converge and no optimal solution is found at all or the algorithm gets stuck in a local minimum or maximum. This is especially true if trajectories are to be optimized that are highly dynamic or if complex dynamic systems are considered.

Over the last decades, a vast number of trajectory optimization problems in aerospace engineering has been solved where the majority of the trajectory optimization problems dealt with point-mass models (see e.g. Refs. [Bulirsch, 1991a], [Bulirsch, 1991b], [Miele, 1986], [Ringertz, 2000], [Schultz, 1987] and [Grimm, 1990]): There, the aircraft is modeled as a point-mass and its motion is optimized. The attitude of the aircraft as well as the rotational motion are not incorporated in the simulation models. Although the resulting trajectories are optimal for the point-mass model, the full dynamic order of the flight system with its attitude and rotational dynamics is not accounted for and it is not guaranteed that the calculated optimal trajectory could be followed by the aircraft in reality. Moreover, in some cases boundary conditions or path constraints with respect to the attitude or the rotational motion of the aircraft may arise so that the treatment of the aircraft as a point-mass model is no longer sufficient and a simulation model of higher fidelity has to be implemented. By doing so, the complexity of the optimization problem is increased dramatically and the optimization faces severe problems that arise with the more accurate modeling of the flight system.

To the author's knowledge, only a few applications of atmospheric flight trajectory optimization exist that are based on higher-fidelity simulation models taking into account the rotational and attitude dynamics of the aircraft. Mostly, only single maneuvers over short time spans are optimized utilizing full 6-Degree of Freedom simulation models. The reason therefore can be seen in the difficulties that arise if high-fidelity simulation models are utilized, especially in the lack of suitable initial guesses for the solution of highly dynamic trajectories in conjunction with high fidelity 6-DoF simulation models. An overview on solved trajectory optimization problems that are based on 6-DoF simulation models is given in chapter 1.2.1, where also some spacecraft trajectory optimization problems are mentioned. At this, no algorithm for the generation of suitable initial guesses for optimization of atmospheric flight trajectories with high-fidelity simulation models was found.

In some cases, not only the controls and the initial states may be subject to optimization but also some additional parameters may be involved that can be related e.g. to the basic design of the aircraft itself or to the fundamental procedure of the flight mission that has to be accomplished. If e.g. the designers of the aircraft or the flight mission planers pursue the same goals as the pilots, the original optimal control problem only has to be augmented by the supplementary parameters and the solution is straight forward. Otherwise, if the decision makers aim at different or even opposing objectives, so-called bilevel programming problems arise with different objectives on the various optimization levels. For example, air race pilots are primarily interested in flying the given course in the minimum possible time and are not concerned about safety aspects as long as no penalties are imposed on them. In contrary, the main focus of the track planners is with regard to the safety of the race track layout, i.e. they want to design the race track such that it will be as safe as possible. For the described scenarios, the aircraft designers or the flight mission planers are first to decide, and then the pilots will make their decisions based on the given aircraft or the prescribed flight mission in order to reach their own objectives. At this, it is assumed that the aircraft designers or the flight mission planners know how their decisions affect the decisions of the pilots.

Thus, a special class of bilevel optimal control problems is established, where an optimal solution of an upper level *parameter optimization problem* depends on the optimal solutions of one or more lower level *optimal control problems*. For the scenarios described above, the upper level optimization problem relates to the decisions of the aircraft designer respectively the flight mission planners while the lower level optimal control problems correspond to the decisions taken by the pilots. Furthermore, equality or inequality constraints on the upper level may be present that depend directly on the optimal trajectories or associated output functions of the lower level optimal control problems.

In the literature, the term *"bilevel optimization problem"* means a bilevel programming problem where the lower level and the upper level optimization problems are represented by standard parameter optimization problems (Ref. [Knauer, 2009]). The term *"bilevel optimal control problem"* refers to bilevel programming problems where both the lower and the upper level optimization problems are given by optimal control problems. The bilevel programming problem described above is a combination of an upper level *parameter optimization problem* and one or more lower level *optimal control problems*. Nevertheless it is termed *"bilevel optimal control problem"* since the overall computational effort for the solution of the entire bilevel programming problem is clearly dominated by the solution of the lower level optimal control problems.

With respect to the solution of bilevel optimization problems, a lot of research on theoretical fundamentals, solutions algorithms and applications has been carried out in the last decades and can be found in the literature (for an overview see e.g. Refs. [Vicente, 1994], [Colson, 2007] and [Dempe, 2003]). In contrary, the solution of bilevel optimal control problems has not gained so much attention up to now, and in the field of aerospace only few applications exist that are outlined in chapter 1.2.2. For the special type of bilevel optimal control problems described above, no applications and no efficient solution methods were found in the literature in the field of atmospheric flight.

Similar types of bilevel programming problems arise in the field of multidisciplinary design optimization of aircraft respectively spacecraft vehicles or by applying the *trajectory decomposition method*. With the trajectory decomposition method, a single trajectory is split up into multiple subarcs. The subarcs are then optimized either sequentially or simultaneously

on the lower level of the bilevel optimization architecture, while the optimizer on the upper level has to ensure the integrity of the trajectory segments at the junction points. At this, all equality and inequality constraints are usually assigned to the lower level optimal control problems and no path constraints apply to the optimization problem on the upper level. Applications of the trajectory decomposition method that have been found in the literature are given in chapter 1.2.3. These applications are mainly limited to the optimization of spacecraft missions or trajectories.

Multidisciplinary design optimization is the task of designing an aircraft or a spacecraft vehicle taking into account multiple disciplines like e.g. weights and sizing, structure, aerodynamics or aircraft performance. Therefore, the aircraft performance can be evaluated either by a mission analysis or by the solution of a trajectory optimization problem. One way to solve the multidisciplinary design task is the decomposition approach that decomposes the optimization problem into multiple sub-problems according to the disciplines involved in the respective design task. Thus, a bilevel programming structure results where the upper level optimizer has to ensure the integrity between the optimal solutions of the multiple sub-problems associated to the disciplines involved in the vehicle design. The upper level optimizer adjusts the design variables and passes them to the lower level optimizers until the overall objective is optimized. Various decomposition methods exist, like e.g. Collaborative Optimization (CO), Enhanced Collaborative Optimization (ECO), Concurrent Subspace Optimization (CSSO) and Bilevel Integrated System Synthesis (BLISS). For an overview, see e.g. Refs. [Brown, 2004] and [Perez, 2004]. For the solution of multidisciplinary design optimization tasks, often response surface models, krigging models or neural networks are used to approximate the subsystems. In chapter 1.2.4, examples for the multidisciplinary design optimization of aircraft or spacecraft vehicles involving trajectory optimization that have been found in the literature are given where the majority of the applications is related to spacecraft vehicle design.

Despite being similar, there are also differences between the type of bilevel optimal control problem specified above and the bilevel programming problems resulting from the decomposition of trajectory optimization problems respectively multidisciplinary design optimization problems. First, the decomposition approach artificially transforms an original standard single-level optimization problem into a bilevel programming problem where the overall objective that is to be optimized remains unchanged. In contrary, the bilevel optimal control problem described above features different objectives on the lower and upper level of the bilevel optimal control problem that may be even contradicting. Furthermore, path constraints on the upper level may exist that involve directly the trajectory or any other time-dependent output function of the lower level optimal control problems. This is usually not the case for the bilevel programming problems obtained by applying the decomposition method.

A crucial point for the solution of any bilevel programming problem is the efficient and accurate computation of the gradient information for the upper level optimization problem. The gradient information is usually obtained by a sensitivity analysis with respect to the optimal solutions of the various sub-problems what is called a post-optimality sensitivity analysis. As can be seen from the literature cited in chapter 1.2, different approaches exist to deal with the post-optimality sensitivity analysis. While for some applications the sensitivity analysis can be avoided by making some simplifying assumptions, for other applications the derivation of the gradient information is straightforward due to the structure of the bilevel programming problem, especially if the main problem parameters and the parameters of the

various subproblems are identical. Alternatively, the sensitivities can be computed approximately or by utilizing numerical differences. Especially numerical differences come along with a high computational cost in addition to the reduced accuracy of the obtained sensitivities. For example, a lower level trajectory optimization problem has to be solved twice with respect to each upper level parameter in order to compute the gradient information for the upper level by means of numerical differences.

1.2 Literature Review – State of the Art

In the following, a review of the literature on the essential topics related to the thesis at hand is given and the current state of the art is depicted.

1.2.1 Trajectory Optimization utilizing 6-Degree-of-Freedom Simulation Models

Hoffman (Ref. [Hoffman, 1991]) optimizes a short-time turning maneuver for a high-alpha fighter aircraft utilizing a 6-Degree-of-Freedom simulation model and the multiple shooting method. The simulation model incorporates quaternions instead of the usually utilized Euler angles, and the objective is the final time required for flying a turning maneuver.

In Ref. [Ciarcià, 2009], a 6-Degree-of-Freedom simulation model is utilized to model the dynamics of an Ekranoplane that is an aircraft designed for using ground effects at extremely low-flight altitudes. An optimal collision avoidance trajectory for a cruising Ekranoplane with regard to an obstacle that is located straight ahead the Ekranoplane is computed. The solution of the trajectory optimization problem was done by the multiple-subarc sequential gradient restoration algorithm.

Pourtakdoust (Ref. [Pourtakdoust, 2009]) determines optimal flight paths for aircraft encountering microburst wind shears during critical flight phases like take-off or landing. The computation of the optimal escape respectively approach strategies is based on a 6-Degree-of-Freedom formulation of the dynamic system and the optimal trajectories are found numerically using a gradient based algorithm.

Optimal aircraft trajectories for terrain-masking flight of an unmanned aerial vehicle are given in Refs. [Ries, 2005] and [Corban, 2007], where the dynamics of the UAV are represented by a non-linear 6-Degree-of-Freedom simulation model. The solutions of the optimal control problems were accomplished by applying the multiple shooting method in conjunction with a sequential quadratic programming algorithm.

Desai (Refs. [Desai, 2005] and [Desai, 2008]) proposes a two-timescale collocation architecture for the solution of a 6-Degree-of-Freedom reentry vehicle trajectory optimization problem. At this, a dense discretization grid is applied to the states associated with the high-frequency rotational dynamics, while only a coarse discretization grid is used for the states corresponding to the slowly varying translational dynamics. Thereby, the size of the overall problem could be reduced significantly, allowing for a more efficient solution of the 6-Degree-of-Freedom trajectory optimization problem by the collocation method.

In the field of trajectory optimization for launch, reentry and orbit vehicles the software tool ASTOS (**A**ero**s**pace **T**rajectory **O**ptimization **S**oftware, formerly named ALTOS) includes the possibility to use dynamic 6-Degree-of-Freedom simulation models for the computation of optimal spacecraft trajectories (Refs. [Well, 1997], [Wiegand, 1999] and [Cremaschi, 2009]).

At this, the flight system is not controlled directly by the moments but the dynamic system model is augmented by a flight-control system. Thus, the control inputs to the dynamic system model are the commanded angle of attack and bank angle while the sideslip angle is set to zero.

The trajectory for a launch vehicle mission that is the Titan IV mission from liftoff to park orbit is optimized in Ref. [Rao, 1996]. Therefore, Rao, Sutter and Hong developed a 6-Degree-of-Freedom trajectory optimization program called 6D TOP that utilizes a full 6-Degree-of-Freedom simulation model incorporating non-linear rotation and attitude equations of motion.

Bollino (Refs. [Bollino, 2006a] and [Bollino, 2006b]) considers 3-Degree-of-Freedom point-mass simulation models as well as full 6-Degree-of-Freedom simulations models for the solution of reentry guidance and trajectory optimization problems for the X-33 reusable launch vehicle. The discretized reentry trajectory optimization problems are solved using the Legendre pseudospectral collocation method.

1.2.2 Bilevel Optimal Control

In Refs. [Raivio, 2000] and [Ehtamo, 2001] pursuit-evasion games are interpreted as bilevel optimal control problems that are the visual identification of an aircraft respectively the escape of an aircraft from a missile encounter. At this, the optimal control problem of the pursuer trying to minimize the distance to the evader is identified with the lower level optimal control problem, while the evader's optimal control problem is regarded as the upper level optimal control problem with the evader trying to maximize the distance to the pursuer. The resulting bilevel optimal control problem is then solved iteratively, where common discretization and nonlinear programming techniques can be applied to solve the two sub-problems that are ordinary optimal control problems. Here, at each iteration step gradient information with respect to the cost function of the lower level optimal control problem is utilized for the solution of the upper level optimal control problem.

For the computation of the gradient of the cost function of the lower level optimal control problem with respect to the terminal position of the pursuer, solely the sensitivities of the cost function respectively the capture condition (i.e. the terminal constraint) of the lower level optimal control problem with respect to the pursuer's final position are required.

Callies (Ref. [Callies, 2000]) determines the optimal ascent trajectory of a hypersonic rocket-powered flight vehicle requiring that in case of a mission abort from every point of the nominal ascent trajectory an emergency landing site can be reached. These additional path constraints are regarded as (secondary) optimal control problems themselves and are solved along with the primary optimal control problem by means of an indirect multiple shooting method.

The problem of simultaneously stabilizing a finite number of dynamic flight systems respectively flight conditions under a single feedback controller is addressed in Ref. [Perez, 2008]. There, a decomposition method is applied that results in a bilevel design optimization architecture. While on the sub-level the stability of the individual plants has to be achieved, the converged top-level optimization problem assures that within each individual subsystem the same single feedback controller is implemented.

For the optimization of chemical processes by means of collocation methods, Tanartkit and Biegler (Ref. [Tanartkit, 1997]) derive a bilevel programming problem by splitting an optimal control problem into an inner and an outer problem, where the inner problem solves the corresponding non-linear programming problem on a fixed mesh while the outer problem is devoted to minimize the objective function utilizing the mesh element lengths as optimization parameters. In the outer problem, the gradients of the objective function with respect to the element lengths are either computed analytically by a sensitivity analysis of the inner problem or numerically by finite differences where the inner problem is re-solved for perturbed element lengths.

1.2.3 Trajectory Decomposition

A trajectory decomposition method for the selection of optimal intermediate targets w.r.t. a spacecraft trajectory optimization problem has been formulated by Petersen et al. (Ref. [Petersen, 1977]). The trajectory is split into natural segments like e.g. ascent, orbital or reentry representing the subproblems within the two-level optimization framework. The master or upper level optimization algorithm coordinates the solutions of the subproblems and ensures the continuity of the entire trajectory at the junction points. The subproblems are solved sequentially and analytical equations based on linearized subproblem equations are used for the evaluation of the master-level gradient.

Rahn (Refs. [Rahn, 1998], [Rahn, 1996a] and [Rahn, 1996b]) applies the trajectory decomposition method to optimize simultaneously the trajectory and the design of a space transportation system for a space flight mission. The entire trajectory is split into multiple flight path segments that are optimized in parallel on the sub-level of the two-level optimization scheme. The various path segments are coordinated by the main-level optimizer by determining subproblem targets such that the global objective is optimized. At this, the gradient information for the main-level optimizer is obtained by means of numerical differences.

Beltracchi (Ref. [Beltracchi, 1992]) solves the ground to mission (all-up) trajectory optimization problem for a two-stage earth-to-orbit launch vehicle by applying the trajectory decomposition method. At this, the trajectory is split up into a booster stage and an upper stage that are regarded as sub-level optimization tasks. For the booster stage, the objective is the maximum throw weight to a park orbit while for the upper stage the objective is the maximum payload that can be transferred from the park into the mission orbit. The main-level optimization problem coordinates the parameters of the park orbit such that the payload that can be transferred to the mission orbit becomes maximal. The objective of the main-level optimization problem is a direct function of the objectives of the two sub-level optimization tasks. Thus, the gradient of the main-level objective can be obtained straightaway by applying the chain rule for differentiation and using the parameter sensitivity derivatives of the sub-level objective functions.

Ledsinger (Refs. [Ledsinger, 2000a], [Ledsinger, 2000b] and [Ledsinger, 1998]) utilizes the Collaborative Optimization framework to optimize branching trajectories of an advanced two-stage-to-orbit launch vehicle where the flight path splits up into two branches, the orbital branch and the booster branch. After booster separation, the orbital branch is represented by the ascent trajectory of the upper stage into the orbit while the booster branch is the trajectory of the booster on its way back to the launching site. Here, a simplified algebraic form for the computation of the gradient information on the upper system level is used that avoids the

computationally expensive post-optimality sensitivity analysis for the optimal solutions of the various subsystem-level optimization problems.

In Ref. [Sugar, 1974], a decomposition technique for the solution of minimum-time optimal control problems by indirect methods is proposed where the original standard single-level optimal control problem is transformed into a three-level optimization problem. Therefore, the original trajectory is decomposed into multiple arcs due to intermediate constraints or discontinuities. On the first level, each arc is optimized given the initial and final states and phase transition times by the second level controller respectively third level optimizer. The coupling of the single arcs in time is accomplished by the second level controller, while the third level optimizer minimizes the overall objective by adjusting the states at the arc boundaries.

1.2.4 Multidisciplinary Design Optimization involving Trajectory Optimization

Mor and Livne (Refs. [Mor, 2006] and [Mor, 2007]) focus on the multidisciplinary optimization of launch and reentry vehicles including the discipline of trajectory optimization. The sensitivities of the optimal solutions for the ascent or reentry trajectories are obtained using either finite differences or the concept of feasible directions. The concept of feasible directions augments the parameter vector for the controls and the states by vehicle design variables and formulates an additional optimization problem that gives the sensitivities of the objective with respect to the additional shape or structural design variables.

Braun applies the Collaborative Optimization architecture to design a launch vehicle involving the disciplines propulsion, weights and sizing, cost and ascent trajectory optimization (Refs. [Braun, 1996] and [Braun, 1997]). The issue of computing the subsystem-level post-optimality sensitivity data is not addressed.

Brown and Olds (Ref. [Brown, 2006]) evaluate various multidisciplinary optimization techniques by solving a reusable launch vehicle design problem. Besides the fixed-point iteration method and the all-at-once technique, three bi-level optimization techniques are considered, namely Bi-Level Integrated Synthesis (BLISS), Collaborative Optimization (CO) as well as Modified Collaborative Optimization (MCO). A propulsion tool, a performance tool and a weights and sizing tool are involved in the multidisciplinary optimization task. Within the performance tool, the program to optimize simulated trajectories (POST) was utilized to analyze and optimize the launch vehicle trajectories. For the solution of the multidisciplinary optimization problem, the outputs of the various tools were replaced by response surface models. The response surface models are generated by a sequence of experiments and describe the dependency of the output (response) variables with respect to one or more input variables by response surface equations. Response surface equations are polynomial functions that are utilized to approximate the complex model response.

Sobieszczanski-Sobieski provides two methods for the generation of sensitivity data of optimal solutions w.r.t. problem parameters that are not subject to optimization (Ref. [Sobieszczanski-Sobieski, 1982]). While the one method derives the sensitivity equations from the Lagrange multiplier equations of the optimization problem, the other method is based on extremum conditions of the penalty functions involved in sequential unconstrained minimization techniques (SUMT). Numerical examples for the application of the sensitivity equations are given namely the optimization with conflicting objectives and the extrapolation

of the optimal solution. Further possible applications are mentioned that are the utilization of the sensitivity equations in order to predict a change in the constraint status or to solve large optimization problems by decomposition into multiple subproblems.

A decomposition method for the optimization of multidisciplinary engineering systems called Bilevel Integrated System Synthesis (BLISS) is given in Refs. [Sobieszczanski-Sobieski, 2000] and [Sobieszczanski-Sobieski, 1998]. The decomposition method is characterized by alternating optimizations on the system level and the subsystem (or discipline) level that are linked by sensitivity information. Two versions of BLISS are given with one version avoiding the computationally expensive post-optimality sensitivity analysis on the subsystem level. The method is tested on an aircraft configuration problem involving aircraft structure, aerodynamics, propulsion and performance, where the performance optimization has the goal to maximize the range using the Breguet range equation.

Braun (Ref. [Braun, 1993]) applies the multi-level decomposition strategy to the complex multidisciplinary design of a reusable, single-stage-to-orbit (SSTO) vehicle. The resulting hierarchical decomposed programming problem is solved using first-order post-optimality sensitivity information. There, the main problem parameters and the parameters of the various subproblems are identical so that the subproblem objective gradients equal the main problem objective gradient.

In Ref. [Braun, 1996], Braun and Kroo develop a Collaborative Optimization framework for the solution of multidisciplinary optimization tasks. They introduce a simplified algebraic form for the computation of the gradient information for the discrepancy constraints on the system level. The simplified algebraic form avoids the computationally expensive post-optimality sensitivity analysis w.r.t. the optimal solutions of the various subsystem-level optimization problems. The decomposition method is applied to optimize a lunar ascent trajectory where the trajectory is split into three subsegments. Within the Collaborative Optimization, the trajectory subsegments represent the subsystem-level optimization problems that are coordinated by the system-level optimization problem with its discrepancy constraints.

Roth develops the Collaborative Optimization technique further, resulting in the so-called Enhanced Collaborative Optimization (ECO) technique (Refs. [Roth, 2008a] and [Roth, 2008b]). The system level optimization problem is an unconstrained minimization problem that ought to ensure consistency between the various sub-level systems while the global objective itself is not involved in the system level optimization problem. Instead, quadratic models of the global objective together with linearized models of all subspace constraints are included in each subspace optimization problem. For the evaluation of the linearized constraint models, a constraint violation minimization (CVM) problem is solved and a post-optimality sensitivity analysis is carried out to give the coefficients for the linearized constraint models. The Enhanced Collaborative Optimization method is applied to an analytic test case and the design of an aircraft family, involving an optimization of the aircraft performance using an aircraft conceptual design tool.

1.3 Contributions of the Thesis

The following aspects are regarded as the main contributions of this thesis to advance beyond the current state of the art in the respective fields:

- **Deployment of a flexible method for the analytical evaluation of the gradient and the hessian of the dynamic flight system**
  For the computation of the Jacobian and the Hessian matrix of the optimal control problem, the first and second order derivative of the dynamic system with respect to the state vector and the control vector has to be evaluated. In chapter 2.2.4, a method is proposed that is based on the block structure of the implemented simulation model and that is very flexible with respect to modifications in the simulation model.
- **Development of a scalable, multi-fidelity fixed-wing aircraft simulation model specifically tailored for optimization tasks**
  The scalable, multi-fidelity simulation model features a sequential structure and takes into account the full dynamic order of the regarded flight systems as well as supplementary subsystems (e.g. actuator dynamics) and environmental influences like e.g. static and convective wind fields. Thus, it is ensured that the resulting trajectories are dynamically realistic and can be flown by the aircraft in reality. Furthermore, the simulation model involves an inversion controller and reference models. The inversion controller also takes into account environmental influences and features a sequential structure identical to that of the simulation model itself. The simulation model is described in detail in chapter 3.
- **Establishment of an algorithm for the solution of complex aircraft trajectory optimization problems**
  The simulation model with its special structure and additional features constitutes the basis for the establishment of the robust and effective process for the solution of rather complex aircraft trajectory optimization problems. The optimization procedure starts with a homotopy for the generation of an initial guess for an optimal point-mass trajectory and ends up with the optimal trajectory for the full, non-linear 6-DoF system dynamics without the necessity for the user to provide any initial guess. The optimization algorithm is given in chapter 4.
- **Development of an efficient algorithm for the solution of the above described bilevel optimal control problem**
  The proposed algorithm is based on a post-optimality sensitivity analysis that utilizes second order derivative information with respect to the lower level optimal control problems. Thus, it allows for a direct computation of the gradient of the upper level optimization problem at each iteration step so that the time consuming evaluation of the gradient of the upper level optimization problem by numerical techniques can be avoided. The algorithm for the efficient solution of the above described bilevel optimal control problem is outlined in chapter 5.

Furthermore, in chapter 2 a framework for the solution of optimal control problems by the direct multiple shooting method is formulated. The framework comprises the discretization of the continuous time optimal control problem, the analytical evaluation of the Jacobian and the Hessian, a suitable scaling of the problem as well as a mesh refinement algorithm.

Finally, in chapter 6 results are shown that originate from the application of the proposed optimization algorithms to an Air Race optimal control problem. The Air Race optimal control problems pose a very challenging task with regard to trajectory optimization since the participating aerobatic aircraft are very agile. Thus, the resulting trajectories are highly dynamic and the full dynamic order of the regarded flight systems has to be taken into account to achieve realistic optimal trajectories.

2

Fundamentals of Optimal Control

For the solution of optimal control problems respectively trajectory optimization problems, one can distinguish between two basically different types of methods, namely the *indirect methods* and the *direct methods*. The indirect methods are based on the theory of calculus of variations (Ref. [Bliss, 1946]), where necessary optimality conditions subject to a local or global minimum (maximum) principle are derived that have to be fulfilled by the optimal solution. For trajectory optimization problems with state and control constraints, this usually leads to the formulation of a multi-point boundary value problem with corresponding state and adjoint differential equations. Then, the resulting multi-point boundary value problems are discretized and solved by an appropriate solution method, e.g. the multiple shooting method (Refs. [Callies, 2000], [von Stryk, 1994]). Since the indirect methods make explicit use of the Hamiltonian, the adjoint variables and the minimum (respectively maximum) principle, this problem-dependent information has to be provided by the user for the solution of an optimal control problem.

By applying direct methods the original infinite-dimensional optimal control problems are transformed into finite-dimensional non-linear parameter optimization problems by discretizing either solely the controls or both the states and the controls. Methods that rely on the discretization of both the states and the controls are referred to as *collocation methods*, whereas the discretization of solely the controls gives rise to the so-called *shooting methods*. With regard to the collocation methods, one can further distinguish between *local* and *global* collocation methods where local collocation methods are based on integration schemes like e.g. the Euler method or the Runge-Kutta method. Global collocation methods make use of polynomials for the discretization of the states and the controls and are also termed *orthogonal* or *pseudospectral* collocation methods. A special type of direct methods are the *inversion based* trajectory optimization methods or the methods of *differential inclusion*, where the controls or some of the states together with the controls are eliminated from the system model and only the remaining states are discretized. With all direct methods, the discretized optimization problems are solved by utilizing non-linear programming methods like e.g. sequential quadratic programming. In contrast to the indirect methods, no information concerning the Hamiltonian, the adjoint variables or the minimum principle has to be provided by the user for the solution of the optimal control problem.

Thus, the fundamental difference between the direct and the indirect methods is the sequence of the optimization and the discretization: while the indirect methods *first optimize, then discretize*, the direct methods *first discretize, then optimize*. Comprehensive surveys on the different methods of trajectory optimization can be found e.g. in Refs. [Betts, 1998], [Betts, 2001], [Riehl, 2006], [Ross, 2002], [von Stryk, 1992], [von Stryk, 1994]. In the

following, the general optimal control problem for problems with only one phase as well as for problems with multiple phases is stated. Then, a framework for the solution of the optimal control problem by the multiple shooting method is established. This framework also comprises an evaluation of the Jacobian and Hessian of the optimal control as well as various mesh refinement procedures.

2.1 The General Optimal Control Problem

In general, a trajectory optimization problem can be stated as follows:

Determine the optimal control history

$$\mathbf{u}_{opt}(t) \in \mathbb{R}^m \tag{2.1}$$

the corresponding optimal state trajectory

$$\mathbf{x}_{opt}(t) \in \mathbb{R}^n \tag{2.2}$$

and possibly real parameters

$$\mathbf{p} \in \mathbb{R}^u \tag{2.3}$$

that minimize the Bolza cost functional

$$J = e\left(\mathbf{x}(t_f), \mathbf{p}, t_f\right) + \int_{t_0}^{t_f} L(\mathbf{x}(t), \mathbf{u}(t), \mathbf{p}, t)dt \tag{2.4}$$

subject to the state dynamics

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), \mathbf{p}, t) \tag{2.5}$$

the initial and final boundary conditions

$$\boldsymbol{\psi}_0\left(\mathbf{x}(t_0), \mathbf{p}, t_0\right) = \mathbf{0} \qquad \boldsymbol{\psi}_0 \in \mathbb{R}^q \qquad q \le m + n \tag{2.6}$$

$$\boldsymbol{\psi}_f\left(\mathbf{x}(t_f), \mathbf{p}, t_f\right) = \mathbf{0} \qquad \boldsymbol{\psi}_f \in \mathbb{R}^p \qquad p \le m + n \tag{2.7}$$

the interior point conditions

$$\mathbf{r}_i\left(\mathbf{x}(t_i), \mathbf{u}(t_i), \mathbf{p}, t_i\right) = \mathbf{0} \qquad i = 1, ..., k \tag{2.8}$$

and the equality and inequality conditions

$$\mathbf{C}_{eq}\left(\mathbf{x}(t), \mathbf{u}(t), \mathbf{p}, t\right) = \mathbf{0} \qquad \mathbf{C}_{eq} \in \mathbb{R}^r \tag{2.9}$$

$$\mathbf{C}_{ineq}\left(\mathbf{x}(t), \mathbf{u}(t), \mathbf{p}, t\right) \le \mathbf{0} \qquad \mathbf{C}_{ineq} \in \mathbb{R}^s \tag{2.10}$$

where m is the number of controls, n the number of states, u the number of real parameters, q the number of initial boundary conditions, p the number of final boundary conditions, k the number of interior point conditions, r the number of equality conditions and s the number of inequality conditions. The cost functional (2.4) is termed *Bolza* cost functional since it comprises an integral term L as well as a final term e. If only the integral term L is present in the cost functional, it is termed *Lagrange* cost functional. Otherwise, if the cost functional consists only of the final term e, it is termed *Mayer* cost functional.

Multi-phase optimal control problems

Multi-phase optimal control problems result if interior point conditions (2.8) are present that cannot be implemented directly for the solution of the optimal control problem. Then, the entire trajectory has to be split up into k+1 phases and the interior point conditions (2.8) are transformed into final boundary conditions for each phase:

$$\mathbf{r}_i\left(\mathbf{x}\left(t_{f,i}\right),\mathbf{u}\left(t_{f,i}\right),\mathbf{p},t_{f,i}\right)=\mathbf{0} \qquad i=1,...,k \tag{2.11}$$

in which $t_{f,i}$ is the final time of the i-th phase. Additionally, the multiple phases have to be connected to the preceding phases to guarantee the continuity of the state and the control time histories at the phase boundaries:

$$\mathbf{x}\left(t_{f,i}\right)-\mathbf{x}\left(t_{0,i+1}\right)=\mathbf{0} \qquad i=1,...,k \tag{2.12}$$

$$\mathbf{u}\left(t_{f,i}\right)-\mathbf{u}\left(t_{0,i+1}\right)=\mathbf{0} \qquad i=1,...,k \tag{2.13}$$

where $t_{0,i}$ is the initial time of the i-th phase. Within the optimization problem, Eqs. (2.12) and (2.13) are treated as supplementary equality conditions.

Furthermore, multi-phase optimal control problems arise if the number of states, controls or real parameters changes within the trajectory optimization problem or if different Lagrange cost functions, path equality constraints or path inequality constraints apply for different phases of the trajectory that is to be optimized. Thus, in its most general form, a multi-phase optimal control problem can be stated as follows:

Determine the optimal control histories

$$\mathbf{u}_{i,opt}\left(t\right)\in\mathbb{R}^{m_i} \tag{2.14}$$

the corresponding optimal state trajectories

$$\mathbf{x}_{i,opt}\left(t\right)\in\mathbb{R}^{n_i} \tag{2.15}$$

and possibly real parameters

$$\mathbf{p}_i\in\mathbb{R}^{u_i} \tag{2.16}$$

that minimize the Bolza cost functional

$$J=\sum_{i=1}^{ph}e_i\left(\mathbf{x}_i\left(t_{f,i}\right),\mathbf{p}_i,t_{f,i}\right)+\sum_{i=1}^{ph}\int_{t_{0,i}}^{t_{f,i}}L_i\left(\mathbf{x}_i\left(t\right),\mathbf{u}_i\left(t\right),\mathbf{p}_i,t\right)dt \tag{2.17}$$

subject to the state dynamics

$$\dot{\mathbf{x}}_i\left(t\right)=\mathbf{f}_i\left(\mathbf{x}_i\left(t\right),\mathbf{u}_i\left(t\right),\mathbf{p}_i,t\right) \tag{2.18}$$

the initial and final boundary conditions

$$\boldsymbol{\psi}_{0,i}\left(\mathbf{x}_i\left(t_{0,i}\right),\mathbf{p}_i,t_{0,i}\right)=\mathbf{0} \qquad \boldsymbol{\psi}_{0,i}\in\mathbb{R}^{q_i} \qquad q_i\le m_i+n_i \tag{2.19}$$

$$\boldsymbol{\psi}_{f,i}\left(\mathbf{x}_i\left(t_{f,i}\right),\mathbf{p}_i,t_{f,i}\right)=\mathbf{0} \qquad \boldsymbol{\psi}_{f,i}\in\mathbb{R}^{p_i} \qquad p_i\le m_i+n_i \tag{2.20}$$

and the equality and inequality conditions

$$\mathbf{C}_{eq,i}\left(\mathbf{x}_i\left(t\right),\mathbf{u}\left(t_i\right),\mathbf{p}_i,t\right)=\mathbf{0} \qquad \mathbf{C}_{eq,i}\in\mathbb{R}^{r_i} \tag{2.21}$$

$$\mathbf{C}_{ineq,i}\left(\mathbf{x}_i(t),\mathbf{u}_i(t),\mathbf{p}_i,t\right)\le\mathbf{0}\qquad\mathbf{C}_{ineq,i}\in\mathbb{R}^{s_i} \tag{2.22}$$

where *ph* is the number of phases. The subscript *i* denotes the state vector **x**, the terminal cost function *e*, the Lagrange cost function *L*, the initial and final boundary conditions $\mathbf{\Psi}_0$ and $\mathbf{\Psi}_f$ and the equality and inequality constraints $\mathbf{C}_{eq}$ and $\mathbf{C}_{ineq}$ of the respective phase *i*.

At the phase transition times t_i, phase transition conditions $\mathbf{g}_i$ enforce prescribed relationships between the states, the controls and the real parameters of the adjacent phases:

$$\mathbf{g}_i\left(\mathbf{x}_{i-1}(t_{f,i-1}),\mathbf{x}_i(t_{0,i}),\mathbf{u}_{i-1}(t_{f,i-1}),\mathbf{u}_i(t_{0,i}),\mathbf{p}_{i-1},\mathbf{p}_i,t_i\right)=\mathbf{0}\qquad i=2,\ldots,ph \tag{2.23}$$

where

$$t_i=t_{f,i-1}=t_{0,i}\qquad i=2,\ldots,ph \tag{2.24}$$

2.2 Framework for the Solution of the Optimal Control Problem

2.2.1 Multiple Shooting Method

As mentioned above, with the shooting method solely the controls are discretized in order to transform the infinite-dimensional optimal control problem into a finite-dimensional optimization problem. The equations of motions are fulfilled by numerical integration of the corresponding ordinary differential equations. The shooting method is also referred to as *sequential* approach (Ref. [Huesman, 2003]), *recursive* approach (Ref. [Büskens, 2000]) or *reduced discretization* approach (Ref. [Gerdts, 2007]), since only the controls are discretized and the states are obtained recursively by integrating the differential equations. Furthermore, it has to be distinguished between the *single* shooting method and *multiple* shooting method. The single shooting method integrates the state equations from the initial point to the terminal point of the trajectory in one sweep. Thus, for a trajectory optimization problem with free final time the parameter vector **z** is made up by the free states $\mathbf{x}_{0,free}$ at the initial time t_0 in addition to the vector **u** of control variables and the free final time t_f:

$$\mathbf{z}=[t_f,\mathbf{x}_{0,free},\mathbf{u}]^T \tag{2.25}$$

In contrast to the single shooting method, the multiple shooting method introduces a relatively small number of so-called multiple shooting nodes $\mathbf{x}_{MS,j}$, j = 1, …, m for the states on a mesh of m grid points:

$$\tau_0=\tau(t_0)<\tau_{MS,1}<\tau_{MS,2}<\ldots<\tau_{MS,m-1}<\tau_{MS,m}<\tau_f=\tau(t_f) \tag{2.26}$$

Here, τ denotes the normalized time and is given by the following expression:

$$\tau(t)=\frac{t-t_0}{t_f-t_0},\qquad t_0\le t\le t_f \tag{2.27}$$

Then, the state equations have to be multiplied by $(t_f - t_0)$ for the integration w.r.t. the normalized time τ:

$$\frac{d\mathbf{x}(t)}{d\tau}=\frac{d\mathbf{x}(t)}{dt}\cdot\frac{dt}{d\tau}=\left(t_f-t_0\right)\cdot\mathbf{f}\left(\mathbf{x}(t),\mathbf{u}(t),\mathbf{p},t\right) \tag{2.28}$$

Consequently, the integration of the state dynamics is not carried out from the initial states $\mathbf{x}_0$ to the end of the trajectory in a single sweep, but is reset to the state values $\mathbf{x}_{MS,j}$ at the multiple shooting nodes. This procedure is depicted in Fig. 1. The state values $\mathbf{x}_{MS,j}$ are additional parameters of the nonlinear programming problem, and for a solution to be optimal the defects at the multiple shooting nodes resulting from the integration of the state dynamics have to be zero. By introducing multiple shooting nodes, the Jacobian of the parameter optimization problems shows a sparse block structure in contrary to the single shooting method that features a dense Jacobian. Thus, the multiple shooting method is well suited for the solution of trajectory optimization problems by applying non-linear programming techniques for large-scale optimization problems that explicitly exploit the sparsity of the Jacobian, like e.g. SNOPT (Ref. [Gill, 2007]). Furthermore, by dividing the trajectory into multiple segments for the integration, the solution of the ordinary differential equations by the numerical integration is less sensitive to the initial conditions and the robustness and stability of the solution of the trajectory optimization problems is increased. Together with the free initial states $\mathbf{x}_{0,free}$ that are also subject to optimization the complete parameter vector $\mathbf{z}$ of an optimal control problem with free final time t_f applying the multiple shooting method reads:

$$\mathbf{z} = [t_f, \mathbf{x}_{0,free}, \mathbf{x}_{MS,1}, \mathbf{x}_{MS,2}, \mathbf{x}_{MS,3}, ..., \mathbf{x}_{MS,m}, \mathbf{u}]^T \tag{2.29}$$

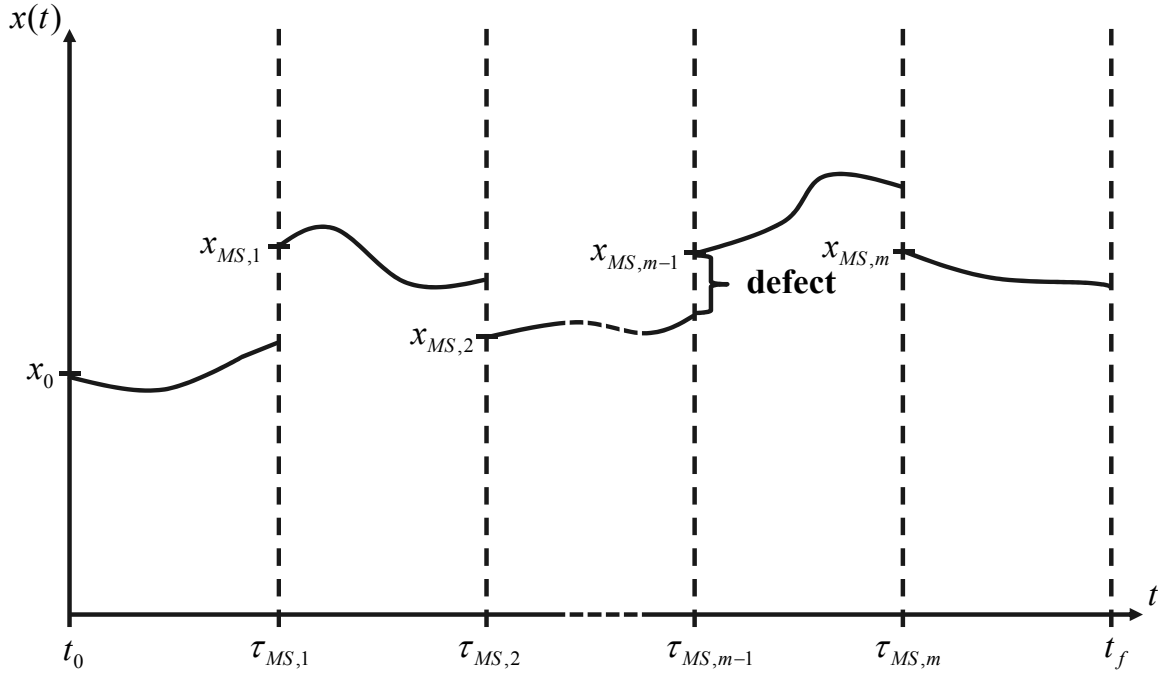

Figure 1. Multiple shooting

2.2.2 Control Discretization

For the discretization of the controls $\mathbf{u}(\tau)$, a mesh of n+1 grid points is chosen such that

$$\tau(t_0) = \tau_0 < \tau_1 < \tau_2 < ... < \tau_{n-1} < \tau_n = \tau_f = \tau(t_f) \tag{2.30}$$

In between the grid points τ_i, $i = 0, \ldots, n$, the approximated controls $\mathbf{u}(\tau)$ are obtained by linear interpolation of the control values $\mathbf{u}_i = \mathbf{u}(\tau_i)$, $i = 0, \ldots, n$ at the grid points:

$$\mathbf{u}(\tau) = \mathbf{u}_i + (\mathbf{u}_{i+1} - \mathbf{u}_i)\frac{\tau - \tau_i}{\tau_{i+1} - \tau_i} \qquad \tau_i \le \tau < \tau_{i+1} \tag{2.31}$$

Alternatively, the control function $\mathbf{u}(\tau)$ can be specified by a *B*-spline representation of degree $k-1$ (respectively order k), where the elementary *B*-splines are defined recursively (Refs. [de Boor, 72] and [Cox, 72]) by the initialization splines for $k = 1$

$$N_{j,1}(\tau) = \begin{cases} 1, & \text{if } \tau \in [\lambda_j, \lambda_{j+1}), \\ 0, & \text{if } \tau \notin [\lambda_j, \lambda_{j+1}), \end{cases} \tag{2.32}$$

and the higher-order splines ($k > 1$)

$$N_{j,k}(\tau) = \frac{\tau - \lambda_j}{\lambda_{j+k-1} - \lambda_j} N_{j,k-1}(\tau) + \frac{\lambda_{j+k} - \tau}{\lambda_{j+k} - \lambda_{j+1}} N_{j+1,k-1}(\tau), \qquad j = 1,\ldots,n+k-1, \tag{2.33}$$

with the auxiliary grid points λ_j, $j = 1, \ldots, n+2k-1$

$$\lambda_j = \begin{cases} \tau_0, & \text{if } 1 \le j \le k, \\ \tau_{j-k}, & \text{if } k+1 \le j \le n+k-1, \\ \tau_f, & \text{if } n+k \le j \le n+2k-1. \end{cases} \tag{2.34}$$

This means that the auxiliary grid given by Eq. (2.34) is identical to the grid given by Eq. (2.30) with a multiplicity of the first and the last grid point equal to the order k of the *B*-spline. In Fig. 2, exemplarily the elementary *B*-splines $N_{1,5}$, $N_{2,5}$ and $N_{3,5}$ on an equidistant grid with $n = 4$ are depicted.

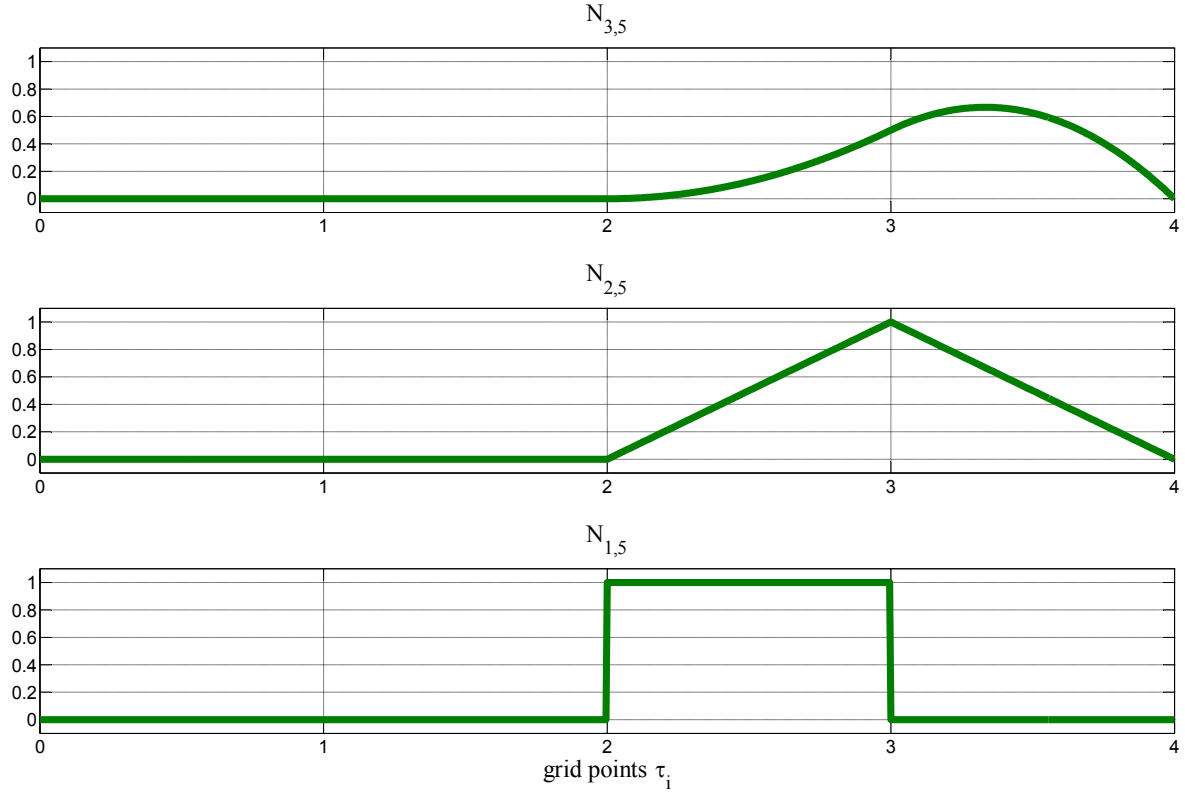

Figure 2. Elementary B-Splines on an equidistant grid with $n = 4$

The control function $u(\tau)$ of order k can then be written as a linear combination of the elementary B-splines $N_{j,k}$:

$$u(\tau) = \sum_{j=1}^{n+k-1} c_j N_{j,k}(\tau) \tag{2.35}$$

where the c_j, $j = 1, \ldots, n+k-1$ are the B-spline coefficients. At this, the vector **u** of the control variables in the parameter vector **z** of the trajectory optimization problem is replaced by the vector **c** of B-spline coefficients c_j:

$$\mathbf{z} = [t_f, \mathbf{x}_{0,free}, \mathbf{x}_{MS,j}, \mathbf{c}]^T \qquad j = 1,\ldots,m \tag{2.36}$$

Utilizing a B-spline representation of order $k = 1$, the controls are piecewise constant and the control values in between the grid points $\lambda_j = \tau_{j-1}$ and $\lambda_{j+k} = \tau_{j+k-1}$ equal the respective B-spline coefficients c_j. For a B-spline representation of order $k = 2$, continuous and piecewise linear controls are obtained identical to the linear interpolation given by Eq. (2.31), where the B-spline coefficients c_j are just the discretized control values at the respective grid points $\lambda_{j+1} = \tau_{j-1}$. Fig. 3 depicts the elementary B-splines of order $k = 2$ and $k = 3$ on a generic, equidistant grid with normalized time τ and $n = 4$. Especially for higher-order approximation of the controls, the utilization of B-splines has the major advantage that the elementary B-splines support the control function $u(\tau)$ only locally (Ref. [Dierckx, 1993]). Thus, the B-spline coefficients c_j influence the control function $u(\tau)$ only in the interval $[\lambda_j , \lambda_{j+k})$ which is not the case with other spline interpolation methods. This results in a specific sparsity pattern of the Jacobian that can also be exploited in order to reduce the computational effort for the non-linear programming technique that is utilized to solve the trajectory optimization problem. It is mentioned that different elements of the control vector **u** may be approximated by B-splines of different order.

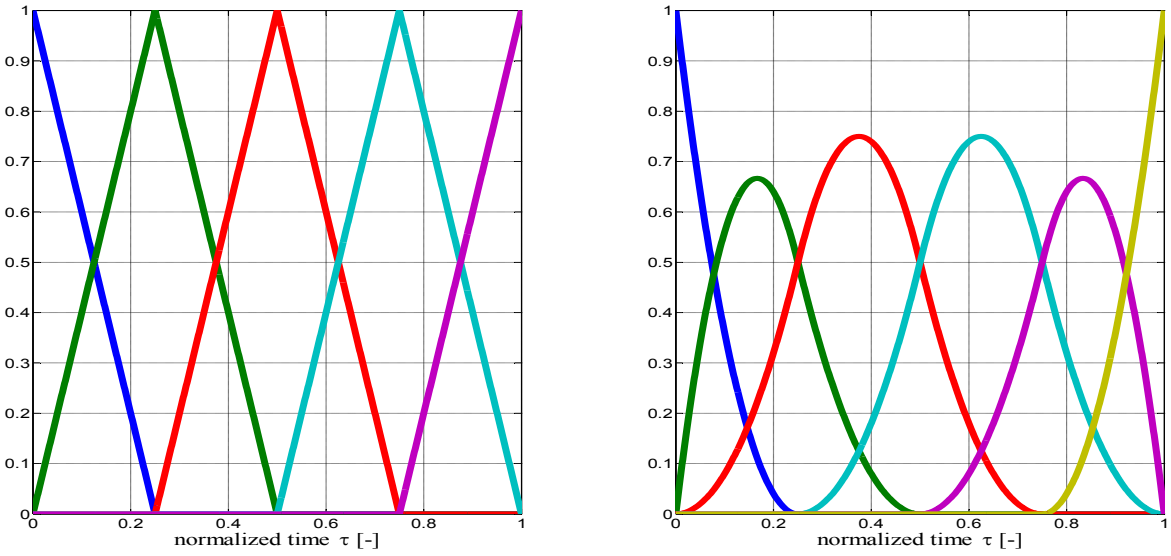

Figure 3. Elementary B-Splines of order $k = 2$ and $k = 3$ with $[\tau_0, \tau_f]=[0,1]$

In addition to the state discretization grid and the control discretization grid, a grid for the evaluation of the path constraints with the normalized time points $\tau_{PC,p}$, $i = 1, \ldots, p$ is introduced:

$$\tau_0 \le \tau_{PC,1} < \tau_{PC,2} < \ldots < \tau_{PC,p-1} < \tau_{PC,p} \le \tau_f \tag{2.37}$$

Furthermore, the parameter vector **z** can be augmented by additional parameters **p** that are also subject to optimization but that do not pertain to the state discretization variables nor the control discretization variables:

$$\mathbf{z} = [t_f, \mathbf{x}_{0,free}, \mathbf{x}_{MS,j}, \mathbf{u}, \mathbf{p}]^T \qquad j = 1,...,m \tag{2.38}$$

By applying the multiple shooting method and the control discretization, the original continuous-time infinite-dimensional optimal control problem stated in chapter 2.1 is transformed into the following finite-dimensional non-linear programming problem: Minimize

$$J = e(\mathbf{x}(\mathbf{z}), \mathbf{z}) + \int_{t_0}^{t_f} L(\mathbf{x}(\mathbf{z}), \mathbf{z}, t)\, dt \tag{2.39}$$

subject to the parameter vector **z** given by Eq. (2.38) such that the following equality and inequality constraints are fulfilled:

$$\mathbf{C} = \begin{pmatrix} \boldsymbol{\psi}_0(\mathbf{x}(\mathbf{z}), \mathbf{z}) = \mathbf{0} \\ \mathbf{C}_{eq}(\mathbf{x}(\mathbf{z}), \mathbf{z}) = \mathbf{0} \\ \mathbf{C}_{ineq}(\mathbf{x}(\mathbf{z}), \mathbf{z}) \geq \mathbf{0} \\ \mathbf{x}(\mathbf{z}, \tau_{j-1}, \tau_j) - \mathbf{x}_{MS,j} = \mathbf{0} \\ \mathbf{r}_i(\mathbf{x}(\mathbf{z}), \mathbf{z}) = \mathbf{0} \\ \boldsymbol{\psi}_f(\mathbf{x}(\mathbf{z}), \mathbf{z}) = \mathbf{0} \end{pmatrix} \tag{2.40}$$

Here, $\mathbf{x}(\mathbf{z}, \tau_{j-1}, \tau_j)$ are the state values resulting from the integration of the equations of motion given by Eq. (2.5) respectively Eq. (2.18) in the interval $[\tau_{j-1}, \tau_j]$:

$$\mathbf{x}(\mathbf{z}, \tau_{j-1}, \tau_j) = \mathbf{x}_{MS,j-1} + \int_{\tau_{j-1}}^{\tau_j} (t_f - t_0) \cdot \mathbf{f}(\mathbf{x}(\mathbf{z}), \mathbf{z}, t)\, d\tau \tag{2.41}$$

The non-linear programming problem defined by Eqs. (2.40) and (2.41) is efficiently solved by software for large-scale non-linear optimization like e.g. SNOPT (Ref. [Gill, 2007]) or IPOPT (Ref. [Wächter, 2009]). Therefore, usually an augmented Lagrange cost function L_0 is defined that features the following form:

$$L_0 = J + \boldsymbol{\mu}^T \mathbf{C} \tag{2.42}$$

where **μ** are the so-called Lagrange multipliers. Necessary conditions for any solution of the nonlinear programming problem to be an optimal one are the so-called *Karush-Kuhn-Tucker* conditions. The *Karush-Kuhn-Tucker* conditions require the complementarity of the Lagrange multipliers for the active constraints, i.e.

$$\mu_a^i \leq 0 \tag{2.43}$$

where μ_i^a is the Lagrange multiplier associated with the *i*-th active constraint.

2.2.3 Multi-Phase Optimal Control Problems

As mentioned in chapter 2.1, optimal control problems can involve multiple phases with changes in the number of states, controls and parameters from one phase to another as well as various numbers of path constraints in the different phases. The various phases may also be

associated with different types of initial and final boundary conditions ψ, terminal cost functions e or integral Lagrange cost functions L. Then, the extended objective and the augmented constraint vector of the multi-phase optimal control problem read:

$$J = \sum_{i=1}^{ph} e_i\left(\mathbf{x}_i\left(\mathbf{z}\right), \mathbf{z}\right) + \sum_{i=1}^{ph} \int_{t_{0,i}}^{t_{f,i}} L_i\left(\mathbf{x}_i\left(\mathbf{z}\right), \mathbf{z}, t\right) dt \tag{2.44}$$

$$\mathbf{C} = \begin{pmatrix} \boldsymbol{\psi}_{0,1}\left(\mathbf{x}_1\left(\mathbf{z}\right), \mathbf{z}\right) = \mathbf{0} \\ \mathbf{C}_{eq,1}\left(\mathbf{x}_1\left(\mathbf{z}\right), \mathbf{z}\right) = \mathbf{0} \\ \mathbf{C}_{ineq,1}\left(\mathbf{x}_1\left(\mathbf{z}\right), \mathbf{z}\right) \geq \mathbf{0} \\ \mathbf{x}_1\left(\mathbf{z}, \tau_{j-1,1}, \tau_{j,1}\right) - \mathbf{x}_{MS,j,1} = \mathbf{0} \\ \boldsymbol{\psi}_{f,1}\left(\mathbf{x}_1\left(\mathbf{z}\right), \mathbf{z}\right) = \mathbf{0} \\ \\ \vdots \\ \\ \boldsymbol{\psi}_{0,ph}\left(\mathbf{x}_{ph}\left(\mathbf{z}\right), \mathbf{z}\right) = \mathbf{0} \\ \mathbf{C}_{eq,ph}\left(\mathbf{x}_{ph}\left(\mathbf{z}\right), \mathbf{z}\right) = \mathbf{0} \\ \mathbf{C}_{ineq,ph}\left(\mathbf{x}_{ph}\left(\mathbf{z}\right), \mathbf{z}\right) \geq \mathbf{0} \\ \mathbf{x}_{ph}\left(\mathbf{z}, \tau_{j-1,ph}, \tau_{j,ph}\right) - \mathbf{x}_{MS,j,ph} = \mathbf{0} \\ \boldsymbol{\psi}_{f,ph}\left(\mathbf{x}_{ph}\left(\mathbf{z}\right), \mathbf{z}\right) = \mathbf{0} \end{pmatrix} \tag{2.45}$$

where ph is the number of phases of the multi-phase trajectory optimization problem. The state vector $\mathbf{x}_i(\mathbf{z}, \tau_{j-1,i}, \tau_{j,i})$ results from the integration of the equations of motion in the time interval $[\tau_{j-1,i}, \tau_{j,i}]$ in phase i:

$$\mathbf{x}_i\left(\mathbf{z}, \tau_{j-1,i}, \tau_{j,i}\right) = \mathbf{x}_{MS,j-1,i} + \int_{\tau_{j-1,i}}^{\tau_{j,i}} \left(t_{f,i} - t_{0,i}\right) \cdot \mathbf{f}_i\left(\mathbf{x}_i\left(\mathbf{z}\right), \mathbf{z}, t\right) d\tau \tag{2.46}$$

Here, the subscript i denotes the state vector $\mathbf{x}$, the terminal cost function e, the Lagrange cost function L, the initial and final boundary conditions $\boldsymbol{\Psi}_0$ and $\boldsymbol{\Psi}_f$ and the equality and inequality constraints $\mathbf{C}_{eq}$ and $\mathbf{C}_{ineq}$ of the respective phase.

Additionally, phase transition conditions $\mathbf{g}_i$ have to be taken into account that enforce prescribed relationships between the states, the controls and the real parameters of the various phases at the phase transition times t_i:

$$\mathbf{g}_i\left(\mathbf{x}_{i-1}\left(\mathbf{z}\right), \mathbf{x}_i\left(\mathbf{z}\right), \mathbf{z}, t_i\right) = \mathbf{0} \qquad i = 2, ..., ph \tag{2.47}$$

where

$$t_i = t_{f,i-1} = t_{0,i} \qquad i = 2, ..., ph \tag{2.48}$$

2.2.4 Gradient, Jacobian and Hessian Evaluation

While all of the programs for the solution of the discretized optimal control problem require the gradient of the objective (Eq. (2.44)) and the Jacobian of the constraint vector (Eq. (2.45)), at least some non-linear optimization programs also necessitate the Hessian of the objective and the constraint vector in order to be able to solve the discretized optimal control problem

efficiently. Here, Jacobian means the first order derivative of the constraint vector with respect to parameter vector **z** of the discretized optimal control problem and Hessian its second order derivative. One possibility would be the evaluation of the Jacobian and the Hessian by numerical techniques like e.g. *finite differences*. Requiring no additional information from the user, the application of finite differences is straight forward, but results in a considerable computational effort and suffers from a lack of accuracy. Thus, the sensitivity equation approach (Refs. [Gerdts, 2007], [Büskens, 2000]) is implemented to allow for an efficient analytical computation of the gradient, the Jacobian and the Hessian for the discretized trajectory optimization problem. In order to be able to apply the sensitivity equation approach, the derivatives of the equations of motion (2.5) with respect to the state vector $\mathbf{x}(t)$**,**

$$\frac{\partial \dot{\mathbf{x}}(t)}{\partial \mathbf{x}^T} = \frac{\partial \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), \mathbf{p}, t)}{\partial \mathbf{x}^T} \tag{2.49}$$

the control vector $\mathbf{u}(t)$**,**

$$\frac{\partial \dot{\mathbf{x}}(t)}{\partial \mathbf{u}^T} = \frac{\partial \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), \mathbf{p}, t)}{\partial \mathbf{u}^T} \tag{2.50}$$

and the parameter vector **p,**

$$\frac{\partial \dot{\mathbf{x}}(t)}{\partial \mathbf{p}^T} = \frac{\partial \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), \mathbf{p}, t)}{\partial \mathbf{p}^T} \tag{2.51}$$

have to be calculated at every evaluation time step t during the integration of the equations of motion of Eq. (2.5). With regard to the quite complicated and extensive simulation model given in chapter 3, the analytical derivation of these derivatives would be a very daunting task. Therefore, an approach has been found that exploits the modular block structure of the simulation model and promises to be very flexible with regard to modifications of the simulation. This approach is illustrated in Fig. 4 for a simplified structure of the simulation model. Instead of deriving the Jacobian and the Hessian of the differential equations (2.5) with respect to the states **x** and the controls **u** for the entire simulation model at once, for each block only the derivatives of its outputs with respect to its inputs are derived and implemented in the simulation model. Due to the modular block structure of the simulation model, the equations that have to be derived analytically are quite succinct. The inputs and outputs of the various blocks of the simulation model are then joint together in order to give the required Jacobian and Hessian for the entire simulation model. Exemplarily, the Jacobian of the output y_{22} of the simulation model depicted in Fig. 4 is given by

$$\begin{aligned}\begin{bmatrix} \frac{\partial y_{22}}{\partial u_{21}} & \frac{\partial y_{22}}{\partial u_{11}} & \frac{\partial y_{22}}{\partial u_{12}} \end{bmatrix} &= \begin{bmatrix} \frac{\partial y_{22}}{\partial u_{21}} & \frac{\partial y_{22}}{\partial y_{11}} \end{bmatrix} \begin{bmatrix} \frac{\partial u_{21}}{\partial u_{21}} & \frac{\partial u_{21}}{\partial u_{11}} & \frac{\partial u_{21}}{\partial u_{12}} \\ \frac{\partial y_{11}}{\partial u_{21}} & \frac{\partial y_{11}}{\partial u_{11}} & \frac{\partial y_{11}}{\partial u_{12}} \end{bmatrix} = \\ &= \begin{bmatrix} \frac{\partial y_{22}}{\partial u_{21}} & \frac{\partial y_{22}}{\partial y_{11}} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \frac{\partial y_{11}}{\partial u_{11}} & \frac{\partial y_{11}}{\partial u_{12}} \end{bmatrix}\end{aligned} \tag{2.52}$$

Within the subsystem f_2, only the derivatives $\partial y_{22}/\partial u_{21}$ and $\partial y_{22}/\partial y_{11}$ are implemented, while the subsystem f_1 provides the derivatives $\partial y_{11}/\partial u_{11}$ and $\partial y_{11}/\partial u_{12}$ that are in turn inputs to the subsystem f_2.

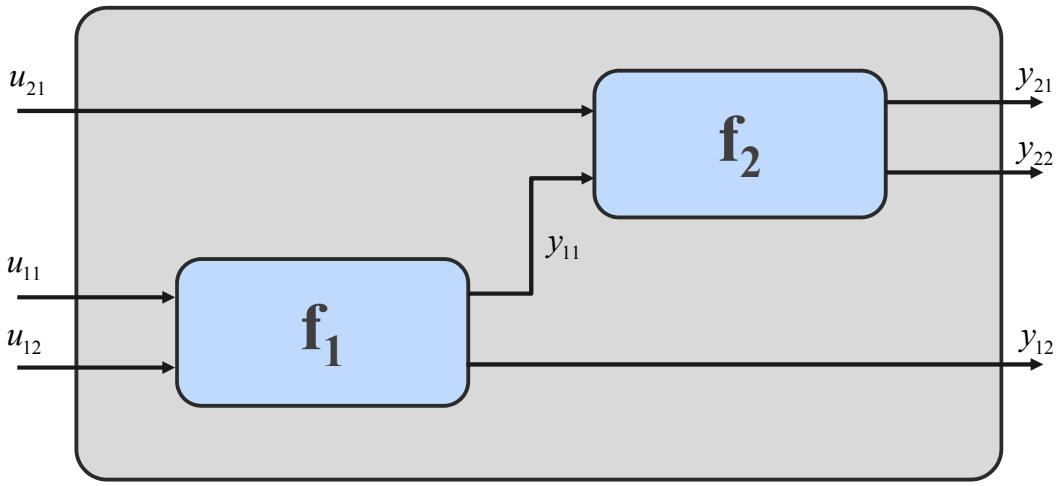

Figure 4. Modular simulation model structure

With regard to simulation models that represent flight system dynamics, usually many transformation matrices between the different reference frames and many matrix-vector products are involved. While the derivation of a vector with respect to another vector ends up in a matrix, the derivation of a matrix with respect to a vector gives a three-dimensional tensor. The second order derivative of a matrix with respect to a vector even yields a four-dimensional tensor. At this, special care has to be taken if derivatives of matrix-vector products are to be implemented in the simulation model. The rules for the differentiation of matrix-vector products respectively Kronecker products can be found e.g. in Ref. [Magnus, 1985]. For the first order derivative, the following relationship holds:

$$\frac{\partial\, vec(\mathbf{Mx})}{\partial\, \mathbf{a}^T} = \left(\mathbf{x}^T \otimes \mathbf{I}_m\right)\frac{\partial\, vec(\mathbf{M})}{\partial\, \mathbf{a}^T} + \mathbf{M}\frac{\partial\, \mathbf{x}}{\partial\, \mathbf{a}^T} \tag{2.53}$$

where *vec()* denotes the vectorization of a matrix, converting the matrix into a column vector by stacking the columns of the matrix vertically. The Kronecker product is an operation between two matrices resulting in a block matrix, where each element of the first matrix is multiplied with the second matrix. In Eq. (2.53), the dimensions of the matrix **M** and the vectors **x** and **a** are:

$$\mathbf{M} \in \mathbb{R}^{m \times r} \tag{2.54}$$

$$\mathbf{x} \in \mathbb{R}^{r \times 1} \tag{2.55}$$

$$\mathbf{a} \in \mathbb{R}^{s \times 1} \tag{2.56}$$

For example, with **M** being a 3×3 matrix, **x** being a 3×1 vector and **a** being a 2×1 vector, Eq. (2.53) evaluates to:

$$\frac{\partial\, vec(\mathbf{Mx})}{\partial\, \mathbf{a}^T} = \left(\mathbf{x}^T \otimes \mathbf{I}_m\right)\begin{bmatrix} \frac{\partial m_{11}}{\partial a_1} & \frac{\partial m_{11}}{\partial a_2} \\ \frac{\partial m_{21}}{\partial a_1} & \frac{\partial m_{21}}{\partial a_2} \\ \frac{\partial m_{31}}{\partial a_1} & \frac{\partial m_{31}}{\partial a_2} \\ \frac{\partial m_{12}}{\partial a_1} & \frac{\partial m_{12}}{\partial a_2} \\ \frac{\partial m_{22}}{\partial a_1} & \frac{\partial m_{22}}{\partial a_2} \\ \frac{\partial m_{23}}{\partial a_1} & \frac{\partial m_{23}}{\partial a_2} \\ \frac{\partial m_{13}}{\partial a_1} & \frac{\partial m_{13}}{\partial a_2} \\ \frac{\partial m_{23}}{\partial a_1} & \frac{\partial m_{23}}{\partial a_2} \\ \frac{\partial m_{33}}{\partial a_1} & \frac{\partial m_{33}}{\partial a_2} \end{bmatrix} + \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix}\begin{bmatrix} \frac{\partial x_1}{\partial a_1} & \frac{\partial x_1}{\partial a_2} \\ \frac{\partial x_2}{\partial a_1} & \frac{\partial x_2}{\partial a_2} \\ \frac{\partial x_3}{\partial a_1} & \frac{\partial x_3}{\partial a_2} \end{bmatrix} \tag{2.57}$$

where the Kronecker product is:

$$\begin{aligned} \left(\mathbf{x}^T \otimes \mathbf{I}_m\right) &= \left(\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} \otimes \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\right) = \\ &= \begin{bmatrix} x_1 & 0 & 0 & x_2 & 0 & 0 & x_3 & 0 & 0 \\ 0 & x_1 & 0 & 0 & x_2 & 0 & 0 & x_3 & 0 \\ 0 & 0 & x_1 & 0 & 0 & x_2 & 0 & 0 & x_3 \end{bmatrix} \end{aligned} \tag{2.58}$$

By expanding Eqs. (2.57) and (2.58) the derivative of the matrix-vector product becomes:

$$\frac{\partial\, \mathbf{Mx}}{\partial\, \mathbf{a}^T} = \begin{bmatrix} \frac{\partial m_{11}x_1}{\partial a_1} + \frac{\partial m_{12}x_2}{\partial a_1} + \frac{\partial m_{13}x_3}{\partial a_1} & \frac{\partial m_{11}x_1}{\partial a_2} + \frac{\partial m_{12}x_2}{\partial a_2} + \frac{\partial m_{13}x_3}{\partial a_2} \\ \frac{\partial m_{21}x_1}{\partial a_1} + \frac{\partial m_{22}x_2}{\partial a_1} + \frac{\partial m_{23}x_3}{\partial a_1} & \frac{\partial m_{21}x_1}{\partial a_2} + \frac{\partial m_{22}x_2}{\partial a_2} + \frac{\partial m_{23}x_3}{\partial a_2} \\ \frac{\partial m_{31}x_1}{\partial a_1} + \frac{\partial m_{32}x_2}{\partial a_1} + \frac{\partial m_{33}x_3}{\partial a_1} & \frac{\partial m_{31}x_1}{\partial a_2} + \frac{\partial m_{32}x_2}{\partial a_2} + \frac{\partial m_{33}x_3}{\partial a_2} \end{bmatrix} \tag{2.59}$$

The second order derivative of the matrix-vector product is obtained by:

$$\begin{aligned}\frac{\partial}{\partial \mathbf{a}^T}\left(vec\left(\frac{\partial vec(\mathbf{Mx})}{\partial \mathbf{a}^T}\right)\right) = &\left[\left(\frac{\partial\, vec(\mathbf{M})}{\partial \mathbf{a}^T}\right)^T \otimes \mathbf{I}_m\right]\cdot \\ &\left(\mathbf{I}_r \otimes \left[\left(\mathbf{I}_m \otimes \mathbf{I}_m\right) vec\left(\mathbf{I}_m\right)\right]\right)\frac{\partial \mathbf{x}}{\partial \mathbf{a}^T} + \\ &\left(\mathbf{I}_s \otimes \left(\mathbf{x}^T \otimes \mathbf{I}_m\right)\right)\frac{\partial}{\partial \mathbf{a}^T}\left(vec\left(\frac{\partial vec(\mathbf{M})}{\partial \mathbf{a}^T}\right)\right) + \\ &\left[\left(\frac{\partial \mathbf{x}}{\partial \mathbf{a}^T}\right)^T \otimes \mathbf{I}_m\right]\frac{\partial\, vec(\mathbf{M})}{\partial \mathbf{a}^T} + \\ &\left[\mathbf{I}_s \otimes \mathbf{M}\right]\frac{\partial}{\partial \mathbf{a}^T}\left(vec\left(\frac{\partial vec(\mathbf{x})}{\partial \mathbf{a}^T}\right)\right)\end{aligned} \tag{2.60}$$

2.2.5 Sensitivity Equations

The sensitivity equations are the first and second order state derivatives with respect to the parameter vector **z** of the discretized optimal control problem and indicate how the state vector **x** at time t is changed if any element of the parameter vector **z** is perturbed. These sensitivities are obtained by integrating the differential equations for the state sensitivities that result from the differentiation of the state dynamics given with respect to the parameter vector **z** (Ref. [Gerdts, 2007]). Given the state dynamics

$$\dot{\mathbf{x}} = \mathbf{f}\left(\mathbf{x}(t), \mathbf{u}(t), t_f, \mathbf{p}\right) \tag{2.61}$$

the first order sensitivity equations w.r.t. the parameter vector **z** evaluate to:

$$\frac{\partial \dot{\mathbf{x}}}{\partial \mathbf{z}^T} = \frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\cdot\frac{\partial \mathbf{x}}{\partial \mathbf{z}^T} + \frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\cdot\frac{\partial \mathbf{u}}{\partial \mathbf{z}^T} + \frac{\partial \mathbf{f}}{\partial t_f}\cdot\frac{\partial t_f}{\partial \mathbf{z}^T} + \frac{\partial \mathbf{f}}{\partial \mathbf{p}^T}\cdot\frac{\partial \mathbf{p}}{\partial \mathbf{z}^T} \tag{2.62}$$

Here, the dot denotes the differentiation of the state vector **x** with respect to the normalized time τ, i.e. the time transformation is already incorporated in the function **f**. The matrices $\partial\mathbf{f}/\partial\mathbf{x}^T$, $\partial\mathbf{f}/\partial\mathbf{u}^T$, $\partial\mathbf{f}/\partial t_f$ and $\partial\mathbf{f}/\partial\mathbf{p}^T$ are the derivatives of the states dynamics with respect to the control vector **u**, the state vector **x,** the final phase time t_f and a parameter vector **p** (see Eqs. (2.49) to (2.51)). The normalized time τ is given by Eq. (2.27). With respect to the parameter vector **z** of Eq. (2.38), the derivative $\partial t_f/\partial\mathbf{z}^T$ and $\partial\mathbf{p}/\partial\mathbf{z}^T$ evaluate to:

$$\begin{aligned}\frac{\partial t_f}{\partial \mathbf{z}^T} &= \left[\frac{\partial t_f}{\partial t_f},\ \frac{\partial t_f}{\partial \mathbf{x}_{0,free}^T},\ \frac{\partial t_f}{\partial \mathbf{x}_{MS,j}^T},\ \frac{\partial t_f}{\partial \mathbf{u}^T},\ \frac{\partial t_f}{\partial \mathbf{p}^T}\right] = \\ &= \left[1,\ \mathbf{0},\ \mathbf{0},\ \mathbf{0},\ \mathbf{0}\right] \qquad\qquad j = 1,\ldots,m\end{aligned} \tag{2.63}$$

$$\frac{\partial \mathbf{p}}{\partial \mathbf{z}^T} = \left[\frac{\partial \mathbf{p}}{\partial t_f}, \quad \frac{\partial \mathbf{p}}{\partial \mathbf{x}_{0,free}^T}, \quad \frac{\partial \mathbf{p}}{\partial \mathbf{x}_{MS,j}^T}, \quad \frac{\partial \mathbf{p}}{\partial \mathbf{u}^T}, \quad \frac{\partial \mathbf{p}}{\partial \mathbf{p}^T} \right] = \tag{2.64}$$

$$= \left[\mathbf{0}, \quad \mathbf{0}, \quad \mathbf{0}, \quad \mathbf{0}, \quad \mathbf{I} \right] \qquad\qquad j = 1, ..., m$$

Accordingly, the derivative $\partial \mathbf{u}(t)/\partial \mathbf{z}^T$ is given by:

$$\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T} = \left[\frac{\partial \mathbf{u}(t)}{\partial t_f}, \quad \frac{\partial \mathbf{u}(t)}{\partial \mathbf{x}_{0,free}^T}, \quad \frac{\partial \mathbf{u}(t)}{\partial \mathbf{x}_{MS,j}^T}, \quad \frac{\partial \mathbf{u}(t)}{\partial \mathbf{u}^T}, \quad \frac{\partial \mathbf{u}(t)}{\partial \mathbf{p}^T} \right] = \tag{2.65}$$

$$= \left[\mathbf{0}, \quad \mathbf{0}, \quad \mathbf{0}, \quad \frac{\partial \mathbf{u}(t)}{\partial \mathbf{u}^T}, \quad \mathbf{0} \right] \qquad\qquad j = 1, ..., m$$

The derivative $\partial \mathbf{u}(t)/\partial \mathbf{u}^T$ depends on the type of discretization that is applied for the transformation of the infinite-dimensional optimal control problem. For linearly interpolated controls, the differentiation of $\mathbf{u}(t)$ with respect to the parameters $\mathbf{u}_i$, $i = 1, \ldots, n$ yields functions that depend solely on the grid points τ_i so that the derivatives $\partial \mathbf{u}(t)/\partial \mathbf{u}_i^T$ have to be computed only once for a specific control grid by the following formulae:

$$\frac{\partial \mathbf{u}(t)}{\partial \mathbf{u}_i^T} = \begin{cases} \mathbf{I}_{n_C} \otimes \left(1 - \dfrac{t - \tau_i}{\tau_{i+1} - \tau_i} \right) & \tau_i \le t < \tau_{i+1} \\ \mathbf{0}_{n_C} & else \end{cases} \tag{2.66}$$

$$\frac{\partial \mathbf{u}(t)}{\partial \mathbf{u}_{i+1}^T} = \begin{cases} \mathbf{I}_{n_C} \otimes \left(\dfrac{t - \tau_i}{\tau_{i+1} - \tau_i} \right) & \tau_i \le t < \tau_{i+1} \\ \mathbf{0}_{n_C} & else \end{cases} \tag{2.67}$$

where n_c is the number of controls and $\mathbf{0}$ the zero matrix. If a *B*-spline representation of degree k is chosen for the discretization of the controls, the derivative matrix $\partial \mathbf{u}(t)/\partial \mathbf{u}^T$ has to be replaced by the derivative matrix $\partial \mathbf{u}(t)/\partial \mathbf{c}^T$. The evaluation of the derivatives $\partial \mathbf{u}(t)/\partial \mathbf{c}^T$ is straightforward, resulting in the corresponding elementary *B*-splines:

$$\frac{\partial \mathbf{u}(t)}{\partial \mathbf{c}^T} = \begin{cases} \mathbf{I}_{n_C} \otimes N_{i,k}(t) & \lambda_i \le t < \lambda_{i+k}, \\ \mathbf{0}_{n_C} & else. \end{cases} \tag{2.68}$$

It is mentioned that the derivative matrix $\partial \mathbf{u}(t)/\partial \mathbf{c}^T$ can be calculated offline before the optimization starts since it does not depend on the actual value of the optimization parameter vector $\mathbf{c}$ for the controls.

The corresponding second order sensitivity equations are obtained by differentiating Eq. (2.62) with respect to the parameter vector **z**

$$\begin{aligned}\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \dot{\mathbf{x}}}{\partial \mathbf{z}^T}\right)\right) &= \frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\cdot\frac{\partial \mathbf{x}}{\partial \mathbf{z}^T}\right)\right)+\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\cdot\frac{\partial \mathbf{u}}{\partial \mathbf{z}^T}\right)\right)+\\ &+\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial t_f}\cdot\frac{\partial t_f}{\partial \mathbf{z}^T}\right)\right)+\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}^T}\cdot\frac{\partial \mathbf{p}}{\partial \mathbf{z}^T}\right)\right)\end{aligned} \tag{2.69}$$

Differentiating the single terms gives:

$$\begin{aligned}\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\cdot\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T}\right)\right) &= \left[\left(\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T}\right)^T\otimes \mathbf{I}_{n_S}\right]\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\right)\right)+\\ &+\left[\mathbf{I}_{n_{\text{var}}}\otimes\frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\right]\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T}\right)\right)\end{aligned} \tag{2.70}$$

$$\begin{aligned}\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\cdot\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T}\right)\right) &= \left[\left(\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T}\right)^T\otimes \mathbf{I}_{n_S}\right]\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\right)\right)+\\ &+\left[\mathbf{I}_{n_{\text{var}}}\otimes\frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\right]\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T}\right)\right)\end{aligned} \tag{2.71}$$

$$\begin{aligned}\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial t_f}\cdot\frac{\partial t_f}{\partial \mathbf{z}^T}\right)\right) &= \left[\left(\frac{\partial t_f}{\partial \mathbf{z}^T}\right)^T\otimes \mathbf{I}_{n_S}\right]\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial t_f}\right)\right)+\\ &+\left[\mathbf{I}_{n_{\text{var}}}\otimes\frac{\partial \mathbf{f}}{\partial t_f}\right]\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial t_f}{\partial \mathbf{z}^T}\right)\right)\end{aligned} \tag{2.72}$$

$$\begin{aligned}\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}^T}\cdot\frac{\partial \mathbf{p}}{\partial \mathbf{z}^T}\right)\right) &= \left[\left(\frac{\partial \mathbf{p}}{\partial \mathbf{z}^T}\right)^T\otimes \mathbf{I}_{n_S}\right]\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}^T}\right)\right)+\\ &+\left[\mathbf{I}_{n_{\text{var}}}\otimes\frac{\partial \mathbf{f}}{\partial \mathbf{p}^T}\right]\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{p}}{\partial \mathbf{z}^T}\right)\right)\end{aligned} \tag{2.73}$$

where n_{var} is the total number of variables of the transformed optimal control problem, i.e. the length of the parameter vector **z**. Here, the second order derivatives of the final time $\partial^2 t_f/\partial \mathbf{z}^{T2}$ and the parameter vector $\partial^2 \mathbf{p}/\partial \mathbf{z}^{T2}$ evaluate to zero. Furthermore, for linearly interpolated controls the second order derivative $\partial^2 \mathbf{u}/\partial \mathbf{z}^{T2}$ is given by a three-dimensional tensor that contains only zeros. By applying the chain rule, the mixed second order derivatives of Eqs. (2.70) to (2.73) are expanded to:

$$\begin{aligned}\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\right)\right) &= \frac{\partial}{\partial \mathbf{x}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\right)\right)\cdot\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T}+\frac{\partial}{\partial \mathbf{u}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\right)\right)\cdot\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T}+\\ &+\frac{\partial}{\partial t_f}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\right)\right)\cdot\frac{\partial t_f}{\partial \mathbf{z}^T}+\frac{\partial}{\partial \mathbf{p}}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\right)\right)\cdot\frac{\partial \mathbf{p}}{\partial \mathbf{z}^T}\end{aligned} \tag{2.74}$$

$$\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\right)\right)=\frac{\partial}{\partial \mathbf{x}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\right)\right)\cdot\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T}+\frac{\partial}{\partial \mathbf{u}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\right)\right)\cdot\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T}+ \\ +\frac{\partial}{\partial t_f}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\right)\right)\cdot\frac{\partial t_f}{\partial \mathbf{z}^T}+\frac{\partial}{\partial \mathbf{p}}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\right)\right)\cdot\frac{\partial \mathbf{p}}{\partial \mathbf{z}^T} \tag{2.75}$$

$$\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial t_f}\right)\right)=\frac{\partial}{\partial \mathbf{x}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial t_f}\right)\right)\cdot\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T}+\frac{\partial}{\partial \mathbf{u}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial t_f}\right)\right)\cdot\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T}+ \\ +\frac{\partial}{\partial t_f}\left(vec\left(\frac{\partial \mathbf{f}}{\partial t_f}\right)\right)\cdot\frac{\partial t_f}{\partial \mathbf{z}^T}+\frac{\partial}{\partial \mathbf{p}}\left(vec\left(\frac{\partial \mathbf{f}}{\partial t_f}\right)\right)\cdot\frac{\partial \mathbf{p}}{\partial \mathbf{z}^T} \tag{2.76}$$

$$\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}^T}\right)\right)=\frac{\partial}{\partial \mathbf{x}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}^T}\right)\right)\cdot\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T}+\frac{\partial}{\partial \mathbf{u}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}^T}\right)\right)\cdot\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T}+ \\ +\frac{\partial}{\partial t_f}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}^T}\right)\right)\cdot\frac{\partial t_f}{\partial \mathbf{z}^T}+\frac{\partial}{\partial \mathbf{p}}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}^T}\right)\right)\cdot\frac{\partial \mathbf{p}}{\partial \mathbf{z}^T} \tag{2.77}$$

Inserting Eqs. (2.70) to (2.77) into Eq. (2.69), one obtains for the second order sensitivity equations:

$$\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \dot{\mathbf{x}}}{\partial \mathbf{z}^T}\right)\right)=\left[\mathbf{I}_{n_{\mathrm{var}}}\otimes\frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\right]\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T}\right)\right)+\left[\left(\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T}\right)^T\otimes\mathbf{I}_{n_S}\right]\cdot \\ \cdot\left\{\frac{\partial}{\partial \mathbf{x}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\right)\right)\cdot\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T}+\frac{\partial}{\partial \mathbf{u}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\right)\right)\cdot\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T}\right\}+ \\ +\left[\mathbf{I}_{n_{\mathrm{var}}}\otimes\frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\right]\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T}\right)\right)+\left[\left(\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T}\right)^T\otimes\mathbf{I}_{n_S}\right]\cdot \\ +\left\{\frac{\partial}{\partial \mathbf{x}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\right)\right)\cdot\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T}+\frac{\partial}{\partial \mathbf{u}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\right)\right)\cdot\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T}\right\} \tag{2.78}$$

where the dependency of the state dynamics **f** w.r.t. the final time t_f and the parameter vector **p** has been omitted for readability reasons. The full version can be found in the appendix (chapter A.1).

With regard to many trajectory optimization problems, the initial and final boundary conditions $\mathbf{\Psi}_0$ and $\mathbf{\Psi}_f$, the equality and inequality constraints $\mathbf{C}_{eq}$ and $\mathbf{C}_{ineq}$ and the interior point conditions $\mathbf{r}_i$ are often combined functions of the states $\mathbf{x}(t)$, the controls $\mathbf{u}(t)$ and possibly the parameter vector **p**. The first order derivative of any function $h = h(\mathbf{x}(t), \mathbf{u}(t), \mathbf{p})$ involving the states **x**, the controls **u** and the parameters **p** with respect to the parameter vector **z** yields:

$$\frac{\partial h}{\partial \mathbf{z}^T}=\frac{\partial h}{\partial \mathbf{u}^T}\cdot\frac{\partial \mathbf{u}}{\partial \mathbf{z}^T}+\frac{\partial h}{\partial \mathbf{x}^T}\cdot\frac{\partial \mathbf{x}}{\partial \mathbf{z}^T}+\frac{\partial h}{\partial \mathbf{p}^T}\cdot\frac{\partial \mathbf{p}}{\partial \mathbf{z}^T} \tag{2.79}$$

where $\partial h/\partial \mathbf{u}^T$, $\partial h/\partial \mathbf{x}^T$ and $\partial h/\partial \mathbf{p}^T$ are the derivatives of the function h with respect to the control vector **u**, the state vector **x** respectively the parameter vector **p**. The second order derivative of function h then becomes:

$$\begin{aligned}\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial h}{\partial \mathbf{z}^T}\right)\right) &= \left[\mathbf{I}_{n_{\text{var}}} \otimes \frac{\partial h}{\partial \mathbf{x}^T}\right]\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{x}}{\partial \mathbf{z}^T}\right)\right) + \left(\frac{\partial \mathbf{x}}{\partial \mathbf{z}^T}\right)^T \cdot \\ &\left\{\frac{\partial}{\partial \mathbf{x}^T}\left(vec\left(\frac{\partial h}{\partial \mathbf{x}^T}\right)\right)\cdot\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T} + \frac{\partial}{\partial \mathbf{u}^T}\left(vec\left(\frac{\partial h}{\partial \mathbf{x}^T}\right)\right)\cdot\frac{\partial \mathbf{u}}{\partial \mathbf{z}^T}\right\} \\ &+ \left[\mathbf{I}_{n_{\text{var}}} \otimes \frac{\partial h}{\partial \mathbf{u}^T}\right]\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{u}}{\partial \mathbf{z}^T}\right)\right) + \left(\frac{\partial \mathbf{u}}{\partial \mathbf{z}^T}\right)^T \cdot \\ &\left\{\frac{\partial}{\partial \mathbf{x}^T}\left(vec\left(\frac{\partial h}{\partial \mathbf{u}^T}\right)\right)\cdot\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T} + \frac{\partial}{\partial \mathbf{u}^T}\left(vec\left(\frac{\partial h}{\partial \mathbf{u}^T}\right)\right)\cdot\frac{\partial \mathbf{u}}{\partial \mathbf{z}^T}\right\}\end{aligned} \tag{2.80}$$

Here again, the derivatives with respect to the parameter vector **p** are left out to increase the readability. Fig. 5 illustrates the sparse block structure of the Jacobian exemplarily for an optimal control problem with two phases, two controls, one multiple shooting node in each phase and with linearly interpolated controls. The triangular form of the blocks results from the fact that for linearly interpolated controls any path constraint $C(\tau_{PC,p})$ is only influenced by a control variable u_i if the evaluation time point $\tau_{PC,p}$ is larger than the control discretization time $\tau_{i\text{-}1}$.

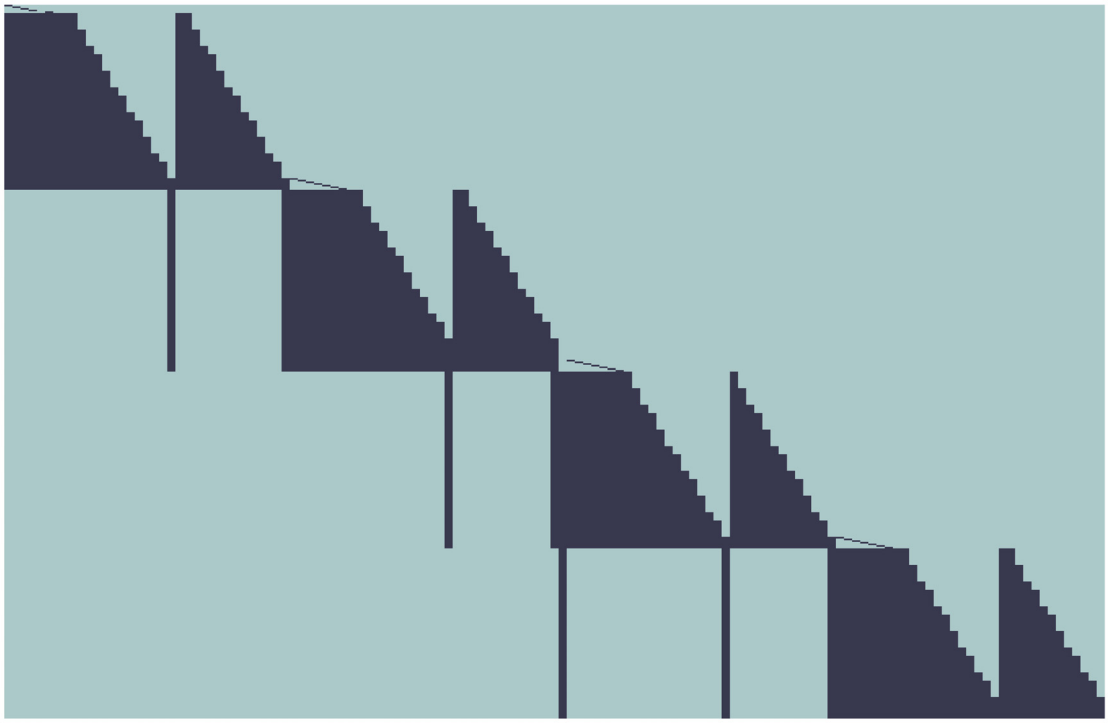

Figure 5. Sparse block structure of the Jacobian

2.2.6 Scaling

For the efficient solution of the finite-dimensional optimal control problem by the applied non-linear programming method, the scaling of the parameter vector **z** as well as the objective vector **F** plays a crucial role (Ref. [Betts, 2001]). It is the goal of the scaling procedure to adjust all elements of the parameter vector, the objective function and the constraint vector to the same order of magnitude. While the scaling of the parameter vector **z** is achieved by

$$\tilde{\mathbf{z}} = \mathbf{M} \cdot \mathbf{z} \tag{2.81}$$

the scaling of the objective vector **F** is done by

$$\widetilde{\mathbf{F}} = \mathbf{T} \cdot \mathbf{F} = \begin{bmatrix} T_J & \mathbf{0} \\ \mathbf{0} & \mathbf{T_C} \end{bmatrix} \cdot \begin{pmatrix} J \\ \mathbf{C} \end{pmatrix} \tag{2.82}$$

where the vector of constraints **C** is given by Eq. (2.40) respectively (2.45). Here, the tilde denotes the scaled entities. The matrices **M** and **T** are the respective scaling matrices, where T_J is the scaling factor associated with the objective J of Eq. (2.39) and $\mathbf{T_C}$ the diagonal scaling matrix for the constraint vector **C** of Eq. (2.40). The scaling factors are chosen such that the objective function and the elements of the parameter and the constraint vector are of the same magnitude. Then, for the scaling of the Jacobian **G** the following equations hold:

$$\widetilde{\mathbf{G}} = \frac{\partial \widetilde{\mathbf{F}}}{\partial \widetilde{\mathbf{z}}^T} = \mathbf{T} \frac{\partial \mathbf{F}}{\partial \mathbf{z}^T} \mathbf{M}^{-1} \tag{2.83}$$

Utilizing the rules for the differentiation of matrix-matrix products given in Ref. [Magnus, 1985], the scaling of the Hessian **H** is:

$$\widetilde{\mathbf{H}} = \left(\mathbf{I}_{n_x} \otimes \mathbf{T}\right)\left[\left(\mathbf{M}^{-1}\right)^T \otimes \mathbf{I}_{n_f}\right] \frac{\partial}{\partial \mathbf{z}^T} \left(vec\left(\frac{\partial \mathbf{F}}{\partial \mathbf{z}^T} \right)^T \right) \mathbf{M}^{-1} \tag{2.84}$$

where n_x is the length of the parameter vector **z** and n_f the length of the objective vector **F**. In some cases it may also be useful to restore the Lagrange multipliers **μ** for the non-scaled discretized optimal control problem from the Lagrange multipliers $\widetilde{\boldsymbol{\mu}}$ associated with the optimal solution of the scaled optimal control problem. From the scaled Lagrange function $\widetilde{L}_0$,

$$\widetilde{L}_0 = \widetilde{J} + \widetilde{\boldsymbol{\mu}} \cdot \widetilde{\mathbf{C}} = T_J \cdot J + \widetilde{\boldsymbol{\mu}} \cdot \mathbf{T_C} \cdot \mathbf{C} \tag{2.85}$$

and the relationship between the scaled and the non-scaled Lagrange functions,

$$\widetilde{L}_0 = T_J \cdot L_0 = T_J \cdot \left(J + \boldsymbol{\mu} \cdot \mathbf{C}\right) \tag{2.86}$$

the following equation for the computation of the non-scaled Lagrange multipliers **μ** results:

$$\boldsymbol{\mu} = \mathbf{T_C} \frac{\widetilde{\boldsymbol{\mu}}}{T_J} \tag{2.87}$$

2.2.7 Estimation of adjoint variables

Based on the sensitivity equations above, the estimation of the adjoint variables $\boldsymbol{\lambda}(t)$ of the optimal control problem is accomplished without much effort, utilizing the Lagrange multipliers **μ** resulting from the optimal solution of the finite-dimensional parameter optimization problem (Ref. [Büskens, 2000]):

$$\boldsymbol{\lambda}(t) = \frac{\partial L_0}{\partial \mathbf{x}(t)} \tag{2.88}$$

Here, L_0 is the augmented Lagrange function of the finite-dimensional parameter optimization problem:

$$L_0 = J + \boldsymbol{\mu}^T \mathbf{C} \tag{2.89}$$

2.2.8 Mesh Refinement

Basically, mesh refinement procedures can be divided into static methods (see e.g. Refs. [Zhao, 2009], [Jain, 2008], [Darby, 2009], [Gong, 2006] and [Betts, 1998]) and dynamic methods (Refs. [Anisi, 2006], [Teo, 2005], [Cuthrell, 1987] and [Vasantharajan, 1990]). Dynamic methods include the grid points as decision variables in the discretized optimal control problem. This means that the spacing of the grid points is not fixed and that the optimal distribution of the grid points is determined during the optimization. At this, the number of optimization variables of the discretized nonlinear programming problem is increased significantly, resulting in a degraded convergence behavior of the nonlinear programming problem and an increase of the time that is required for the computation of the optimal solution. In contrary to dynamic methods, static methods first compute the optimal solution of the nonlinear programming problem for a fixed number of grid points with predetermined distribution. Then, the current optimal solution is utilized to refine the mesh either by moving the current grid points or by inserting and deleting certain grid points. Thus, the dimension of the original nonlinear programming problem remains unchanged utilizing a static mesh refinement procedure. In the following, a static mesh refinement method is considered that adapts the mesh by inserting respectively deleting grid points based on a weighting function that is derived from the time histories of the controls.

For this weighting function w either the first order time derivative $\dot{u}$ of the respective control or alternatively the curvature κ of the control time history is utilized. Regarding linear interpolated controls and the current mesh distribution

$$\tau_0 = \tau_1 < \tau_2 < \ldots < \tau_{n-1} < \tau_n = \tau_f \tag{2.90}$$

with the corresponding control values

$$u_i, \qquad i = 1, \ldots, n \tag{2.91}$$

the first order time derivatives $\dot{u}_i$ of the control time history are obtained by

$$\dot{u}_i = \frac{u_{i+1} - u_i}{t_{i+1} - t_i}, \qquad i = 1, \ldots, n-1 \tag{2.92}$$

with the corresponding time points $\dot{t}_i$

$$\dot{t}_i = \frac{1}{2}\left(t_i + t_{i+1}\right), \qquad i = 1, \ldots, n-1 \tag{2.93}$$

Alternatively, for linear interpolated controls it is also possible to establish a weighting function based on the angle between two adjacent segments of the discretized controls (see Fig. 6):

$$\Delta\alpha_i = \arctan\left(\frac{u_{i+1} - u_i}{t_{i+1} - t_i}\right) - \arctan\left(\frac{u_i - u_{i-1}}{t_i - t_{i-1}}\right), \qquad i = 2, \ldots, n-1 \tag{2.94}$$

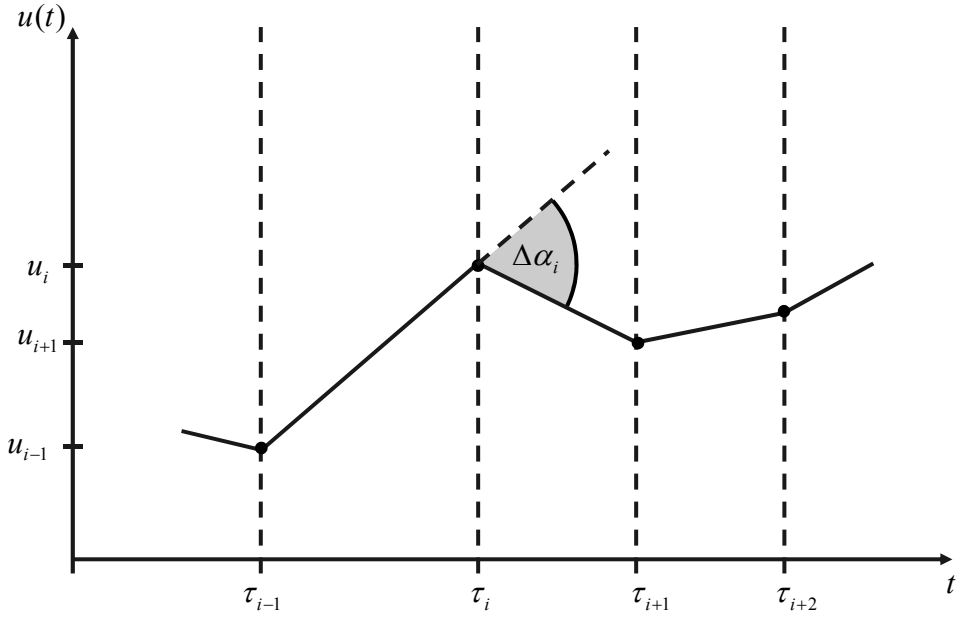

Figure 6. Angle between adjacent control segments

For any function f that is at least twice differentiable, the curvature κ is defined as

$$\kappa = \frac{\left| \ddot{f} \right|}{\left| \left(1 + \dot{f}^2\right)^{3/2} \right|} \tag{2.95}$$

Thus, for a linear interpolation of the controls, the curvature κ_i can be calculated by

$$\kappa_i = \frac{\left| \ddot{u}_i \right|}{\left| \left(1 + \dot{u}_i^{\,2}\right)^{3/2} \right|} \tag{2.96}$$

where the first order time derivative is given by Eq. (2.92). The second order time derivative $\ddot{u}_i$ of the controls can be approximated by (Ref. [Zhao, 2009]):

$$\ddot{u}_i \cong \frac{\dot{u}_{i+1} - \dot{u}_i}{\dot{t}_{i+1} - \dot{t}_i}, \qquad i = 1, \ldots, n-2 \tag{2.97}$$

The corresponding time points $\ddot{t}_i$ are given by

$$\ddot{t}_i = \frac{1}{2}\left(\dot{t}_i + \dot{t}_{i+1}\right), \qquad i = 1, \ldots, n-2 \tag{2.98}$$

Approximate values for the first order time derivative $\dot{u}$ of the controls at the nodes $\ddot{t}_i$ of the second order time derivative function $\ddot{u}$ can be obtained e.g. by a spline interpolation. Once the weighting function has been computed, the mesh is refined in the following way: Those grid points where the weighting function w lies below a certain threshold $w_{\min}$ defined by the user are removed from the grid. At grid points where the weighting function w is larger than a defined threshold $w_{\max}$, the segments directly before and behind the respective grid point are split up into two sub-segments by inserting additional grid points before and behind the respective grid point. Next, control variables u_i are detected that are at their lower or upper bounds and where the control time history enters or leaves its lower or upper boundary.

ALGORITHM I	
1	Given t_i, u_i, $i = 1, \ldots, n$ compute the weighting function w either by Eq. (2.92), by Eq. (2.94) or Eq. (2.96)
2	Refine the control mesh
2A	Delete those grid points where the weighting function w is below a certain threshold $w_{\min}$
2B	Insert grid points where the weighting function w is above a certain threshold $w_{\max}$
2C	Insert grid points where controls leave or enter bounds
2D	Insert grid points where path constraints become active or inactive
3	Refine the path constraint mesh

Table 1. Mesh Refinement Algorithm

Additional grid points are inserted before and behind the detected grid points in order to determine as exact as possible the time point at which the control enters or leaves one of its boundaries. Furthermore, at time points where inequality path constraints become active or inactive, grid points are added to all control meshes to enable a precise determination of the time point when the path constraints get active or inactive.

Besides the control grids, the path constraint grids are refined, too. Therefore, it is checked in between the current path constraint grid points where the path constraints are violated. At the time points that show the maximum violation of the path constraints, additional grid points are added to the path constraint grids.

In Table 1, an overview of the algorithm for the refinement of the control mesh and the path constraint grids is given.

3

Optimization Simulation Model

3.1 Overview

The following chapter describes the development of a scalable, multi-fidelity simulation model that is specifically tailored for optimization tasks. This simulation model provides the basis for the establishment of a robust and effective process for the solution of complex trajectory optimization problems because of its novel, sequential structure that is a scalable inner loop followed by an outer loop as depicted in Fig. 7. For the implementation of the simulation model and the optimization tasks, the full dynamic order of the regarded flight system is taken into account. Therefore it is ensured that the trajectory found by the optimization later on is dynamically realistic and can be followed by the aircraft in reality.

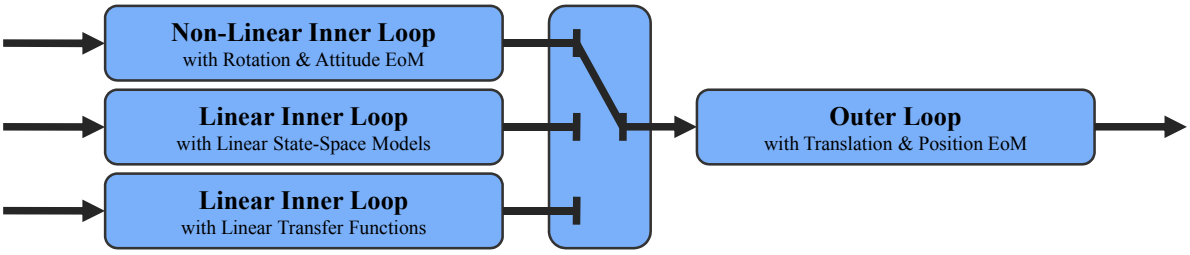

Figure 7. Simulation Model Structure with Scalable Inner Loop

The outer loop contains the nonlinear translation equations of motion as well as the position propagation equations and can therefore be considered as a quasi non-linear point mass simulation model. The inner loop represents the rotation equation of motions and the attitude dynamics of the flight system under consideration. At this, the equations of motion are formulated such that the inner loop and the outer loop can be simulated sequentially. This sequential structure follows the structure of the physical causal chains of flight systems. Physical causal chains describe the causal relationships covering flight system dynamics, e.g. between the deflections of the control surfaces and the resulting changes in the states of the aircraft. To give an example, the causal chain between an elevator deflection η and the resulting change in the altitude h of the aircraft is depicted in Fig. 8. A deflection η of the elevator control surface invokes a pitching moment M around the y-axis of the aircraft. This moment in turn effects a pitch acceleration $\dot{q}$ and therefore via an integration a pitch rate q. This pitch rate induces a time rate change $\dot{\alpha}$ of the angle of attack. An integration of this angle of attack time rate change results in a change $\Delta\alpha$ of the aerodynamic angle of attack that in turn leads to a variation ΔL of the aerodynamic lift force perpendicular to the flight path, i.e. in the direction of the z-axis of the aircraft. The variation ΔL of the lift then produces an acceleration that results in a load factor increment Δn_z parallel to the lift increment, finally

leading to a time rate change of the flight-path climb angle $\dot{\gamma}$. Integrating this change, a climb angle γ results that causes a time rate change of the altitude $\dot{h}$ of the aircraft. By a further integration, the change of the altitude yields the actual altitude h of the flight system vehicle. The portion of the causal chain from an elevator deflection to a load factor change is represented by the inner loop of the simulation model, while the part from a change in the load factor to a variation in the aircraft's altitude is assigned to the outer loop.

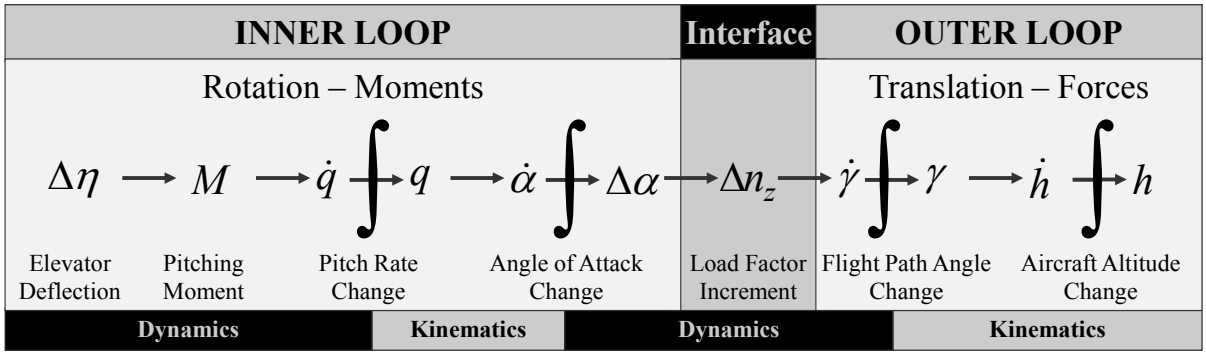

Figure 8. Causal Chain from an Elevator Deflection to a Change in the Altitude of the Aircraft

The modeling fidelity of the inner loop is scalable and can be increased from a low level of simulation model complexity represented by linear transfer functions for the load factors and the roll rate to a more sophisticated complexity level with linear state space models for the longitudinal respectively the lateral dynamics of the aircraft and finally to the most accurate flight system modeling featuring full non-linear rotation equations of motion and attitude propagation equations. At this juncture, the alternative equations for the different depths of modeling for the inner loop are formulated and implemented in such a way that the interface between the inner and the outer loop always remains the same regardless of the depth of modeling that is chosen, i.e. the modeling of the outer loop is not affected by the selected model complexity level of the inner loop. Therefore, the rotation equations of motion and the attitude propagation equations are formulated with respect to the Kinematic Flight-Path Frame K and not with respect to the local geodetic North-East-Down Reference Frame as habitual. This fundamental difference is illustrated in Fig. 9. The interface between the inner and outer loop is represented by the aerodynamic load factors $(\vec{\mathbf{n}}_A)_K$ in the Intermediate Kinematic Flight-Path Reference Frame $\bar{K}$.

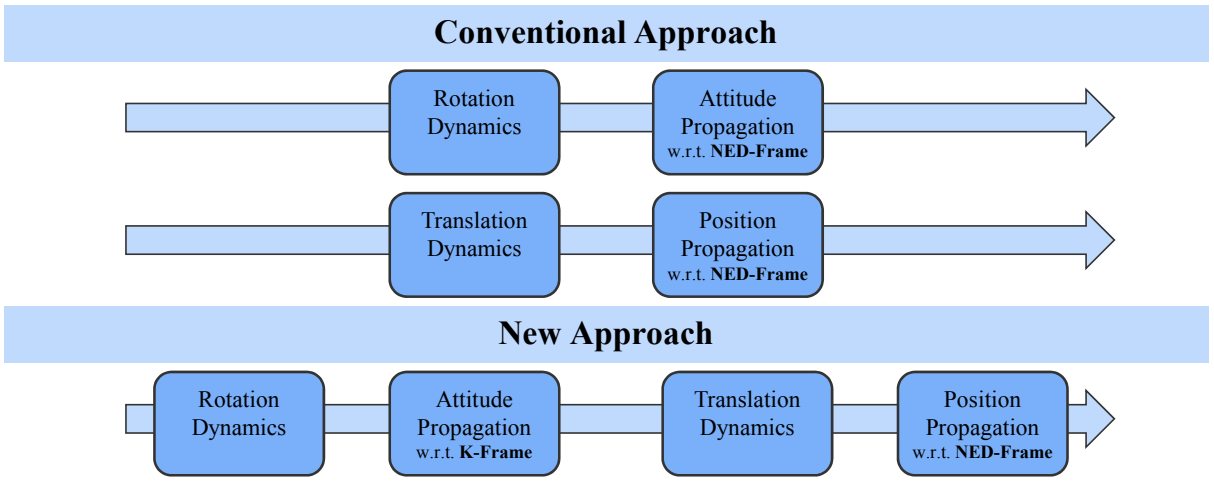

Figure 9. Conventional and New Simulation Model Structure

By the unified structure of the simulation model, the complexity of the simulation model can be scaled to varying levels of fidelity in a nearly continuous manner and it can be easily adapted to various requirements of a specific simulation or optimization task like total duration of the optimization process, operational robustness, level of complexity, accuracy of the optimized trajectory or the specific data that are available for the modeling of the flight system that is to be simulated. This is depicted in Fig. 10. Thus, the simulation model allows for an easy transition between the accuracy of the simulation respectively the optimization results and the overall time needed for the optimization process.

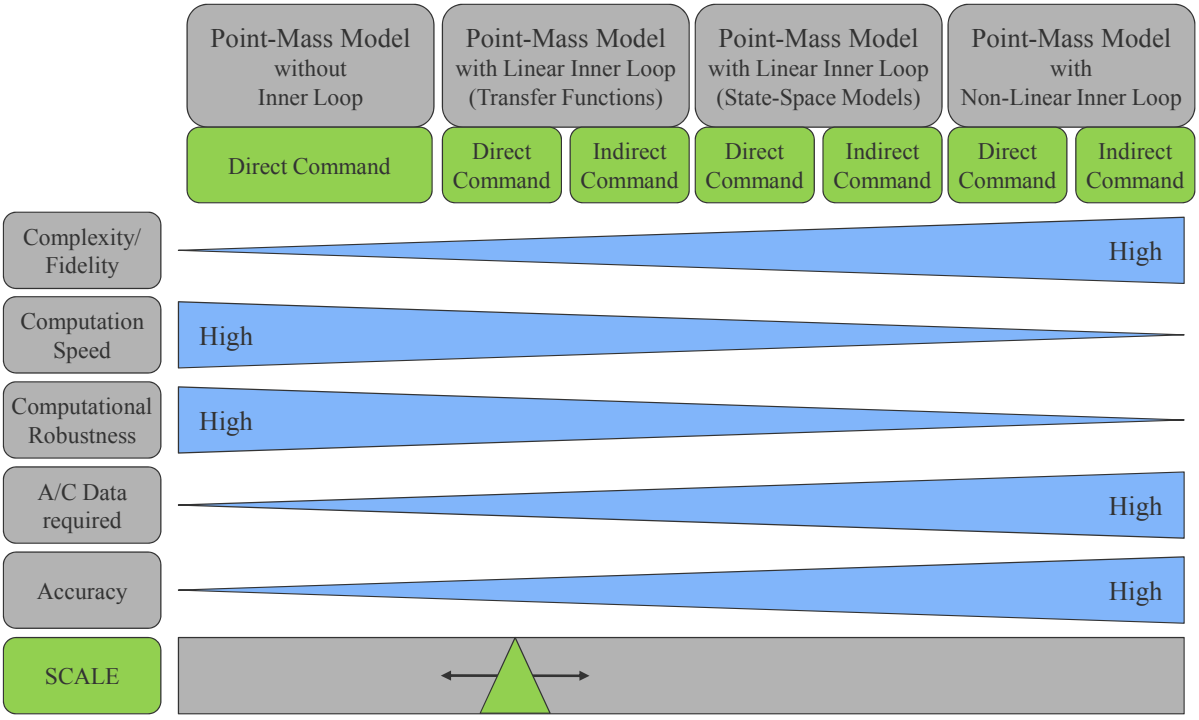

Figure 10. Scalable, Multi-Fidelity Simulation Model

Besides the rigid body dynamics, further subsystems have to be taken into account in the simulation model in order to achieve simulated and thus optimal trajectories that are compatible to the true dynamic order of the flight system and thus as realistic as possible. Therefore, for example actuator dynamics and the impact of static and dynamic properties of the atmosphere surrounding the aircraft are incorporated in the simulation model. Since especially wind is supposed to have a great influence on optimal trajectories, the influence of static, time variant and convective wind fields is included in the equations of motion to provide a maximum level of realism in the reproduction of the environment.

In addition to flight system modeling itself, a controller that is based on the principle of dynamic inversion of the physical causal chain of flight systems is implemented in the simulation model. For all depths of modeling included in the scalable simulation model, the equations of motion are inverted so that e.g. for a given trajectory the required control surface deflections can be computed. Feeding these control surface deflections forward to the simulation model will then again result in the predetermined trajectory. As another example, by utilizing only the inversion controller for the inner loop, the full non-linear 6-Degree of Freedom simulation model can be simulated making use of the same virtual controls as for the non-linear point mass model without any inner loop and inversion controller i.e. the aerodynamic angle of attack α_A, the aerodynamic sideslip angle β_A, the first order time

derivative of the aerodynamic bank angle $\dot{\mu}_A$ and the thrust lever position δ_T. This represents another essential feature of the simulation model required for the development of the optimization process to be described later. For the inversion controller to work, not only the reference values themselves but also higher order derivatives of the respective reference values are required. Therefore, the simulation model is augmented by reference models of the appropriate order to produce these derivatives in a consistent manner.

The exact dynamic inversion of a causal chain of any flight system is only possible for flight system dynamics with minimum phase behavior (see chapter 3.3.1). Thus, besides the numerical computation errors, any non-minimum phase behavior will cause the actual values of the simulation model to diverge from the reference values. In order to cope with these effects and to guarantee a precise tracking of the reference values, error feedbacks on all levels of the simulation model are implemented to attenuate possible deviations. Therefore, the various states are fed back to the highest possible derivatives of their corresponding reference values to allow the simulation model to react to any occurring deviation as fast as possible.

In the following, the equations of motion for the different depths of modeling of a rigid-body aircraft based on the underlying reference coordinate systems as well as the equations for the corresponding inversion controllers are stated, with a proper inclusion of the influences of environmental conditions, especially static and convective wind fields, and the forces and moments acting on the aircraft that result from the gravitational force, the airflow surrounding the aircraft and the propulsion system. Furthermore, the equations for the implemented error feedbacks are outlined and an overview of the various simulation model modes with the respective inputs and outputs is given.

3.2 Rigid-Body Equations of Motion

In this section, the rigid-body differential equations of motion that describe the time rate change of the states of the rigid-body aircraft in the respective degrees of freedom are given for the different depths of modeling. The differential equations of motion can be subdivided into equations for the description of the translational motion and the rotational motion and into differential equations for the determination of the position and the attitude in space. As shown in Fig. 9, in a conventional approach (Refs. [Stevens, 1992], [Philips, 2004], [Etkin, 1996], [Etkin, 2005] and [Roskam, 2001]) the rotational and attitude dynamics are in general modeled in a manner parallel to the translational and position dynamics, so that the attitude dynamics are given with respect to the *NED*-Reference Frame utilizing the set of Euler-angles that are the azimuth angle Ψ, the inclination angle Θ and the bank angle Φ. With the sequential approach, the rotational and attitude dynamics denoted as inner loop dynamics are modeled in series to the outer loop dynamics that are the translation and position equations of motion. Thus, the rotational and attitude equations of motion are not formulated with respect to the *NED*-Reference Frame but with respect to the trajectory reference frame or more precisely the Intermediate Kinematic Flight-Path Frame $\bar{K}$. The sequential approach allows one to switch between different depths of modeling for the inner loop without the necessity for any modification of the modeling of the outer loop since the differential equations are formulated in such a way that the interface between the inner and outer loop always remains the same regardless of the type of modeling for the inner loop. In the following, besides the differential equations for the translation equations of motion and the

position propagation equations, the equations for three different depths of modeling for the inner loop are described: first, linear transfer functions for the load factors and the roll rate representing the characteristic dynamics of the inner loop, second, linear state-space models for the longitudinal and the lateral motion of the aircraft, and third the full, non-linear rotation equations of motion and attitude propagation equations representing the highest model fidelity respectively depth of modeling.

Detailed derivations of the rigid-body equations of motion can be found in the literature e.g. in Refs. [Holzapfel, 2009b], [Stevens, 1992], [Philips, 2004], [Etkin, 1996], [Etkin, 2005], [Roskam, 2001], [Schmidt, 1998], [Brockhaus, 2001], [McRuer, 1990], [Nelson, 1997], [Russell, 2003], [Boiffier, 1998], [Blakelock, 1991], [Hancock, 1995] and [McCormick, 1994].

3.2.1 Scope of Validity

For the modeling of the flight system, a trade-off between the model complexity on the one side and the external validity of the implemented model on the other side has to be done and certain simplifying assumptions have to be made. This approach is legitimated by the affirmation that all requirements that are necessary for the implementation of the simplifying assumptions without any significant implications on the validity of the implemented models are fulfilled. The flight systems that are mainly taken into consideration in this work are aircraft of a relatively high stiffness with a relative small fuel consumption compared to the total mass of the aircraft. Furthermore, the speed range is limited to subsonic speeds. Thus, the following assumptions can be made:

- The vehicle is assumed to feature a quasi constant mass for the derivation of the equations of motion, i.e. the change in the linear momentum of the vehicle due to dynamic changes in the system mass is negligible. Therefore, the mass is considered quasi-stationary.
- For the derivation of the equations of motion, the vehicle is assumed to be a rigid body, i.e. that relative changes in the position of mass elements inside the system are not accounted for and that the mass distribution is considered quasi-stationary.
- The reference point is assumed to coincidence with the center of gravity point of the aircraft.
- The *ECI*-Reference Frame is considered to be a valid Euclidean frame, i.e. a system where the residual acceleration is negligible so that Newton's 2nd law may be applied.
- The angular speed $\tilde{\boldsymbol{\omega}}^{IE}$ of the *ECEF*-Reference Frame with respect to the *ECI*-Frame is assumed to be constant in both absolute value and direction. This means that the slow variations in the Earth's rotational axis and rotation rate are neglected.

3.2.2 Position Propagation Equations

There are various possibilities to select a triple of states that can be used to describe the position of an aircraft in space. E.g. the aircraft's position can either be given with respect to the *ECEF*-Frame or with respect to a Navigation-Frame *N*. The associated equations for the position propagation with respect to the *ECEF*-Frame are given by Eq. (3.1) where the position of a point in space is specified according to the World Geodetic System WGS84

(Ref. [NN, 2000]) utilizing two angles and the altitude above the reference ellipsoid namely the geodetic longitude λ, the geodetic latitude μ and the geodetic altitude h:

$$\begin{pmatrix} \dot{\lambda} \\ \dot{\mu} \\ \dot{h} \end{pmatrix}_O^E = \begin{pmatrix} \dfrac{\left(V_K^G\right)_K^E \cdot \sin\chi_K^G \cdot \cos\gamma_K^G}{\left(N_\mu + (h)_O^E\right) \cdot \cos(\mu)_O^E} \\ \dfrac{\left(V_K^G\right)_K^E \cdot \cos\chi_K^G \cdot \cos\gamma_K^G}{M_\mu + (h)_O^E} \\ \left(V_K^G\right)_K^E \cdot \sin\gamma_K^G \end{pmatrix} \tag{3.1}$$

$$N_\mu = \frac{a}{\sqrt{1 - e^2 \sin^2(\mu)_O^E}} \tag{3.2}$$

$$M_\mu = N_\mu \cdot \frac{1 - e^2}{1 - e^2 \sin^2(\mu)_O^E} \tag{3.3}$$

In Eq. (3.1) V_K denotes the kinematic velocity of the aircraft, χ_K the kinematic course angle and γ_K the kinematic flight path inclination angle. The radius of curvature in the prime vertical N_μ and the meridian radius of curvature M_μ are computed from the semi-major axis length a of the reference ellipsoid and the first eccentricity e, where the first eccentricity e has to be calculated from the flattening f:

$$e^2 = 2f - f^2 \tag{3.4}$$

The flattening f is defined as:

$$f = \frac{a - b}{a} \tag{3.5}$$

with b being the length of the semi-minor axis of the reference ellipsoid. In WGS84, the values for the semi-major axis length a and the semi-minor axis length b are set to *6378137,0 m* respectively *6356752,3142 m.* Fig. 11 depicts the reference ellipsoid of WGS84 as well as the associated values.

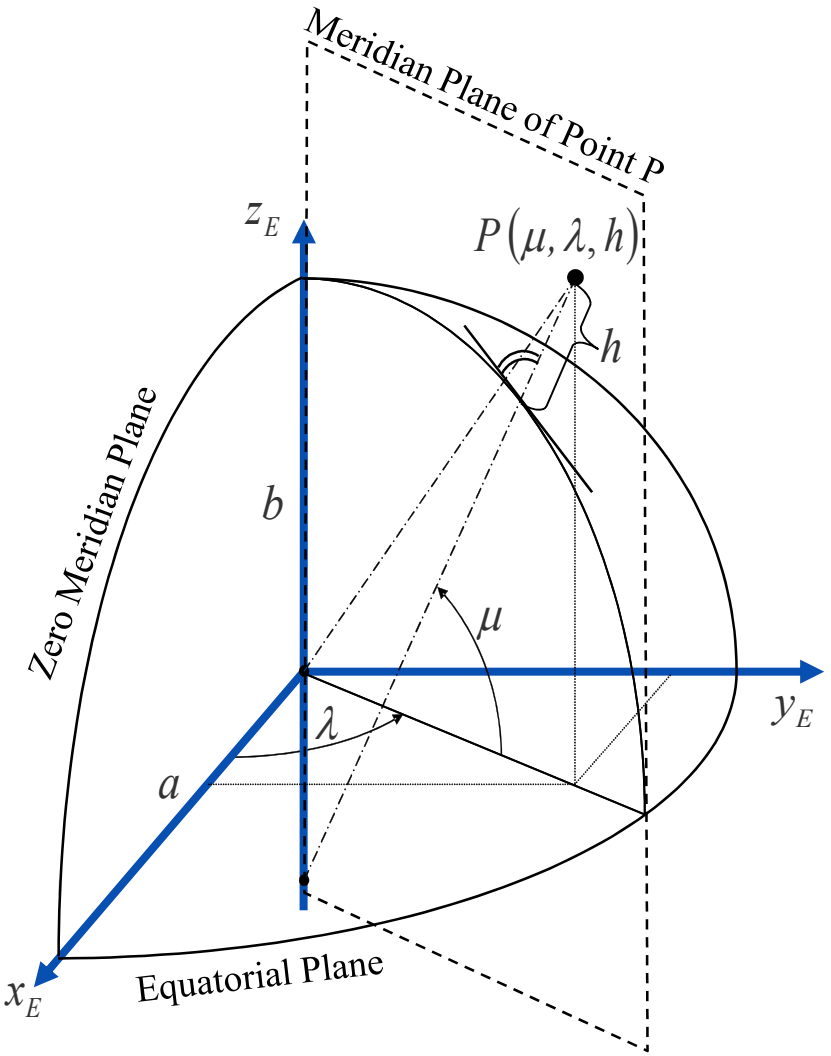

Figure 11. WGS84 Reference Ellipsoid

Thus, the position state vector of the flight system consists of the geodetic latitude μ, the geodetic longitude λ and the geodetic height h. A Navigation-Frame N can be derived from the *NED*-Frame and is used to specify the position of an aircraft in a local Cartesian coordinate system with its x-axis rotated to an arbitrary heading. The origin of a Navigation-Frame N is fixed to a certain location on the Earth surface and the frame itself is rotated about the navigation angle ψ_N around the z-axis of the *NED*-Frame. The corresponding position equations of motion with respect to such a Navigation-Frame N are given by Eq. (3.6), where the matrix $\mathbf{M}_{NO}$ is the transformation matrix between the Navigation-Frame N and the *NED*-Frame:

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix}_N^E = \mathbf{M}_{NO} \cdot \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix}_O^E = \mathbf{M}_{NO} \cdot \begin{pmatrix} u_K^G \\ v_K^G \\ w_K^G \end{pmatrix}_O^E = \mathbf{M}_{NO} \cdot \begin{pmatrix} V_K^G \cdot \cos\chi_K^G \cdot \cos\gamma_K^G \\ V_K^G \cdot \sin\chi_K^G \cdot \cos\gamma_K^G \\ -V_K^G \cdot \sin\gamma_K^G \end{pmatrix}_O \tag{3.6}$$

$$\mathbf{M}_{NO} = \begin{pmatrix} \cos\psi_N & \sin\psi_N & 0 \\ -\sin\psi_N & \cos\psi_N & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.7}$$

Here, the position state vector is made up of the three states *x*, *y* and *z* that are the coordinates in the local Navigation-Frame *N*. If the Navigation-Frame *N* coincides with the *NED*-Frame (i.e. $\psi_N = 0$), *x* represents the coordinate position in a northward direction, *y* the coordinate position in an eastward direction and *z* the coordinate position in the downward direction. The local Navigation-Frame *N* is well-suited to generate an easily interpreted graphically image of trajectories that are characterized by small geographic extents as it is the case for Red Bull Air Races.

3.2.3 Translation Equations of Motion

The basis for the formulation of the differential equations for dynamic systems is Newton's 2nd law that states that the rate of change of the linear momentum $\vec{\mathbf{p}}$ with respect to an inertial (i.e. not accelerated) frame is proportional to the sum of external forces $\Sigma\vec{\mathbf{F}}$ acting on the vehicle (Ref. [Holzapfel, 2009b]):

$$\sum \vec{\mathbf{F}}^G = \left(\frac{d}{dt}\right)^I \vec{\mathbf{p}}^G = \left(\frac{d}{dt}\right)^I \int_m \vec{\mathbf{V}}^I\left(\vec{\mathbf{x}}^P\right) \cdot dm \tag{3.8}$$

For Newton's 2nd law to be valid the velocity $\vec{\mathbf{V}}^I$ of an arbitrary mass element *P* at the position $\vec{\mathbf{x}}^P$ has to be given with respect to a reference frame that is not accelerated, i.e. the velocity has to be given with respect to an Inertial or an Euclidean frame *I*. The integration is carried out over all mass elements of the considered flight system vehicle. Assuming that the mass of the aircraft is quasi-stationary, i.e. $\dot{m} = dm/dt = 0$, Eq. (3.8) gives:

$$\sum \vec{\mathbf{F}}^G = \left(\frac{d}{dt}\right)^I \vec{\mathbf{p}}^G = \int_m \vec{\mathbf{a}}^{II}\left(\vec{\mathbf{x}}^P\right) \cdot dm \tag{3.9}$$

where $\vec{\mathbf{a}}^{II}$ is the total acceleration of the vehicle. Assuming that the aircraft is a rigid body, i.e.

$$\dot{\vec{\mathbf{r}}}^{RP} = 0, \tag{3.10}$$

the total acceleration of an arbitrary point *R* of the vehicle w.r.t. the *ECI*-Frame is:

$$\begin{aligned} \vec{\mathbf{a}}^{II} = &\left(\dot{\vec{\mathbf{V}}}_K^R\right)^{EO} + \left(\vec{\boldsymbol{\omega}}_K^{EO}\right) \times \left(\vec{\mathbf{V}}_K^R\right)^E + 2 \cdot \left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left(\vec{\mathbf{V}}_K^R\right)^E + \left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left[\left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left(\vec{\mathbf{r}}^R\right)\right] \\ &+ \left[\left(\dot{\vec{\boldsymbol{\omega}}}_K^{IB}\right)^O + \left(\vec{\boldsymbol{\omega}}_K^{IO}\right) \times \left(\vec{\boldsymbol{\omega}}_K^{OB}\right)\right] \times \left(\vec{\mathbf{r}}^{RP}\right) + \left(\vec{\boldsymbol{\omega}}_K^{IB}\right) \times \left[\left(\vec{\boldsymbol{\omega}}_K^{IO}\right) \times \left(\vec{\mathbf{r}}^{RP}\right)\right] \end{aligned} \tag{3.11}$$

Here, $\vec{\boldsymbol{\omega}}^{EO}$ denotes the transport rate, i.e. the rotational rate between the *ECEF*-Frame and the *NED*-Frame and $\vec{\boldsymbol{\omega}}^{IE}$ the rotational rate of the Earth. In Eq. (3.11) the *ECI*-Frame is chosen as reference frame since together with the transformation given by Eqs. (3.21) respectively (3.22) one finally obtains the differential equations for the kinematic flight-path course angle χ_K and the kinematic flight-path inclination angle γ_K. Thus, the kinematic flight-path course angle χ_K and the kinematic flight-path inclination angle γ_K are states of the simulation model. This is essential for the sequential structure of the simulation model. Inserting Eq. (3.11) into Eq. (3.9), one obtains:

$$\begin{aligned} \sum \vec{\mathbf{F}}^G = \int_m \Big\{ &\left(\dot{\vec{\mathbf{V}}}_K^R\right)^{EO} + \left(\vec{\boldsymbol{\omega}}_K^{EO}\right) \times \left(\vec{\mathbf{V}}_K^R\right)^E + 2 \cdot \left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left(\vec{\mathbf{V}}_K^R\right)^E + \left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left[\left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left(\vec{\mathbf{r}}^R\right)\right] + \\ &+ \left[\left(\dot{\vec{\boldsymbol{\omega}}}_K^{IB}\right)^O + \left(\vec{\boldsymbol{\omega}}_K^{IO}\right) \times \left(\vec{\boldsymbol{\omega}}_K^{OB}\right)\right] \times \left(\vec{\mathbf{r}}^{RP}\right) + \left(\vec{\boldsymbol{\omega}}_K^{IB}\right) \times \left[\left(\vec{\boldsymbol{\omega}}_K^{IO}\right) \times \left(\vec{\mathbf{r}}^{RP}\right)\right] \Big\} \cdot dm \end{aligned} \tag{3.12}$$

Now the constant terms can be extracted from the integral:

$$\begin{aligned}\sum \vec{\mathbf{F}}^{G} = \int_m dm \cdot \Big\{ &\left(\dot{\vec{\mathbf{V}}}_K^R\right)^{EO} + \left(\vec{\boldsymbol{\omega}}_K^{EO}\right) \times \left(\vec{\mathbf{V}}_K^R\right)^{E} + 2 \cdot \left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left(\vec{\mathbf{V}}_K^R\right)^{E} \\ &+ \left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left[\left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left(\vec{\mathbf{r}}^{R}\right)\right] \Big\} + \left[\left(\dot{\vec{\boldsymbol{\omega}}}_K^{IB}\right)^{O} + \left(\vec{\boldsymbol{\omega}}_K^{IO}\right) \times \left(\vec{\boldsymbol{\omega}}_K^{OB}\right)\right] \times \int_m \left(\vec{\mathbf{r}}^{RP}\right) dm + \\ &+ \left(\vec{\boldsymbol{\omega}}_K^{IB}\right) \times \left[\left(\vec{\boldsymbol{\omega}}_K^{IO}\right) \times \int_m \left(\vec{\mathbf{r}}^{RP}\right) dm\right]\end{aligned} \tag{3.13}$$

The position vector from the reference point *R* to an arbitrary point *P* is then split up into:

$$\vec{\mathbf{r}}^{RP} = \vec{\mathbf{r}}^{RG} + \vec{\mathbf{r}}^{GP} \tag{3.14}$$

where *G* is the center of gravity. Inserting Eq. (3.14) into Eq. (3.13) gives:

$$\begin{aligned}\sum \vec{\mathbf{F}}^{G} = m \cdot \Big\{ &\left(\dot{\vec{\mathbf{V}}}_K^R\right)^{EO} + \left(\vec{\boldsymbol{\omega}}_K^{EO}\right) \times \left(\vec{\mathbf{V}}_K^R\right)^{E} + 2 \cdot \left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left(\vec{\mathbf{V}}_K^R\right)^{E} \\ &+ \left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left[\left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left(\vec{\mathbf{r}}^{R}\right)\right] \Big\} + \left[\left(\dot{\vec{\boldsymbol{\omega}}}_K^{IB}\right)^{O} + \left(\vec{\boldsymbol{\omega}}_K^{IO}\right) \times \left(\vec{\boldsymbol{\omega}}_K^{OB}\right)\right] \times \int_m \left(\vec{\mathbf{r}}^{RG} + \vec{\mathbf{r}}^{GP}\right) dm + \\ &+ \left(\vec{\boldsymbol{\omega}}_K^{IB}\right) \times \left[\left(\vec{\boldsymbol{\omega}}_K^{IO}\right) \times \int_m \left(\vec{\mathbf{r}}^{RG} + \vec{\mathbf{r}}^{GP}\right) dm\right]\end{aligned} \tag{3.15}$$

With the definition of the center of gravity,

$$\int_m \left(\vec{\mathbf{r}}^{GP}\right) \cdot dm = 0 \tag{3.16}$$

Eq. (3.15) simplifies to:

$$\begin{aligned}\sum \vec{\mathbf{F}}^{G} = m \cdot \Big\{ &\left(\dot{\vec{\mathbf{V}}}_K^R\right)^{EO} + \left(\vec{\boldsymbol{\omega}}_K^{EO}\right) \times \left(\vec{\mathbf{V}}_K^R\right)^{E} + 2 \cdot \left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left(\vec{\mathbf{V}}_K^R\right)^{E} \\ &+ \left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left[\left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left(\vec{\mathbf{r}}^{R}\right)\right] \Big\} + \left[\left(\dot{\vec{\boldsymbol{\omega}}}_K^{IB}\right)^{O} + \left(\vec{\boldsymbol{\omega}}_K^{IO}\right) \times \left(\vec{\boldsymbol{\omega}}_K^{OB}\right)\right] \times \left(\vec{\mathbf{r}}^{RG}\right) \int_m dm + \\ &+ \left(\vec{\boldsymbol{\omega}}_K^{IB}\right) \times \left[\left(\vec{\boldsymbol{\omega}}_K^{IO}\right) \times \left(\vec{\mathbf{r}}^{RG}\right) \int_m dm\right]\end{aligned} \tag{3.17}$$

where the constant position vector from the reference point *R* to the center of gravity *G* has been extracted from the integral. Furthermore, assuming that the reference point *R* coincidences with the center of gravity *G* of the aircraft, i.e.

$$\vec{\mathbf{r}}^{RG} = 0 \tag{3.18}$$

one gets:

$$\begin{aligned}\sum \vec{\mathbf{F}}^{G} = m \cdot \Big\{ &\left(\dot{\vec{\mathbf{V}}}_K^G\right)^{EO} + \left(\vec{\boldsymbol{\omega}}_K^{EO}\right) \times \left(\vec{\mathbf{V}}_K^G\right)^{E} + 2 \cdot \left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left(\vec{\mathbf{V}}_K^G\right)^{E} \\ &+ \left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left[\left(\vec{\boldsymbol{\omega}}_K^{IE}\right) \times \left(\vec{\mathbf{r}}^{G}\right)\right] \Big\}\end{aligned} \tag{3.19}$$

Thus, based on the simplifying assumptions made above, the following translation equations of motion for the simulation model with the components given in the Kinematic Flight-Path Reference Frame *K* result:

$$\left(\dot{\vec{\mathbf{V}}}_K^G\right)_K^{EO} = \begin{pmatrix} \dot{u}_K^G \\ \dot{v}_K^G \\ \dot{w}_K^G \end{pmatrix}_K^{EO} = \frac{1}{m}\left(\sum \vec{\mathbf{F}}^G\right)_K - \left(\vec{\boldsymbol{\omega}}_K^{EO}\right)_K \times \left(\vec{\mathbf{V}}_K^G\right)_K^E - 2 \cdot \left(\vec{\boldsymbol{\omega}}_K^{IE}\right)_K \times \left(\vec{\mathbf{V}}_K^G\right)_K^E - \left(\vec{\boldsymbol{\omega}}_K^{IE}\right)_K \times \left[\left(\vec{\boldsymbol{\omega}}_K^{IE}\right)_K \times \left(\vec{\mathbf{r}}^G\right)_K\right] \tag{3.20}$$

Here, m denotes the mass of the aircraft. The translation equations of motion of the aircraft are described by the first order time derivatives of the states u_K, v_K and w_K, the components of the kinematic velocity vector $\vec{\mathbf{V}}_K$ given in the Kinematic Flight-Path Reference Frame. The resulting sum of external forces $\Sigma\vec{\mathbf{F}}$ acting on the aircraft is primarily composed of the gravitational force, the aerodynamic forces as well as the propulsion forces. Since the choice of the system's states is not unique, the translation equations of motion can also be formulated with respect to the states kinematic velocity V_K, the kinematic flight-path climb angle γ_K and the flight-path course angle χ_K. Therefore, the following transformation of the velocity vector $\vec{\mathbf{V}}_K$ has to be done:

$$\left(\dot{\vec{\mathbf{V}}}_K^G\right)_K^{EO} = \left(\dot{\vec{\mathbf{V}}}_K^G\right)_K^{EK} + \left(\vec{\boldsymbol{\omega}}_K^{OK}\right)_K \times \left(\vec{\mathbf{V}}_K^G\right)_K^E \tag{3.21}$$

$$\left(\dot{\vec{\mathbf{V}}}_K^G\right)_K^{EO} = \begin{pmatrix} \dot{V}_K^G \\ 0 \\ 0 \end{pmatrix}_K^{EK} + \begin{pmatrix} -\dot{\chi}_K^G \cdot \sin\gamma_K^G \\ \dot{\gamma}_K^G \\ \dot{\chi}_K^G \cdot \cos\gamma_K^G \end{pmatrix}_K \times \begin{pmatrix} V_K^G \\ 0 \\ 0 \end{pmatrix}_K^E = \begin{pmatrix} \dot{V}_K^G \\ \dot{\chi}_K^G \cdot V_K^G \cdot \cos\gamma_K^G \\ -V_K^G \cdot \dot{\gamma}_K^G \end{pmatrix}_K^{EO} \tag{3.22}$$

Equating the preceding equations (3.20) and (3.22) and solving for the first order time derivatives of the states kinematic velocity V_K, kinematic flight-path climb angle γ_K and flight-path course angle χ_K, the translation equations of motion become:

$$\begin{pmatrix} \dot{V}_K^G \\ \dot{\chi}_K^G \\ \dot{\gamma}_K^G \end{pmatrix}_K^{EO} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \dfrac{1}{V_K^G \cdot \cos\gamma_K^G} & 0 \\ 0 & 0 & -\dfrac{1}{V_K^G} \end{bmatrix} \cdot \left\{ \frac{1}{m}\left(\sum \vec{\mathbf{F}}^G\right)_K - \left(\vec{\boldsymbol{\omega}}_K^{EO}\right)_K \times \left(\vec{\mathbf{V}}_K^G\right)_K^E - 2 \cdot \left(\vec{\boldsymbol{\omega}}_K^{IE}\right)_K \times \left(\vec{\mathbf{V}}_K^G\right)_K^E - \left(\vec{\boldsymbol{\omega}}_K^{IE}\right)_K \times \left[\left(\vec{\boldsymbol{\omega}}_K^{IE}\right)_K \times \left(\vec{\mathbf{r}}^G\right)_K\right] \right\} \tag{3.23}$$

Assuming that merely flight over flat, non-rotating Earth is considered, the rotational rates $\vec{\boldsymbol{\omega}}^{IE}$ and $\vec{\boldsymbol{\omega}}^{EO}$ equal zero, resulting in the following simplified translation equations of motion:

$$\dot{V}_K^G = \frac{\left(F_x^G\right)_K}{m} \tag{3.24}$$

$$\dot{\chi}_K^G = \frac{\left(F_y^G\right)_K}{m \cdot V_K^G \cdot \cos\gamma_K^G} \tag{3.25}$$

$$\dot{\gamma}_K^G = -\frac{\left(F_z^G\right)_K}{m \cdot V_K^G} \tag{3.26}$$

Formulating the translation equation of motions in such a way means that a singularity emerges when the aircraft reaches a kinematic flight-path angle of $\gamma_K = \pm 90°$. This singularity is similar to the singularity that occurs when the inclination angle Θ equals ±90° if the set of Euler-angles is used to describe the attitude of an aircraft. For simulating and optimizing aerobatic and all attitude trajectories including aerobatic maneuvers like the Half Cuban Eight, it is mandatory to find a way to cope with this singularity. Thus, the simulation model has to be modified so that the occurrence of these singularities has no influence on the performance of the simulation model. Regarding Fig. 12, the sequential approach for the implementation of the simulation model follows the path from the *NED*-Frame to the Kinematic Flight-Path Frame *K* respectively the Intermediate Kinematic Flight-Path Frame $\bar{K}$ to the Body-Fixed Frame *B* for a complete description of the attitude of the aircraft. Thus, five angles are required as states that are the kinematic flight-path bank angle μ_K and the kinematic attitude angles angle of attack α_K and angle of sideslip β_K besides the kinematic flight-path angles γ_K and χ_K.

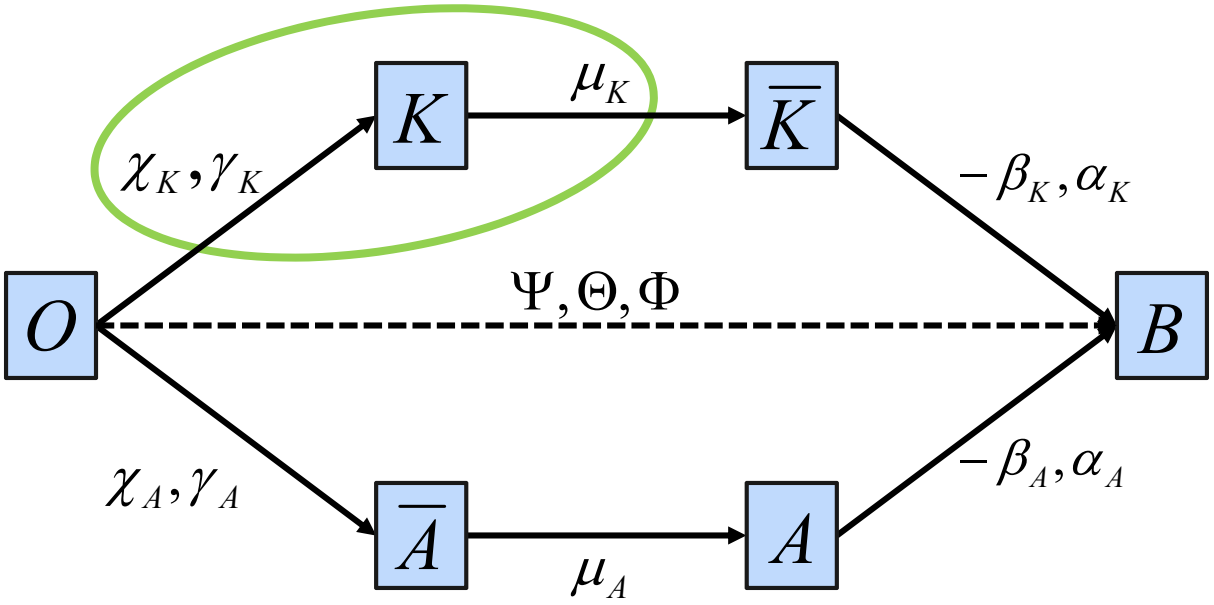

Figure 12. Coordinate Systems

At this, the differential equations for the kinematic flight-path bank angle μ_K and the kinematic attitude angles α_K and β_K are given by the attitude dynamics (chapter 3.2.4, Eq. (3.43)). Instead of regarding the states resulting from the attitude dynamics as a separate set of states, the kinematic flight-path bank angle μ_K is regarded as a state belonging to the kinematic flight-path angles γ_K and χ_K so that a set of states is formed by the kinematic flight-path angles γ_K and χ_K and the kinematic flight-path bank angle μ_K (see Fig. 12). Then, the translation equations of motion can be formulated using the four quaternions q_0, q_1, q_2 and q_3 (Refs. [Stevens, 1992], [Philips, 2004]) instead of the three kinematic flight-path angles γ_K, χ_K and μ_K. By utilizing the quaternions, singularities that occur for climb angles of ±90° are avoided and the flight-path angles can be determined without ambiguity. The translation equations of motion using the four quaternions then read:

$$\begin{pmatrix}\dot{q}_0\\ \dot{q}_1\\ \dot{q}_2\\ \dot{q}_3\end{pmatrix}=\frac{1}{2}\begin{bmatrix}-q_1 & -q_2 & -q_3 & q_0\\ q_0 & -q_3 & q_2 & q_1\\ q_3 & q_0 & -q_1 & q_2\\ -q_2 & q_1 & q_0 & q_3\end{bmatrix}\cdot\begin{pmatrix}\left(\omega_{K,x}^{O\bar{K}}\right)_{\bar{K}}\\ \left(\omega_{K,y}^{O\bar{K}}\right)_{\bar{K}}\\ \left(\omega_{K,z}^{O\bar{K}}\right)_{\bar{K}}\\ 2k\lambda\end{pmatrix} \tag{3.27}$$

Here it is mentioned that the differential equation for the kinematic flight-path bank angle μ_K resulting from Eq. (3.43) is not integrated as a state but acts as input into Eq. (3.30) for the calculation of the angular rate $(\vec{\boldsymbol{\omega}}^{OK})_{\bar{K}}$. Since the quaternions introduce an additional state, the dynamic system is now over-determined and thus constrained. The algebraic constraint to be met is that the square sum of the quaternions equals 1 (Eq. (3.28)). As after a number of performed integration steps the build-up of computational errors will eventually cause the quaternion vector to take a non-unit length and thus violate the constraint so that the vector is not suited to describe the underlying set of flight-path angles any more. Thus, an additional constraint has to be enforced on the quaternions:

$$\left(q_0^2+q_1^2+q_2^2+q_3^2\right)=1 \tag{3.28}$$

The constraint is enforced by accounting for the square sum error λ computed by Eq. (3.29) in the quaternion update equations (3.27) in the fourth column. The square sum error λ acts as an integration drift correction gain that attempts to drive the quaternion vector back to unit length if any deviation results from computational errors during integration (Refs. [Pamadi, 1998] and [Rolfe, 1986]).

$$\lambda=1-\left(q_0^2+q_1^2+q_2^2+q_3^2\right) \tag{3.29}$$

In Eq. (3.27), the constant k represents a correction step factor and may be chosen such that $kh < 1$ for a fixed integration step size h (Ref. [Rolfe, 1986]). For $kh = 1$, the entire correction step is performed within one iteration cycle, whereas for $0 < kh < 1$, the deviation decreases exponentially.

As can be seen from the quaternion update equations (3.27) the angular rate $(\vec{\boldsymbol{\omega}}^{OK})_{\bar{K}}$ between the *NED*-Frame and the Intermediate Kinematic Flight-Path Frame $\bar{K}$ has to be determined which can be done in two alternative ways: By the first alternative, the first order time derivatives of the kinematic flight-path climb angle $\dot{\gamma}_K$ and course angle $\dot{\chi}_K$ together with the first order time derivative of the kinematic flight-path bank angle $\dot{\mu}_K$ are used to calculate the required angular rate $(\vec{\boldsymbol{\omega}}^{OK})_{\bar{K}}$ with the help of the following equation that can be derived from the appropriate strap-down equation (Refs. [Stevens, 2003] and [Holzapfel, 2009b]):

$$\left(\vec{\boldsymbol{\omega}}_K^{O\bar{K}}\right)_{\bar{K}}=\begin{pmatrix}\dot{\mu}_K^G-\dot{\chi}_K^G\sin\gamma_K^G\\ \dot{\chi}_K^G\cos\gamma_K^G\sin\mu_K^G+\dot{\gamma}_K^G\cos\mu_K^G\\ \dot{\chi}_K^G\cos\gamma_K^G\cos\mu_K^G-\dot{\gamma}_K^G\sin\mu_K^G\end{pmatrix}_{\bar{K}} \tag{3.30}$$

By the second alternative, at first the *y*- and *z*-component of the angular rate $(\vec{\boldsymbol{\omega}}^{OK})_K$ are calculated by putting Eq. (3.22) equal to Eq. (3.31):

$$\left(\dot{\vec{\mathbf{V}}}_K^G\right)_K^{EO} = \begin{pmatrix} \dot{V}_K^G \\ 0 \\ 0 \end{pmatrix}_K^{EK} + \begin{pmatrix} \omega_{K,x}^{OK} \\ \omega_{K,y}^{OK} \\ \omega_{K,z}^{OK} \end{pmatrix}_K \times \begin{pmatrix} V_K^G \\ 0 \\ 0 \end{pmatrix}_K^E = \begin{pmatrix} \dot{V}_K^G \\ V_K^G \cdot \omega_{K,z}^{OK} \\ -V_K^G \cdot \omega_{K,y}^{OK} \end{pmatrix}_K^{EO} \tag{3.31}$$

where the angular rate $(\vec{\boldsymbol{\omega}}^{OK})_K$ is given by:

$$\left(\vec{\boldsymbol{\omega}}_K^{OK}\right)_K = \begin{pmatrix} -\dot{\chi}_K^G \sin\gamma_K^G \\ \dot{\gamma}_K^G \\ \dot{\chi}_K^G \cos\gamma_K^G \end{pmatrix}_K \tag{3.32}$$

This gives the following relationships for the *y*- and *z*-component of the angular rate $(\vec{\boldsymbol{\omega}}^{OK})_K$:

$$\left(\omega_{K,y}^{OK}\right)_K = \dot{\gamma}_K^G \tag{3.33}$$

$$\left(\omega_{K,z}^{OK}\right)_K = \dot{\chi}_K^G \cdot \cos\gamma_K^G \tag{3.34}$$

The *x*-component of the angular rate $(\vec{\boldsymbol{\omega}}^{OK})_K$ can then be calculated using the relationship between the *x*- and *z*-component of the angular rate $(\vec{\boldsymbol{\omega}}^{OK})_K$ (Eq. (3.32)):

$$\left(\omega_{K,x}^{OK}\right)_K = -\dot{\chi}_K^G \cdot \sin\gamma_K^G = -\frac{\left(\omega_{K,z}^{OK}\right)_K}{\cos\gamma_K^G} \cdot \sin\gamma_K^G = -\left(\omega_{K,z}^{OK}\right)_K \cdot \tan\gamma_K^G \tag{3.35}$$

Together with the first order time derivative of the kinematic flight-path bank angle $\dot{\mu}_K$, finally the required angular rate $(\vec{\boldsymbol{\omega}}^{O\bar{K}})_{\bar{K}}$ can be computed:

$$\left(\vec{\boldsymbol{\omega}}_K^{O\bar{K}}\right)_{\bar{K}} = \mathbf{M}_{\bar{K}K}\left(\vec{\boldsymbol{\omega}}_K^{OK}\right)_K + \left(\vec{\boldsymbol{\omega}}_K^{K\bar{K}}\right)_{\bar{K}} = \mathbf{M}_{\bar{K}K}\begin{pmatrix} \omega_{K,x}^{OK} \\ \omega_{K,y}^{OK} \\ \omega_{K,z}^{OK} \end{pmatrix}_K + \begin{pmatrix} \dot{\mu}_K^G \\ 0 \\ 0 \end{pmatrix}_{\bar{K}} \tag{3.36}$$

where the transformation matrix $\mathbf{M}_{\bar{K}K}$ is given by:

$$\mathbf{M}_{\bar{K}K} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\mu_K & \sin\mu_K \\ 0 & -\sin\mu_K & \cos\mu_K \end{bmatrix} \tag{3.37}$$

After the integration of the quaternions, the corresponding flight path angles can be re-calculated by:

$$\chi_K^G = \arctan\left(\frac{2(q_1q_2 + q_0q_3)}{q_0^2 + q_1^2 - q_2^2 - q_3^2}\right) \tag{3.38}$$

$$\gamma_K^G = \arcsin\left(-2(q_1q_3 - q_0q_2)\right) \tag{3.39}$$

$$\mu_K^G = \arctan\left(\frac{2(q_2q_3 + q_0q_1)}{q_0^2 - q_1^2 - q_2^2 + q_3^2}\right) \tag{3.40}$$

Regardless of the use of the quaternions, the flight-path climb angle γ_K has to be limited to values between -90° and +90° to avoid singularities while calculating the angular rate $(\vec{\boldsymbol{\omega}}^{OK})_K$, but it has only to be limited for this calculation and not globally as it would have been the case with the utilization of the flight-path climb angle γ_K as a state itself. Another advantage of the quaternions compared to the usage of the flight-path angles is the uniqueness of the quaternions, i.e. that the flight-path angles can be determined from the quaternions without ambiguity. Furthermore, the values of the flight-path bank angle μ_K and of the flight-path course angle χ_K switch automatically when the flight-path climb angle γ_K reaches ±90° e.g. when flying a looping or a Half Cuban Eight. Of course, if there are no aerobatic maneuvers that might cause climb angles of ±90°, the original translation equations of motion (Eq. (3.23) respectively Eqs. (3.24) to (3.26)) can be utilized. Given the case that the four quaternions are utilized, the state vector associated to the translation equations of motion comprises the kinematic velocity V_K and the four quaternions q_0, q_1, q_2 and q_3, else the kinematic velocity V_K is augmented by the three kinematic flight-path angles γ_K, χ_K and μ_K to give the full translation state vector. With regard to the trajectory optimization problem, the utilization of the kinematic flight-path angles improves the stability of the optimization problem since the number of states and thus the dimension of the optimization problem is reduced. Furthermore, the implementation of path constraints, initial or final boundary conditions is achieved more easily with the three kinematic flight-path angles.

3.2.4 Attitude Propagation Equations

In contrast to conventional flight system models (Refs. [Stevens, 1992], [Philips, 2004], [Etkin, 1996], [Etkin, 2005] and [Roskam, 2001]), in this work the attitude of the flight vehicle is not specified with respect to the *NED*-Frame by the Euler-angles azimuth Ψ, inclination Θ and bank angle Φ but with respect to the trajectory respectively the Intermediate Kinematic Flight-Path Frame $\bar{K}$ that is the Kinematic Flight-Path Frame K rotated by the kinematic bank-angle μ_K around its x-axis. Thus the attitude states are given by the angles between the Body-Fixed Frame B and the Intermediate Kinematic Flight-Path Frame $\bar{K}$. The associated kinematic attitude angles are α_K and β_K. The respective equations of motion for the attitude dynamics can be derived by the so called strap-down equation (Refs. [Stevens, 2003], [Holzapfel, 2009a]):

$$\left(\boldsymbol{\Omega}_K^{KB}\right)_{KK} = \dot{\mathbf{M}}_{KB} \cdot \mathbf{M}_{BK} \tag{3.41}$$

$$\left(\vec{\boldsymbol{\omega}}_K^{KB}\right)_K = \begin{pmatrix} \omega_{K,x}^{KB} \\ \omega_{K,y}^{KB} \\ \omega_{K,z}^{KB} \end{pmatrix}_K = \begin{pmatrix} \dot{\mu}_K^G + \dot{\alpha}_K^G \cdot \sin\beta_K^G \\ \dot{\alpha}_K^G \cdot \cos\beta_K^G \cdot \cos\mu_K^G + \dot{\beta}_K^G \cdot \sin\mu_K^G \\ \dot{\alpha}_K^G \cdot \cos\beta_K^G \cdot \sin\mu_K^G - \dot{\beta}_K^G \cdot \cos\mu_K^G \end{pmatrix}_K \tag{3.42}$$

$$\begin{pmatrix} \dot{\alpha}_K^G \\ \dot{\beta}_K^G \\ \dot{\mu}_K^G \end{pmatrix}_K = \begin{pmatrix} \dfrac{1}{\cos\beta_K^G} \cdot \left(\omega_{K,y}^{KB} \cdot \cos\mu_K^G + \omega_{K,z}^{KB} \cdot \sin\mu_K^G\right) \\ \omega_{K,y}^{KB} \cdot \sin\mu_K^G - \omega_{K,z}^{KB} \cdot \cos\mu_K^G \\ \omega_{K,x}^{KB} - \tan\beta_K^G \cdot \left(\omega_{K,y}^{KB} \cdot \cos\mu_K^G + \omega_{K,z}^{KB} \cdot \sin\mu_K^G\right) \end{pmatrix}_K \tag{3.43}$$

The angular rate $\vec{\boldsymbol{\omega}}^{KB}$ that is required for the above equations can be computed by Eqs. (3.44) and (3.45), where the angular rate $\vec{\boldsymbol{\omega}}^{OK}$ is an output of the translation equations of motion and

the angular rate $\vec{\boldsymbol{\omega}}^{IB}$ is given by the rotation equations of motion described later on (chapter 3.2.5):

$$\left(\vec{\boldsymbol{\omega}}_K^{KB}\right)_K = \mathbf{M}_{KB} \cdot \left(\vec{\boldsymbol{\omega}}_K^{OB}\right)_B - \left(\vec{\boldsymbol{\omega}}^{OK}\right)_K \tag{3.44}$$

$$\left(\vec{\boldsymbol{\omega}}_K^{OB}\right)_B = \left(\vec{\boldsymbol{\omega}}_K^{IB}\right)_B - \mathbf{M}_{BO} \cdot \left(\vec{\boldsymbol{\omega}}^{IE}\right)_O - \mathbf{M}_{BO} \cdot \left(\vec{\boldsymbol{\omega}}^{EO}\right)_O \tag{3.45}$$

If quaternions are used for the translation equations of motion, the first order time derivative of the flight-path bank angle $\dot{\mu}_K$ is not considered as an attitude state and thus not directly integrated but used to compute the angular rate $\vec{\boldsymbol{\omega}}^{O\bar{K}}$ in Eq. (3.30) respectively Eq. (3.36). Therefore, the first order time derivative of the flight-path bank angle $\dot{\mu}_K$ is an output of the attitude propagation system and the states associated with the flight vehicle's attitude are just the kinematic angle of attack α_K and the kinematic angle of sideslip β_K.

3.2.5 Rotation Equation of Motions

Analogous to the linear momentum, the derivation of the differential equations for the rotational degrees of freedom is also based on Newton's 2nd law. Just like the linear momentum, the angular momentum is defined for an arbitrary differential mass element *P* of the vehicle and then integrated over the complete vehicle mass. While the principle of conservation of the linear momentum poses a relationship between the translational motion of a vehicle and the external forces acting on this vehicle, the principle of conservation of the angular momentum draws the relationship between the rotational motion of an aircraft and the sum of external moments: it indicates that the time rate of change of the angular momentum $\vec{\mathbf{H}}$ of a dynamic system is proportional to the sum of external moments $\Sigma\vec{\mathbf{M}}$ acting on the dynamic system:

$$\sum\left(\vec{\mathbf{M}}^O\right) = \left(\frac{d}{dt}\right)^I \vec{\mathbf{H}}^O = \left(\frac{d}{dt}\right)^I \int_m \vec{\mathbf{r}}^P\left(\vec{\mathbf{x}}^P\right) \times \vec{\mathbf{V}}^I\left(\vec{\mathbf{x}}^P\right) dm \tag{3.46}$$

where $\vec{\mathbf{r}}^P$ represents the position vector and $\vec{\mathbf{V}}^I$ the velocity of an arbitrary mass element *P*. In Eq. (3.46) the reference point for the angular momentum and thus for the external moments is the center of the Earth since the *ECI*-Frame has been identified as a legitimate Euclidean Frame. With the simplifying assumptions made above and the choice of the center of gravity *G* of the aircraft as reference point for the determination of the external moments, the sum of external moments $\Sigma\vec{\mathbf{M}}$ acting at the center of gravity *G* and denoted in the Body-Fixed Reference Frame *B* evaluates to (Ref. [Holzapfel, 2009b]):

$$\sum\left(\vec{\mathbf{M}}^G\right)_B = \left(\mathbf{I}^G\right)_{BB} \cdot \left(\dot{\vec{\boldsymbol{\omega}}}_K^{IB}\right)_B^B + \left(\vec{\boldsymbol{\omega}}_K^{IB}\right)_B \times \left[\left(\mathbf{I}^G\right)_{BB} \cdot \left(\vec{\boldsymbol{\omega}}_K^{IB}\right)_B\right] \tag{3.47}$$

Here, $\mathbf{I}^G$ denotes the inertia tensor of the aircraft with respect to the center of gravity point *G*. Solving Eq. (3.47) for the first order time derivative of the angular rate $\vec{\boldsymbol{\omega}}^{IB}$ between the Body-Fixed Frame *B* and the *ECI*-Frame *I*, the following differential equations for the rotational motion of the aircraft denoted in the Body-Fixed Reference Frame *B* result:

$$\left(\dot{\vec{\boldsymbol{\omega}}}_K^{IB}\right)_B^B = \left(\mathbf{I}^G\right)_{BB}^{-1} \cdot \left\{\sum\left(\vec{\mathbf{M}}^G\right)_B - \left(\vec{\boldsymbol{\omega}}_K^{IB}\right)_B \times \left[\left(\mathbf{I}^G\right)_{BB} \cdot \left(\vec{\boldsymbol{\omega}}_K^{IB}\right)_B\right]\right\} \tag{3.48}$$

Thus, the states of the rotational motion of the aircraft are the components of the rotational rate $\vec{\boldsymbol{\omega}}^{IB}$ between the Body-Fixed Reference Frame *B* and the *ECI*-Frame *I*, namely the roll

rate p_K, the pitch rate q_K and the yaw rate r_K, and Eq. (3.48) provides the corresponding differential equations for the determination of the state derivatives.

3.2.6 Linear State-Space Models

Alternatively to the full, non-linear attitude propagation equations and rotational equation of motions stated above, the attitude and rotational dynamics can be simulated utilizing a state-consistent but simplified approach with linear state-space models incorporated in the optimization simulation model. The input quantities to these state-space models are the control surface deflections, the outputs are the rotational and attitude motions of the aircraft, i.e. the linear state-space models feature the same inputs and outputs as the nonlinear model with the nonlinear attitude and rotation equations of motion. These state-space models are more accurate than the transfer functions for the inner loop presented in chapter 3.2.7, but they are still only linear state-space models and do not represent the full non-linear behavior of the rotational and attitude dynamics of an aircraft. But they are easier to implement than the full non-linear rotational and attitude equations of motion that have been depicted in the preceding chapters. The linear state-space models represent a linear approximation of the rotation and attitude dynamics around a steady-state condition. In contrast to the single-input single-output transfer functions, they allow to take into account coupling effects.

Basically, linear state-space models for an arbitrary flight system can be obtained by linearizing the non-linear equations of motion of the flight system with respect to a specific, quasi-steady-state reference or trim condition (Refs. [Roskam, 2001] and [Holzapfel, 2009c]). Linear state-space models are usually derived for the analytical investigation of flying qualities of an aircraft like e.g. stability or controllability or for the design of flight control laws. It is important to mention that the resulting linear substitute equations of motion are only valid in the vicinity of the regarded steady-state flight condition. Therefore the investigation of flying qualities and the development of control laws is also limited to the close vicinity of the considered trim condition. Within the state-space models, the states that effect a motion of the aircraft solely in the vertical plane are assigned to the longitudinal dynamics, while states that lead to a motion in the horizontal plane are assigned to the lateral dynamics. Setting the states of the lateral motion to zero in the differential equations of the longitudinal dynamics and vice versa, the longitudinal motion can be investigated decoupled from the lateral dynamics. The resulting state-space models for the longitudinal as well as the lateral motion are given in appendix A (Refs. [Roskam, 2001] and [Holzapfel, 2009c]).

As can be seen from the linearized state-space models for the longitudinal motion (appendix A, Eq. (A.22)), the state variable for the position x is not coupled to the differential equations of the remaining longitudinal states since the according elements of the system matrix are zero. This means that the derivatives of the remaining state variables are independent of the actual position x: they are dynamically decoupled from the position x. Since for conventional aircraft the derivatives $C_{L\dot{\alpha}}$, $C_{D\dot{\alpha}}$ as well as $C_{m\dot{\alpha}}$ are quite small in reality, the substitute derivatives $Z_{\dot{\alpha}}$, $X_{\dot{\alpha}}$ and $M_{\dot{\alpha}}$ can be set to zero. Furthermore, the substitute derivatives with respect to a change in the altitude h, i.e. X_h, Z_h and M_h can also be neglected to a first approximation, so that the differential equation of the altitude h may also be regarded as dynamically decoupled from the remaining linearized equations of motion. Taking into account the dynamic decoupling for the position x and the altitude h, the simplified linear state-space model for the longitudinal motion given by Eq. (A.22) in appendix A results. Investigating the eigen values of the longitudinal motion, two basic eigen motion forms can

be observed that are the short-period motion and the phygoid motion. Here, the natural frequency of the short-period motion is by far larger than the natural frequency of the phygoid motion. Furthermore, the primary states of the phygoid motion that are the velocity V_K and the flight-path inclination angle γ_K are only slightly involved in the short-period dynamics, so that the simplified state-space model for the longitudinal motion can be split up into a state-space model representing the short-period motion (Eq. (3.49)) and a state-space model for the phygoid motion, where the primary states regarding the short-period motion are the pitch rate q_K and the angle of attack α_K (Refs. [Roskam, 2001], [Holzapfel, 2009c] and [Stevens, 1992]). This means that only the rotational and attitude dynamics are accounted for with negligible backward influence from the translation dynamics that are implemented in the nonlinear outer loop.

Consequently, for the longitudinal dynamics of the aircraft's inner loop the appropriate linear state-space model that represents the characteristic short-period motion of an aircraft is chosen. As mentioned above, the two states that are mainly involved in the short-period motion are the kinematic angle of attack α_K and the pitch rate q_K, thus a second-order state-space model is implemented in the simulation model that describes the short-period coupling between the kinematic angle of attack α_K and pseudo-pitch rate q_K^*, while the according control input is the elevator deflection η:

$$\begin{aligned}\begin{pmatrix}\dot{\alpha}_K^G\\ \dot{q}_K^*\end{pmatrix} &= \begin{bmatrix}Z_\alpha & Z_q+1\\ M_\alpha & M_q\end{bmatrix}\cdot\begin{pmatrix}\alpha_A^G\\ q_K^*\end{pmatrix}+\begin{pmatrix}Z_\eta\\ M_\eta\end{pmatrix}\cdot\eta_{CMD}\\ &= \begin{bmatrix}0 & Z_q+1\\ 0 & M_q\end{bmatrix}\cdot\begin{pmatrix}\alpha_K^G\\ q_K^*\end{pmatrix}+\begin{bmatrix}Z_\alpha & Z_\eta\\ M_\alpha & M_\eta\end{bmatrix}\cdot\begin{pmatrix}\alpha_A^G\\ \eta_{CMD}\end{pmatrix}\\ &= \mathbf{A}_{longitudinal}\cdot\begin{pmatrix}\alpha_K^G\\ q_K^*\end{pmatrix}+\mathbf{B}_{longitudinal}\cdot\begin{pmatrix}\alpha_A^G\\ \eta_{CMD}\end{pmatrix}\end{aligned} \tag{3.49}$$

Here, in analogy to the nonlinear rotation and attitude dynamics the first order time derivatives of the kinematic angle of attack α_K and the pitch rate q_K^* are a function of the *aerodynamic* angle of attack α_A and not of the kinematic angle of attack α_K. Furthermore the pitch rate q_K is termed pseudo-pitch rate since the pitch rate resulting from the linear state space model (3.49) is consistent with the real pitch rate only at the linearization point of the state space model, i.e. with the angle of attack equal to zero. Otherwise, for a steady-state flight with a constant angle of attack deviating from the linearization point and not being equal to zero, a pitch rate q_K would result from the steady-state flight requirement $\dot{q}_K$=0, contradicting the steady-state flight assumption. Thus, the pitch rate can only be regarded as the second state in a second order linear state-space model. The real pitch rate has to be restored by Eq. (3.58) given below. Regarding the lateral dynamics of an aircraft, the dependencies on the first order time derivative of the sideslip angle $\dot{\beta}_K$ can be neglected for conventional configurations and the corresponding substitute derivatives $N_{\dot{\beta}}$, $Y_{\dot{\beta}}$ and $L_{\dot{\beta}}$ can approximately be set to zero in the system matrix of the lateral state-space model. Thus, the four state variables of the lateral motion roll rate p_K, yaw rate r_K, sideslip angle β_K and the bank angle Φ are dynamically decoupled from the remaining lateral states that are the position variable y and the azimuth angle Ψ. Furthermore, for small steady-state pitch angles Θ_0 (and thus $tan(\Theta_0)$ being approximately zero) and with the influence of the bank angle Φ in the differential equation of the sideslip angle β_K being negligible, the differential equation for the bank angle Φ can also be regarded as dynamically decoupled from the remaining linearized

equations of motion, so that the simplified state-space model for the lateral motion given by Eq. (3.50) results.

This linearized state-space model for the lateral dynamics is characterized by two eigen motion forms that are the roll motion and the dutch-roll mode. The states that are primarily incorporated in the dutch-roll motion are the sideslip angle β_K and the yaw rate r_K, while the roll mode incorporates mainly the roll rate p_K (Refs. [Roskam, 2001], [Holzapfel, 2009c] and [Stevens, 1992]).

Thus, for the modeling of the inner loop lateral dynamics a third order approach with the states pseudo-roll rate p_K^*, pseudo-yaw rate r_K^* and kinematic angle of sideslip β_K is implemented in order to represent the appropriate dynamics, where the inputs are the aileron surface deflection ξ and the rudder surface deflection ζ:

$$\begin{aligned}
\begin{pmatrix} \dot{p}_K^* \\ \dot{r}_K^* \\ \dot{\beta}_K^G \end{pmatrix} &= \begin{bmatrix} L_p & L_r & L_\beta \\ N_p & N_r & N_\beta \\ Y_p & Y_r - 1 & Y_\beta \end{bmatrix} \cdot \begin{pmatrix} p_K^* \\ r_K^* \\ \beta_A^G \end{pmatrix} + \begin{bmatrix} L_\xi & L_\zeta \\ N_\xi & N_\zeta \\ Y_\xi & Y_\zeta \end{bmatrix} \cdot \begin{pmatrix} \xi_{CMD} \\ \zeta_{CMD} \end{pmatrix} \\
&= \begin{bmatrix} L_p & L_r & 0 \\ N_p & N_r & 0 \\ Y_p & Y_r - 1 & 0 \end{bmatrix} \cdot \begin{pmatrix} p_K^* \\ r_K^* \\ \beta_K^G \end{pmatrix} + \begin{bmatrix} L_\xi & L_\zeta & L_\beta \\ N_\xi & N_\zeta & N_\beta \\ Y_\xi & Y_\zeta & Y_\beta \end{bmatrix} \cdot \begin{pmatrix} \xi_{CMD} \\ \zeta_{CMD} \\ \beta_A^G \end{pmatrix} \\
&= \mathbf{A}_{lateral} \cdot \begin{pmatrix} p_K^* \\ r_K^* \\ \beta_K^G \end{pmatrix} + \mathbf{B}_{lateral} \cdot \begin{pmatrix} \xi_{CMD} \\ \zeta_{CMD} \\ \beta_A^G \end{pmatrix}
\end{aligned} \tag{3.50}$$

In total, the linear state-space models for the longitudinal and the lateral motion incorporate five states in order to guarantee state-consistency in comparison to the full, non-linear rotation and attitude equations of motion.

In Eqs. (3.49) and (3.50), the system matrices **A** and the control matrices **B** are made up of the dimensional force derivatives *Z* and *Y* respectively the aerodynamic moment coefficients *L*, *M* and *N* with respect to the various states and controls, where the derivatives are functions of the actual aircraft configuration (i.e. mass *m*, location of the center of gravity and inertia), the aerodynamic velocity V_A and the air density ρ (Ref. [Holzapfel, 2009c]). Exemplarily, the equation for the aerodynamic force derivative Z_α due to a change in the angle of attack α is given below, while the formulae for the remaining force and moment derivatives can be found in appendix A:

$$Z_\alpha = -\frac{\bar{q} \cdot S}{m \cdot V_A} \cdot \left[C_{L\alpha} + C_{D|_0} \right] \tag{3.51}$$

Here, *S* is the aircraft's wing reference area, $C_{L\alpha}$ the variation of the aircraft's lift coefficient with the angle of attack α_A and $C_{D|_0}$ the drag coefficient of the aircraft in the steady-state reference flight condition utilized for linearization:

$$C_{D|_0} = C_{D0} + k \cdot \left(C_{L0} + C_{L\alpha} \cdot \alpha_A - C_{L,C_{D0}} \right)^2 \tag{3.52}$$

$\bar{q}$ is the dynamic pressure calculated from the air density ρ and the aerodynamic velocity V_A:

$$\bar{q} = \frac{1}{2} \cdot \rho \cdot \left(V_A^G \right)^2 \tag{3.53}$$

As can be seen from Eqs. (3.49) and (3.50), the *kinematic* angle of attack α_K and the *kinematic* angle of sideslip β_K are chosen as states in order to give a full description of the aircraft following the path from the *NED*-Frame to the Body-Fixed Frame *B* via the Kinematic Flight-Path Frame *K* and Intermediate Kinematic Flight-Path Frame $\overline{K}$ due to the sequential implementation of the simulation model. In addition, the rotation equations of motion given in chapter 3.2.5 incorporate *kinematic* body-angular rates, so that *kinematic* attitude angles result from the attitude equations of motion. While *kinematic* attitude angles are chosen as states, the *aerodynamic* attitude angles α_A and β_A are regarded as inputs to the state-space models. This is done to take into account the analogy to the full, non-linear 6-Degree of Freedom simulation model where the derivatives α_K and β_K are evidently not a function of the kinematic angles α_K and β_K but a function of the aerodynamic angles α_A and β_A that have to be utilized to compute the aerodynamic moments for the rotation equations of motion (3.48).

Furthermore, the angular rates p_K^*, q_K^* and r_K^* are marked with a star to accentuate that these angular rates are in fact not the real angular body rates. As one can see from Eq. (3.49), a change in the pitch rate $\dot{q}_K$ and consequently a pitch rate q_K would result for any arbitrary angle of attack α_A that does not equal zero. But e.g. for a steady-state straight flight in trim condition, i.e. a trimmed flight condition with the pitch rate q_K being constantly zero, an angle of attack α_A is required that does not equal zero. Thus, the linearized longitudinal state-space model (3.49) cannot be used to compute the pitch rate change $\dot{q}_K$ since this would not lead to a trimmed straight horizontal flight. Instead, the real angular body rates p_K, q_K and r_K between the Body-Fixed Frame *B* and the *NED*-Reference Frame respectively the *ECI*-Frame have to be restored with the help of the following equations:

$$\begin{pmatrix} p_K \\ q_K \\ r_K \end{pmatrix}_B = \left(\vec{\boldsymbol{\omega}}_K^{\,OB}\right)_B = \left(\vec{\boldsymbol{\omega}}_K^{\,O\overline{K}}\right)_B + \left(\vec{\boldsymbol{\omega}}_K^{\,\overline{K}B}\right)_B = \mathbf{M}_{B\overline{K}} \cdot \left(\vec{\boldsymbol{\omega}}_K^{\,O\overline{K}}\right)_{\overline{K}} + \left(\vec{\boldsymbol{\omega}}_K^{\,\overline{K}B}\right)_B \tag{3.54}$$

$$\left(\vec{\boldsymbol{\omega}}_K^{\,IB}\right)_B = \left(\vec{\boldsymbol{\omega}}_K^{\,IE}\right)_B + \left(\vec{\boldsymbol{\omega}}_K^{\,EO}\right)_B + \left(\vec{\boldsymbol{\omega}}_K^{\,OB}\right)_B \tag{3.55}$$

with

$$\left(\vec{\boldsymbol{\omega}}_K^{\,\overline{K}B}\right)_B = \begin{pmatrix} \dot{\beta}_K^G \cdot \sin\alpha_K^G \\ \dot{\alpha}_K^G \\ -\dot{\beta}_K^G \cdot \cos\alpha_K^G \end{pmatrix}_B \tag{3.56}$$

$$\left(\vec{\boldsymbol{\omega}}_K^{\,O\overline{K}}\right)_{\overline{K}} = \begin{pmatrix} \dot{\mu}_K^G - \dot{\chi}_K^G \sin\gamma_K^G \\ \dot{\chi}_K^G \cos\gamma_K^G \sin\mu_K^G + \dot{\gamma}_K^G \cos\mu_K^G \\ \dot{\chi}_K^G \cos\gamma_K^G \cos\mu_K^G - \dot{\gamma}_K^G \sin\mu_K^G \end{pmatrix}_{\overline{K}} \tag{3.57}$$

so that

$$\begin{pmatrix} p_K \\ q_K \\ r_K \end{pmatrix}_B = \mathbf{M}_{B\overline{K}} \cdot \begin{pmatrix} \dot{\mu}_K^G - \dot{\chi}_K^G \sin\gamma_K^G \\ \dot{\chi}_K^G \cos\gamma_K^G \sin\mu_K^G + \dot{\gamma}_K^G \cos\mu_K^G \\ \dot{\chi}_K^G \cos\gamma_K^G \cos\mu_K^G - \dot{\gamma}_K^G \sin\mu_K^G \end{pmatrix}_{\overline{K}} + \begin{pmatrix} \dot{\beta}_K^G \cdot \sin\alpha_K^G \\ \dot{\alpha}_K^G \\ -\dot{\beta}_K^G \cdot \cos\alpha_K^G \end{pmatrix}_B \tag{3.58}$$

For the implementation of the linear state-space models, the absolute values and the rates of the state variables have to be limited in order to adjust the dynamics of the state-space models to the dynamics of the non-linear inner loop. Especially the angle of attack α_A has to be

restricted to prevent the aircraft from stalling, while the control surface deflections ξ, η and ζ have to be limited due to the given geometry of the aircraft, i.e. the maximum possible physical control surface deflections:

$$\alpha_{A,\min} \le \alpha_A(t) \le \alpha_{A,\max} \tag{3.59}$$

$$\xi_{\min} \le \xi(t) \le \xi_{\max} \tag{3.60}$$

$$\eta_{\min} \le \eta(t) \le \eta_{\max} \tag{3.61}$$

$$\zeta_{\min} \le \zeta(t) \le \zeta_{\max} \tag{3.62}$$

The computation of the aerodynamic angle of attack α_A is given by Eq. (3.106). Here it is mentioned that implementing any limiters into the state-space models means that these state-space models also incorporate non-linearities although they are linear in principle. Fig. 13 shows the principal layout for the implementation of a limiter, where the saturation flag indicates whether the respective signal is saturated or not.

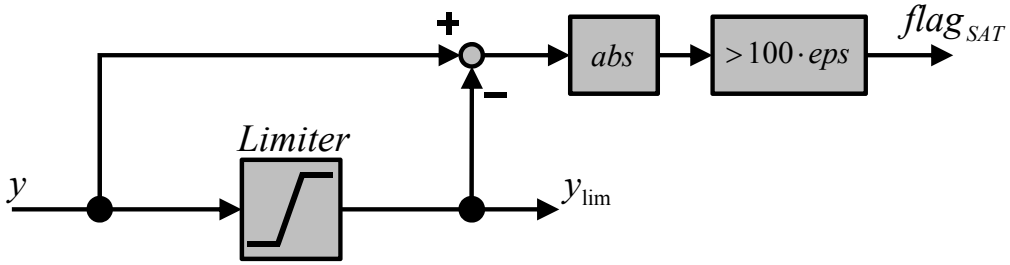

Figure 13. Limiter with Saturation Flag

3.2.7 Linear Transfer Functions

The most simplified approach for modeling the dynamics represented by the inner loop, i.e. the attitude and rotational dynamics, that still preserves the true dynamic order of this inner loop but is not state-consistent any more is the utilization of linear transfer functions for the load factors and the roll rate. This inner loop can be utilized to represent the rotational and attitude dynamics of an aircraft in a rudimentary manner retaining the correct order for the case that only little aircraft data are available. For this purpose, the dynamics of the normal load factor n_z and the dynamics of the lateral load factor n_y given in the Aerodynamic Reference Frame A are approximated by a second order time behavior, whereas the dynamics of the pseudo-roll rate p_K^* feature a first order time behavior. Here again the roll rate is marked with a star to indicate that this roll rate does not correspond to the true roll rate of the aircraft. In order to guarantee a good approximation of the correct aircraft dynamics, the dynamics for the lift build-up use the same dynamics as the short-period oscillation and the build-up of the side force corresponds to the dutch roll dynamics. Furthermore, for the roll rate the simple, decoupled roll dynamics are used:

$$\left(n_{A,y}^G\right)_A = \frac{\omega_{0,DR}^2}{s^2 + 2\cdot\zeta_{DR}\cdot\omega_{0,DR}\cdot s + \omega_{0,DR}^2}\left(n_{A,y}^G\right)_{A,CMD} \tag{3.63}$$

$$\left(n_{A,z}^G\right)_A = \frac{\omega_{0,SP}^2}{s^2 + 2\cdot\zeta_{SP}\cdot\omega_{0,SP}\cdot s + \omega_{0,SP}^2}\left(n_{A,z}^G\right)_{A,CMD} \tag{3.64}$$

$$p_K^* = \frac{1}{T_{Roll} \cdot s + 1} \cdot p_{K,CMD}^* \tag{3.65}$$

Alternatively, the linear transfer functions for the load factors n_y and n_z can also account for a non-minimum phase behavior of the flight system by the utilization of the following, slightly modified equations for the transfer functions:

$$\left(n_{A,y}^G\right)_A = \frac{-\left(T_y \cdot s - 1\right) \cdot \omega_{0,DR}^2}{s^2 + 2 \cdot \zeta_{DR} \cdot \omega_{0,DR} \cdot s + \omega_{0,DR}^2} \left(n_{A,y}^G\right)_{A,CMD} \tag{3.66}$$

$$\left(n_{A,z}^G\right)_A = \frac{-\left(T_z \cdot s - 1\right) \cdot \omega_{0,SP}^2}{s^2 + 2 \cdot \zeta_{SP} \cdot \omega_{0,SP} \cdot s + \omega_{0,SP}^2} \left(n_{A,z}^G\right)_{A,CMD} \tag{3.67}$$

Considering the transfer function (3.67) for the load factor in the direction of the aircraft's z-axis, this transfer function accounts for the fact that an aircraft initially moves downward when the pilot pulls the stick before it begins to ascend as desired by the pilot's control input. This effect is depicted in Fig. 14.

The various parameters included in the load factor transfer functions that are the natural frequencies ω_0, the relative damping ζ and the time constants T depend on the current aerodynamic velocity V_A of the aircraft, the actual aircraft mass m and the air density ρ and are therefore scheduled with regard to the current flight condition. The pseudo-roll rate p_K^* also has to be limited depending on the current aircraft mass m and dynamic pressure $\bar{q}$. The respective values can be derived from the analytical dependency of the modes on the various parameters. The transformation of the above stated transfer functions for the roll rate respectively the load factors into state-space models of first order gives:

$$\frac{d}{dt}\begin{pmatrix} n_{A,y}^G \\ \dot{n}_{A,y}^G \end{pmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_{0,DR}^2 & -2 \cdot \zeta_{DR} \cdot \omega_{0,DR} \end{bmatrix} \begin{pmatrix} n_{A,y}^G \\ \dot{n}_{A,y}^G \end{pmatrix} + \begin{bmatrix} 0 \\ \omega_{0,DR}^2 \end{bmatrix} n_{A,y,CMD}^G \tag{3.68}$$

$$\frac{d}{dt}\begin{pmatrix} n_{A,z}^G \\ \dot{n}_{A,z}^G \end{pmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_{0,SP}^2 & -2 \cdot \zeta_{SP} \cdot \omega_{0,SP} \end{bmatrix} \begin{pmatrix} n_{A,z}^G \\ \dot{n}_{A,z}^G \end{pmatrix} + \begin{bmatrix} 0 \\ \omega_{0,SP}^2 \end{bmatrix} n_{A,z,CMD}^G \tag{3.69}$$

$$\dot{p}_K^* = \frac{1}{T_{Roll}} \cdot \left(p_{K,CMD}^* - p_K^*\right) \tag{3.70}$$

Thus, the state vector for the linear transfer functions comprises the load factors $n_{A,y}$ and $n_{A,z}$, the first order time derivatives of the load factors $\dot{n}_{A,y}$ and $\dot{n}_{A,z}$ as well as the roll rate p_K^*. For the load-factor transfer functions taking into account non-minimum phase effects (Eqs. (3.66) and (3.67)), the following state-space models result:

$$\frac{d}{dt}\begin{pmatrix} n_{A,y}^G \\ \dot{n}_{A,y}^G \end{pmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_{0,DR}^2 & -2 \cdot \zeta_{DR} \cdot \omega_{0,DR} \end{bmatrix} \begin{pmatrix} n_{A,y}^G \\ \dot{n}_{A,y}^G \end{pmatrix} + \begin{bmatrix} 0 & 0 \\ \omega_{0,DR}^2 & -T_y \omega_{0,DR}^2 \end{bmatrix} \begin{pmatrix} n_{A,y,CMD}^G \\ \dot{n}_{A,y,CMD}^G \end{pmatrix} \tag{3.71}$$

$$\frac{d}{dt}\begin{pmatrix} n_{A,z}^G \\ \dot{n}_{A,z}^G \end{pmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_{0,SP}^2 & -2 \cdot \zeta_{SP} \cdot \omega_{0,SP} \end{bmatrix} \begin{pmatrix} n_{A,z}^G \\ \dot{n}_{A,z}^G \end{pmatrix} + \begin{bmatrix} 0 & 0 \\ \omega_{0,SP}^2 & -T_z \omega_{0,SP}^2 \end{bmatrix} \begin{pmatrix} n_{A,z,CMD}^G \\ \dot{n}_{A,z,CMD}^G \end{pmatrix} \tag{3.72}$$

Limitations with respect to the load factors n_y and n_z can arise due the maximum achievable lift coefficient $C_{L,max}$, the maximum achievable side force coefficient $C_{Y,max}$, the maximum allowable load capacity $n_{max,structure}$ of the aircraft structure or the maximum load factors $n_{max,pilot}$ the pilot can sustain:

$$n^G_{A,y,\min} \le n^G_{A,y} \le n^G_{A,y,\max} \tag{3.73}$$

$$n^G_{A,z,\min} \le n^G_{A,z} \le n^G_{A,z,\max} \tag{3.74}$$

where

$$n^G_{A,y,\min} = \max\left(\frac{\overline{q} \cdot S}{mg} C_{Y,\min}, n_{y,\min,\text{structure}}, n_{y,\min,\text{pilot}} \right) \tag{3.75}$$

$$n^G_{A,y,\max} = \min\left(\frac{\overline{q} \cdot S}{mg} C_{Y,\max}, n_{y,\max,\text{structure}}, n_{y,\max,\text{pilot}} \right) \tag{3.76}$$

and

$$n^G_{A,z,\min} = \max\left(\frac{\overline{q} \cdot S}{mg} C_{L,\min}, n_{z,\min,\text{structure}}, n_{z,\min,\text{pilot}} \right) \tag{3.77}$$

$$n^G_{A,z,\max} = \min\left(\frac{\overline{q} \cdot S}{mg} C_{L,\max}, n_{z,\max,\text{structure}}, n_{z,\max,\text{pilot}} \right) \tag{3.78}$$

Additionally to the minimum and maximum values for the load factors n_y and n_z, limits for their first and second order time derivatives as well as limits regarding the roll rate p_K and its first order time derivative can be taken into account for the modeling respectively the implementation of the inner loop utilizing linear transfer functions in order to adjust the dynamics of the linear transfer functions to the dynamics of the non-linear inner loop:

$$\dot{n}^G_{A,y,\min} \le \dot{n}^G_{A,y} \le \dot{n}^G_{A,y,\max} \tag{3.79}$$

$$\ddot{n}^G_{A,y,\min} \le \ddot{n}^G_{A,y} \le \ddot{n}^G_{A,y,\max} \tag{3.80}$$

$$\dot{n}^G_{A,z,\min} \le \dot{n}^G_{A,z} \le \dot{n}^G_{A,z,\max} \tag{3.81}$$

$$\ddot{n}^G_{A,z,\min} \le \ddot{n}^G_{A,z} \le \ddot{n}^G_{A,z,\max} \tag{3.82}$$

$$-p_{K,\max} \le p_K \le p_{K,\max} \tag{3.83}$$

$$-\dot{p}_{K,\max} \le \dot{p}_K \le \dot{p}_{K,\max} \tag{3.84}$$

As for the linear state-space models above, it is mentioned that incorporating these limits in the simulation model introduces non-linearities into the transfer functions that are in principle linear.

For the transfer functions, the natural frequency $\omega_{0,SP}$ and the damping ratio $\zeta_{0,SP}$ of the short-period motion, the natural frequency $\omega_{0,DR}$ and the damping ratio $\zeta_{0,DR}$ of the dutch-roll mode and the time constant T_{Roll} of the roll mode are required since these quantities determine the oscillation of the load factor build-up respectively the roll-rate build-up. These values are also only valid for a specific reference point and thus depend on the actual flight condition where the variables are the air density ρ, the aerodynamic velocity V_A and the aircraft mass m.

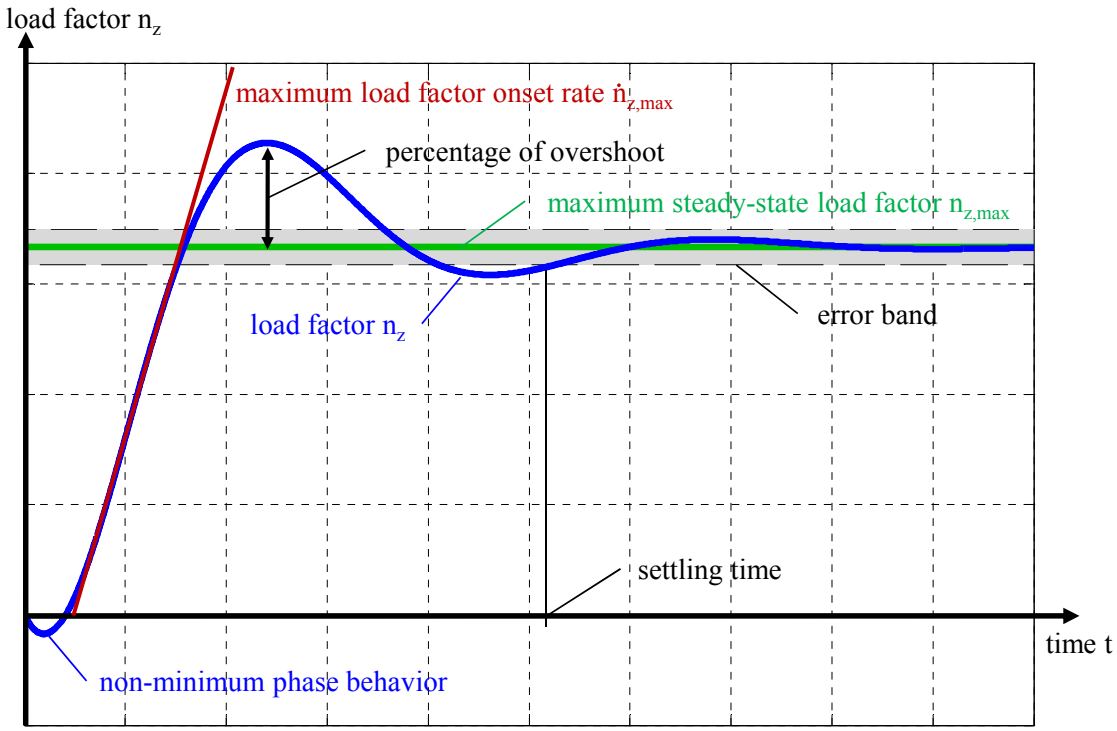

Figure 14. Load Factor n_z Time History showing Non-Minimum Phase Behavior

With the values for the load factors n_y and n_z given, the appurtenant force coefficients C_L and C_Y can be restored:

$$C_L = \frac{\left(n^G_{A,z}\right)_A \cdot mg}{\bar{q} \cdot S} \tag{3.85}$$

$$C_Y = \frac{\left(n^G_{A,y}\right)_A \cdot mg}{\bar{q} \cdot S} \tag{3.86}$$

Given the assumption that the aerodynamic lift coefficient C_L depends only on the angle of attack α_A and that the aerodynamic side force coefficient C_Y is solely a function of the sideslip angle β_A of the aircraft, the approximate angle of attack α_A as well as the angle of sideslip β_A can be obtained.

3.2.8 Aircraft Attitude described by Euler Angles

By the kinematic angle of attack α_K and the kinematic sideslip angle β_K, the attitude of the aircraft is described with respect to the Intermediate Kinematic Flight-Path Frame $\bar{K}$. Usually, the attitude of the aircraft is given with respect to the *NED*-Reference Frame, utilizing the set of Euler angles azimuth Ψ, inclination Θ and bank angle Φ. This representation is derived in the following for the simulation model described above.

The Euler angles can be restored from the transformation matrix $\mathbf{M}_{BO}$ between the Body-Fixed Frame B and the *NED*-Reference Frame:

$$\mathbf{M}_{BO} = \begin{bmatrix} c\Psi_K c\Theta_K & s\Psi_K c\Theta_K & -s\Theta_K \\ c\Psi_K s\Theta_K s\Phi_K - s\Psi_K c\Phi_K & s\Psi_K s\Theta_K s\Phi_K + c\Psi_K c\Phi_K & c\Theta_K s\Phi_K \\ c\Psi_K s\Theta_K c\Phi + s\Psi_K s\Phi_K & s\Psi_K s\Theta_K c\Phi_K - c\Psi_K s\Phi_K & c\Theta_K c\Phi_K \end{bmatrix} \tag{3.87}$$

Here, *s* is the abbreviation for *sin*() and *c* means the abbreviation for *cos*(). The value for the transformation matrix $\mathbf{M}_{BO}$ is determined by the following relationship:

$$\mathbf{M}_{BO} = \mathbf{M}_{B\bar{K}} \cdot \mathbf{M}_{\bar{K}O} \tag{3.88}$$

Equating element (1,1) with element (1,2) of the transformation matrix $\mathbf{M}_{BO}$ given by Eq. (3.87), the kinematic azimuth angle Ψ_K evaluates to:

$$\Psi_K = \arctan\left(\frac{\mathbf{M}_{BO}(1,2)}{\mathbf{M}_{BO}(1,1)}\right) \tag{3.89}$$

The kinematic inclination angle Θ_K is obtained from element (1,3) of the transformation matrix $\mathbf{M}_{BO}$:

$$\Theta_K = -\arcsin\left(\mathbf{M}_{BO}(1,3)\right) \tag{3.90}$$

Finally, elements (2,3) and (3,3) of the matrix $\mathbf{M}_{BO}$ between the Body-Fixed Frame *B* and the *NED*-Reference Frame give the kinematic flight-path bank angle Φ_K:

$$\Phi_K = \arctan\left(\frac{\mathbf{M}_{BO}(2,3)}{\mathbf{M}_{BO}(3,3)}\right) \tag{3.91}$$

3.2.9 Wind Inclusion and Kinematic Relationships

The aerodynamic forces and moments are a function of the relative motion of the aircraft with respect to the surrounding air. Given the seldom case that there is no atmospheric motion and thus no wind at all, the kinematic velocity V_K and the kinematic angles resulting from the integration of the differential equation of motions equal the aerodynamic velocity V_A respectively the aerodynamic angles α_A, β_A, etc. Otherwise, the aerodynamic values differ from the kinematic values and the kinematic relationships have to be accounted for in the simulation model. Thus, wind influences like static and convective wind fields play an important role when simulating aircraft trajectories and therefore a proper inclusion of these environmental issues in the simulation model is mandatory to provide a reproduction of the real environmental conditions that is as realistic as possible.

For the optimization simulation model, static, time dependent and convective wind terms given in the *NED*-Frame are regarded as sufficient modeling accuracy. The aerodynamic velocity of the aircraft's center of gravity with its components denoted in the *NED*-Frame can then be calculated as (Ref. [Brockhaus, 2001]):

$$\left(\vec{\mathbf{V}}_A^{\,G}\right)_O^E = \left(\vec{\mathbf{V}}_K^{\,G}\right)_O^E - \left(\vec{\mathbf{V}}_W^{\,G}\right)_O^E \tag{3.92}$$

Here, $\vec{\mathbf{V}}_W$ represents the wind field given in the *NED*-Reference Frame as a function of time *t* and position $\vec{\mathbf{r}}$. Then, the first order time derivative of the aerodynamic velocity vector is as follows:

$$\left(\dot{\vec{\mathbf{V}}}_A^{\,G}\right)_O^{EO} = \left(\dot{\vec{\mathbf{V}}}_K^{\,G}\right)_O^{EO} - \left(\dot{\vec{\mathbf{V}}}_W^{\,G}\right)_O^{EO} \tag{3.93}$$

The first order time derivative of the wind velocity vector $\vec{\mathbf{V}}_W$ with respect to the *NED*-Frame can be calculated by taking the total derivative:

$$\left(\dot{\vec{\mathbf{V}}}_W^{\,G}\right)_O^{EO} = \left(\frac{\partial}{\partial t}\right)^O \left(\vec{\mathbf{V}}_W^{\,G}\right)_O^E + \left[\nabla^O \left(\vec{\mathbf{V}}_W^{\,G}\right)_O^E\right] \cdot \left(\vec{\mathbf{V}}_K^{\,G}\right)_O^E \tag{3.94}$$

In case that the velocity $\vec{\mathbf{V}}_W$ is assumed to be time-constant, this equation reduces to:

$$\left(\dot{\vec{\mathbf{V}}}_W^G\right)_O^{EO} = \left[\nabla^O\left(\vec{\mathbf{V}}_W^G\right)_O^E\right] \cdot \left(\vec{\mathbf{V}}_K^G\right)_O^E \tag{3.95}$$

With respect to the *NED*-Frame, the convective wind field is given by the gradient matrix $\nabla\vec{\mathbf{V}}_W$ of the wind velocity $\vec{\mathbf{V}}_W$ relative to the *ECEF*-Frame:

$$\left[\nabla^O\left(\vec{\mathbf{V}}_W^G\right)_O^E\right] = \begin{bmatrix} \frac{\partial\left(u_W^G\right)_O^E}{\partial(x)_O} & \frac{\partial\left(u_W^G\right)_O^E}{\partial(y)_O} & \frac{\partial\left(u_W^G\right)_O^E}{\partial(z)_O} \\ \frac{\partial\left(v_W^G\right)_O^E}{\partial(x)_O} & \frac{\partial\left(v_W^G\right)_O^E}{\partial(y)_O} & \frac{\partial\left(v_W^G\right)_O^E}{\partial(z)_O} \\ \frac{\partial\left(w_W^G\right)_O^E}{\partial(x)_O} & \frac{\partial\left(w_W^G\right)_O^E}{\partial(y)_O} & \frac{\partial\left(w_W^G\right)_O^E}{\partial(z)_O} \end{bmatrix} \tag{3.96}$$

With the help of the aerodynamic velocity $\vec{\mathbf{V}}_A$, the aerodynamic flight-path course angle χ_A and the aerodynamic flight-path climb angle γ_A as well as the absolute aerodynamic velocity V_A can be restored:

$$\left(\vec{\mathbf{V}}_A^G\right)_O^E = \begin{pmatrix} u_A^G \\ v_A^G \\ w_A^G \end{pmatrix}_O^E = \begin{pmatrix} V_A^G \cdot \cos\chi_A^G \cdot \cos\gamma_A^G \\ V_A^G \cdot \sin\chi_A^G \cdot \cos\gamma_A^G \\ -V_A^G \cdot \sin\gamma_A^G \end{pmatrix}_O^E \tag{3.97}$$

$$V_A^G = \left\|\left(\vec{\mathbf{V}}_A^G\right)_O^E\right\|_2 = \sqrt{\left[\left(u_A^G\right)_O^E\right]^2 + \left[\left(v_A^G\right)_O^E\right]^2 + \left[\left(w_A^G\right)_O^E\right]^2} \tag{3.98}$$

$$\chi_A^G = \arctan\left(\frac{\left(v_A^G\right)_O^E}{\left(u_A^G\right)_O^E}\right) \tag{3.99}$$

$$\gamma_A^G = -\arctan\left(\frac{\left(w_A^G\right)_O^E}{\sqrt{\left[\left(u_A^G\right)_O^E\right]^2 + \left[\left(v_A^G\right)_O^E\right]^2}}\right) \tag{3.100}$$

where the aerodynamic flight-path course angle χ_A and the aerodynamic flight-path climb angle γ_A give the transformation from the *NED*-Reference Frame O to the Intermediate Aerodynamic Frame $\bar{A}$. The kinematic angle of attack α_K and the kinematic sideslip angle β_K are used to determine the transformation matrix $\mathbf{M}_{B\bar{K}}$ between the Intermediate Kinematic Flight-Path Frame $\bar{K}$ and the Body-Fixed Frame B that in turn gives the transformation matrix $\mathbf{M}_{BO}$ between the Body-Fixed Frame B and the *NED*-Frame with the transformation matrix $\mathbf{M}_{\bar{K}O}$ given by the kinematic flight-path angles respectively the quaternions:

$$\mathbf{M}_{BO} = \mathbf{M}_{B\bar{K}} \cdot \mathbf{M}_{\bar{K}O} = \begin{bmatrix} \cos\alpha_K \cdot \cos\beta_K & -\cos\alpha_K \cdot \sin\beta_K & -\sin\alpha_K \\ \sin\beta_K & \cos\beta_K & 0 \\ \sin\alpha_K \cdot \cos\beta_K & -\sin\alpha_K \cdot \sin\beta_K & \cos\alpha_K \end{bmatrix} \cdot \mathbf{M}_{\bar{K}O} \tag{3.101}$$

$$\mathbf{M}_{\bar{K}O} = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2\cdot(q_1 \cdot q_2 + q_0 \cdot q_3) & 2\cdot(q_1 \cdot q_3 - q_0 \cdot q_2) \\ 2\cdot(q_1 \cdot q_2 - q_0 \cdot q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2\cdot(q_2 \cdot q_3 + q_0 \cdot q_1) \\ 2\cdot(q_1 \cdot q_3 + q_0 \cdot q_2) & 2\cdot(q_2 \cdot q_3 - q_0 \cdot q_1) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \tag{3.102}$$

With the help of the transformation matrix $\mathbf{M}_{BO}$, the aerodynamic velocity $\vec{\mathbf{V}}_A$ of the center of gravity with respect to the *ECEF*-Frame and its components denoted in the Body-Fixed Frame *B* can be restored:

$$\left(\vec{\mathbf{V}}_A^G\right)_B^E = \mathbf{M}_{BO} \cdot \left(\vec{\mathbf{V}}_A^G\right)_O^E = \mathbf{M}_{BO} \cdot \left[\left(\vec{\mathbf{V}}_K^G\right)_O^E - \left(\vec{\mathbf{V}}_W^G\right)_O^E\right] \tag{3.103}$$

This aerodynamic velocity $\vec{\mathbf{V}}_A$ in turn allows for the calculation of the absolute aerodynamic velocity V_A, the aerodynamic angle of attack α_A and the aerodynamic angle of sideslip β_A:

$$\left(\vec{\mathbf{V}}_A^G\right)_B^E = \begin{pmatrix} u_A^G \\ v_A^G \\ w_A^G \end{pmatrix}_B^E = \mathbf{M}_{BA} \cdot \left(\vec{\mathbf{V}}_A^G\right)_A^E = \mathbf{M}_{BA} \cdot \begin{pmatrix} V_A^G \\ 0 \\ 0 \end{pmatrix}_A^E = \begin{pmatrix} V_A^G \cdot \cos\alpha_A^G \cdot \cos\beta_A^G \\ V_A^G \cdot \sin\beta_A^G \\ V_A^G \cdot \sin\alpha_A^G \cdot \cos\beta_A^G \end{pmatrix}_B^E \tag{3.104}$$

$$V_A^G = \left\| \left(\vec{\mathbf{V}}_A^G\right)_B^E \right\|_2 = \sqrt{\left[\left(u_A^G\right)_B^E\right]^2 + \left[\left(v_A^G\right)_B^E\right]^2 + \left[\left(w_A^G\right)_B^E\right]^2} \tag{3.105}$$

$$\alpha_A^G = \arctan\left(\frac{\left(w_A^G\right)_B^E}{\left(u_A^G\right)_B^E}\right) \tag{3.106}$$

$$\beta_A^G = \arctan\left(\frac{\left(v_A^G\right)_B^E}{\sqrt{\left[\left(u_A^G\right)_B^E\right]^2 + \left[\left(w_A^G\right)_B^E\right]^2}}\right) \tag{3.107}$$

Finally, the aerodynamic bank μ_A angle is computed with the help of the transformation matrices between the Aerodynamic Reference Frame *A* and the Intermediate Aerodynamic Reference Frame $\bar{A}$ that is the Aerodynamic Reference Frame *A* rotated by the aerodynamic bank angle μ_A around its *x*-axis:

$$\mathbf{M}_{\bar{A}A}\left(\mu_A^G\right) = \mathbf{M}_{\bar{A}N}\left(\chi_A^G, \gamma_A^G\right) \cdot \mathbf{M}_{NK}\left(\chi_K^G, \gamma_K^G\right) \cdot \mathbf{M}_{KB}\left(\mu_K^G, \alpha_K^G, \beta_K^G\right) \cdot \mathbf{M}_{BA}\left(\alpha_A^G, \beta_A^G\right) \tag{3.108}$$

$$\mu_A^G = \arctan\left(\frac{\mathbf{M}_{\bar{A}A}(2,2)}{\mathbf{M}_{\bar{A}A}(3,2)}\right) \tag{3.109}$$

3.2.10 External Forces and Moments

For the derivation of the translation and rotation equations of motion in the preceding chapters, the sum of external forces and the sum of external moments acting on the aircraft have been related to the translational respectively the rotational accelerations of the aircraft by applying Newton's 2nd law. Subsequently, the external forces and moments will be examined in detail, where the sum of forces is split up into the categories gravitational force $\vec{\mathbf{F}}_G$, aerodynamic forces $\vec{\mathbf{F}}_A$ and propulsion force $\vec{\mathbf{F}}_P$ and the moments are divided into the categories aerodynamic moments $\vec{\mathbf{M}}_A$ and propulsive moments $\vec{\mathbf{M}}_P$:

$$\sum \left(\vec{\mathbf{F}}^{G}\right)_{K} = \left(\vec{\mathbf{F}}_{A}^{G}\right)_{K} + \left(\vec{\mathbf{F}}_{P}^{G}\right)_{K} + \left(\vec{\mathbf{F}}_{G}^{G}\right)_{K} = \mathbf{M}_{KA}\left(\vec{\mathbf{F}}_{A}^{G}\right)_{A} + \mathbf{M}_{KB}\left(\vec{\mathbf{F}}_{P}^{G}\right)_{B} + \left(\vec{\mathbf{F}}_{G}^{G}\right)_{K} \tag{3.110}$$

$$\sum \left(\vec{\mathbf{M}}^{G}\right)_{B} = \left(\vec{\mathbf{M}}_{A}^{G}\right)_{B} + \left(\vec{\mathbf{M}}_{P}^{G}\right)_{B} \tag{3.111}$$

where $\mathbf{M}_{KA}$ and $\mathbf{M}_{KB}$ are the transformation matrices between the Kinematic Flight-Path Reference Frame K and the Aerodynamic Reference Frame A respectively the Body-Fixed Reference Frame B.

3.2.10.1. Gravitational Force

Since flight trajectories with relatively low altitudes and with a limited geographic extent are to be considered, the decrease in the gravitational force with increasing flight altitude as well as inhomogeneities in the Earth's gravitational field are not accounted for in the simulation model. Thus, the absolute gravitational force acting on the aircraft is considered to be a constant value. Since the point of application of the gravitational force coincides with the center of gravity of the aircraft, no moments are induced by the attraction force. Denoting the gravitational force in the Kinematic Flight-Path Frame K, the following expression results:

$$\left(\vec{\mathbf{F}}_{G}^{G}\right)_{K} = \begin{pmatrix} F_{G,x} \\ F_{G,y} \\ F_{G,z} \end{pmatrix}_{K} = \mathbf{M}_{KO} \cdot \left(\vec{\mathbf{F}}_{G}^{G}\right)_{O} = \mathbf{M}_{KO} \cdot \begin{pmatrix} 0 \\ 0 \\ m \cdot g \end{pmatrix}_{O} \tag{3.112}$$

3.2.10.2. Aerodynamic Forces and Moments

The aerodynamic forces and moments acting on the aircraft are invoked by the airflow surrounding the flight vehicle. For the calculation of the aerodynamic forces, it is assumed that the aerodynamic reference point coincides with the center of gravity of the aircraft. Then, the aerodynamic forces and moments with respect to the center of gravity G denoted in the Aerodynamic Frame A respectively the Body-Fixed Frame B are computed by the following formulae:

$$\left(\vec{\mathbf{F}}_{A}^{G}\right)_{A} = \begin{pmatrix} -D \\ Q \\ -L \end{pmatrix}_{A} = \bar{q} \cdot S \cdot \begin{pmatrix} -C_{D} \\ C_{Q} \\ -C_{L} \end{pmatrix}_{A} \tag{3.113}$$

$$\left(\vec{\mathbf{M}}_{A}^{G}\right)_{B} = \begin{pmatrix} L \\ M \\ N \end{pmatrix}_{B} = \bar{q} \cdot S \cdot \begin{pmatrix} s \cdot C_{l} \\ \bar{c} \cdot C_{m} \\ s \cdot C_{n} \end{pmatrix}_{B} \tag{3.114}$$

where D denotes the aerodynamic drag force, Q the aerodynamic force in the direction of the y-axis of the Aerodynamic Frame A and L the lift force. The aerodynamic moments L, M and N denote the rolling moment, the pitching moment and the yawing moment respectively. The aerodynamic force and moment coefficients are primarily functions of the aerodynamic angles α_A and β_A, the control surface deflections ξ, η and ζ and the dimensionless aerodynamic angular rates $\tilde{p}_A$, $\tilde{q}_A$ and $\tilde{r}_A$:

$$\tilde{p}_{A} = \frac{p_{A} \cdot b}{2 \cdot V_{A}} \tag{3.115}$$

$$\widetilde{q}_A = \frac{q_A \cdot \bar{c}}{2 \cdot V_A} \tag{3.116}$$

$$\widetilde{r}_A = \frac{r_A \cdot b}{2 \cdot V_A} \tag{3.117}$$

The aerodynamic angular rates p_A, q_A and r_A are the elements of the aerodynamic rotation vector $\vec{\boldsymbol{\omega}}_A^{AB}$ between the aircraft and the surrounding air denoted in the Body-Fixed Frame B:

$$\left(\vec{\boldsymbol{\omega}}_A^{AB}\right)_B = \begin{pmatrix} p_A \\ q_A \\ r_A \end{pmatrix}_B \tag{3.118}$$

Analogous to the aerodynamic velocity $\vec{\mathbf{V}}_A$, the aerodynamic rotation vector $\vec{\boldsymbol{\omega}}_A$ can in general be derived from the difference between the kinematic rotation vector $\vec{\boldsymbol{\omega}}_K$ and the wind rotation vector $\vec{\boldsymbol{\omega}}_W$ of the surrounding atmosphere (Ref. [Brockhaus, 2001]):

$$\vec{\boldsymbol{\omega}}_A = \vec{\boldsymbol{\omega}}_K - \vec{\boldsymbol{\omega}}_W \tag{3.119}$$

Thus, the aerodynamic rotation vector $\vec{\boldsymbol{\omega}}_A^{AB}$ is obtained by subtracting the rotation vector $\vec{\boldsymbol{\omega}}_W^{OA}$ of the circumfluent air relative to the *NED*-Frame from the rotation vector $\vec{\boldsymbol{\omega}}_K^{OB}$ of the flight vehicle relative to the *NED*-Frame:

$$\left(\vec{\boldsymbol{\omega}}_A^{AB}\right)_B = \left(\vec{\boldsymbol{\omega}}_K^{OB}\right)_B - \left(\vec{\boldsymbol{\omega}}_W^{OA}\right)_B \tag{3.120}$$

The rotation vector $\vec{\boldsymbol{\omega}}_W^{OA}$ of the surrounding air relative to the *NED*-Frame is given by Eq. (3.121), utilizing the non-diagonal elements of the gradient matrix $\nabla\vec{\mathbf{V}}_W$ of the wind velocity $\vec{\mathbf{V}}_W$ given by Eq. (3.96) (Ref. [Brockhaus, 2001]).

$$\left(\vec{\boldsymbol{\omega}}_W^{OA}\right)_O = \frac{1}{2}\,\mathbf{rot}_O\left(\vec{\mathbf{V}}_W^G\right)_O^E = \frac{1}{2}\left[\vec{\nabla}^O \times \left(\vec{\mathbf{V}}_W^G\right)_O^E\right] = \frac{1}{2}\begin{bmatrix} \dfrac{\partial\left(w_W^G\right)_O^E}{\partial(y)_O} - \dfrac{\partial\left(v_W^G\right)_O^E}{\partial(z)_O} \\ \dfrac{\partial\left(u_W^G\right)_O^E}{\partial(z)_O} - \dfrac{\partial\left(w_W^G\right)_O^E}{\partial(x)_O} \\ \dfrac{\partial\left(v_W^G\right)_O^E}{\partial(x)_O} - \dfrac{\partial\left(u_W^G\right)_O^E}{\partial(y)_O} \end{bmatrix} \tag{3.121}$$

3.2.10.3. Propulsion Forces and Moments

The installed propulsion system of a flight vehicle generates a thrust force acting at the thrust reference point. In general, this thrust reference point is not identical to the center of gravity and the thrust direction is not aligned with the Body-Fixed Reference Frame's x-axis. The net thrust generated by the propulsion system can be split up into an inlet impulse T_I and an outlet impulse T_O (Fig. 15) acting at the reference points T_I and T_O, where the inlet impulse is parallel to the x-axis of the Aerodynamic Frame A. The direction of the outlet impulse with respect to the Body-Fixed Frame B is considered as constant and is given by the two mounting angles of an aircraft's engine that are the angle κ in the body-fixed xy-plane and the thrust elevation angle σ. For the depiction of the outlet impulse T_O, a Propulsion Frame P is introduced that is the Body-Fixed Frame B rotated by the two mounting angles κ and σ of the aircraft's engine, so that consequently the x-axis of the Propulsion Frame P is aligned with the

direction of the outlet impulse T_O. Especially for air-breathing propulsion systems, table data for the outlet impulse are provided instead of table data for the net forces and moments.

The net thrust force with its components given in the Body-Fixed Frame B is the sum of the outlet impulse T_I and the inlet impulse T_O:

$$\begin{aligned}\left(\vec{\mathbf{F}}_P^G\right)_B &= \left(\vec{\mathbf{F}}_{P,T_I}^{T_I}\right)_B + \left(\vec{\mathbf{F}}_{P,T_O}^{T_O}\right)_B \\ &= \mathbf{M}_{BA}\cdot\left(\vec{\mathbf{F}}_{P,T_I}^{T_I}\right)_A + \mathbf{M}_{BP}\cdot\left(\vec{\mathbf{F}}_{P,T_O}^{T_O}\right)_P \\ &= \mathbf{M}_{BA}\cdot\begin{pmatrix}-T_I\\0\\0\end{pmatrix}_A + \mathbf{M}_{BP}\cdot\begin{pmatrix}T_O\\0\\0\end{pmatrix}_P\end{aligned} \tag{3.122}$$

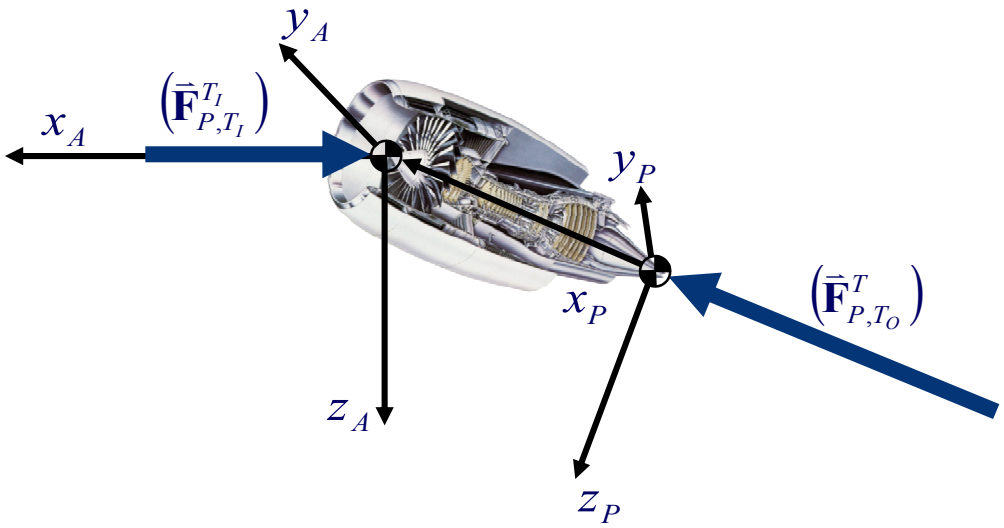

Figure 15. Depiction of Inlet Impulse and Outlet Impulse

$\mathbf{M}_{BP}$ is the transformation matrix between the Body-Fixed Reference Frame B and the Propulsion Frame P:

$$\mathbf{M}_{BP} = \begin{bmatrix}\cos\kappa\cdot\cos\sigma & -\sin\kappa & \cos\kappa\cdot\sin\sigma\\ \sin\kappa\cdot\cos\sigma & \cos\kappa & \sin\kappa\cdot\sin\sigma\\ -\sin\sigma & 0 & \cos\sigma\end{bmatrix} \tag{3.123}$$

Because of the lever arms between the aircraft's center of gravity point G and the thrust reference points T_I and T_O, the inlet impulse T_I and the outlet impulse T_O also induce a supplementary moment around the center of gravity point G of the flight system. Furthermore, components of the propulsion system rotating at high speeds like e.g. shafts induce gyroscopic moments $\vec{\mathbf{M}}_{Gyro}$ also acting on the aircraft. These moments can be taken into account in the total propulsion moments $\vec{\mathbf{M}}_P$ acting at the aircraft's center of gravity point:

$$\left(\vec{\mathbf{M}}_P^G\right)_B = \left(\vec{\mathbf{M}}_{Gyro}^T\right)_B + \left(\vec{\mathbf{r}}^{GT_I}\right)_B \times \left(\vec{\mathbf{F}}_{P,T_I}^{T_I}\right)_B + \left(\vec{\mathbf{r}}^{GT_O}\right)_B \times \left(\vec{\mathbf{F}}_{P,T_O}^{T_O}\right)_B \tag{3.124}$$

If the propulsion system is not a tilt-rotor system nor a system with thrust vector control, i.e. if the Propulsion Frame P is fixed relative to the Body-Fixed Reference Frame B, the gyroscopic moments are (Ref. [Holzapfel, 2009b]):

$$\left(\vec{\mathbf{M}}_{Gyro}^T\right)_B = \left(\mathbf{I}^{G_{Rot}}\right)_{BB}\cdot\left(\dot{\vec{\boldsymbol{\omega}}}_K^{PRot}\right)_B^P + \left(\vec{\boldsymbol{\omega}}_K^{IB}\right)_B \times \left[\left(\mathbf{I}^{G_{Rot}}\right)_{BB}\cdot\left(\vec{\boldsymbol{\omega}}_K^{PRot}\right)_B\right] \tag{3.125}$$

Here, *Rot* denotes the Rotor Reference Frame that is fixed to the rotor and thus rotates with the rotor. The *x*-axis of the Rotor Reference Frame *Rot* is aligned with the *x*-axis of the Propulsion Frame *P* and the rotation vector $\vec{\boldsymbol{\omega}}_K^{PRot}$ is given by

$$\left(\vec{\boldsymbol{\omega}}_K^{PRot}\right)_P = \begin{bmatrix} \omega_{K,x}^{PRot} \\ 0 \\ 0 \end{bmatrix}_P \tag{3.126}$$

For the calculation of the thrust force T that is a function of the commanded thrust lever position $\delta_{T,CMD}$, one has to bear in mind that due to the engine dynamics the thrust force cannot perform discrete changes even if a step change in the thrust lever position is commanded. For this reason, it is much more realistic to model the relationship between the commanded thrust lever position $\delta_{T,CMD}$ and the thrust force T as a linear transfer function of first order depending on the maximum possible thrust for any given altitude and velocity:

$$T = T_{\max} \cdot \delta_T = T_{\max} \cdot \frac{1}{T_\delta \cdot s + 1} \cdot \delta_{T,CMD} \tag{3.127}$$

This transfer function can also be rewritten as a differential equation:

$$\dot{\delta}_T = \frac{1}{T_\delta}\left(\delta_{T,CMD} - \delta_T\right) \tag{3.128}$$

The engine time constant T_δ can be adjusted to simulate the real engine dynamics in a realistic manner. With the effective thrust lever position δ_T a further state is added to the non-linear simulation model. Within the simulation model, the effective thrust lever position δ_T has to be limited so that on the one hand the thrust force cannot take negative values and on the other hand the thrust force cannot exceed the maximum available thrust force T_{max}:

$$\delta_T \in [0,1] \tag{3.129}$$

Furthermore, the first order time derivative of the effective thrust lever position can be limited to achieve a realistic reproduction of the characteristic dynamics of the propulsion system in the simulation model:

$$\dot{\delta}_{T,\min} \le \dot{\delta}_T \le \dot{\delta}_{T,\max} \tag{3.130}$$

3.2.11 Actuator Dynamics

Additionally to the basic flight system components that have been considered so far, for a realistic simulation of the aircraft it is also necessary to take into account further characteristic dynamics that are directly enclosed in the physical causal chain of the flight system. In this conjunction, one important aspect that has to be considered are the dynamics associated with the actuators installed in the aircraft. Even in case of an aerobatic aircraft without any actuators installed, the pilot will not be able to move the control surfaces at infinite speed because of the pilot's limited maximum physical motion speed that in turn is further reduced by the counteracting aerodynamic forces acting on the control surfaces. Thus, it is more realistic to assume that discrete jumps in the acceleration of the control surfaces are possible instead of discrete changes in the velocity or even in the position of the control surfaces. This justifies the modeling of the actuator dynamics by a substituted mechanical system of second

order. The according linear transfer functions between the commanded control surface deflections and the achieved deflection values have the following form:

$$y = \frac{\omega_0^2}{s^2 + 2 \cdot \zeta \cdot \omega_0 \cdot s + \omega_0^2} u_{CMD} \tag{3.131}$$

Here, the input u_{CMD} represents the commanded control surface deflection while the output y gives the effectively achieved value. This transfer function can also be cast in state-space form:

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & -2 \cdot \zeta \cdot \omega_0 \end{bmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{bmatrix} 0 \\ \omega_0^2 \end{bmatrix} \cdot u_{CMD} \tag{3.132}$$

$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \tag{3.133}$$

Because the actuator dynamics are modeled as second-order mechanical systems, the aileron deflection ξ, the elevator surface deflection η and the rudder deflection ζ augment the state-vector of the optimization simulation model by totally six states that are the effective control surface deflections ξ, η and ζ and their first order time derivatives $\dot{\xi}$, $\dot{\eta}$ and $\dot{\zeta}$. Within the simulation model, these states are limited to take into account the maximum achievable control surface deflections and rates of the regarded flight system and to simulate this flight system as precise as possible. Furthermore, if the maximum occurring control moments shall be limited, the accelerations, i.e. the second order time derivatives $\ddot{\xi}$, $\ddot{\eta}$ and $\ddot{\zeta}$ of the various control surface deflections have to be restricted.

3.2.12 Static Atmosphere

The atmospheric conditions are calculated in accordance to the International Standard Atmosphere DIN ISO 2533 (Ref. [NN, 1975]). Deviations from the norm standard atmosphere can be taken into account by an adjustment of the respective norm reference values p_0 and T_0 regarding the polytropic troposphere layer:

$$p_0^* = p_0 + \Delta p_{ISA} \tag{3.134}$$

$$T_0^* = T_0 + \Delta T_{ISA} \tag{3.135}$$

In this manner, deviations from the norm standard atmosphere given in the form *"ISA+25°C"* as it could be the case for air races taking place on a hot day e.g. in Abu Dhabi are taken into account for the simulation and optimization of the flight trajectories. If any deviations from the norm standard atmosphere are existent, the corresponding reference values T_{11}^*, T_{20}^*, p_{11}^* and p_{20}^* for the isothermal lower stratosphere layer respectively the polytropic upper stratosphere layer have to be calculated from T_0^* and p_0^*.

3.3 Feedback Linearization as Plant Transformation

Additionally to the simulation model with its different depths of modeling described above, for all subsystems inverse simulation models are implemented so that for given reference values the required command values that force the simulation model to track the prescribed reference values can be calculated. This inversion is based on the principle of nonlinear

dynamic inversion (Refs. [Slotine, 1991], [Holzapfel, 2004], [Holzapfel, 2009d] and [Khalil, 2001]) of the flight dynamics equations.

For the dynamic inversion, the equations of motion respectively the propagation equations implemented in the various subsystems are inverted so that the inputs that in turn will produce the desired outputs can be computed. It has to be mentioned that the inversion is only possible for minimum phase systems, i.e. systems that have all their transfer function zeros in the left-hand side of the complex plane. If any system or subsystem incorporates a non-minimum phase behavior, only the minimum phase part of the subsystem can be inverted to yield an inverse system that is also stable and causal.

The application of dynamic inversion to flight control tasks has a long tradition (Refs. [Snell, 1991], [Snell, 1992] and [Lane, 1988]) and has been successfully implemented in different experimental programs simulating a broad range of aircraft with various tasks (Ref. [Calise, 2000]). Over the time, a lot of modifications have been made to the basic concept, like the addition of adaptive terms to cancel the inversion error, a concept that has originally been demonstrated for robots. Basically the principle of dynamic inversion can be stated as follows: for a system with relative degree of one the dynamic inversion of the plant computes the required control inputs u_{CMD} with respect to any given reference trajectory $v = \dot{y}_{REF}$ such that the model plant reacts with the desired output trajectory $\dot{y} = \dot{y}_{REF}$.

For the full non-linear 6-Degree-of-Freedom simulation model plant, a change in the altitude results from a certain deflection of the elevator control surface over a chain of integrations. On the other side, for a desired altitude profile, the necessary elevator deflections can be computed by inverting the equations of motion of the simulation plant. If the computed control surface deflections are then commanded to the simulation model, the aircraft follows the desired reference trajectory.

As can be seen from Fig. 8 in chapter 3.1, there are at least four integrations between the control surface deflections and the resulting position of the aircraft if no actuator dynamics are taken into account, so the relative degree of the overall flight system is four. Thus, not only the reference time history itself but also higher order time derivatives are required as input to the inverse simulation model in order to be able to calculate the command values that have to be fed into the simulation model so that the simulation model follows a given reference time history. Since there are two integrations in the causal chain between the commanded control surface deflections and the resulting changes in the aerodynamic attitude angles respectively aerodynamic load factors (see Fig. 8 in chapter 3.1), for the dynamic inversion of the inner loop attitude and rotational dynamics the second order time derivatives of the reference load factors $\ddot{\mathbf{n}}$ are required. This in turn means that the third order time derivatives of the flight-path velocity $\dddot{V}$, the climb angle $\dddot{\gamma}$ and the course angle $\dddot{\chi}$ have to be computed thus demanding for the fourth order time derivative of the position vector $\vec{\mathbf{x}}^{(4)}$. In the following the derivation of the required reference values is depicted, starting from a given reference trajectory $\vec{\mathbf{x}}(\tau)$. For this reference trajectory, it is essential to be at least four times differentiable with respect to the trajectory parameter τ.

Since the inversion controller that is based on the principle of dynamic inversion is a substantial part of the simulation model and thus of the developed optimization algorithm, theoretical foundations on the principle of nonlinear dynamic inversion respectively input-output linearization are reviewed in chapter 3.3.1. More detailed information on the principle of dynamic inversion can also be found in Refs. [Slotine, 1991], [Holzapfel, 2004],

[Holzapfel, 2009d] or [Khalil, 2001]. The input to the outermost part of the inversion controller is a reference trajectory that the dynamic system ought to follow. Thus, in chapter 3.3.2 the method for computing the required reference values from a given input trajectory is depicted.

The dynamic inversion part of the simulation model is built up as a cascaded structure, whereas the reference signals for a particular subsystem are calculated from the signals given by the next outer subsystem. The derivation of the reference values for the distinct subsystems is explained in chapters 3.3.3 to 3.3.12, starting from the outermost system, the subsystem for the generation of the reference values for the kinematic flight-path variables, and then going inwards to the subsystem for the generation of the moments respectively the control surface reference values. Here, the major task of the different subsystems is to generate reference signals up to the respective required derivative order with respect to a given reference trajectory. For the computation of these reference values by the particular subsystems, not only the reference commands themselves coming from the next outer subsystem are required as input signals but also their derivatives with respect to time up to a certain order.

3.3.1 Non-Linear Dynamic Inversion - Theoretical Fundamentals

The primary objective of non-linear dynamic inversion is to find a non-linear transformation of the state vector of the form

$$\mathbf{z} = \boldsymbol{\Phi}(\mathbf{x}) \tag{3.136}$$

so that the transformed system shows a linear input-output behavior. Therefore, the principle of non-linear dynamic inversion is also called *exact input-output linearization*. It can be termed exact, since the linearization is done without any approximations or simplifications of the underlying dynamic system.

In the following, non-linear multi-input multi-output (MIMO) systems are considered that are of the state-space form

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{G}(\mathbf{x}) \cdot \mathbf{u} \tag{3.137}$$

$$\mathbf{y} = \mathbf{h}(\mathbf{x}) \tag{3.138}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the state vector, $\mathbf{u} \in \mathbb{R}^m$ the control input vector and $\mathbf{y} \in \mathbb{R}^m$ the output vector of the dynamic system. The functions $\mathbf{f}$: $\mathbf{x} \longrightarrow \mathbb{R}^n$, $\mathbf{G}$: $\mathbf{u} \longrightarrow \mathbb{R}^m$ and $\mathbf{h}$: $\mathbf{x} \longrightarrow \mathbb{R}^m$ that are functions of the *n*-dimensional state vector have to be sufficiently smooth, i.e. that they have to be continuously differentiable up to a certain differentiation order. Since the control input vector $\mathbf{u}$ appears linear in the system (3.137), the system is called input affine. Furthermore, the system is square, i.e. the number of input variables equals the number of outputs. The matrix $\mathbf{G}$ is comprised of *m* vector fields $\mathbf{g}_i$:

$$\mathbf{G}(\mathbf{x}) = \begin{bmatrix} \mathbf{g}_1(\mathbf{x}) & \cdots & \mathbf{g}_m(\mathbf{x}) \end{bmatrix} \tag{3.139}$$

The vector fields $\mathbf{f}$, $\mathbf{g}_i$ and $\mathbf{h}$ in turn consist of non-linear functions of the state vector $\mathbf{x}$:

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} f_1(\mathbf{x}) & \cdots & f_n(\mathbf{x}) \end{bmatrix}^T \tag{3.140}$$

$$\mathbf{g}_i(\mathbf{x}) = \begin{bmatrix} g_{i,1}(\mathbf{x}) & \cdots & g_{i,n}(\mathbf{x}) \end{bmatrix}^T \quad i = 1, \ldots, m \tag{3.141}$$

$$\mathbf{h}(\mathbf{x}) = \begin{bmatrix} h_1(\mathbf{x}) & \cdots & h_m(\mathbf{x}) \end{bmatrix}^T \tag{3.142}$$

Now, the basic approach of the dynamic inversion principle in order to linearize the input-output behavior of the regarded system is to differentiate the outputs repeatedly with respect to time until the control **u** appears. Then, the control **u** can be designed such that the non-linearities of the system are canceled out. A necessary pre-condition in order to be able to accomplish this task is the existence of a well-defined relative degree *r* of the considered system that is defined as follows:

Definition: A non-linear multi-input multi-output system of the form (3.137) to (3.138) is said to have a **vectorial relative degree** $\{r_1, \ldots, r_m\}$ at a point $\mathbf{x}_0$ if

$$L_{\mathbf{g}_j} L_{\mathbf{f}}^k h_i(\mathbf{x}) = 0 \quad \forall 1 \le j \le m, k < r_i - 1, 1 \le i \le m \tag{3.143}$$

with **x** being in the neighborhood **Ω** of $\mathbf{x}_0$. Furthermore, the *m* x *m* decoupling matrix **A** defined by Eq. (3.144) must not be singular at $\mathbf{x} = \mathbf{x}_0$, which also implies that the decoupling matrix **A** is invertible.

$$\mathbf{A}(\mathbf{x}) = \begin{bmatrix} L_{\mathbf{g}_1} L_{\mathbf{f}}^{r_1-1} h_1(\mathbf{x}) & L_{\mathbf{g}_2} L_{\mathbf{f}}^{r_1-1} h_1(\mathbf{x}) & \cdots & L_{\mathbf{g}_m} L_{\mathbf{f}}^{r_1-1} h_1(\mathbf{x}) \\ L_{\mathbf{g}_1} L_{\mathbf{f}}^{r_2-1} h_2(\mathbf{x}) & L_{\mathbf{g}_2} L_{\mathbf{f}}^{r_2-1} h_2(\mathbf{x}) & \cdots & L_{\mathbf{g}_m} L_{\mathbf{f}}^{r_2-1} h_2(\mathbf{x}) \\ \vdots & \vdots & \ddots & \vdots \\ L_{\mathbf{g}_1} L_{\mathbf{f}}^{r_m-1} h_m(\mathbf{x}) & L_{\mathbf{g}_2} L_{\mathbf{f}}^{r_m-1} h_m(\mathbf{x}) & \cdots & L_{\mathbf{g}_m} L_{\mathbf{f}}^{r_m-1} h_m(\mathbf{x}) \end{bmatrix} \tag{3.144}$$

For the vectorial relative degree, the following relation holds:

$$r = r_1 + \ldots + r_m = \sum_{i=1}^{m} r_i \le n \tag{3.145}$$

In (3.143) and (3.144), the so-called Lie derivatives $L_f h$ are used that are defined as follows:

Definition: Given a real-valued, smooth scalar function h: $\mathbb{R}^n \longrightarrow \mathbb{R}$ and a real-valued, smooth vector field **f**: $\mathbb{R}^n \longrightarrow \mathbb{R}^n$, the **Lie derivative** $L_f h$ of h with respect to **f** is defined as the derivative of the function h along the vector field **f**:

$$L_{\mathbf{f}} h = \left(\frac{\partial h(\mathbf{x})}{\partial \mathbf{x}} \right)^T \cdot \mathbf{f}(\mathbf{x}) \tag{3.146}$$

i.e. the Lie derivative is the gradient of h with respect to **x** projected onto the vector field **f**. Since the resulting Lie derivative is again a scalar value, higher order Lie derivatives with respect to the same vector field can be computed by recursion:

$$L_{\mathbf{f}}^k h(\mathbf{x}) = \frac{\partial \left(L_{\mathbf{f}}^{k-1} h(\mathbf{x}) \right)}{\partial \mathbf{x}} \cdot \mathbf{f}(\mathbf{x}) \tag{3.147}$$

with

$$L_{\mathbf{f}}^0 h(\mathbf{x}) = h(\mathbf{x}). \tag{3.148}$$

If another vector field **g**: $\mathbb{R}^n \longrightarrow \mathbb{R}^n$ is given, the recursive execution of the Lie derivative with respect to **f** and **g** gives:

$$L_{\mathbf{g}} L_{\mathbf{f}} h(\mathbf{x}) = \frac{\partial \left(L_{\mathbf{f}} h(\mathbf{x}) \right)}{\partial \mathbf{x}} \cdot \mathbf{g}(\mathbf{x}) \tag{3.149}$$

Finally, the Lie derivative can also be applied with respect to a matrix **G** as given by Eq. (3.139):

$$L_{\mathbf{G}}h = \left(L_{\mathbf{g}_1}h \quad \ldots \quad L_{\mathbf{g}_m}h\right) \tag{3.150}$$

which means that the gradient of the function h is projected on the vector fields $\mathbf{g}_i$ that form the columns of the matrix **G**.

For the system given by (3.137) and (3.138), the relative degree r_i of each single output y_i can be determined by differentiating the respective output with respect to time repeatedly until a control variable u_i appears directly in the r_i-th order time derivative of the output y_i:

$$\dot{y}_i = \frac{dy_i}{dt} = \left(\frac{\partial h_i}{\partial \mathbf{x}}\right)^T \cdot \dot{\mathbf{x}} = \left(\frac{\partial h_i}{\partial \mathbf{x}}\right)^T \cdot \mathbf{f}(\mathbf{x}) + \left(\frac{\partial h_i}{\partial \mathbf{x}}\right)^T \cdot \mathbf{G}(\mathbf{x}) \cdot \mathbf{u} = L_{\mathbf{f}}h_i + \overbrace{L_{\mathbf{G}}h_i}^{=0}\mathbf{u} \tag{3.151}$$

$$\ddot{y}_i = \left(\frac{\partial L_{\mathbf{f}}h_i}{\partial \mathbf{x}}\right)^T \cdot \mathbf{f}(\mathbf{x}) + \left(\frac{\partial L_{\mathbf{f}}h_i}{\partial \mathbf{x}}\right)^T \cdot \mathbf{G}(\mathbf{x}) \cdot \mathbf{u} = L_{\mathbf{f}}^2 h_i + \overbrace{L_{\mathbf{G}}L_{\mathbf{f}}h_i}^{=0} \cdot \mathbf{u} \tag{3.152}$$

$$y_i^{(r_i)} = \left(\frac{\partial L_{\mathbf{f}}^{r_i-1}h_i}{\partial \mathbf{x}}\right)^T \cdot \mathbf{f}(\mathbf{x}) + \left(\frac{\partial L_{\mathbf{f}}^{r_i-1}h_i}{\partial \mathbf{x}}\right)^T \cdot \mathbf{G}(\mathbf{x}) \cdot \mathbf{u} = L_{\mathbf{f}}^{r_i}h_i + \overbrace{L_{\mathbf{G}}L_{\mathbf{f}}^{r_i-1}h_i}^{\neq 0} \cdot \mathbf{u} \tag{3.153}$$

Here it is assumed that the control vector **u** at a point $\mathbf{x} = \mathbf{x}_0$ has no influence on the first r_i-1 time derivatives, so that the corresponding Lie derivatives are zero. This implies that the r_i-th order time derivative of the output y_i is the first time derivative order that can directly be influenced by the control input **u**, while the lower order time derivatives do not depend on the control vector **u** in a direct manner and thus cannot be influenced forthright. Thus, the relative degree r_i of an output y_i provides the lowest differentiation order of the considered output that can be directly prescribed by a control input which means that the r_i-th order time derivative of this output is proper with respect to the control input. Furthermore, the relative degree r_i can be regarded as a measure for the minimum possible time delay for the output y_i to react to any change $\Delta\mathbf{u}$ in the control input since the lower order time derivatives of the output y_i result from integration of its r_i-th order time derivative. In case of a linear transfer function the relative degree r_i of the system matches the pole excess, i.e. the difference between the degree of the denominator polynomial and the degree of the numerator polynomial.

As mentioned before, the existence of a well-defined relative degree r as well as the non-singularity of the decoupling matrix **A** are necessary conditions in order to perform an input-output linearization of a specific system. If these conditions are fulfilled, the transformation (3.136) can be carried out so that the resulting system will feature a linear input-output relationship. This is illustrated in the following, where for a system with multiple outputs at first the individual outputs are treated separately before finally the transformations for the single outputs are combined together to give the transformation for the entire multi-output system.

Exploiting the fact that for the output y_i with relative degree r_i the first r_i-1 Lie derivatives do not depend on the control **u**, i.e.

$$y_i^{(k)} = L_{\mathbf{f}}^k h_i(\mathbf{x}) \qquad k = 1, \ldots, r_i - 1 \tag{3.154}$$

the system can be transformed to

$$z_1^i = \xi_1^i = \Phi_1^i(\mathbf{x}) = y_i = L_{\mathbf{f}}^0 h_i(\mathbf{x}) = h_i(\mathbf{x}) \tag{3.155}$$

$$z_2^i = \xi_2^i = \Phi_2^i(\mathbf{x}) = \frac{d\Phi_1^i}{dt} = \dot{y}_i = L_{\mathbf{f}}^1 h_i(\mathbf{x}) = L_{\mathbf{f}} h_i(\mathbf{x}) \tag{3.156}$$

$$\vdots$$

$$z_{r_i-1}^i = \xi_{r_i-1}^i = \Phi_{r_i-1}^i(\mathbf{x}) = \frac{d\Phi_{r_i-2}^i}{dt} = y_i^{(r_i-2)} = L_{\mathbf{f}}^{r_i-2} h_i(\mathbf{x}) \tag{3.157}$$

$$z_{r_i}^i = \xi_{r_i}^i = \Phi_{r_i}^i(\mathbf{x}) = \frac{d\Phi_{r_1-1}^i}{dt} = y_i^{(r_i-1)} = L_{\mathbf{f}}^{r_i-1} h_i(\mathbf{x}). \tag{3.158}$$

Thus, the transformed state $\boldsymbol{\xi}^i$ vector for the i-th output is

$$\boldsymbol{\xi}^i = \begin{bmatrix} \xi_1^i & \cdots & \xi_{r_i}^i \end{bmatrix}^T \tag{3.159}$$

Joining together the transformed state vectors $\boldsymbol{\xi}^i$ for the different outputs gives the following state vector $\boldsymbol{\xi}$ for the first $r = r_1 + \ldots + r_m$ coordinates:

$$\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\xi}^1 \\ \vdots \\ \boldsymbol{\xi}^m \end{bmatrix} \tag{3.160}$$

If the state vector $\boldsymbol{\xi}$ obtained by the transformation contains less elements than the original state vector $\mathbf{x}$ comprising in total n elements, n-r additional coordinates $\eta_1, \ldots, \eta_{n-r}$ have to be found in order to ensure that the transformation (3.136) is a local diffeomorphism and thus a valid coordinate transformation. Formally, the definition of a diffeomorphism reads:

Definition: A function $\boldsymbol{\Phi}(\mathbf{x})$: $\mathbb{R}^n \longrightarrow \mathbb{R}^n$ is called a **diffeomorphism** if $\boldsymbol{\Phi}(\mathbf{x})$ is bijective, i.e. the function has to be invertible so that a function $\boldsymbol{\Phi}^{-1}(\mathbf{x})$ exists with

$$\boldsymbol{\Phi}^{-1}(\boldsymbol{\Phi}(\mathbf{x})) = \mathbf{x} \qquad \forall \mathbf{x} \in \mathbb{R}^n \tag{3.161}$$

and if $\boldsymbol{\Phi}(\mathbf{x})$ and $\boldsymbol{\Phi}^{-1}(\mathbf{x})$ are smooth, continuously differentiable mappings, i.e. that all partial derivatives have to exist and must be continuous:

$$\boldsymbol{\Phi}(\mathbf{x}) \in C^1 \tag{3.162}$$

$$\boldsymbol{\Phi}^{-1}(\mathbf{z}) \in C^1 \tag{3.163}$$

If the necessary conditions are fulfilled only locally at a point $\mathbf{x}_0$, and not globally in $\mathbb{R}^n$, the diffeomorphism is said to be a *local diffeomorphism*. In case of a local diffeomorphism, the Jacobi-matrix

$$\nabla\boldsymbol{\Phi}(\mathbf{x}) = \frac{\partial\boldsymbol{\Phi}(\mathbf{x})}{\partial\mathbf{x}} \tag{3.164}$$

of the mapping $\boldsymbol{\Phi}(\mathbf{x})$ is regular, i.e. invertible.

For the Jacobian $\partial\boldsymbol{\Phi}/\partial\mathbf{x}$ to be invertible, the row vectors of these matrix have to be linearly independent. This implies that for the selection of the remaining n-r coordinates $\eta_1, \ldots, \eta_{n-r}$ the linear independence of the derivatives $d\Phi_i$, $i = r+1, \ldots, n$ among themselves as well as their linear independence with respect to the r coordinates given by Eq. (3.160) have to be taken into account:

$$z_{r+1} = \eta_1 = \Phi_{r+1}(\mathbf{x}) \tag{3.165}$$

$$\vdots$$

$$z_n = \eta_{n-r} = \Phi_n(\mathbf{x}) \tag{3.166}$$

The full length state vector of the transformed system then reads:

$$\begin{aligned}\mathbf{z} &= \begin{bmatrix} z_1 & \cdots & z_n \end{bmatrix}^T \\ &= \begin{bmatrix} \Phi_1^1(\mathbf{x}) & \cdots & \Phi_{r_1}^1(\mathbf{x}) & \cdots & \Phi_1^m(\mathbf{x}) & \cdots & \Phi_{r_m}^m(\mathbf{x}) & \Phi_{r+1}(\mathbf{x}) & \cdots & \Phi_n(\mathbf{x}) \end{bmatrix}^T \\ &= \begin{bmatrix} \xi_1^1 & \cdots & \xi_{r_1}^1 & \cdots & \xi_1^m & \cdots & \xi_{r_m}^m & \eta_1 & \cdots & \eta_{n-r} \end{bmatrix}^T\end{aligned} \tag{3.167}$$

with the corresponding transformation being defined by Eq. (3.136). According to this transformation, each of the first r states corresponds to the first order time derivative of the preceding state:

$$z_i = \dot{z}_{i-1} \tag{3.168}$$

Then, substituting the state vector $\mathbf{x}$ by the inverse of the transformation, i.e. $\mathbf{x} = \mathbf{\Phi}^{-1}(\mathbf{z})$ and introducing the fraction state vectors $\boldsymbol{\xi} = [\xi_1, \ldots, \xi_r]^T$ and $\boldsymbol{\eta} = [\eta_1, \ldots, \eta_{n-r}]^T$ so that $\mathbf{z} = [\boldsymbol{\xi},\boldsymbol{\eta}]^T$, the following system dynamics result with one block for each output y_i, $i = 1, \ldots, m$

$$\dot{\xi}_1^i = \xi_2^i \tag{3.169}$$

$$\vdots$$

$$\dot{\xi}_{r_i}^i = b_i(\boldsymbol{\xi}, \boldsymbol{\eta}) + \sum_{j=1}^{m} a_{ij}(\boldsymbol{\xi}, \boldsymbol{\eta}) \cdot u_j \tag{3.170}$$

and one additional block

$$\dot{\boldsymbol{\eta}} = \mathbf{q}(\boldsymbol{\xi}, \boldsymbol{\eta}) + \sum_{j=1}^{m} \mathbf{p}_j(\boldsymbol{\xi}, \boldsymbol{\eta}) \cdot u_j = \mathbf{q}(\boldsymbol{\xi}, \boldsymbol{\eta}) + \mathbf{P}(\boldsymbol{\xi}, \boldsymbol{\eta}) \cdot \mathbf{u} \tag{3.171}$$

where

$$b_i(\mathbf{z}) = L_{\mathbf{f}}^{r_i} h_i\left(\mathbf{\Phi}^{-1}(\mathbf{z})\right) \qquad i = 1, \ldots, m \tag{3.172}$$

$$a_{ij}(\mathbf{z}) = L_{\mathbf{g}_j} L_{\mathbf{f}}^{r_i - 1} h_i\left(\mathbf{\Phi}^{-1}(\mathbf{z})\right) \qquad i, j = 1, \ldots, m \tag{3.173}$$

$$q_i(\mathbf{z}) = L_{\mathbf{f}} \Phi_i\left(\mathbf{\Phi}^{-1}(\mathbf{z})\right) \qquad i = r+1, \ldots, n \tag{3.174}$$

$$\mathbf{p}_i^T(\mathbf{z}) = L_{\mathbf{G}} \Phi_i\left(\mathbf{\Phi}^{-1}(\mathbf{z})\right) \qquad i = r+1, \ldots, n \tag{3.175}$$

and

$$\mathbf{P}(\mathbf{z}) = \begin{bmatrix} \mathbf{p}_{r+1}^T \\ \vdots \\ \mathbf{p}_n^T \end{bmatrix}. \tag{3.176}$$

For the outputs of the transformed system, the following relationships hold:

$$y_i = h_i\left(\mathbf{\Phi}^{-1}(\mathbf{z})\right) = \xi_1^i \qquad i = 1, ..., m \tag{3.177}$$

Based on the state-transformation derived so far, a non-linear state feedback can be found that gives the desired linear input-output behavior. Therefore, only equations (3.170) that contain the highest order derivatives of the various outputs are combined together:

$$\left[y_i^{(r_i)}\right] = \mathbf{b}(\mathbf{x}) + \mathbf{A}(\mathbf{x}) \cdot \mathbf{u} = \mathbf{b}(\boldsymbol{\xi}, \boldsymbol{\eta}) + \mathbf{A}(\boldsymbol{\xi}, \boldsymbol{\eta}) \cdot \mathbf{u} \qquad i = 1, ..., m \tag{3.178}$$

where

$$\mathbf{A}(\mathbf{x}) = \begin{bmatrix} L_{\mathbf{g}_1} L_{\mathbf{f}}^{r_1-1} h_1(\mathbf{x}) & L_{\mathbf{g}_2} L_{\mathbf{f}}^{r_1-1} h_1(\mathbf{x}) & \cdots & L_{\mathbf{g}_m} L_{\mathbf{f}}^{r_1-1} h_1(\mathbf{x}) \\ L_{\mathbf{g}_1} L_{\mathbf{f}}^{r_2-1} h_2(\mathbf{x}) & L_{\mathbf{g}_2} L_{\mathbf{f}}^{r_2-1} h_2(\mathbf{x}) & \cdots & L_{\mathbf{g}_m} L_{\mathbf{f}}^{r_2-1} h_2(\mathbf{x}) \\ \vdots & \vdots & \ddots & \vdots \\ L_{\mathbf{g}_1} L_{\mathbf{f}}^{r_m-1} h_m(\mathbf{x}) & L_{\mathbf{g}_2} L_{\mathbf{f}}^{r_m-1} h_m(\mathbf{x}) & \cdots & L_{\mathbf{g}_m} L_{\mathbf{f}}^{r_m-1} h_m(\mathbf{x}) \end{bmatrix} \tag{3.179}$$

$$\mathbf{b}(\mathbf{x}) = \begin{bmatrix} L_{\mathbf{f}}^{r_1} h_1(\mathbf{x}) \\ L_{\mathbf{f}}^{r_2} h_2(\mathbf{x}) \\ \vdots \\ L_{\mathbf{f}}^{r_m} h_m(\mathbf{x}) \end{bmatrix}. \tag{3.180}$$

In addition to the linearization of the input-output behavior of the dynamic system, non-linear dynamic inversion has the goal to decouple the system dynamics between the inputs and the outputs, i.e. each control input shall only influence a single output. Therefore, pseudo-controls $\boldsymbol{\nu} = [\nu_1, \ldots, \nu_m]^T$ are introduced for the transformed system, where the number of pseudo-controls has to equal the number of the system outputs in order to allow for a decoupling of the input-output dynamics. Then, a non-linear state-feedback of the following form can be chosen:

$$\mathbf{u} = \boldsymbol{\alpha}(\mathbf{x}) + \boldsymbol{\beta}(\mathbf{x}) \cdot \boldsymbol{\nu} \tag{3.181}$$

Substituting Eq. (3.181) into the transformed system (3.178), the dynamics of the closed-loop system are:

$$\left[y_i^{(r_i)}\right] = \mathbf{b}(\mathbf{x}) + \mathbf{A}(\mathbf{x}) \cdot \left[\boldsymbol{\alpha}(\mathbf{x}) + \boldsymbol{\beta}(\mathbf{x}) \cdot \boldsymbol{\nu}\right] \tag{3.182}$$

Setting the coefficients of the non-linear state-feedback (3.181) to

$$\boldsymbol{\alpha}(\mathbf{x}) = -\mathbf{A}^{-1}(\mathbf{x}) \cdot \mathbf{b}(\mathbf{x}) \tag{3.183}$$

respectively

$$\boldsymbol{\beta}(\mathbf{x}) = \mathbf{A}^{-1}(\mathbf{x}) \tag{3.184}$$

the system dynamics (3.182) reduce to

$$\left[y_i^{(r_i)}\right] = \boldsymbol{\nu} \tag{3.185}$$

The actual controls $\mathbf{u}$ that have to be fed forward into the original dynamic system can then be restored from the pseudo-controls $\boldsymbol{\nu}$ and the implemented feedback gains $\boldsymbol{\alpha}(\mathbf{x})$ and $\boldsymbol{\beta}(\mathbf{x})$:

$$\mathbf{u} = \mathbf{A}^{-1}(\mathbf{x}) \cdot \left[\boldsymbol{\nu} - \mathbf{b}(\mathbf{x})\right] \tag{3.186}$$

Thus, the original system featuring coupled and non-linear dynamics has been substituted by a transformed system with decoupled and linear dynamics between the pseudo-controls and the system outputs, where the i-th output y_i results directly from the i-th pseudo-control ν_i after r_i integrations as can be seen from (3.185), i.e. that the r_i-th derivative of the i-th output y_i can be set directly by the i-th pseudo-control ν_i. Fig. 16 shows the principle layout of the system that ideally results from an exact input-output linearization respectively dynamic inversion.

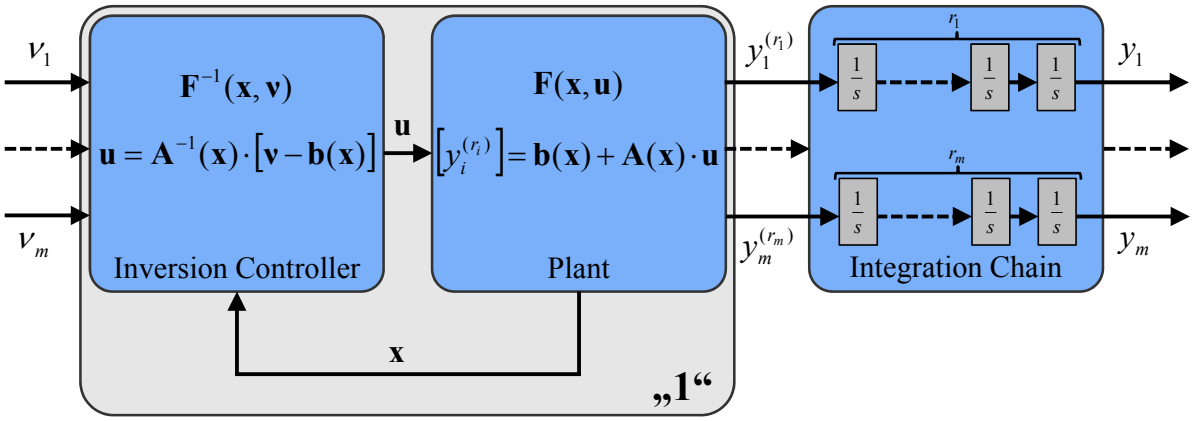

Figure 16. Input-Output Linearization

The relationships between the input quantities u_i and the r_i-th order time derivatives of the output quantities y_i are given by a system of non-linear algebraic equations. These equations are determined by the vector fields $\mathbf{f}$, $\mathbf{g}_1$, …, $\mathbf{g}_m$ which describe the dynamics of the considered system and depend on the actual control inputs and state values of the system. In principle, the exact input-output linearization is identical to solving the system equations $\mathbf{F}(\mathbf{x}, \mathbf{u})$ for the control inputs $\mathbf{u}$ for the example of a system of relative degree 1.

In the following chapters, the principle of input-output linearization respectively non-linear dynamic inversion is applied to the rigid body equations of motion introduced in chapter 3.2. At this, the input-output linearization is not applied to the dynamic flight system on the whole, but it is made use of its sequential structure. Starting with the outermost system, i.e. the position propagation equations, the inputs to these equations that are the kinematic flight-path variables are considered as control inputs. Applying the principle of dynamic inversion, these variables can then be calculated from a given reference trajectory $\vec{\mathbf{x}}_{REF}$ (see chapter 3.3.3). Now, feeding the computed flight-path variables directly into the position propagation equations would again result in the reference trajectory as output. Next, the input-output linearization is applied to the translation equations of motion, where now the kinematic flight-path variables are regarded as the output while the load factors in the kinematic flight-path frame K now represent the inputs and thus the controls (chapter 3.3.4). Given the reference values for the kinematic flight-path variables, the load factors can be computed that would result in the prescribed reference values of the kinematic flight-path variables if they were fed into the translation equations of motion. This procedure (see Fig. 17) is repeated until the innermost subsystem is reached that incorporates the rotation equation of motions.

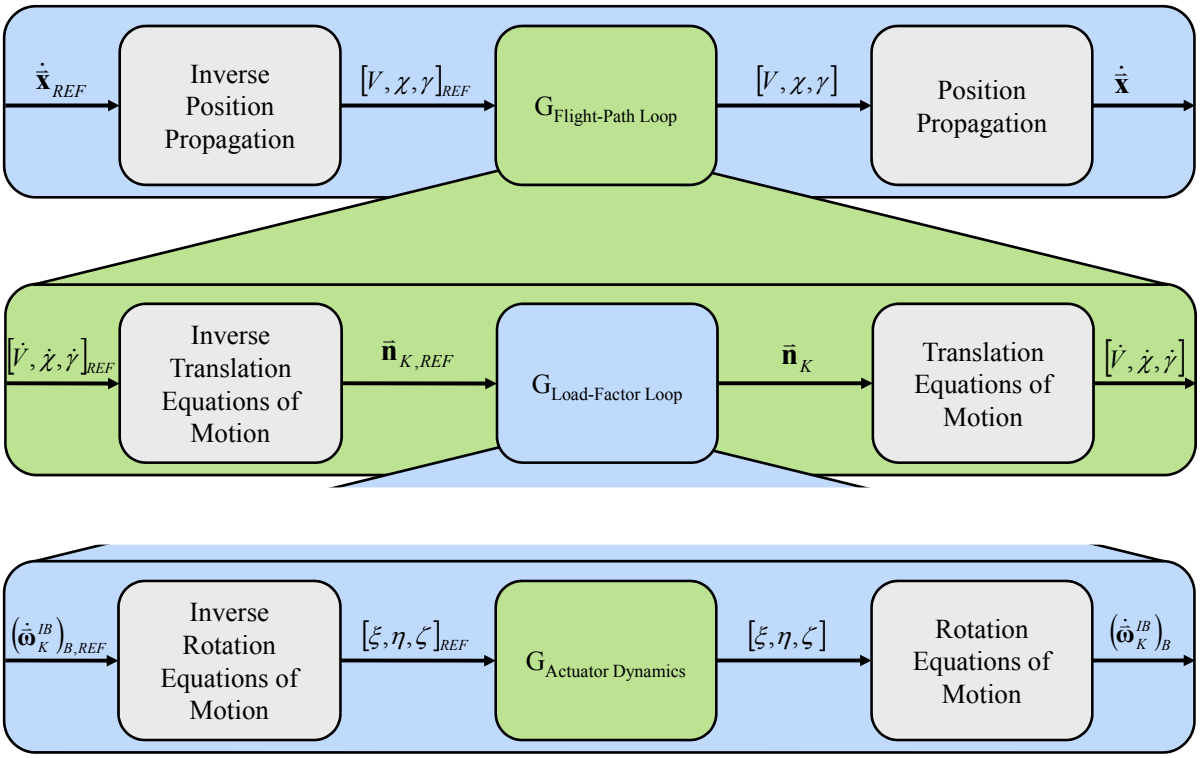

Figure 17. Structure of the Applied Input-Output Linearization

As can be seen from Fig. 17, from the viewpoint of the outer loops the respective inner loops are seen as actuator dynamics (with transfer functions G) since the inner loops impress some dynamics onto the respective controls. Given the case that all input-output linearizations are exact, i.e. $G = 1$ for all inner loops and that no actuator dynamics are present ($G_A = 1$), the complete dynamic inversion is exact and any reference trajectory will result in exactly the same output trajectory. In reality, it is an impossible task to give an exact reproduction of the real dynamic behavior of a system by mathematical equations out of numerous reasons that can be

- parameter uncertainties,
- modeling uncertainties,
- neglected dynamics,
- external perturbations,
- errors or delay when measuring certain quantities,
- or quantities that are not measurable at all.

Thus, for real-world applications of the input-output linearization only approximate systems can be utilized whose dynamic behavior differs from the behavior of the real system. Regarding an input-output linearization for a simulation model that is only run on computers, most of the above mentioned problems are non-existent. Besides the actuator dynamics (that could also be input-output linearized), differences between the reference values and the resulting output quantities appear only due to numerical imprecision or inherent non-minimum phase dynamics of the considered flight system that cannot be treated by the dynamic inversion. As mentioned in Ref. [Slotine, 1991], if the "internal dynamics is stable […], our tracking control design problem has indeed been solved. Otherwise, the […] tracking controller is practically meaningless, because the instability of the internal dynamics would imply undesirable phenomena […]" (Ref. [Slotine, 1991, p. 218]). As this, "the internal dynamics is stable if the plant zeros are in the left-hand plane, i.e., if the plant is "minimum-

phase" (Ref. [Slotine, 1991, p. 222]). If the input-output dynamics of the actuators were also linearized, the reference trajectory would have to be given up to its sixth order time derivative assuming that second-order linear transfer functions were implemented for the actuator dynamics as proposed in chapter 3.2.11.

3.3.2 Reference Trajectory

One basic mode of the whole simulation model is the trajectory following mode, i.e. the geometry of the desired trajectory is given as input to the inverse simulation model and the simulation model ought to follow this trajectory as closely as possible respecting the given limitations of the aircraft dynamics. Therefore, the reference trajectory for the *x*-, *y*- and *z*-position in the Navigation Frame has to be specified:

$$\vec{\mathbf{x}}_{REF} = \begin{pmatrix} x_{REF}(\tau) \\ y_{REF}(\tau) \\ z_{REF}(\tau) \end{pmatrix}_N \tag{3.187}$$

Here the trajectory is parameterized with respect to the parameter τ that does not necessarily have to be identical to the simulation time t. As mentioned above, the reference trajectory has to be at least four times continuously differentiable, thus it could for example be given as a quintic spline.

With the given reference trajectory and the actual position of the aircraft, the reference point on the trajectory can be computed with the reference point being the point on the reference trajectory at which the aircraft ideally should be at a particular time. Since the reference point is specified as the nearest point on the reference trajectory from the current aircraft position (see Fig. 18), the reference point is the perpendicular foot point on the trajectory from the current aircraft position.

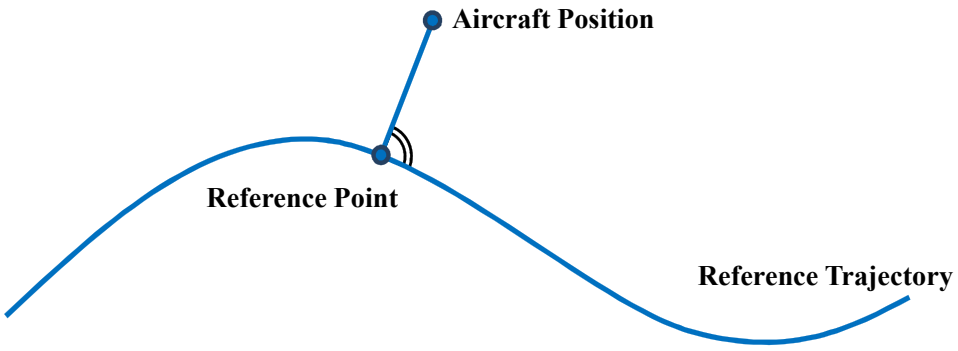

Figure 18. Calculation of the Aircraft's Reference Point

Thus, the vector from the current aircraft position to the reference point is orthogonal to the tangent at the trajectory in the reference point, so that the actual reference point can be calculated by

$$\vec{\mathbf{x}}'_{REF}(\tau_{REF}) \circ \left(\vec{\mathbf{x}}_{REF}(\tau_{REF}) - \vec{\mathbf{x}}\right) = 0 \tag{3.188}$$

In case that this equation cannot be solved analytically for the parameter τ_{REF} of the actual reference point $\vec{\mathbf{x}}_{REF}(\tau_{REF})$, it has to be solved numerically, what can be accomplished e.g. by the Newton algorithm (Refs. [Kiusalaas, 2005] and [Quarteroni, 2000]) or alternatively by the Laguerre algorithm (Ref. [Kiusalaas, 2005]) if the reference trajectory is given in polynomial form. Here, $\vec{\mathbf{x}}'$ denotes the derivative of the position vector with respect to the parameter τ.

Within the inverse simulation model, the reference trajectory is the input to the path reference subsystem that contains the dynamic inversion of the position propagation equations and is introduced below. For this subsystem the derivatives of the current reference position vector of the aircraft up to the fourth order with respect to time are required. Thus, if the parameter τ is not identical to the time variable t, the derivatives with respect to time have to be computed within the reference trajectory subsystem. Therefore, the derivatives of the position vector with respect to the parameter τ up to the fourth order are derived and a scalar pseudo-velocity $\overline{V}$ is introduced. Next, the derivatives of this pseudo-velocity $\overline{V}$ up to the third order with respect to the parameter τ are obtained by

$$\overline{V} = \left\| \vec{\mathbf{x}}'(\tau_{REF}) \right\| \tag{3.189}$$

$$\overline{V}' = \frac{\vec{\mathbf{x}}'(\tau_{REF}) \circ \vec{\mathbf{x}}''(\tau_{REF})}{\overline{V}} \tag{3.190}$$

$$\overline{V}'' = \frac{\vec{\mathbf{x}}''(\tau_{REF}) \circ \vec{\mathbf{x}}''(\tau_{REF}) + \vec{\mathbf{x}}'(\tau_{REF}) \circ \vec{\mathbf{x}}'''(\tau_{REF}) - \overline{V}'^2}{\overline{V}} \tag{3.191}$$

$$\overline{V}''' = \frac{3 \cdot \vec{\mathbf{x}}''(\tau_{REF}) \circ \vec{\mathbf{x}}'''(\tau_{REF}) + \vec{\mathbf{x}}'(\tau_{REF}) \circ \vec{\mathbf{x}}^{(4)}(\tau_{REF}) - 3 \cdot \overline{V}' \cdot \overline{V}''}{\overline{V}} \tag{3.192}$$

In combination with the current kinematic aircraft velocity V_K and its time derivatives, the above computed pseudo velocity $\overline{V}$ and its derivatives with respect to the trajectory parameter τ can now be used to compute the required derivatives of the position vector $\vec{\mathbf{x}}(t)$ with respect to time up to the fourth order:

$$\dot{\vec{\mathbf{x}}} = \frac{d\vec{\mathbf{x}}}{d\tau} \cdot \frac{d\tau}{dt} = \frac{d\vec{\mathbf{x}}}{d\tau} \dot{\tau} \tag{3.193}$$

$$\ddot{\vec{\mathbf{x}}} = \dot{\tau}^2 \frac{d^2\vec{\mathbf{x}}}{d\tau^2} + \ddot{\tau} \frac{d\vec{\mathbf{x}}}{d\tau} \tag{3.194}$$

$$\dddot{\vec{\mathbf{x}}} = \dot{\tau}^3 \frac{d^3\vec{\mathbf{x}}}{d\tau^3} + 3 \cdot \ddot{\tau} \cdot \dot{\tau} \frac{d^2\vec{\mathbf{x}}}{d\tau^2} + \dddot{\tau} \frac{d\vec{\mathbf{x}}}{d\tau} \tag{3.195}$$

$$\vec{\mathbf{x}}^{(4)} = \dot{\tau}^4 \frac{d^4\vec{\mathbf{x}}}{d\tau^4} + 6 \cdot \ddot{\tau} \cdot \dot{\tau}^2 \frac{d^3\vec{\mathbf{x}}}{d\tau^3} + 4 \cdot \dddot{\tau} \cdot \dot{\tau} \frac{d^2\vec{\mathbf{x}}}{d\tau^2} + 3 \cdot \ddot{\tau}^2 \frac{d^2\vec{\mathbf{x}}}{d\tau^2} + \tau^{(4)} \frac{d\vec{\mathbf{x}}}{d\tau} \tag{3.196}$$

$$\frac{d\tau}{dt} = \frac{d\tau}{dx} \cdot \frac{dx}{dt} = \frac{V_K^G}{\overline{V}} = \dot{\tau} \tag{3.197}$$

$$\frac{d^2\tau}{dt^2} = \frac{\overline{V} \cdot \dot{V}_K^G - \dot{\tau} \cdot V_K^G \cdot \overline{V}'}{\overline{V}^2} \tag{3.198}$$

$$\frac{d^3\tau}{dt^3} = \frac{\ddot{V}_K^G \cdot \overline{V}^2 - \overline{V}\left(\dot{\tau}^2 \cdot V_K^G \cdot \overline{V}'' + \overline{V}'\left(2 \cdot \dot{\tau} \cdot \dot{V}_K^G + \ddot{\tau} \cdot V_K^G\right)\right) + 2 \cdot \dot{\tau}^2 \cdot V_K^G \cdot \overline{V}'^2}{\overline{V}^3} \tag{3.199}$$

$$\begin{aligned} \frac{d^4\tau}{dt^4} = \frac{1}{\overline{V}^4}\Big[& -6 \cdot V_K^G \cdot \overline{V}'^3 \cdot \dot{\tau}^3 + 6 \cdot \dot{\tau} \cdot \overline{V} \cdot \overline{V}'\left(\dot{\tau}^2 \cdot V_K^G \cdot \overline{V}'' + \overline{V}'\left(\dot{\tau} \cdot \dot{V}_K^G + V_K^G \cdot \ddot{\tau}\right)\right) + \\ & + \dddot{V}_K^G \cdot \overline{V}^3 - \overline{V}^2\left(3 \cdot \dot{\tau} \cdot \overline{V}''\left(\dot{\tau} \cdot \dot{V}_K^G + V_K^G \cdot \ddot{\tau}\right) + V_K^G \cdot \dot{\tau}^3 \cdot \overline{V}''' + \right. \\ & \left. + \overline{V}'\left(3 \cdot \dot{\tau} \cdot \ddot{V}_K^G + 3 \cdot \dot{V}_K^G \cdot \ddot{\tau} + \dddot{\tau} \cdot V_K^G\right)\right)\Big] \end{aligned} \tag{3.200}$$

3.3.3 Reference Kinematic Flight-Path Values

The path reference subsystem contains the inverted position propagation equations. Here, the reference kinematic flight-path signals namely the kinematic velocity V_K, the kinematic flight-path climb angle γ_K and the kinematic flight-path course angle χ_K are computed up to their third order time derivatives from the reference trajectory. Therefore, the path reference subsystem requires as input the reference trajectory as well as the derivatives of this reference trajectory with respect to time up to the fourth order.

The computation of the required reference values is accomplished as follows:

$$V_{K,REF}^{G} = \left\| \left(\dot{\vec{\mathbf{x}}} \right)_N^N \right\|_2 = \sqrt{\left(\dot{\vec{\mathbf{x}}} \right)_N^N \circ \left(\dot{\vec{\mathbf{x}}} \right)_N^N} = \sqrt{(\dot{x})^2 + (\dot{y})^2 + (\dot{z})^2} \tag{3.201}$$

$$V_{K,HOR,REF}^{G} = \left\| \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} \right\| = \sqrt{(\dot{x})^2 + (\dot{y})^2} \tag{3.202}$$

$$\gamma_{K,REF}^{G} = -\arctan \frac{\dot{z}}{V_{K,HOR}^{G}} \tag{3.203}$$

$$\chi_{K,REF}^{G} = \arctan \frac{\dot{y}}{\dot{x}} \tag{3.204}$$

Within the inverse simulation model, these flight-path signals are in turn utilized for the computation of the reference load factor values and their time derivatives up to the second order. Therefore, not only the flight-path signals themselves but also their derivatives with respect to time up to the third order are necessary. The derivatives of the kinematic flight-path signals can be derived by differentiation of the above stated Eqs. (3.201) to (3.204):

$$\dot{V}_{K,REF}^{G} = \frac{\left(\dot{\vec{\mathbf{x}}} \right)_N^N \circ \left(\ddot{\vec{\mathbf{x}}} \right)_N^{NN}}{V_K^G} \tag{3.205}$$

$$\ddot{V}_{K,REF}^{G} = \frac{\left(\ddot{\vec{\mathbf{x}}} \right)_N^{NN} \circ \left(\ddot{\vec{\mathbf{x}}} \right)_N^{NN} + \left(\dot{\vec{\mathbf{x}}} \right)_N^N \circ \left(\dddot{\vec{\mathbf{x}}} \right)_N^{NNN} - \left(\dot{V}_K^G \right)^2}{V_K^G} \tag{3.206}$$

$$\dddot{V}_{K,REF}^{G} = \frac{3 \cdot \left(\ddot{\vec{\mathbf{x}}} \right)_N^{NN} \circ \left(\dddot{\vec{\mathbf{x}}} \right)_N^{NNN} + \left(\dot{\vec{\mathbf{x}}} \right)_N^N \circ \left(\vec{\mathbf{x}}^{(4)} \right)_N^{NNNN} - 3 \cdot \dot{V}_K^G \cdot \ddot{V}_K^G}{V_K^G} \tag{3.207}$$

$$\dot{V}_{K,HOR,REF}^{G} = \frac{\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} \circ \begin{pmatrix} \ddot{x} \\ \ddot{y} \end{pmatrix}}{V_{K,HOR}^{G}} \tag{3.208}$$

$$\ddot{V}_{K,HOR,REF}^{G} = \frac{\begin{pmatrix} \ddot{x} \\ \ddot{y} \end{pmatrix} \circ \begin{pmatrix} \ddot{x} \\ \ddot{y} \end{pmatrix} + \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} \circ \begin{pmatrix} \dddot{x} \\ \dddot{y} \end{pmatrix} - \left(\dot{V}_{K,HOR}^{G} \right)^2}{V_{K,HOR}^{G}} \tag{3.209}$$

$$\dddot{V}_{K,HOR,REF}^{G} = \frac{3 \cdot \begin{pmatrix} \ddot{x} \\ \ddot{y} \end{pmatrix} \circ \begin{pmatrix} \dddot{x} \\ \dddot{y} \end{pmatrix} + \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} \circ \begin{pmatrix} x^{(4)} \\ y^{(4)} \end{pmatrix} - 3 \cdot \dot{V}_{K,HOR}^{G} \cdot \dot{V}_{K,HOR}^{G}}{V_{K,HOR}^{G}} \tag{3.210}$$

$$\dot{\gamma}^G_{K,REF} = \frac{\dot{z}\cdot\dot{V}^G_{K,HOR} - V^G_{K,HOR}\cdot\ddot{z}}{\left(V^G_K\right)^2} \tag{3.211}$$

$$\ddot{\gamma}^G_{K,REF} = \frac{\dot{z}\cdot\ddot{V}^G_{K,HOR} - V^G_{K,HOR}\cdot\dddot{z}}{\left(V^G_K\right)^2} - 2\cdot\dot{\gamma}^G_K\frac{\dot{V}^G_K}{V^G_K} \tag{3.212}$$

$$\begin{aligned}\dddot{\gamma}^G_{K,REF} = {} & \frac{\ddot{z}\cdot\dddot{V}^G_{K,HOR} - \dot{V}^G_{K,HOR}\cdot\dddot{z} + \dot{z}\cdot\ddot{V}^G_{K,HOR} - V^G_{K,HOR}\cdot z^{(4)}}{\left(V^G_K\right)^2} - \\ & - \frac{2\left(\left(\dot{V}^G_K\right)^2\cdot\dot{\gamma}^G_K + V^G_K\left(\ddot{V}^G_K\cdot\dot{\gamma}^G_K + 2\cdot\dot{V}^G_{K,REF}\cdot\ddot{\gamma}^G_K\right)\right)}{\left(V^G_K\right)^2}\end{aligned} \tag{3.213}$$

$$\dot{\chi}^G_{K,REF} = \frac{\dot{x}\cdot\ddot{y} - \dot{y}\cdot\ddot{x}}{\left(V^G_{K,HOR}\right)^2} \tag{3.214}$$

$$\ddot{\chi}^G_{K,REF} = \frac{\dot{x}\cdot\dddot{y} - \dot{y}\cdot\dddot{x}}{\left(V^G_{K,HOR}\right)^2} - 2\cdot\dot{\chi}^G_K\frac{\dot{V}^G_{K,HOR}}{V^G_{K,HOR}} \tag{3.215}$$

$$\begin{aligned}\dddot{\chi}^G_{K,REF} = {} & \frac{\ddot{x}\cdot\dddot{y} - \ddot{y}\cdot\dddot{x} + \dot{x}\cdot y^{(4)} - \dot{y}\cdot x^{(4)} - 2\cdot\left(\dot{V}^G_{K,HOR}\right)^2\cdot\dot{\chi}^G_K}{\left(V^G_{K,HOR}\right)^2} - \\ & - \frac{2\cdot\left(\ddot{V}^G_{K,HOR}\cdot\dot{\chi}^G_K + 2\cdot\dot{V}^G_{K,HOR}\cdot\ddot{\chi}^G_K\right)}{V^G_{K,HOR}}\end{aligned} \tag{3.216}$$

3.3.4 Reference Load Factors in the Kinematic Flight-Path Frame *K*

The reference load factors in the Kinematic Flight-Path Frame *K* are obtained by the inversion of the translation equations of motion that describe the translational dynamics of the vehicle with respect to the surface of the Earth and are the basis for the control of the path dynamics variables. Physically, to follow a desired trajectory, forces are to be generated that counter the inertia forces associated with the desired curvature of the trajectory as well as the gravitational force. Primary virtual controls which can be utilized are mainly the lift (controlled by the aerodynamic angle of attack α_A and the dynamic pressure $\bar{q}$) to produce forces perpendicular to the flight-path climb angle γ_K and the thrust to generate forces in the velocity direction. Secondary force generators in other directions are the aerodynamic angle of sideslip β_A to produce side forces and thus to quicken changes in the lateral plane of curvature, the flaps to produce direct lift force changes at low bandwidth, speed- or air-brakes to increase the drag and thus the deceleration potential of the aircraft and finally, if available, thrust vector controls.

Specified in the Kinematic Flight-Path Frame *K* (i.e. the path-axis system), the required total forces for maneuvers over a non-flat, rotating Earth are:

$$\begin{aligned}\left(\bar{\mathbf{F}}^G\right) = m\cdot\Big\{ & \left(\dot{\bar{\mathbf{V}}}^G_K\right)^{EO}_K + \left(\bar{\boldsymbol{\omega}}^{EO}_K\right)_K\times\left(\bar{\mathbf{V}}^G_K\right)^E_K + 2\cdot\left(\bar{\boldsymbol{\omega}}^{IE}_K\right)_K\times\left(\bar{\mathbf{V}}^G_K\right)^E_K \\ & + \left(\bar{\boldsymbol{\omega}}^{IE}_K\right)_K\times\left[\left(\bar{\boldsymbol{\omega}}^{IE}_K\right)_K\times\left(\bar{\mathbf{r}}^G\right)_K\right]\Big\}\end{aligned} \tag{3.217}$$

In case that only maneuvers over a flat, non-rotating Earth are considered, the equations for the required forces simplify to:

$$\left(F_x^G\right)_K = m \cdot \dot{V}_K^G + m \cdot g \cdot \sin \gamma_K^G \tag{3.218}$$

$$\left(F_y^G\right)_K = m \cdot V_K^G \cdot \cos \gamma_K^G \cdot \dot{\chi}_K^G \tag{3.219}$$

$$\left(F_z^G\right)_K = -m \cdot V_K^G \cdot \dot{\gamma}_K^G - m \cdot g \cdot \cos \gamma_K^G \tag{3.220}$$

Thus, the required load factors in the *K*-Frame have to be calculated from the reference flight-path signals and their derivatives, which is done by the following equations that are derived from Eqs. (3.218) to (3.220) with the load factor n_z having the opposite direction as the sum of forces in *z*-direction:

$$\left(n_x^G\right)_{K,REF} = \frac{\left(F_x^G\right)_K}{m \cdot g} = \frac{\dot{V}_K^G}{g} + \sin \gamma_K^G \tag{3.221}$$

$$\left(n_y^G\right)_{K,REF} = \frac{\left(F_y^G\right)_K}{m \cdot g} = \frac{V_K^G \cdot \dot{\chi}_K^G \cdot \cos \gamma_K^G}{g} \tag{3.222}$$

$$\left(n_z^G\right)_{K,REF} = \frac{-\left(F_z^G\right)_K}{m \cdot g} = \cos \gamma_K^G + \frac{V_K^G \cdot \dot{\gamma}_K^G}{g} \tag{3.223}$$

As mentioned above, for the reference load factors not only the reference signals themselves but also their first and second order time derivatives are necessary. These derivatives have to be computed by differentiating Eqs. (3.221) to (3.223) with respect to time:

$$\left(\dot{n}_x^G\right)_{K,REF}^K = \frac{\ddot{V}_K^G}{g} + \dot{\gamma}_K^G \cdot \cos \gamma_K^G \tag{3.224}$$

$$\begin{aligned}\left(\dot{n}_y^G\right)_{K,REF}^K = \frac{1}{g}\Big(&\dot{V}_K^G \cdot \dot{\chi}_K^G \cdot \cos \gamma_K^G - V_K^G \cdot \dot{\chi}_K^G \cdot \dot{\gamma}_K^G \cdot \sin \gamma_K^G + \\ &+ V_K^G \cdot \ddot{\chi}_K^G \cdot \cos \gamma_K^G\Big)\end{aligned} \tag{3.225}$$

$$\left(\dot{n}_z^G\right)_{K,REF}^K = -\dot{\gamma}_K^G \cdot \sin \gamma_K^G + \frac{\dot{V}_K^G \cdot \dot{\gamma}_K^G + V_K^G \cdot \ddot{\gamma}_K^G}{g} \tag{3.226}$$

$$\left(\ddot{n}_x^G\right)_{K,REF}^{KK} = \frac{\dddot{V}_K^G}{g} + \ddot{\gamma}_K^G \cdot \cos \gamma_K^G - \left(\dot{\gamma}_K^G\right)^2 \cdot \sin \gamma_K^G \tag{3.227}$$

$$\begin{aligned}\left(\ddot{n}_y^G\right)_{K,REF}^{KK} = \frac{1}{g}\Big[&\cos \gamma_K^G \left(V_K^G \left(\dddot{\chi}_K^G - \left(\dot{\gamma}_K^G\right)^2 \cdot \dot{\chi}_K^G\right) + 2 \cdot \dot{V}_K^G \cdot \ddot{\chi}_K^G + \ddot{V}_K^G \cdot \dot{\chi}_K^G\right) - \\ &- \sin \gamma_K^G \left(V_K^G \left(2 \cdot \dot{\gamma}_K^G \cdot \ddot{\chi}_K^G + \ddot{\gamma}_K^G \cdot \dot{\chi}_K^G\right) + 2 \cdot \dot{V}_K^G \cdot \dot{\gamma}_K^G \cdot \dot{\chi}_K^G\right)\Big]\end{aligned} \tag{3.228}$$

$$\left(\ddot{n}_z^G\right)_{K,REF}^{KK} = -\dddot{\gamma}_K^G \cdot \sin \gamma_K^G - \left(\dot{\gamma}_K^G\right)^2 \cdot \cos \gamma_K^G + \frac{\ddot{V}_K^G \cdot \dot{\gamma}_K^G}{g} + \frac{2 \cdot \dot{V}_K^G \cdot \ddot{\gamma}_K^G + V_K^G \cdot \dddot{\gamma}_K^G}{g} \tag{3.229}$$

From the preceding equations it can be seen that not only the first and second order derivatives of the flight-path climb angle γ_K, the azimuth angle χ_K and the kinematic velocity V_K have to be computed, but also the third order time derivatives of the reference flight-path values.

3.3.5 Reference Load Factors in the Intermediate Flight-Path Frame $\bar{K}$

In the Intermediate Kinematic Flight-Path Frame $\bar{K}$, also the kinematic flight-path bank angle μ_K can be utilized to control the direction of the lift force in addition to the virtual controls that can be used in the Kinematic Flight-Path Frame K. Concerning the generation of the reference load factors in the $\bar{K}$-Frame two alternatives are possible, namely a coordinated roll dynamics mode and an external kinematic roll dynamics mode where the roll dynamics are commanded externally in addition to the reference load factors in the K-Frame.

By the coordinated roll dynamics or coordinated turning mode, the reference load factors and the reference flight-path bank angle in the $\bar{K}$-Frame are calculated from the reference load factors in the K-Frame. At this, the side force corresponding to the y-component of the load factors in the $\bar{K}$-Frame is assumed to be zero which means that the necessary side force for a turn can be produced only by rotating the lift vector in the yz-plane what is done by the flight-path bank angle μ_K. The equations for the calculation of the reference values are

$$\left(n_x^G\right)_{\bar{K},REF} = \left(n_x^G\right)_K \tag{3.230}$$

$$\left(n_y^G\right)_{\bar{K},REF} = 0 \tag{3.231}$$

$$\left(n_z^G\right)_{\bar{K},REF} = \sqrt{\left(\left(n_z^G\right)_K\right)^2 + \left(\left(n_y^G\right)_K\right)^2} \tag{3.232}$$

$$\mu_{K,REF}^G = \arctan\left(\frac{\left(n_y^G\right)_K}{\left(n_z^G\right)_K}\right) \tag{3.233}$$

The first and second order time derivatives of the load factors and roll rate reference values that are also necessary can be calculated by differentiating the above equations, yielding

$$\left(\dot{n}_x^G\right)_{\bar{K},REF}^{\bar{K}} = \left(\dot{n}_x^G\right)_K^K \tag{3.234}$$

$$\left(\dot{n}_y^G\right)_{\bar{K},REF}^{\bar{K}} = 0 \tag{3.235}$$

$$\left(\dot{n}_z^G\right)_{\bar{K},REF}^{\bar{K}} = \frac{\left(n_y^G\right)_K \cdot \left(\dot{n}_y^G\right)_K^K + \left(n_z^G\right)_K \cdot \left(\dot{n}_z^G\right)_K^K}{\left(n_z^G\right)_{\bar{K}}} \tag{3.236}$$

$$\dot{\mu}_{K,REF}^G = \frac{\left(n_z^G\right)_K \cdot \left(\dot{n}_y^G\right)_K^K - \left(n_y^G\right)_K \cdot \left(\dot{n}_z^G\right)_K^K}{\left(\left(n_z^G\right)_{\bar{K}}\right)^2} \tag{3.237}$$

$$\left(\ddot{n}_x^G\right)_{\bar{K},REF}^{\bar{K}\bar{K}} = \left(\ddot{n}_x^G\right)_K^{KK} \tag{3.238}$$

$$\left(\ddot{n}_y^G\right)_{\bar{K},REF}^{\bar{K}\bar{K}} = 0 \tag{3.239}$$

$$\left(\ddot{n}_z^G\right)_{\bar{K},REF}^{\overline{KK}} = \frac{\left(\left(\dot{n}_y^G\right)_K^K\right)^2 + \left(\left(\dot{n}_z^G\right)_K^K\right)^2 + \left(n_y^G\right)_K \cdot \left(\ddot{n}_y^G\right)_K^{KK} + \left(n_z^G\right)_K \cdot \left(\ddot{n}_z^G\right)_K^{KK} - \left(\left(\dot{n}_z^G\right)_{\bar{K}}^{\bar{K}}\right)^2}{\left(n_z^G\right)_{\bar{K}}} \tag{3.240}$$

$$\ddot{\mu}_{K,REF}^G = \frac{\left(n_z^G\right)_K \cdot \left(\ddot{n}_y^G\right)_K^{KK} - \left(n_y^G\right)_K \cdot \left(\ddot{n}_z^G\right)_K^{KK}}{\left(\left(n_z^G\right)_{\bar{K}}\right)^2} - 2 \cdot \dot{\mu}_K^G \cdot \frac{\left(\dot{n}_z^G\right)_{\bar{K}}^{\bar{K}}}{\left(n_z^G\right)_{\bar{K}}} \tag{3.241}$$

With the alternative mode that is the external roll dynamics mode the kinematic roll dynamics are commanded externally in addition to the load factors. This is necessary for certain flight maneuvers e.g. for flying through a knife edge gate with a kinematic flight-path bank angle μ_K equal to 90°. In such a case, the side force is used to compensate the external roll dynamics. So the flight-path bank angle command and its derivatives do not result from the load factors in the *K*-Frame as in the previous mode, but they are commanded directly and the load factors in the $\bar{K}$-Frame are calculated by the following equations, using the transformation matrix between the $\bar{K}$-Frame and *K*-Frame and its derivatives:

$$\left(\vec{\mathbf{n}}^G\right)_{\bar{K},REF} = \mathbf{M}_{\bar{K}K}\left(\mu_K^G\right) \cdot \left(\vec{\mathbf{n}}^G\right)_K \tag{3.242}$$

$$\left(\dot{\vec{\mathbf{n}}}^G\right)_{\bar{K},REF}^{\bar{K}} = \mathbf{M}_{\bar{K}K}\left(\mu_K^G\right) \cdot \left(\dot{\vec{\mathbf{n}}}^G\right)_K^K + \dot{\mathbf{M}}_{\bar{K}K}\left(\mu_K^G, \dot{\mu}_K^G\right) \cdot \left(\vec{\mathbf{n}}^G\right)_K \tag{3.243}$$

$$\begin{aligned}\left(\ddot{\vec{\mathbf{n}}}^G\right)_{\bar{K},REF}^{\overline{KK}} = {} & \mathbf{M}_{\bar{K}K}\left(\mu_K^G\right) \cdot \left(\ddot{\vec{\mathbf{n}}}^G\right)_K^{KK} + 2 \cdot \dot{\mathbf{M}}_{\bar{K}K}\left(\mu_K^G, \dot{\mu}_K^G\right) \cdot \left(\dot{\vec{\mathbf{n}}}^G\right)_K^K \\ & + \ddot{\mathbf{M}}_{\bar{K}K}\left(\mu_K^G, \dot{\mu}_K^G, \ddot{\mu}_K^G\right) \cdot \left(\vec{\mathbf{n}}^G\right)_K\end{aligned} \tag{3.244}$$

where the transformation matrix $\mathbf{M}_{\bar{K}K}$ and its derivatives are:

$$\mathbf{M}_{\bar{K}K} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\mu_K^G & \sin\mu_K^G \\ 0 & -\sin\mu_K^G & \cos\mu_K^G \end{bmatrix} \tag{3.245}$$

$$\dot{\mathbf{M}}_{\bar{K}K} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -\sin\mu_K^G & \cos\mu_K^G \\ 0 & -\cos\mu_K^G & -\sin\mu_K^G \end{bmatrix} \cdot \dot{\mu}_K^G \tag{3.246}$$

$$\ddot{\mathbf{M}}_{\bar{K}K} = -\mathbf{M}_{\bar{K}K} \cdot \left(\dot{\mu}_K^G\right)^2 + \begin{bmatrix} 0 & 0 & 0 \\ 0 & -\sin\mu_K^G & \cos\mu_K^G \\ 0 & -\cos\mu_K^G & -\sin\mu_K^G \end{bmatrix} \cdot \ddot{\mu}_K^G \tag{3.247}$$

For both the coordinated roll dynamics and the external roll dynamics not only the reference load factor values in the *K*-Frame themselves but also their first and second order time derivatives have to be given as inputs.

3.3.6 Reference Load Factors in the Aerodynamic Frame *A*

Within this subsystem of the inverse simulation model, the reference load factors in the Aerodynamic Frame *A* are generated. Therefore, the reference load factors and their

derivatives given in the $\bar{K}$-Frame have to be transformed into reference load factor values in the Aerodynamic Frame A in order to account for the influence of the wind. This is accomplished by

$$\left(\vec{\mathbf{n}}^{G}\right)_{A,REF} = \mathbf{M}_{A\bar{K}}\left(\vec{\mathbf{n}}^{G}\right)_{\bar{K}} \tag{3.248}$$

$$\left(\dot{\vec{\mathbf{n}}}^{G}\right)^{A}_{A,REF} = \mathbf{M}_{A\bar{K}}\left(\dot{\vec{\mathbf{n}}}^{G}\right)^{\bar{K}}_{\bar{K}} + \left(\vec{\boldsymbol{\omega}}^{A\bar{K}}_{K}\right)_{A} \times \left(\vec{\mathbf{n}}^{G}\right)_{A} \tag{3.249}$$

$$\begin{aligned}\left(\ddot{\vec{\mathbf{n}}}^{G}\right)^{AA}_{A,REF} = \mathbf{M}_{A\bar{K}}\left(\ddot{\vec{\mathbf{n}}}^{G}\right)^{\bar{K}\bar{K}}_{\bar{K}} &+ \left(\vec{\boldsymbol{\omega}}^{A\bar{K}}_{K}\right)_{A} \times \left(\dot{\vec{\mathbf{n}}}^{G}\right)^{A}_{A} + \left(\vec{\boldsymbol{\omega}}^{A\bar{K}}_{K}\right)_{A} \times \left(\mathbf{M}_{A\bar{K}}\left(\dot{\vec{\mathbf{n}}}^{G}\right)^{\bar{K}}_{\bar{K}}\right) \\ &- \left(\dot{\vec{\boldsymbol{\omega}}}^{\bar{K}A}_{K}\right)^{\bar{K}}_{A} \times \left(\vec{\mathbf{n}}^{G}\right)_{A}\end{aligned} \tag{3.250}$$

In case that there is no wind, the Intermediate Kinematic Flight-Path Frame $\bar{K}$ is identical to the Aerodynamic Reference Frame A and the above computations can be omitted. As one might notice from the above equations, the transformation matrix $\mathbf{M}_{A\bar{K}}$ between the Intermediate Kinematic Flight-Path Reference Frame $\bar{K}$ and the Aerodynamic Frame A as well as the rotational rate $\vec{\boldsymbol{\omega}}^{A\bar{K}}$ and its first order time derivative between those two reference frames require the kinematic flight-path bank angle μ_K up to its second order time derivative. The kinematic flight-path bank angle and its derivatives can only be calculated from the aerodynamic flight-path bank angle if the aerodynamic and kinematic attitude angles and their derivatives are already known. Since this is obviously not the case at this point of the inverse simulation model, an alternative inverse simulation model subsystem has been implemented that circumvents this drawback, that is the external *aerodynamic* roll dynamics mode for the calculation of the reference load factors in the Aerodynamic Frame A. This mode is similar to the external *kinematic* roll dynamics mode but instead of the kinematic flight-path bank angle μ_K, here the aerodynamic bank angle μ_A can be commanded externally in addition to the load factors in the Kinematic Flight-Path Reference Frame K. The corresponding equations read as follows:

$$\left(\vec{\mathbf{n}}^{G}\right)_{A,REF} = \mathbf{M}_{AK}\left(\vec{\mathbf{n}}^{G}\right)_{K} \tag{3.251}$$

$$\left(\dot{\vec{\mathbf{n}}}^{G}\right)^{A}_{A,REF} = \mathbf{M}_{AK}\left(\dot{\vec{\mathbf{n}}}^{G}\right)^{K}_{K} + \left(\vec{\boldsymbol{\omega}}^{AK}_{K}\right)_{A} \times \left(\vec{\mathbf{n}}^{G}\right)_{A} \tag{3.252}$$

$$\begin{aligned}\left(\ddot{\vec{\mathbf{n}}}^{G}\right)^{AA}_{A,REF} = \mathbf{M}_{AK}\left(\ddot{\vec{\mathbf{n}}}^{G}\right)^{KK}_{K} &+ \left(\vec{\boldsymbol{\omega}}^{AK}_{K}\right)_{A} \times \left(\dot{\vec{\mathbf{n}}}^{G}\right)^{A}_{A} + \left(\vec{\boldsymbol{\omega}}^{AK}_{K}\right)_{A} \times \left(\mathbf{M}_{AK}\left(\dot{\vec{\mathbf{n}}}^{G}\right)^{K}_{K}\right) \\ &+ \left(\dot{\vec{\boldsymbol{\omega}}}^{AK}_{K}\right)^{K}_{A} \times \left(\vec{\mathbf{n}}^{G}\right)_{A}\end{aligned} \tag{3.253}$$

Thus, the reference values for the load factors in the Intermediate Kinematic Flight-Path Reference Frame $\bar{K}$ do not have to be computed at all and the kinematic flight-path bank angle μ_K is not required as input so far. Since the aerodynamic flight-path bank angle μ_A ought to be less descriptive than the kinematic flight-path bank angle μ_K, the external aerodynamics roll dynamics mode might also be less descriptive than the external kinematic roll dynamics mode. This imposes a drawback only in case that the flight-path bank angle is commanded externally by a human but not in case that a computer prescribes the time history for the flight-path bank angle as it is the case for the optimization tasks. Moreover, as described later on the external aerodynamic roll dynamics mode gives way to the development of an optimization procedure that is more integrated and continuous than it would be the case with the external kinematic roll dynamics mode.

3.3.7 Reference Aerodynamic Attitude Angles

After deriving the necessary reference load factors in the Aerodynamic Frame A, the corresponding aerodynamic reference attitude angles that are the aerodynamic angle of attack α_A and the aerodynamic angle of sideslip β_A that are primarily utilized to control the aerodynamic forces respectively the aerodynamic load factors can be computed.

Therefore, at first the aerodynamic coefficients have to be calculated from the aerodynamic load factors and their derivatives given in the Aerodynamic Reference Frame A. For the computation of the derivatives of the force coefficients C_L, C_Y and C_D, only the load factors and the velocity are regarded as time-dependent. This leads to the following formulae for the lift coefficient C_L and its derivatives:

$$C_{L,REF} = \frac{2mg}{\rho S} \cdot \left(\frac{\left(n_{A,z}^{G}\right)_A}{\left(V_A^G\right)^2} \right) \tag{3.254}$$

$$\dot{C}_{L,REF} = \frac{2mg}{\rho S} \cdot \left(\frac{\left(\dot{n}_{A,z}^{G}\right)_A^A}{\left(V_A^G\right)^2} - \frac{2 \cdot \dot{V}_A^G \cdot \left(n_{A,z}^{G}\right)_A}{\left(V_A^G\right)^3} \right) \tag{3.255}$$

$$\begin{aligned} \ddot{C}_{L,REF} = \frac{2mg}{\rho S} \cdot \left(\frac{\left(\ddot{n}_{A,z}^{G}\right)_A^{AA}}{\left(V_A^G\right)^2} - \frac{4 \cdot \dot{V}_A^G \cdot \left(\dot{n}_{A,z}^{G}\right)_A^A}{\left(V_A^G\right)^3} - \frac{2 \cdot \ddot{V}_A^G \cdot \left(n_{A,z}^{G}\right)_A}{\left(V_A^G\right)^3} + \right. \\ \left. + \frac{6 \cdot \left(\dot{V}_A^G\right)^2 \cdot \left(n_{A,z}^{G}\right)_A}{\left(V_A^G\right)^4} \right) \end{aligned} \tag{3.256}$$

The equations for the aerodynamic force coefficients C_Y and C_D are analogous.

Given the assumption that the lift force derivative C_L is only a function of the aerodynamic angle of attack α_A, the reference values for the aerodynamic angle of attack α_A and its derivatives are obtained by:

$$\alpha_{A,REF}^{G} = \alpha_A^G\left(C_L\right) \tag{3.257}$$

$$\dot{\alpha}_{A,REF}^{G} = \frac{\partial \alpha_A^G\left(C_L\right)}{\partial t} = \left.\frac{\partial \alpha_A^G\left(C_L\right)}{\partial C_L}\right|_{C_L} \cdot \dot{C}_L \tag{3.258}$$

$$\ddot{\alpha}_{A,REF}^{G} = \left.\frac{\partial^2 \alpha_A^G\left(C_L\right)}{\partial C_L^2}\right|_{C_L} \cdot \left(\dot{C}_L\right)^2 + \left.\frac{\partial \alpha_A^G\left(C_L\right)}{\partial C_L}\right|_{C_L} \cdot \ddot{C}_L \tag{3.259}$$

Furthermore, assuming that the side force derivative C_Y is only a function of the sideslip angle β_A, the corresponding equations for the respective reference values of the aerodynamic sideslip angle β_A are:

$$\beta_{A,REF}^{G} = \beta_A^G\left(C_Y\right) \tag{3.260}$$

$$\dot{\beta}_{A,REF}^{G} = \frac{\partial \beta_A^G\left(C_Y\right)}{\partial t} = \left.\frac{\partial \beta_A^G\left(C_Y\right)}{\partial C_Y}\right|_{C_Y} \cdot \dot{C}_Y \tag{3.261}$$

$$\ddot{\beta}_{A,REF}^{G} = \left.\frac{\partial^2 \beta_A^G(C_Y)}{\partial C_Y^2}\right|_{C_Y} \cdot \left(\dot{C}_Y\right)^2 + \left.\frac{\partial \beta_A^G(C_Y)}{\partial C_Y}\right|_{C_Y} \cdot \ddot{C}_Y \tag{3.262}$$

As mentioned above, non-minimum phase effects like e.g. the generation of a downward lift force due to an upward deflection of the elevator cannot be taken into account for the calculation of the aerodynamic attitude angles in the inverse simulation model. Finally, the required thrust force T can be computed utilizing the x-component of the aerodynamic load factor vector $\vec{\mathbf{n}}_A$:

$$T = \left(n_{A,x}^{G}\right)_A \cdot mg + D \tag{3.263}$$

Here it is assumed that the thrust force T is aligned with the x-axis of the Aerodynamic Reference Frame A. For the calculation of the drag force D, a quadratic drag polar as given by Eq. (6.18) can be utilized:

$$D = 0.5 \cdot \rho \cdot \left(V_A^G\right)^2 \cdot S \cdot \left(C_{D0} + k_L \cdot \left(C_L - C_{L,C_{D0}}\right)^2 + k_Y \cdot \left(C_Y\right)^2\right) \tag{3.264}$$

where the reference angle of attack $\alpha_{A,REF}$ is given by Eq. (3.257) and where the reference sideslip angle $\beta_{A,REF}$ is given by Eq. (3.260). Finally, the reference thrust lever position $\delta_{T,REF}$ can be derived from the thrust force T depending on the propulsion model. For example, if for the thrust force computation Eq. (6.24) is implemented, the reference thrust lever position $\delta_{T,REF}$ evaluates to:

$$\delta_{T,REF} = \frac{T}{T_{ref}} \cdot \left(\frac{\rho_{ref}}{\rho}\right)^{n_\rho} \cdot \left(\frac{V_{ref}}{V_A^G}\right)^{n_V} \tag{3.265}$$

where T_{ref} is the engine's reference thrust, V_{ref} the reference velocity, ρ_{ref} the reference air density and n_ρ the density exponent. The exponent n_V gives the dependency of the thrust w.r.t. the aerodynamic velocity, for propeller-driven aircraft this exponent equals -1. The higher order derivatives for the reference thrust lever position $\delta_{T,REF}$ can be obtained by differentiating the above equations.

Without the assumptions made above, the computation of the reference values for the aerodynamic attitude angles α_A and β_A and the thrust lever position δ_T would not be that straight forward: before the aerodynamic coefficients can be calculated, the load factors $\vec{\mathbf{n}}_P$ induced by the propulsion force have to be subtracted from the total reference load factor values to obtain the load factors $\vec{\mathbf{n}}_A$ that have to be produced solely by the aerodynamic forces. Therefore, the aerodynamic load factors $\vec{\mathbf{n}}_A$ and their derivatives can be computed as:

$$\left(\vec{\mathbf{n}}_A^G\right)_{A,REF} = \left(\vec{\mathbf{n}}^G\right)_A - \frac{1}{mg} \cdot \mathbf{M}_{AB}\left(\vec{\mathbf{F}}_P^G\right)_B = \left(\vec{\mathbf{n}}^G\right)_A - \mathbf{M}_{AB}\left(\vec{\mathbf{n}}_P^G\right)_B \tag{3.266}$$

$$\left(\dot{\vec{\mathbf{n}}}_A^G\right)_{A,REF}^{A} = \left(\dot{\vec{\mathbf{n}}}^G\right)_A^A - \mathbf{M}_{AB}\left(\dot{\vec{\mathbf{n}}}_P^G\right)_B^B - \left(\vec{\boldsymbol{\omega}}_K^{AB}\right)_A \times \left(\mathbf{M}_{AB}\left(\vec{\mathbf{n}}_P^G\right)_B\right) \tag{3.267}$$

$$\begin{aligned}\left(\ddot{\vec{\mathbf{n}}}_A^G\right)_{A,REF}^{AA} = {} & \left(\ddot{\vec{\mathbf{n}}}^G\right)_A^{AA} - \mathbf{M}_{AB}\left(\ddot{\vec{\mathbf{n}}}_P^G\right)_B^{BB} - \left(\vec{\boldsymbol{\omega}}_B^{AB}\right)_A \times \left[\left(\vec{\boldsymbol{\omega}}_K^{AB}\right)_A \times \left(\mathbf{M}_{AB}\left(\vec{\mathbf{n}}_P^G\right)_B\right)\right] \\ & - 2 \cdot \left(\vec{\boldsymbol{\omega}}_B^{AB}\right)_A \times \left(\mathbf{M}_{AB}\left(\dot{\vec{\mathbf{n}}}_P^G\right)_B^B\right) - \left(\dot{\vec{\boldsymbol{\omega}}}_B^{AB}\right)_A^B \times \left(\mathbf{M}_{AB}\left(\vec{\mathbf{n}}_P^G\right)_B\right)\end{aligned} \tag{3.268}$$

A major drawback of the above stated equations is the fact that the aerodynamic attitude angles α_A and β_A for the calculation of the transformation matrix $\mathbf{M}_{AB}$ between the Aerodynamic Reference Frame A and the Body-Fixed Frame B as well as the thrust lever

position δ_T for the calculation of the propulsion force are required. But these reference values are in turn calculated from the aerodynamic load factors given by the above equations. There are various possibilities to cope with this problem: for simulation tasks, one could take the reference values for the aerodynamic attitude angles and the thrust lever position from the preceding time step to approximately compute the required reference values for the aerodynamic load factors in the actual time step. Alternatively, the corresponding values for the aerodynamic attitude angles and the thrust lever position from the simulation model plant itself could be used and fed back to calculate the aerodynamic load factor reference values. Finally, the aerodynamic attitude angles and the thrust lever position could be calculated iteratively. Therefore, e.g. a Newton Algorithm can be used to solve the following equation:

$$\frac{m \cdot g}{\bar{q} \cdot S} \cdot \left(\bar{\mathbf{n}}^{G}\right)_{A,REF} - \begin{pmatrix} C_D\left(\alpha_A, \beta_A, V_A\right) \\ C_Y\left(\alpha_A, \beta_A, V_A\right) \\ C_L\left(\alpha_A, \beta_A, V_A\right) \end{pmatrix}_A - \frac{1}{\bar{q} \cdot S} \cdot \mathbf{M}_{AB}\left(\alpha_A, \beta_A\right) \cdot \left(\bar{\mathbf{F}}_P^G\left(\delta_T\right)\right)_B = \mathbf{0} \tag{3.269}$$

where C_D, C_Y and C_L are the aerodynamic force coefficients given in the Aerodynamic Reference Frame A. Eq. (3.269) can then be solved iteratively to give the aerodynamic attitude angle angles α_A and β_A together with the thrust lever position δ_T. At this, the force coefficients can also be taken from appropriate look-up tables.

Accordingly, the higher order time derivatives of the aerodynamic angle of attack α_A, the aerodynamic sideslip angle β_A and the thrust lever position δ_T can be computed by deriving Eq. (3.269) with respect to time and solving the resulting equations iteratively for the higher order derivatives.

3.3.8 Reference Kinematic Attitude Angles

With the reference values for the aerodynamic angle of attack α_A and the aerodynamic angle of sideslip β_A specified, one can now calculate the according reference values for the kinematic angle of attack α_K and the kinematic angle of sideslip β_K including their derivatives by the below equations where foremost the kinematic velocity $\vec{\mathbf{V}}_K$ with its components given in the Body-Fixed Frame B has to be established:

$$\left(\bar{\mathbf{V}}_A^G\right)_{B,REF}^E = \begin{pmatrix} V_A^G \cdot \cos\alpha_A^G \cdot \cos\beta_A^G \\ V_A^G \cdot \sin\beta_A^G \\ V_A^G \cdot \sin\alpha_A^G \cdot \cos\beta_A^G \end{pmatrix}_B^E \tag{3.270}$$

$$\left(\bar{\mathbf{V}}_K^G\right)_{B,REF}^E = \left(\bar{\mathbf{V}}_A^G\right)_{B,REF}^E + \mathbf{M}_{BO}\left(\bar{\mathbf{V}}_W^G\right)_O^E \tag{3.271}$$

Furthermore, the first and second order derivative with respect to the Body-Fixed Frame B of the kinematic velocity $\vec{\mathbf{V}}_K$ denoted in the Body-Fixed Frame B are necessary for the calculation of the kinematic attitude angles:

$$\left(\dot{\bar{\mathbf{V}}}_K^G\right)_{B,REF}^{EB} = \left(\dot{\bar{\mathbf{V}}}_A^G\right)_{B,REF}^{EB} + \mathbf{M}_{BN}\left(\dot{\bar{\mathbf{V}}}_W^G\right)_N^{EN} + \left(\bar{\boldsymbol{\omega}}_K^{BN}\right)_B \times \mathbf{M}_{BN}\left(\bar{\mathbf{V}}_W^G\right)_N^E \tag{3.272}$$

$$\begin{aligned} \left(\ddot{\bar{\mathbf{V}}}_K^G\right)_{B,REF}^{EBB} = {} & \left(\ddot{\bar{\mathbf{V}}}_A^G\right)_{B,REF}^{EBB} + \mathbf{M}_{BN}\left(\ddot{\bar{\mathbf{V}}}_W^G\right)_N^{ENN} + \left(\dot{\bar{\boldsymbol{\omega}}}_K^{BN}\right)_B^B \times \mathbf{M}_{BN}\left(\bar{\mathbf{V}}_W^G\right)_N^E \\ & + 2 \cdot \left(\bar{\boldsymbol{\omega}}_K^{BN}\right)_B \times \mathbf{M}_{BN}\left(\dot{\bar{\mathbf{V}}}_W^G\right)_N^{EN} + \left(\bar{\boldsymbol{\omega}}_K^{BN}\right)_B \times \left[\left(\bar{\boldsymbol{\omega}}_K^{BN}\right)_B \times \mathbf{M}_{BN}\left(\bar{\mathbf{V}}_W^G\right)_N^E\right] \end{aligned} \tag{3.273}$$

The kinematic velocity $\vec{\mathbf{V}}_K$ denoted in the Body-Fixed Frame B is related to the kinematic attitude angles as given by Eq. (3.274):

$$\left(\vec{\mathbf{V}}_K^G\right)_{B,REF}^E = \begin{pmatrix} V_K^G \cdot \cos\alpha_K^G \cdot \cos\beta_K^G \\ V_K^G \cdot \sin\beta_K^G \\ V_K^G \cdot \sin\alpha_K^G \cdot \cos\beta_K^G \end{pmatrix}_B^E \tag{3.274}$$

Solving Eq. (3.274) for the kinematic angle of attack α_K and the kinematic sideslip angle β_K gives:

$$\alpha_{K,REF}^G = \arctan\left(\frac{\left(w_K^G\right)_B^E}{\left(u_K^G\right)_B^E}\right) \tag{3.275}$$

$$\beta_{K,REF}^G = \arctan\left(\frac{\left(v_K^G\right)_B^E}{\sqrt{\left[\left(u_K^G\right)_B^E\right]^2 + \left[\left(w_K^G\right)_B^E\right]^2}}\right) \tag{3.276}$$

The first and the second order time derivatives of the kinematic angle of attack α_K and the angle of sideslip β_K can be restored by differentiating the above equations and are as follows:

$$\dot{\alpha}_{K,REF}^G = \frac{\left(u_K^G\right)_B^E \cdot \left(\dot{w}_K^G\right)_B^{EB} - \left(w_K^G\right)_B^E \cdot \left(\dot{u}_K^G\right)_B^{EB}}{\left(V_{K,VER}^G\right)^2} \tag{3.277}$$

$$\dot{\beta}_{K,REF}^G = \frac{V_{K,VER}^G \cdot \left(\dot{v}_K^G\right)_B^{EB} - \left(v_K^G\right)_B^E \cdot \dot{V}_{K,VER}^G}{\left(V_K^G\right)^2} \tag{3.278}$$

$$\ddot{\alpha}_{K,REF}^G = \frac{\left(u_K^G\right)_B^E \cdot \left(\ddot{w}_K^G\right)_B^{EBB} - \left(w_K^G\right)_B^E \cdot \left(\ddot{u}_K^G\right)_B^{EBB}}{\left(V_{K,VER}^G\right)^2} - \frac{2 \cdot \dot{\alpha}_{K,REF}^G \cdot \dot{V}_{K,VER}^G}{V_{K,VER}^G} \tag{3.279}$$

$$\ddot{\beta}_{K,REF}^G = \frac{V_{K,VER}^G \cdot \left(\ddot{v}_K^G\right)_B^{EBB} - \left(v_K^G\right)_B^E \cdot \ddot{V}_{K,VER}^G}{\left(V_K^G\right)^2} - \frac{2 \cdot \dot{\beta}_{K,REF}^G \cdot \dot{V}_K^G}{V_K^G} \tag{3.280}$$

In Eqs. (3.277) to (3.280), V_K denotes the total kinematic velocity and $V_{K,VER}$ the total kinematic velocity of the aircraft in the vertical plane calculated from the kinematic velocity vector $\vec{\mathbf{V}}_K$ with its components given in the Body-Fixed Reference Frame B:

$$V_{K,VER}^G = \sqrt{\left[\left(u_K^G\right)_B^E\right]^2 + \left[\left(w_K^G\right)_B^E\right]^2} \tag{3.281}$$

$$V_K^G = \sqrt{\left[\left(u_K^G\right)_B^E\right]^2 + \left[\left(v_K^G\right)_B^E\right]^2 + \left[\left(w_K^G\right)_B^E\right]^2} \tag{3.282}$$

3.3.9 Reference Angular Rates

Given the reference values for the kinematic attitude angles that are the kinematic angle of attack α_K and the kinematic sideslip angle β_K and their derivatives, the necessary reference signals for the angular rate $\vec{\boldsymbol{\omega}}^{KB}$ can then be derived by the inversion of the attitude propagation equations (3.43), leading to the following expression:

$$\left(\vec{\boldsymbol{\omega}}_K^{KB}\right)_{K,REF} = \begin{pmatrix} \dot{\mu}_K^G + \dot{\alpha}_K^G \cdot \sin\beta_K^G \\ \dot{\alpha}_K^G \cdot \cos\beta_K^G \cdot \cos\mu_K^G + \dot{\beta}_K^G \cdot \sin\mu_K^G \\ \dot{\alpha}_K^G \cdot \cos\beta_K^G \cdot \sin\mu_K^G - \dot{\beta}_K^G \cdot \cos\mu_K^G \end{pmatrix}_K \tag{3.283}$$

The first order derivative of the angular rate $\dot{\vec{\boldsymbol{\omega}}}^{KB}$ can be computed by differentiating the above equation, whereat the second order time derivatives of the kinematic angle of attack $\ddot{\alpha}_K$ and of the kinematic sideslip angle $\ddot{\beta}_K$ are necessary and therefore have to be computed by the subsystem for the generation of the kinematic reference attitude values described above in chapter 3.3.8:

$$\left(\dot{\vec{\boldsymbol{\omega}}}_K^{KB}\right)_{K,REF}^K = \begin{pmatrix} \ddot{\mu}_K^G + \ddot{\alpha}_K^G \cdot \sin\beta_K^G + \dot{\alpha}_K^G \cdot \cos\beta_K^G \cdot \dot{\beta}_K^G \\ \\ \ddot{\alpha}_K^G \cdot \cos\beta_K^G \cdot \cos\mu_K^G - \dot{\alpha}_K^G \cdot \sin\beta_K^G \cdot \dot{\beta}_K^G \cdot \cos\mu_K^G \\ -\dot{\alpha}_K^G \cdot \cos\beta_K^G \cdot \sin\mu_K^G \cdot \dot{\mu}_K^G + \ddot{\beta}_K^G \cdot \sin\mu_K^G + \dot{\beta}_K^G \cdot \cos\mu_K^G \cdot \dot{\mu}_K^G \\ \\ \ddot{\alpha}_K^G \cdot \cos\beta_K^G \cdot \sin\mu_K^G - \dot{\alpha}_K^G \cdot \sin\beta_K^G \cdot \dot{\beta}_K^G \cdot \sin\mu_K^G \\ +\dot{\alpha}_K^G \cdot \cos\beta_K^G \cdot \cos\mu_K^G \cdot \dot{\mu}_K^G - \ddot{\beta}_K^G \cdot \cos\mu_K^G - \dot{\beta}_K^G \cdot \sin\mu_K^G \cdot \dot{\mu}_K^G \end{pmatrix}_K \tag{3.284}$$

By use of the computed reference values for the angular rate $\vec{\boldsymbol{\omega}}^{KB}$ and its first order time derivative, the angular rate $\vec{\boldsymbol{\omega}}^{IB}$ and its first order derivative can then be restored as follows:

$$\left(\vec{\boldsymbol{\omega}}_K^{IB}\right)_{K,REF} = \left(\vec{\boldsymbol{\omega}}_K^{IO}\right)_K + \left(\vec{\boldsymbol{\omega}}_K^{OK}\right)_K + \left(\vec{\boldsymbol{\omega}}_K^{KB}\right)_K \tag{3.285}$$

$$\left(\dot{\vec{\boldsymbol{\omega}}}_K^{IB}\right)_{K,REF}^K = \left(\dot{\vec{\boldsymbol{\omega}}}_K^{EO}\right)_K^K + \left(\dot{\vec{\boldsymbol{\omega}}}_K^{OK}\right)_K^K + \left(\dot{\vec{\boldsymbol{\omega}}}_K^{KB}\right)_K^K \tag{3.286}$$

Since for the calculation of the reference values for the total sum of the moments the first order time derivative of the angular rate $\dot{\vec{\boldsymbol{\omega}}}^{IB}$ with respect to the Body-Fixed Frame B and its components denoted in the Body-Fixed Frame B is essential, this derivative of the angular rate is calculated by

$$\left(\dot{\vec{\boldsymbol{\omega}}}_K^{IB}\right)_{B,REF}^B = \mathbf{M}_{BK}\left(\dot{\vec{\boldsymbol{\omega}}}_K^{IB}\right)_K^B = \mathbf{M}_{BK}\left(\dot{\vec{\boldsymbol{\omega}}}_K^{IB}\right)_K^K + \mathbf{M}_{BK}\left[\left(\vec{\boldsymbol{\omega}}_K^{BK}\right)_K \times \left(\vec{\boldsymbol{\omega}}_K^{IB}\right)_K\right] \tag{3.287}$$

3.3.10 Reference Control Surface Deflections

With the reference values for the angular rate $\vec{\boldsymbol{\omega}}^{IB}$ and its first order time derivative given by Eqs. (3.285) and (3.287), the required moments that are necessary to produce the desired angular rates and that have to be commanded by the dynamic inversion system can be computed by inversion of the rotation equations of motion that are given by Eq. (3.48):

$$\left(\vec{\mathbf{M}}_T^G\right)_{B,REF} = \left(\mathbf{I}^G\right)_{BB} \cdot \left(\dot{\vec{\boldsymbol{\omega}}}_K^{IB}\right)_B^B + \left(\vec{\boldsymbol{\omega}}_K^{IB}\right)_B \times \left(\mathbf{I}^G\right)_{BB} \cdot \left(\vec{\boldsymbol{\omega}}_K^{IB}\right)_B \tag{3.288}$$

In general, the relationship between the moments and the control surface deflections is quite complicated and cannot be solved analytically for the required control surface deflections. Then, an incremental approach can be chosen where at first the difference between the actual moments and the commanded reference moments has to be determined by:

$$\left(\Delta\vec{\mathbf{M}}^G\left(\Delta\eta, \Delta\xi, \Delta\varsigma\right)\right)_{B,REF} = \left(\vec{\mathbf{M}}_T^G\right)_{B,REF} - \left(\vec{\mathbf{M}}^G\left(\eta_0, \xi_0, \varsigma_0\right)\right)_B \tag{3.289}$$

Since now only incremental changes in the moments are demanded, these can be assumed to be small and therefore the incremental changes can be linearized with respect to the control surface deflections η, ξ and ζ. In the following, $\mathbf{u}$ represents the vector of those control surface deflections:

$$\left(\Delta\vec{\mathbf{M}}^{G}\left(\Delta\mathbf{u}\right)\right)_{B,REF} = \frac{\partial\left(\vec{\mathbf{M}}^{G}\right)_{B}}{\partial\mathbf{u}}\Delta\mathbf{u}_{REF} = \mathbf{B}\cdot\Delta\mathbf{u}_{REF} \tag{3.290}$$

Now the required changes in the control surface deflections can be computed and later be added to the actual control surface deflections in order to obtain the total reference values for the control surface deflections that represent the output of the complete dynamic inversion part of the simulation model and simultaneously act as input to the full 6-Degree of Freedom simulation model:

$$\Delta\mathbf{u}_{REF} = \mathbf{B}^{-1}\cdot\left(\Delta\vec{\mathbf{M}}^{G}\left(\Delta\mathbf{u}\right)\right)_{B,REF} \tag{3.291}$$

$$\mathbf{u}_{REF} = \mathbf{u}_{0} + \Delta\mathbf{u}_{REF} \tag{3.292}$$

If the actual moments can be separated into moments solely due to the control surface deflections and into moments that are only produced by the body angular rates and the attitude angles, Eq. (3.292) can be rewritten into:

$$\begin{aligned}\mathbf{u}_{REF} &= \mathbf{u}_{0} + \mathbf{B}^{-1}\cdot\left(\Delta\vec{\mathbf{M}}^{G}\left(\Delta\mathbf{u}\right)\right)_{B,REF} = \\ &= \mathbf{u}_{0} + \mathbf{B}^{-1}\cdot\left[\left(\vec{\mathbf{M}}_{T}^{G}\right)_{B,REF} - \left(\vec{\mathbf{M}}^{G}\left(\eta_{0},\xi_{0},\varsigma_{0}\right)\right)_{B} - \left(\vec{\mathbf{M}}^{G}\left(\tilde{p}_{A},\tilde{q}_{A},\tilde{r}_{A},\alpha_{A},\beta_{A}\right)\right)_{B}\right]\end{aligned} \tag{3.293}$$

If the aerodynamics moments are computed utilizing Eqs. (6.21) to (6.23) of chapter 6.1, the moments that are solely generated by the control surface deflections are:

$$\left(\vec{\mathbf{M}}^{G}\left(\eta_{0},\xi_{0},\varsigma_{0}\right)\right)_{B} = \mathbf{B}\cdot\mathbf{u}_{0} \tag{3.294}$$

where

$$\mathbf{B} = \bar{q}\cdot S\cdot\begin{bmatrix} s\cdot C_{l\xi} & 0 & s\cdot C_{l\varsigma} \\ 0 & \bar{c}\cdot C_{m\eta} & 0 \\ s\cdot C_{n\xi} & 0 & s\cdot C_{n\varsigma}\end{bmatrix}_{B} \tag{3.295}$$

Inserting Eq. (3.294) into Eq. (3.293), the reference control surface deflections evaluate to

$$\begin{aligned}\mathbf{u}_{REF} &= \mathbf{u}_{0} + \mathbf{B}^{-1}\cdot\left[\left(\vec{\mathbf{M}}_{T}^{G}\right)_{B,REF} - \mathbf{B}\cdot\mathbf{u}_{0} - \left(\vec{\mathbf{M}}^{G}\left(\tilde{p}_{A},\tilde{q}_{A},\tilde{r}_{A},\alpha_{A},\beta_{A}\right)\right)_{B}\right] = \\ &= \mathbf{B}^{-1}\cdot\left[\left(\vec{\mathbf{M}}_{T}^{G}\right)_{B,REF} - \left(\vec{\mathbf{M}}^{G}\left(\tilde{p}_{A},\tilde{q}_{A},\tilde{r}_{A},\alpha_{A},\beta_{A}\right)\right)_{B}\right]\end{aligned} \tag{3.296}$$

where

$$\left(\vec{\mathbf{M}}^{G}\left(\tilde{p}_{A},\tilde{q}_{A},\tilde{r}_{A},\alpha_{A},\beta_{A}\right)\right)_{B} = \bar{q}\cdot S\cdot\begin{bmatrix} s\cdot\left(C_{lp}\cdot\tilde{p}_{A} + C_{lr}\cdot\tilde{r}_{A} + C_{l\beta}\cdot\beta_{A}\right) \\ \bar{c}\cdot\left(C_{m0} + C_{m\alpha}\cdot\alpha_{A} + C_{mq}\cdot\tilde{q}_{A}\right) \\ s\cdot\left(C_{np}\cdot\tilde{p}_{A} + C_{nr}\cdot\tilde{r}_{A} + C_{n\beta}\cdot\beta_{A}\right)\end{bmatrix}_{B}. \tag{3.297}$$

3.3.11 Reference Values for the Linear State-Space Models

Besides the full, non-linear rotational and attitude dynamics, the principle of dynamic inversion can also be applied to the simplified, linearized state-space models representing the

rotational and attitude dynamics that have been described in chapter 3.2.6. This is illustrated in the following.

The dynamic model for the generation of the reference values uses the linearized dynamics of longitudinal and lateral motion of the aircraft. As described above, the longitudinal motion is described by the following short period approximation:

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{\alpha}_K^G \\ \dot{q}_K^* \end{pmatrix} = \begin{bmatrix} 0 & Z_q + 1 \\ 0 & M_q \end{bmatrix} \cdot \begin{pmatrix} \alpha_K^G \\ q_K^* \end{pmatrix} + \begin{bmatrix} Z_\alpha & Z_\eta \\ M_\alpha & M_\eta \end{bmatrix} \cdot \begin{pmatrix} \alpha_A^G \\ \eta_{CMD} \end{pmatrix} = \mathbf{A} \cdot \mathbf{x} + \mathbf{B} \cdot \mathbf{u} \tag{3.298}$$

$$y = \begin{pmatrix} 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} \alpha_K^G \\ q_K^* \end{pmatrix} = \mathbf{c}^T \cdot \mathbf{x} \tag{3.299}$$

Here again it has to be mentioned that the first order time derivative of the kinematic angle of attack $\dot{\alpha}_K$ is a function of the aerodynamic angle of attack α_A in analogy to the full, non-linear 6-Degree of Freedom simulation model.

For the longitudinal motion, the output y of the linearized dynamic model equals the kinematic angle of attack α_K. Since the elevator deflection η does not appear directly in this output y nor in its first order derivative $\dot{y}$ (after substituting $\dot{\mathbf{x}}$ by Eq. (3.298) and with Z_η being close to zero), the second order derivative $\ddot{y}$ of the output y has to be computed to apply the principle of dynamic inversion. Therefore, the output y is said to be of relative degree two.

By solving Eq. (3.298) for the elevator command η_{CMD}, the following equations for the computation of the commanded elevator deflection η_{CMD} result:

$$\eta_{CMD} = \frac{1}{M_\eta}\left(\dot{q}_K^* - M_\alpha \alpha_A^G - M_q q_K^*\right) \tag{3.300}$$

with

$$q_K^* = \frac{1}{Z_q + 1}\left(\dot{\alpha}_K^G - Z_\alpha \alpha_A^G\right) \tag{3.301}$$

$$\dot{q}_K^* = \frac{1}{Z_q + 1}\left(\ddot{\alpha}_K^G - \dot{Z}_\alpha \alpha_A^G - Z_\alpha \dot{\alpha}_A^G - \dot{Z}_q q_K^*\right) \tag{3.302}$$

where the first order time derivatives of the force coefficients are obtained by deriving the respective coefficients with respect to time. For example, the first order time derivative of the force coefficient Z_α evaluates to:

$$\dot{Z}_\alpha = -\frac{\rho \cdot S}{2 \cdot m}\left(\dot{V}_A \cdot \left[C_{L\alpha} + C_{D|_0}\right] + V_A \cdot \dot{C}_{D|_0}\right) \tag{3.303}$$

where

$$\dot{C}_{D|_0} = 2 \cdot k \cdot \left(C_{L0} + C_{L\alpha} \cdot \alpha_A - C_{L,C_{D0}}\right) \cdot C_{L\alpha} \cdot \dot{\alpha}_A \tag{3.304}$$

For the lateral motion a third order model with the states pseudo-roll rate p_K^*, pseudo-yaw rate r_K^* and kinematic angle of sideslip β_K is implemented, where the first order time derivative of the kinematic sideslip angle $\dot{\beta}_K$ is a function of the aerodynamic angle of sideslip β_A analogous to the full, non-linear 6-Degree of Freedom simulation model plant:

$$\dot{\mathbf{x}} = \begin{pmatrix} \dot{p}_K^* \\ \dot{r}_K^* \\ \dot{\beta}_K^G \end{pmatrix} = \begin{bmatrix} L_p & L_r & 0 \\ N_p & N_r & 0 \\ Y_p & Y_r - 1 & 0 \end{bmatrix} \cdot \begin{pmatrix} p_K^* \\ r_K^* \\ \beta_K^G \end{pmatrix} + \begin{bmatrix} L_\xi & L_\zeta & L_\beta \\ N_\xi & N_\zeta & N_\beta \\ Y_\xi & Y_\zeta & Y_\beta \end{bmatrix} \cdot \begin{pmatrix} \xi_{CMD} \\ \zeta_{CMD} \\ \beta_A^G \end{pmatrix} = \mathbf{A} \cdot \mathbf{x} + \mathbf{B} \cdot \mathbf{u} \tag{3.305}$$

$$\mathbf{y} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} p_K^* \\ r_K^* \\ \beta_K \end{pmatrix} = \begin{pmatrix} \mathbf{c}_1^T \\ \mathbf{c}_2^T \end{pmatrix} \cdot \mathbf{x} \tag{3.306}$$

The outputs of the lateral motion dynamic model are the pseudo-roll rate p_K^* and the kinematic angle of sideslip β_K. For the pseudo-roll rate output, the input appears for the first time in the first order derivative, whereas for the angle of sideslip, the input can be seen primarily in the second order derivative with the side force derivatives Y_ξ and Y_ζ being close to zero. Thus, the roll rate output is of relative degree one, the angle of sideslip output is of relative degree two.

Solving Eq. (3.305) for the aileron command ξ and the rudder command ζ gives the following equations for the calculation of the reference values for the aileron and rudder surface deflections:

$$\xi_{CMD} = \frac{\dot{r}_K^* - N_p p_K^* - N_r r_K^* - N_\beta \beta_A^G + \frac{N_\zeta}{L_\zeta}\left(L_p p_K^* - \dot{p}_K^* + L_r r_K^* + L_\beta \beta_A^G\right)}{N_\xi - \frac{L_\xi}{L_\zeta} N_\zeta} \tag{3.307}$$

$$\zeta_{CMD} = \frac{\dot{r}_K^* - N_p p_K^* - N_r r_K^* - N_\beta \beta_A^G + \frac{N_\xi}{L_\xi}\left(L_p p_K^* - \dot{p}_K^* + L_r r_K^* + L_\beta \beta_A^G\right)}{N_\zeta - \frac{L_\zeta}{L_\xi} N_\xi} \tag{3.308}$$

with

$$r_K^* = \frac{1}{Y_r - 1}\left(\dot{\beta}_K^G - Y_p p_K^* - Y_\beta \beta_A^G\right) \tag{3.309}$$

$$\dot{r}_K^* = \frac{1}{Y_r - 1}\left(\ddot{\beta}_K^G - \dot{Y}_p p_K^* - Y_p \dot{p}_K^* - \dot{Y}_r r_K^* - \dot{Y}_\beta \beta_A^G - Y_\beta \dot{\beta}_A^G\right) \tag{3.310}$$

The reference value for the pseudo-roll rate $p_{K,REF}^*$ is set equal to the first order derivative of the reference value for the kinematic flight-path bank angle $\dot{\mu}_{K,REF}$:

$$p_K^* = \left(\dot{\mu}_K^G\right)_K \tag{3.311}$$

Accordingly, the first order time derivative of the pseudo-roll rate reference value $\dot{p}_{K,REF}^*$ is set equal to the second order derivative of the reference value for the kinematic flight-path bank angle $\ddot{\mu}_{K,REF}$:

$$\dot{p}_{K,REF}^* = \left(\ddot{\mu}_K^G\right)_{K,REF} \tag{3.312}$$

As one might observe from the equations stated above, again not only the reference signals themselves are necessary for the dynamic inversion of the linear state-space models, but also

the second order time derivatives of the reference input values. The output of the described subsystem containing the inverted linear state-space models are the reference control surface deflections η, ξ and ζ that in turn act as input to the hybrid simulation model that is the non-linear point-mass model augmented by the linear state-space models representing the inner loop dynamics.

3.3.12 Reference Values for the Linear Transfer Functions

The principle of dynamic inversion can also be applied to the most simplified hybrid simulation model where the attitude and rotational dynamics are modeled as linear transfer functions for the load factors and the roll rate respectively. This procedure is described in the following for the case that the flight system is not modeled as a non-minimum phase system.

As mentioned above, in the kinematic model the dynamics of the normal load factor n_z and the dynamics of the lateral load factor n_y are approximated by a second order time behavior, while the dynamics of the load factor n_x and the dynamics of the roll rate p_K feature a first order time behavior. In order to guarantee a good approximation of the correct aircraft dynamics, the dynamics for the lift build-up use the same dynamics as the short period oscillation and the build-up of the side force corresponds to the dutch roll dynamics. Furthermore, for the roll rate, the decoupled first order roll dynamics are used and for the thrust, a specific engine time constant is defined.

For example, the dynamics for the normal load factor n_z are given by the following second order linear transfer function:

$$\left(n_z^G(s)\right)_A = \frac{\omega_{0,SP}^2}{s^2 + 2 \cdot \zeta_{SP} \cdot \omega_{0,SP} \cdot s + \omega_{0,SP}^2} \left(n_z^G(s)\right)_{A,CMD} \tag{3.313}$$

A paramount goal of the inverse simulation model together with the simulation model plant is to follow a given trajectory as close as possible. Therefore, a dynamic inversion control structure has been chosen for the inner loop control system to achieve this goal. By applying the dynamic inversion control principle, the linear transfer functions for the load factor and roll rate dynamics are inverted and the highest derivative order of the load factor is replaced with the respective reference value to provide optimal trajectory following of the simulation model. Thus, the resulting equation for the computation of the reference value for the normal load factor n_z is given by

$$\left(n_z^G\right)_{A,CMD} = \frac{1}{\omega_{0,SP}^2} v + \frac{2 \cdot \zeta_{SP}}{\omega_{0,SP}} \left(\dot{n}_z^G\right)_A^A + \left(n_z^G\right)_A \tag{3.314}$$

where the new virtual input v, the so-called pseudo-control, is then replaced by the second order derivative of the load factor reference value to guarantee perfect trajectory following:

$$\left(n_z^G\right)_{A,CMD} = \frac{1}{\omega_{0,SP}^2} \left(\ddot{n}_z^G\right)_{A,REF}^{AA} + \frac{2 \cdot \zeta_{SP}}{\omega_{0,SP}} \left(\dot{n}_z^G\right)_A^A + \left(n_z^G\right)_A \tag{3.315}$$

As can be seen from the above equation, not only the reference values themselves are required for the inner loop dynamic inversion, but the second order derivatives of the specific reference values. The equations for the side force n_y, the load factor in the x-direction n_x and the roll rate p_K are derived just in the same manner. The reference value for the commanded roll rate p_K is obtained by:

$$p_{K,CMD} = T_{Roll} \cdot \dot{p}_K + p_K \tag{3.316}$$

3.4 Reference Models

As mentioned before, for the inverse simulation model not only the time history of the reference values themselves but also higher order time derivatives of these reference values are essential for the calculation of the inputs to the simulation model that will produce the desired outputs. Because of its inherent dynamics, the respective flight system under consideration will not be able to follow arbitrary time histories concerning the reference values that are fed into the inverse simulation model. This means that the differentiation order of the reference values' time histories has to take into account the dynamic order of the causal chain between the considered inputs and outputs. As can be seen from the corresponding physical causal chain (Fig. 8), there are two integrations between the commanded elevator deflection η_{CMD} and the resulting angle of attack α. Therefore, the second order time derivative with respect to time of the angle of attack α is a direct function of the elevator deflection command η_{CMD}:

$$\ddot{\alpha}(t) = f(\eta_{CMD}(t)) \tag{3.317}$$

Thus, the resulting time history for the angle of attack α is twice differentiable with respect to time. On the other hand this implies that for the calculation of the elevator deflection command by inversion of Eq. (3.317) not only the time histories for the angle of attack α and its first order time derivative but also the time history for the second order time derivative of the angle of attack are necessary:

$$\eta_{REF}(t) = f^{-1}(\alpha(t), \dot{\alpha}(t), \ddot{\alpha}(t)) \tag{3.318}$$

This implies that the reference time history for the angle of attack cannot be discrete nor linear but has to be at least twice differentiable with respect to time to give a time history for the commanded elevator deflection.

Each subsystem of the inverse simulation model requires as input specific reference values and their derivatives up to a certain order for the calculation of the respective output reference values and their derivatives up to a specified order. E.g. the subsystem for the calculation of the reference load factor values in the *K*-Frame requires the flight-path kinematic variables and their time derivatives up to the order of three to be able to compute the reference load factors and their second order time derivatives. These input reference values can either be taken from the next outer subsystem that would be the path reference subsystem or they can be commanded directly. In the latter case it has to be guaranteed that the time history for the flight-path values that is fed into the subsystem for the generation of the reference load factors is smooth enough so that the dynamics of the flight system can track the prescribed reference values.

One possibility to guarantee the required differentiation order is the utilization of a model-based approach on the basis of reference models. Reference models generate signals that are continuously differentiable where the relative degree of those signals corresponds to the dynamic order of the regarded flight system. These reference models contain linear transfer functions of the respective degree, where the input is an arbitrary time history for any reference value. This time history can also be discrete or linear and does not have to take into account the in fact required differentiation order. The resulting output of these reference

models is the corresponding time history of this reference value that is of the necessary differentiation order such that the reference value fed forward to the inverse simulation model already takes into account the dynamic order of the physical causal chain under consideration. In the following, exemplified reference models up to the order of two are given, i.e. that the resulting time history for the respective reference value is twice continuously differentiable with respect to time.

3.4.1 First Order Reference Model

Direct command values for the body angular rates, that are the roll rate p_K, the pitch rate q_K and the yaw rate r_K have to be fed through first order reference models in order to generate time histories that are smooth enough, i.e. that the resulting time histories can once be continuously differentiated with respect to time. This is guaranteed by a linear transfer function of the order of one:

$$p_{K,REF} = \frac{1}{T_R \cdot s + 1} p_{K,CMD} \tag{3.319}$$

The transfer functions for the pitch rate q_K and the yaw rate r_K are analogous. The basic layout of a first order reference model is shown in Fig. 19. The time constants of the various reference models have to be adapted so that the reference time history and its first order time derivative can be followed by the plant and thus an accurate tracking is guaranteed. The above linear transfer function can also be written as a first order differential equation:

$$\dot{p}_{K,REF} = \frac{1}{T_R}\left(p_{K,CMD} - p_{K,REF}\right) \tag{3.320}$$

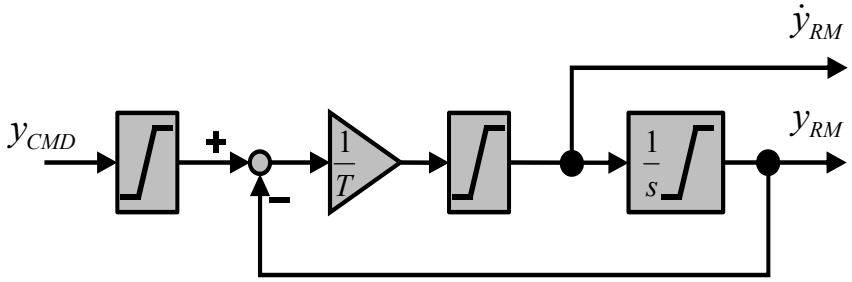

Figure 19. First Order Reference Model

With each first order reference model, an additional state is added to the state vector of the simulation model. Thus, with the three reference models for the body angular rates p_K, q_K and r_K the three states $p_{K,REF}$, $q_{K,REF}$ and $r_{K,REF}$ are added to the simulation model state vector. In Fig. 20, the response curve for a first order reference model due to a step input is shown.

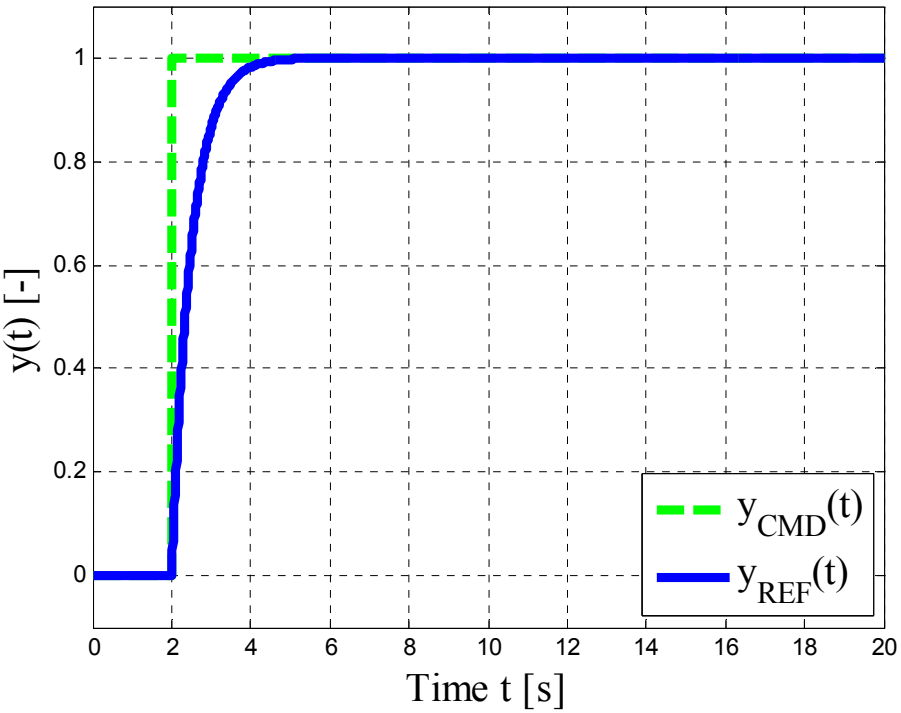

Figure 20. Step Response of 1st Order Reference Model

3.4.2 Second Order Reference Model

For the reference load factors $\vec{\mathbf{n}}_{REF}$, the reference attitude angles α_{REF} and β_{REF} as well as the reference flight-path bank angle μ_{REF} the time histories for the reference values have to be at least twice continuously differentiable with respect to time to take into account the dynamic order of the respective flight system. Thus, direct load factor commands or direct commands for the attitude angles are delayed by second order linear transfer functions in order to enable the simulation model to track the prescribed reference values. For example, the reference model for the aerodynamic angle of attack α_A consists of the following second order linear transfer function:

$$\alpha_A^G = \frac{\omega_0^2}{s^2 + 2 \cdot \zeta \cdot \omega_0 \cdot s + \omega_0^2} \alpha_{A,CMD}^G \tag{3.321}$$

The transfer functions for the load factors $\vec{\mathbf{n}}$ and the angles β and μ feature the same structure. Here again, the parameters of the linear transfer functions that are the relative damping ζ and the natural frequency ω_0 have to be adjusted in such a way to allow for an accurate tracking of the reference values by the simulation model plant. Written in state-space form, the linear transfer function (3.321) reads:

$$\frac{\partial}{\partial t}\begin{pmatrix} \alpha_A^G \\ \dot{\alpha}_A^G \end{pmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & -2 \cdot \zeta \cdot \omega_0 \end{bmatrix} \cdot \begin{pmatrix} \alpha_A^G \\ \dot{\alpha}_A^G \end{pmatrix} + \begin{bmatrix} 0 \\ \omega_0^2 \end{bmatrix} \cdot \alpha_{A,CMD}^G \tag{3.322}$$

$$y = \alpha_A^G = \begin{bmatrix} 1 & 0 \end{bmatrix} \cdot \begin{pmatrix} \alpha_A^G \\ \dot{\alpha}_A^G \end{pmatrix} \tag{3.323}$$

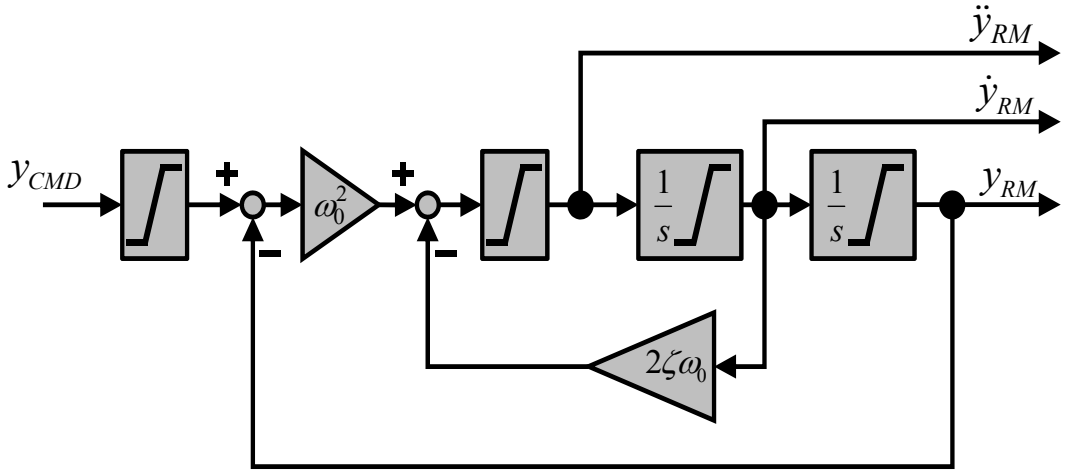

Figure 21. Second Order Reference Model

The reference model-based approach for the generation of sufficiently smooth reference values augments the simulation model state vector by two states for every single reference model, i.e. the simulation model state vector is supplemented in total by six additional states. In case that the load factor reference values are generated by the respective reference models, the six extra states are the three reference load factors $n_{x,REF}$, $n_{y,REF}$ and $n_{z,REF}$ and their first order time derivatives $\dot{n}_{x,REF}$, $\dot{n}_{y,REF}$ and $\dot{n}_{z,REF}$. Fig. 21 shows the basic layout of a second order reference model, while in Fig. 22 the corresponding step response curve is depicted.

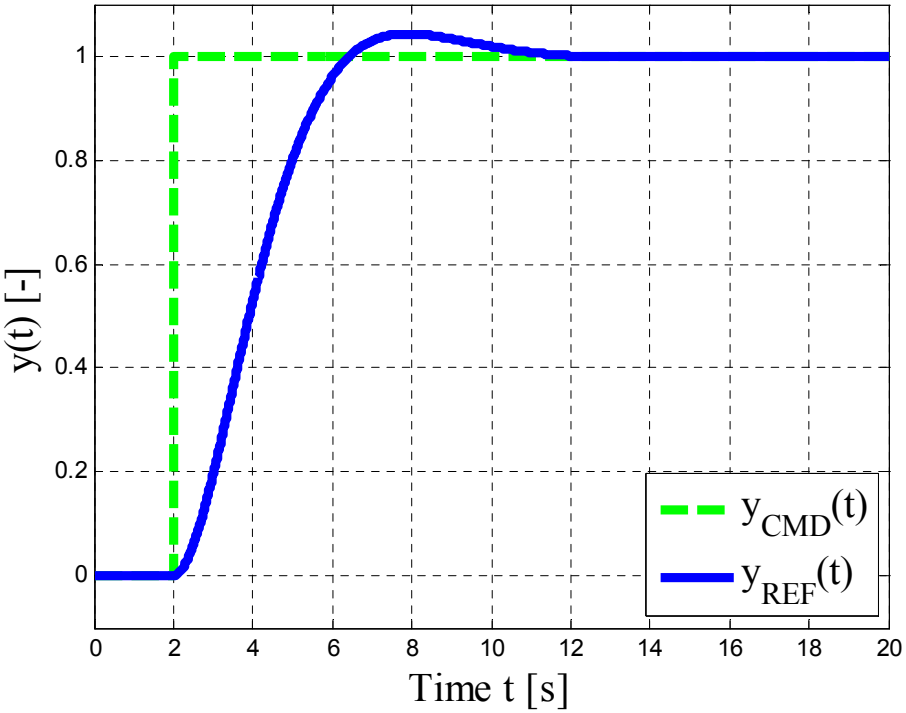

Figure 22. Step Response of 2nd Order Reference Model

3.5 Error Feedbacks as Stabilizing Controls

In order to guarantee a precise tracking of the reference values given by the inversion controllers provided in chapter 3.3, error feedbacks on all levels of the simulation model have to be implemented to eliminate deviations of the actual values from the reference values. For

the simulation model, there are various reasons why the resulting trajectories may diverge from the input reference trajectories: besides numerical computation errors that occur because of the limited computational accuracy and the round-off errors that are inherent to the machine used for simulation respectively optimization, further reasons are the actuator dynamics and the non-minimum phase behavior of the flight system dynamics. While the actuator dynamics are not incorporated in the inverse simulation model at all, the non-minimum phase part of the respective flight system cannot be inverted as mentioned before. Thus, in general the reference commands generated by the inverse simulation model will result in output trajectories of the simulation model that differ from the input reference trajectories not only by the computational errors but also by additional errors caused by the actuator dynamics and the non-minimum phase part of the flight system dynamics if these effects are taken into account in the simulation model itself. In order to avoid growing deviations between the reference trajectories and the resulting trajectories induced by the effects outlined above, error feedbacks on all levels of the simulation model are implemented in the simulation model.

To allow the error feedback controllers to react to any possible deviations from the reference values as fast as possible, the deviations of the respective state values of the simulation model and also of the higher order time derivatives of the considered values are fed back to the highest order time derivative of the corresponding reference value of the inverse simulation model. The lower derivatives of the reference signals are used directly as input reference values for the next subsystem of the dynamic inversion part of the simulation model. Since all derivatives of a specific signal are utilized for the error feedback, the error dynamics can be set to any desired dynamic order by adjusting the feedback gains of the respective control loops.

For the implementation of the error feedback control loops, parts of the simulation model have to be extended to allow for the computation of not only the first order time derivatives themselves but also of certain higher order time derivatives. E.g. not only the first order time derivative of the kinematic flight-path course angle but also its second order time derivative will be fed back to the third order time derivative of the given reference time history of the flight-path course angle. Therefore, this second order time derivative has to be computed in addition to its first order time derivative that results from the translation equations of motion. The corresponding equations can be obtained by differentiating the equations of motion and the propagation equations depicted in chapter 3.2 with respect to time. In the following, the implemented error feedback control laws for the different levels of the simulation model are given. Furthermore, the error dynamics for the aerodynamic angle of attack control loop are derived exemplarily.

3.5.1 Trajectory Deviation Control Loop

For the trajectory deviation control, a Trajectory Frame *T* is defined with its origin located at the current footpoint on the reference trajectory calculated by Eq. (3.188). The orientation of the Trajectory Frame *T* is then obtained by two rotations of the *NED*-Frame around its *z*-axis respectively its *y*-axis by the reference values for the kinematic course angle χ_K and the kinematic climb angle γ_K corresponding to the actual footpoint on the reference trajectory. The transformation matrix $\mathbf{M}_{TO}$ between the *NED*-Frame and the corresponding rotational rates are:

$$\mathbf{M}_{TO} = \begin{bmatrix} \cos\chi_K^G \cos\gamma_K^G & \sin\chi_K^G \cos\gamma_K^G & -\sin\gamma_K^G \\ -\sin\chi_K^G & \cos\chi_K^G & 0 \\ \cos\chi_K^G \sin\gamma_K^G & \sin\chi_K^G \sin\gamma_K^G & \cos\gamma_K^G \end{bmatrix} \tag{3.324}$$

$$\left(\vec{\boldsymbol{\omega}}^{TO}\right)_T = \begin{bmatrix} \dot{\chi}_K^G \cdot \sin\gamma_K^G \\ -\dot{\gamma}_K^G \\ -\dot{\chi}_K^G \cdot \cos\gamma_K^G \end{bmatrix} \tag{3.325}$$

$$\left(\dot{\vec{\boldsymbol{\omega}}}^{TO}\right)_T^T = \begin{bmatrix} \ddot{\chi}_K^G \cdot \sin\gamma_K^G + \dot{\chi}_K^G \cdot \cos\gamma_K^G \cdot \dot{\gamma}_K^G \\ -\ddot{\gamma}_K^G \\ -\ddot{\chi}_K^G \cdot \cos\gamma_K^G + \dot{\chi}_K^G \cdot \sin\gamma_K^G \cdot \dot{\gamma}_K^G \end{bmatrix} \tag{3.326}$$

$$\left(\ddot{\vec{\boldsymbol{\omega}}}^{TO}\right)_T^{TT} = \begin{bmatrix} \sin\gamma_K^G \cdot \left(\dddot{\chi}_K^G - \dot{\chi}_K^G \cdot \left(\dot{\gamma}_K^G\right)^2\right) + \cos\gamma_K^G \cdot \left(2 \cdot \ddot{\chi}_K^G \cdot \dot{\gamma}_K^G + \dot{\chi}_K^G \cdot \ddot{\gamma}_K^G\right) \\ -\dddot{\gamma}_K^G \\ -\cos\gamma_K^G \cdot \left(\dddot{\chi}_K^G - \dot{\chi}_K^G \cdot \left(\dot{\gamma}_K^G\right)^2\right) + \sin\gamma_K^G \cdot \left(2 \cdot \ddot{\chi}_K^G \cdot \dot{\gamma}_K^G + \dot{\chi}_K^G \cdot \ddot{\gamma}_K^G\right) \end{bmatrix} \tag{3.327}$$

Given the case that the aircraft is perfectly following the reference trajectory, the Trajectory Frame *T* coincidences with the Kinematic Flight-Path Frame *K*. Then, the trajectory deviation vector $\Delta\vec{\mathbf{r}} = [\Delta x, \Delta y, \Delta z]$ and its time derivatives up to the third order are transformed from the *NED*-Frame into the Trajectory Frame *T* by the following equations:

$$\left(\Delta\vec{\mathbf{r}}\right)_T = \mathbf{M}_{TO}\left(\Delta\vec{\mathbf{r}}\right)_O \tag{3.328}$$

$$\left(\Delta\dot{\vec{\mathbf{r}}}\right)_T^T = \mathbf{M}_{TO}\left(\Delta\dot{\vec{\mathbf{r}}}\right)_O^O + \left(\vec{\boldsymbol{\omega}}^{TO}\right)_T \times \left(\Delta\vec{\mathbf{r}}\right)_T \tag{3.329}$$

$$\begin{aligned} \left(\Delta\ddot{\vec{\mathbf{r}}}\right)_T^{TT} &= \mathbf{M}_{TO}\left(\Delta\ddot{\vec{\mathbf{r}}}\right)_O^{OO} + \left(\vec{\boldsymbol{\omega}}^{TO}\right)_T \times \left[\mathbf{M}_{TO}\left(\Delta\dot{\vec{\mathbf{r}}}\right)_O^O\right] \\ &\quad + \left(\dot{\vec{\boldsymbol{\omega}}}^{TO}\right)_T^T \times \left(\Delta\vec{\mathbf{r}}\right)_T + \left(\vec{\boldsymbol{\omega}}^{TO}\right)_T \times \left(\Delta\dot{\vec{\mathbf{r}}}\right)_T^T \end{aligned} \tag{3.330}$$

$$\begin{aligned} \left(\Delta\dddot{\vec{\mathbf{r}}}\right)_T^{TTT} &= \mathbf{M}_{TO}\left(\Delta\dddot{\vec{\mathbf{r}}}\right)_O^{OOO} + 2 \cdot \left(\vec{\boldsymbol{\omega}}^{TO}\right)_T \times \left[\mathbf{M}_{TO}\left(\Delta\ddot{\vec{\mathbf{r}}}\right)_O^{OO}\right] \\ &\quad + \left(\dot{\vec{\boldsymbol{\omega}}}^{TO}\right)_T^T \times \left[\mathbf{M}_{TO}\left(\Delta\dot{\vec{\mathbf{r}}}\right)_O^O\right] + \left(\vec{\boldsymbol{\omega}}^{TO}\right)_T \times \left[\left(\vec{\boldsymbol{\omega}}^{TO}\right)_T \times \left[\mathbf{M}_{TO}\left(\Delta\dot{\vec{\mathbf{r}}}\right)_O^O\right]\right] \\ &\quad + \left(\ddot{\vec{\boldsymbol{\omega}}}^{TO}\right)_T^{TT} \times \left(\Delta\vec{\mathbf{r}}\right)_T + 2 \cdot \left(\dot{\vec{\boldsymbol{\omega}}}^{TO}\right)_T^T \times \left(\Delta\dot{\vec{\mathbf{r}}}\right)_T^T + \left(\vec{\boldsymbol{\omega}}^{TO}\right)_T \times \left(\Delta\ddot{\vec{\mathbf{r}}}\right)_T^{TT} \end{aligned} \tag{3.331}$$

For the desired feedback dynamics in the horizontal plane, first order error dynamics have been implemented to guide the aircraft back onto the reference trajectory:

$$T_y \cdot \left(\Delta\dot{y}\right)_T + \left(\Delta y\right)_T = 0 \tag{3.332}$$

where the time constant T_y can be utilized to adjust the dynamics of the trajectory deviation controller. From the position propagation equations, the first order time derivative of the deviation Δy evaluates to:

$$\Delta\dot{y} = V_K^G \cdot \sin\Delta\chi_K^G \cdot \cos\Delta\gamma_K^G \tag{3.333}$$

For small angles $\Delta\chi_K$ and $\Delta\gamma_K$, the following approximations can be made:

$$\cos\Delta\gamma_K^G \cong 1 \tag{3.334}$$

$$\sin \Delta\chi_K^G \approx \Delta\chi_K^G \tag{3.335}$$

Then, Eq. (3.333) simplifies to

$$\Delta\dot{y} = V_K^G \cdot \Delta\chi_K^G \tag{3.336}$$

Inserting Eq. (3.336) into Eq. (3.332) and solving for $\Delta\chi_K$ gives

$$\Delta\chi_K^G = -\frac{(\Delta y)_T}{V_K^G \cdot T_y} \tag{3.337}$$

For the kinematic flight-path reference loop that follows the trajectory deviation control loop, time derivatives of the kinematic course angle χ_K up to the third order are required. The higher order derivatives therefore are obtained by differentiating the course angle deviation $\Delta\chi_K$:

$$\Delta\dot{\chi}_K^G = -\frac{1}{T_y}\left(-\frac{\dot{V}_K^G \cdot (\Delta y)_T}{\left(V_K^G\right)^2} + \frac{(\Delta\dot{y})_T}{V_K^G}\right) \tag{3.338}$$

$$\Delta\ddot{\chi}_K^G = -\frac{1}{T_y}\left(\frac{\left(2\cdot\left(\dot{V}_K^G\right)^2 - V_K^G \cdot \ddot{V}_K^G\right)\cdot(\Delta y)_T}{\left(V_K^G\right)^3} - 2\frac{\dot{V}_K^G \cdot (\Delta\dot{y})_T}{\left(V_K^G\right)^2} + \frac{(\Delta\ddot{y})_T}{V_K^G}\right) \tag{3.339}$$

$$\begin{aligned}\Delta\dddot{\chi}_K^G = -\frac{1}{T_y}\Bigg(&\frac{\left(-6\cdot\left(\dot{V}_K^G\right)^3 + 6\cdot V_K^G \cdot \dot{V}_K^G \cdot \ddot{V}_K^G - \dddot{V}_K^G \cdot \left(V_K^G\right)^2\right)\cdot(\Delta y)_T}{\left(V_K^G\right)^4} + \\ &+ 3\frac{\left(2\cdot\left(\dot{V}_K^G\right)^2 - V_K^G \cdot \ddot{V}_K^G\right)\cdot(\Delta\dot{y})_T}{\left(V_K^G\right)^3} - 3\frac{\dot{V}_K^G \cdot (\Delta\ddot{y})_T}{\left(V_K^G\right)^2} + \frac{(\Delta\dddot{y})_T}{V_K^G}\Bigg)\end{aligned} \tag{3.340}$$

For the vertical dynamics, the same feedback dynamics as for the horizontal error feedback controller have been chosen:

$$T_z \cdot (\Delta\dot{z})_T + (\Delta z)_T = 0 \tag{3.341}$$

where the time constant T_z can be utilized to adjust the dynamics of the trajectory deviation controller. From the position propagation equations, the first order time derivative of the vertical deviation Δz is:

$$\Delta\dot{z} = -V_K^G \cdot \sin \Delta\gamma_K^G \tag{3.342}$$

Applying the assumption that

$$\sin \Delta\gamma_K^G \cong \Delta\gamma_K^G \tag{3.343}$$

and inserting Eq. (3.342) into Eq. (3.341), one obtains for the deviation of the kinematic flight-path inclination angle γ_K:

$$\Delta\gamma_K^G = \frac{(\Delta z)_T}{V_K^G \cdot T_z} \tag{3.344}$$

Again, the higher order derivatives are required:

$$\Delta\dot{\gamma}_K^G = \frac{1}{T_z}\left(-\frac{\dot{V}_K^G \cdot (\Delta z)_T}{\left(V_K^G\right)^2} + \frac{(\Delta\dot{z})_T}{V_K^G}\right) \tag{3.345}$$

$$\Delta\ddot{\gamma}_K^G = \frac{1}{T_z}\left(\frac{\left(2\cdot\left(\dot{V}_K^G\right)^2 - V_K^G\cdot\ddot{V}_K^G\right)\cdot\left(\Delta z\right)_T}{\left(V_K^G\right)^3} - 2\frac{\dot{V}_K^G\cdot\left(\Delta\dot{z}\right)_T}{\left(V_K^G\right)^2} + \frac{\left(\Delta\ddot{z}\right)_T}{V_K^G}\right) \tag{3.346}$$

$$\begin{aligned}\Delta\dddot{\gamma}_K^G = \frac{1}{T_z}\Bigg(&\frac{\left(-6\cdot\left(\dot{V}_K^G\right)^3 + 6\cdot V_K^G\cdot\dot{V}_K^G\cdot\ddot{V}_K^G - \dddot{V}_K^G\cdot\left(V_K^G\right)^2\right)\cdot\left(\Delta z\right)_T}{\left(V_K^G\right)^4} + \\ &+3\frac{\left(2\cdot\left(\dot{V}_K^G\right)^2 - V_K^G\cdot\ddot{V}_K^G\right)\cdot\left(\Delta\dot{z}\right)_T}{\left(V_K^G\right)^3} - 3\frac{\dot{V}_K^G\cdot\left(\Delta\ddot{z}\right)_T}{\left(V_K^G\right)^2} + \frac{\left(\Delta\dddot{z}\right)_T}{V_K^G}\Bigg)\end{aligned} \tag{3.347}$$

Finally, the reference values for the kinematic course angle χ_K respectively the kinematic flight-path bank angle γ_K are corrected by the course angle and climb angle deviations obtained by Eqs. (3.337) to (3.340) respectively Eqs. (3.344) to (3.347) to guide the aircraft back onto the reference trajectory:

$$\chi_{K,CMD}^G = \chi_{K,REF}^G - \Delta\chi_K^G \tag{3.348}$$

$$\dot{\chi}_{K,CMD}^G = \dot{\chi}_{K,REF}^G - \Delta\dot{\chi}_K^G \tag{3.349}$$

$$\ddot{\chi}_{K,CMD}^G = \ddot{\chi}_{K,REF}^G - \Delta\ddot{\chi}_K^G \tag{3.350}$$

$$\dddot{\chi}_{K,CMD}^G = \dddot{\chi}_{K,REF}^G - \Delta\dddot{\chi}_K^G \tag{3.351}$$

$$\gamma_{K,CMD}^G = \gamma_{K,REF}^G - \Delta\gamma_K^G \tag{3.352}$$

$$\dot{\gamma}_{K,CMD}^G = \dot{\gamma}_{K,REF}^G - \Delta\dot{\gamma}_K^G \tag{3.353}$$

$$\ddot{\gamma}_{K,CMD}^G = \ddot{\gamma}_{K,REF}^G - \Delta\ddot{\gamma}_K^G \tag{3.354}$$

$$\dddot{\gamma}_{K,CMD}^G = \dddot{\gamma}_{K,REF}^G - \Delta\dddot{\gamma}_K^G \tag{3.355}$$

3.5.2 Kinematic Flight-Path Variables Control Loop

In the kinematic flight-path variables control loop, deviations between the actual values for the kinematic flight-path variables and the respective reference values are fed back to the highest order time derivatives of the flight-path variables that are the second order time derivatives of the absolute kinematic velocity V_K and the third order time derivatives of the kinematic course angle χ_K as well as the kinematic flight-path climb angle γ_K:

$$\ddot{V}_{K,CMD}^G = \ddot{V}_{K,REF}^G + K_{\dot{V}}\left(\dot{V}_{K,REF}^G - \dot{V}_K^G\right) + K_V\left(V_{K,REF}^G - V_K^G\right) \tag{3.356}$$

$$\dddot{\gamma}_{K,CMD}^G = \dddot{\gamma}_{K,REF}^G + K_{\ddot{\gamma}}\left(\ddot{\gamma}_{K,REF}^G - \ddot{\gamma}_K^G\right) + K_{\dot{\gamma}}\left(\dot{\gamma}_{K,REF}^G - \dot{\gamma}_K^G\right) + K_{\gamma}\left(\gamma_{K,REF}^G - \gamma_K^G\right) \tag{3.357}$$

$$\dddot{\chi}_{K,CMD}^G = \dddot{\chi}_{K,REF}^G + K_{\ddot{\chi}}\left(\ddot{\chi}_{K,REF}^G - \ddot{\chi}_K^G\right) + K_{\dot{\chi}}\left(\dot{\chi}_{K,REF}^G - \dot{\chi}_K^G\right) + K_{\chi}\left(\chi_{K,REF}^G - \chi_K^G\right) \tag{3.358}$$

So far, the coupling between the kinematic flight-path climb angle γ_K and the kinematic course angle χ_K via the kinematic flight-path bank angle μ_K has not yet been considered for the implementation of the flight-path variables control loop. The coupling implies that at flight-path bank angles μ_K close to zero a faster reaction of the aircraft in the vertical plane is possible while at bank angles close to 90°, a faster reaction in the horizontal plane can be achieved. For a further improvement of the controller this coupling could also be taken into

account for the error feedbacks of the kinematic flight-path inclination angle γ_K respectively the kinematic course angle χ_K by an adjustment of the gains depending on the flight-path bank angle μ_K.

3.5.3 Kinematic Load Factors Control Loop

The kinematic load factors control loop contains the feedbacks of the deviations between the reference values for the kinematic load factors and the actual values of the kinematic load factors. The deviations of the load factors are fed back to the highest order time derivatives of the commanded kinematic load factors that are the first order time derivatives for the load factor $n_{K,x}$ and the second order time derivatives for the load factors $n_{K,y}$ respectively $n_{K,z}$:

$$\dot{n}_{K,x,CMD} = \dot{n}_{K,x,REF} + K_{nx}\left(n_{K,x,REF} - n_{K,x}\right) \tag{3.359}$$

$$\ddot{n}_{K,y,CMD} = \ddot{n}_{K,y,REF} + K_{n\dot{y}}\left(\dot{n}_{K,y,REF} - \dot{n}_{K,y}\right) + K_{ny}\left(n_{K,y,REF} - n_{K,y}\right) \tag{3.360}$$

$$\ddot{n}_{K,z,CMD} = \ddot{n}_{K,z,REF} + K_{n\dot{z}}\left(\dot{n}_{K,z,REF} - \dot{n}_{K,z}\right) + K_{nz}\left(n_{K,z,REF} - n_{K,z}\right) \tag{3.361}$$

For the load factors $\vec{\mathbf{n}}_K$ in the Intermediate Kinematic Flight-Path Frame $\bar{K}$ respectively the load factors $\vec{\mathbf{n}}_A$ in the Aerodynamic Reference Frame A, the same error feedbacks can be implemented.

3.5.4 Aerodynamic Attitude Angles Control Loop

As for the kinematic load factors control loop, the deviations between the reference values and the actual values of the aerodynamic attitude angles and their first order time derivatives are fed back to the second order time derivatives of the aerodynamic angle of attack α_A, the aerodynamic sideslip angle β_A and the aerodynamic flight-path bank angle μ_A in order to avoid any divergences between the reference values and the resulting simulation model plant values:

$$\ddot{\alpha}_{A,CMD} = \ddot{\alpha}_{A,REF} + K_{\dot{\alpha}}\left(\dot{\alpha}_{A,REF} - \dot{\alpha}_A\right) + K_{\alpha}\left(\alpha_{A,REF} - \alpha_A\right) \tag{3.362}$$

$$\ddot{\beta}_{A,CMD} = \ddot{\beta}_{A,REF} + K_{\dot{\beta}}\left(\dot{\beta}_{A,REF} - \dot{\beta}_A\right) + K_{\beta}\left(\beta_{A,REF} - \beta_A\right) \tag{3.363}$$

$$\ddot{\mu}_{A,CMD} = \ddot{\mu}_{A,REF} + K_{\dot{\mu}}\left(\dot{\mu}_{A,REF} - \dot{\mu}_A\right) + K_{\mu}\left(\mu_{A,REF} - \mu_A\right) \tag{3.364}$$

Regarding the kinematic attitude angles α_K, β_K and μ_K, identical error feedbacks can be applied.

In the following, a differential equation that describes the error dynamics with respect to the aerodynamic angle of attack α_A is derived. Therefore, at first the control error e is defined as (Ref. [Holzapfel, 2009d]):

$$e = \alpha_{A,REF} - \alpha_A \tag{3.365}$$

For the input-output linearized simulation model, the relationship between the commanded and the resulting second order time derivative of the aerodynamic angle of attack α_A is given by Eq. (3.366), where Δ_α represents an error term due to model uncertainties, modeling errors and/or numerical integration drift:

$$\ddot{\alpha}_A = \ddot{\alpha}_{A,CMD} + \Delta_\alpha \tag{3.366}$$

Inserting Eq. (3.362) into Eq. (3.366) gives:

$$\ddot{\alpha}_A - \ddot{\alpha}_{A,REF} = \Delta_\alpha + K_{\dot{\alpha}}\left(\dot{\alpha}_{A,REF} - \dot{\alpha}_A\right) + K_\alpha\left(\alpha_{A,REF} - \alpha_A\right) \tag{3.367}$$

Substituting the difference between the reference value and the resulting value for the aerodynamic angle of attack α_A by the error e defined in Eq. (3.365) and resorting the terms, the following differential equation for the error e results:

$$\ddot{e} + K_{\dot{\alpha}}\dot{e} + K_\alpha e = -\Delta_\alpha \tag{3.368}$$

The error dynamic with respect to the aerodynamic angle of attack α_A is excited by the error term Δ_α and can be adjusted by the coefficients $K_{\dot{\alpha}}$ and K_α. The evolution of the error e with time is limited if

- the real parts of the solution of the characteristic polynomial of the error dynamics (3.368) are negative,
- the error term Δ_α is limited and
- the initial deviation between the reference value $\ddot{\alpha}_{A,CMD}$ and the actual value $\ddot{\alpha}_A$ is limited.

3.5.5 Body Angular Rates Control Loop

Since the highest order time derivative of the body angular rates that is incorporated in the simulation model is the first order time derivative, only the deviations between the reference body angular rates and the actual values of the angular rates are fed back to the first order time derivative to penalize any deviations between the reference values and the simulation model output:

$$\dot{p}_{K,CMD} = \dot{p}_{K,REF} + K_p\left(p_{K,REF} - p_K\right) \tag{3.369}$$

$$\dot{q}_{K,CMD} = \dot{q}_{K,REF} + K_q\left(q_{K,REF} - q_K\right) \tag{3.370}$$

$$\dot{r}_{K,CMD} = \dot{r}_{K,REF} + K_r\left(r_{K,REF} - r_K\right) \tag{3.371}$$

3.6 Simulation Modes

As mentioned before, the simulation model features a novel architecture where the attitude and rotational dynamics are modeled in a serial manner to the translational and position dynamics in contrast to the widespread conventional simulation model architecture where the rotation equations of motion and the attitude propagation equations are modeled in a parallel manner to the translation equations of motion and the position propagation equations of motion (Refs. [Stevens, 1992], [Philips, 2004], [Etkin, 1996], [Etkin, 2005] and [Roskam, 2001]). This basic difference is depicted in Fig. 9. With this serial model architecture and the afore mentioned different depths of modeling for the inner loop, i.e. the rotational and the attitude flight dynamics, the simulation model allows for many different simulation modes and for a switching between these simulation modes. In this chapter, the most important simulation modes that are fundamental for the newly developed optimization procedure described later on are figured out. Besides the states that are utilized in the respective simulation modes the inherent characteristics like e.g. accuracy, complexity, computation time, etc. of these simulation modes are given.

3.6.1 Point-Mass Simulation Mode

The pure point-mass simulation mode (Fig. 23) is the most simplifying approach for simulating the flight system dynamics since with the non-linear point-mass simulation model only the position propagation equations and the translations equations of motion are taken into account for the simulation but no rotational and attitude dynamics. Therefore, the states of the simulation model are the aircraft position values x, y and z, the absolute kinematic velocity V_K, the thrust lever position δ_T and the kinematic flight-path variables course angle χ_K, climb angle γ_K and bank angle μ_K respectively the four quaternions q_0, q_1, q_2 and q_3. The controls are the aerodynamic angle of attack α_A, the aerodynamic angle of sideslip β_A and the first order time derivative of the aerodynamic bank angle $\dot{\mu}_A$ as well as the commanded thrust lever position $\delta_{T,CMD}$.

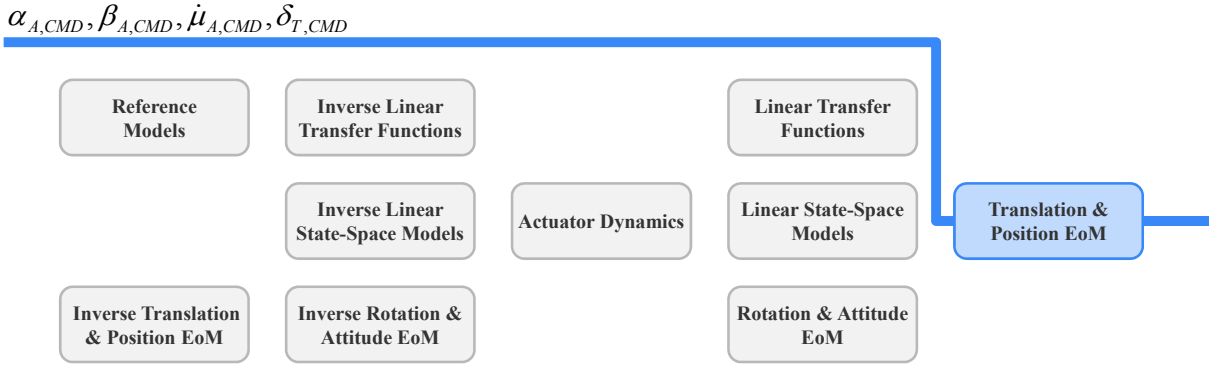

Figure 23. Point-Mass Simulation Mode

$$\mathbf{x} = \left[x, y, z, V_K, q_0, q_1, q_2, q_3, \delta_T\right]^T \tag{3.372}$$

$$\mathbf{u} = \left[\alpha_{A,CMD}, \beta_{A,CMD}, \dot{\mu}_{A,CMD}, \delta_{T,CMD}\right]^T \tag{3.373}$$

Here, the first order time derivative of the aerodynamic bank angle $\dot{\mu}_A$ is chosen as control input instead of the first order time derivative of the kinematic bank angle $\dot{\mu}_K$. Furthermore, within the simulation model the first order time derivative of the aerodynamic bank angle $\dot{\mu}_A$ then directly acts as input to the translation equations of motion, so that the four quaternions effectively represent the three flight-path angles that are the kinematic course angle χ_K, the kinematic path inclination angle γ_K and the **aerodynamic** bank angle μ_A.

Then, the load factors that are calculated in the Aerodynamic Reference Frame A from the aerodynamic angle of attack $\alpha_{A,CMD}$ and the aerodynamic sideslip angle $\beta_{A,CMD}$ can be directly transformed to the Kinematic Flight-Path Frame K utilizing the aerodynamic bank angle μ_A together with the aerodynamic and kinematic course angle χ_A and χ_K as well as the aerodynamic and kinematic inclination angle γ_A and γ_K (see Fig. 24). If the kinematic bank angle μ_K had been chosen, this would not have been possible since in this case the aerodynamic bank angle μ_A could not be restored. On the other hand, using the aerodynamic bank angle μ_A together with the commanded attitude angles $\alpha_{A,CMD}$ and $\beta_{A,CMD}$, even the kinematic angle of attack α_K and the kinematic sideslip angle β_K can be calculated making use of the equations given in chapter 3.3.8. Thus, finally the kinematic bank angle μ_K can be computed and the circle depicted in Fig. 24 gets closed.

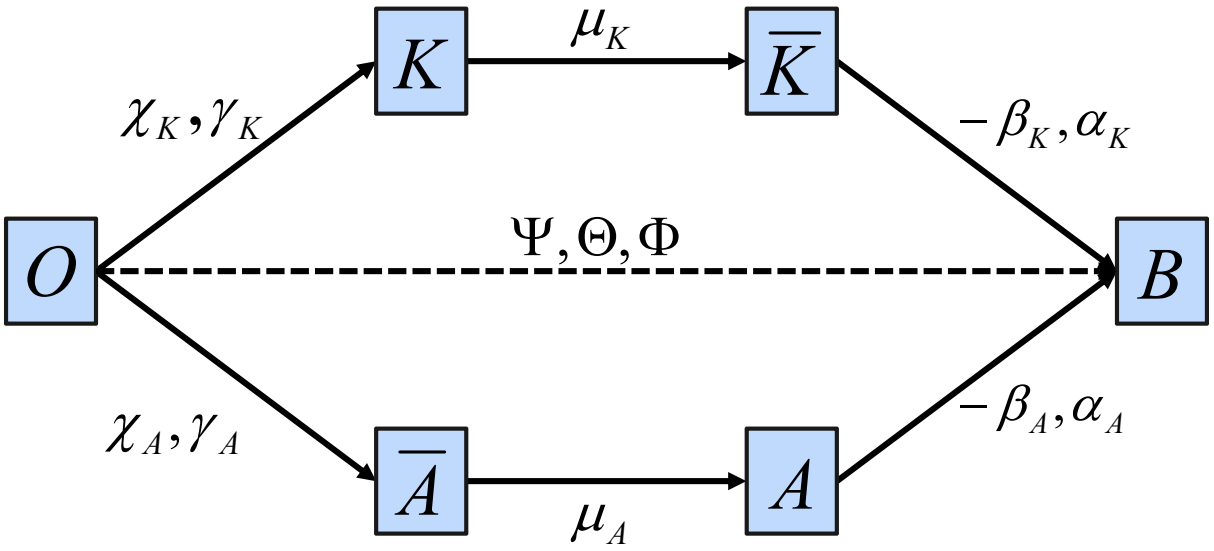

Figure 24. Coordinate Systems

3.6.2 Hybrid Simulation Mode with Linear Transfer Functions

Since the aerodynamic angle of attack α_A, the aerodynamic angle of sideslip β_A and the first order time derivative of the aerodynamic bank angle $\dot{\mu}_A$ are the controls of the pure point-mass simulation models, arbitrary time histories for these control signals could be commanded and these time histories do not necessarily take into account the real dynamic order of the flight system. E.g. in reality there cannot occur any discrete jumps in the aerodynamic angle of attack α_A nor in the first order time derivative of the kinematic bank angle $\dot{\mu}_A$ and also the time rate of change of these signals is limited due to the inherent dynamics of the real flight vehicle. One approach to come closer to reality and to generate more realistic time histories for the angle of attack α_A or the bank angle μ_A is the utilization of second order linear transfer functions for the load factor build-up respectively first order linear transfer functions for the build-up of the first order time derivative of the flight-path bank angle μ_A as depicted in Fig. 25.

In comparison to the pure point-mass simulation mode the dynamic order of the flight system dynamics is increased since the rotational and attitude dynamics of the flight system are represented by linear transfer functions for the load factors and the roll rate. If only the point-mass simulation mode, i.e. the outer loop without any inner loop is used for simulation, the load factors are the directly commanded values and therefore it would be possible that there are leaps in the load factor curves that cannot occur in reality. With the linear inner loop comprising transfer functions for the load-factors and the roll rate, the maximum achievable build-up rates of these quantities are limited and the progression of the load factor and roll rate curves become much more realistic.

Additionally to the states of the pure point-mass simulation model, the following states are added to the simulation model state vector: the load factors n_y and n_z, the first order time derivatives of these load factors $\dot{n}_y$ and $\dot{n}_z$ and the first order time derivative of the kinematic flight-path bank angle $\dot{\mu}_K$ respectively the roll rate p_K. Besides the commanded thrust lever position $\delta_{T,CMD}$, the controls are the attitude angles $\alpha_{A,CMD}$ and $\beta_{A,CMD}$ and the first order time derivative of the flight-path bank angle $\dot{\mu}_{A,CMD}$.

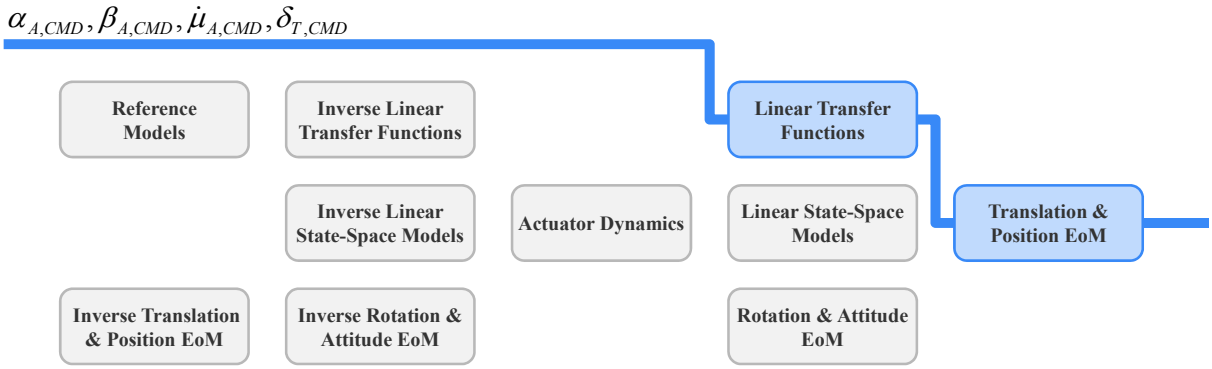

Figure 25. Hybrid Simulation Mode with Linear Transfer Functions

$$\mathbf{x} = \left[x, y, z, V_K, q_0, q_1, q_2, q_3, n_{y,A}, \dot{n}_{y,A}, n_{z,A}, \dot{n}_{z,A}, p_K^*, \delta_T\right]^T \tag{3.374}$$

$$\mathbf{u} = \left[\alpha_{A,CMD}, \beta_{A,CMD}, \dot{\mu}_{A,CMD}, \delta_{T,CMD}\right]^T \tag{3.375}$$

Here again, the first order time derivative of the aerodynamic bank angle $\dot{\mu}_A$ is the preferred control input since for the hybrid simulation mode with linear transfer functions in principle the same statements hold as for the pure point-mass simulation mode stated in the preceding chapter.

3.6.3 Hybrid Simulation Mode with State-Space Models

This hybrid simulation mode utilizes the point-mass simulation model supplemented by the linearized state-space models for the longitudinal and lateral motion of the aircraft (Fig. 26) to represent the rotational and attitude dynamics of the flight system in a more realistic manner than the linear transfer functions. Additionally, the actuator dynamics are incorporated in this simulation mode. Thus, the states for the simulation with the hybrid simulation mode with state space models are, in addition to the states of the pure point-mass simulation model, the kinematic attitude angles angle of attack α_K and sideslip angle β_K and the pseudo-roll rates p_K^*, q_K^* and r_K^*. The actuator dynamics introduce six extra states, namely the elevator position η, the aileron surface deflection ξ and the rudder position ζ as well as their respective first order time derivatives $\dot{\xi}$, $\dot{\eta}$ and $\dot{\zeta}$. The controls are the thrust lever position $\delta_{T,CMD}$ as for the pure point-mass simulation model augmented by the commanded deflections of the elevator η_{CMD}, the aileron ξ_{CMD} and the rudder ζ_{CMD}.

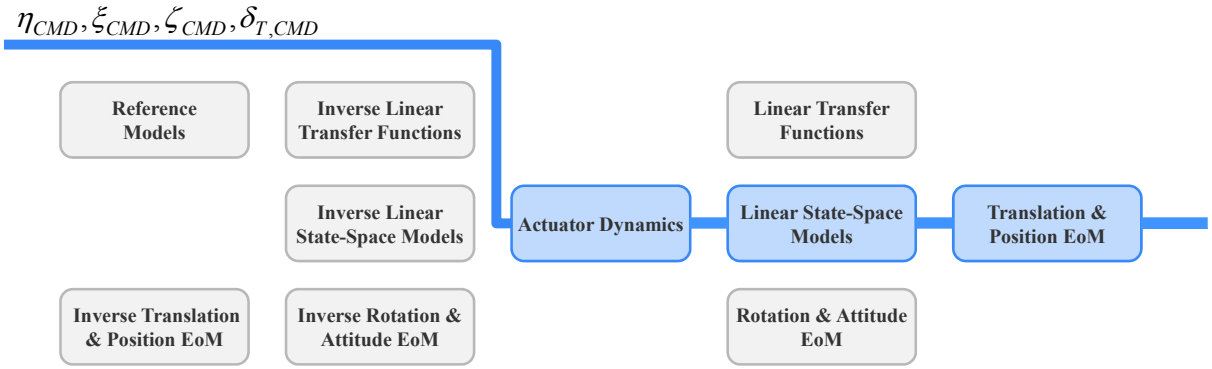

Figure 26. Hybrid Simulation Mode with Linear State-Space Models

$$\mathbf{x} = \left[x, y, z, V_K, \alpha_K, \beta_K, q_0, q_1, q_2, q_3, p_K^*, q_K^*, r_K^*, \delta_T, \eta, \dot{\eta}, \xi, \dot{\xi}, \zeta, \dot{\zeta}\right]^T \tag{3.376}$$

$$\mathbf{u} = \left[\eta_{CMD}, \xi_{CMD}, \zeta_{CMD}, \delta_{T,CMD}\right]^T \tag{3.377}$$

3.6.4 Full, Non-Linear Simulation Mode

The full, non-linear simulation mode illustrated in Fig. 27 is the simulation mode that comes closest to reality since the point-mass simulation model is augmented by the full non-linear rotational and attitude dynamics. Thus, the states related to this simulation mode are the angle of attack α_K, the sideslip angle β_K and the real body angular rates that are the roll rate p_K, the pitch rate q_K and the yaw rate r_K plus the states originating from the pure point-mass simulation model. As for the hybrid simulation model with linear state-space models, the actuator dynamics introduce the six supplementary states for the control surface deflections η, ξ, ζ and their first order time derivatives $\dot{\xi}$, $\dot{\eta}$ and $\dot{\zeta}$. The controls are also identical to the controls of the hybrid simulation model with linear state-space models and comprise the commanded thrust lever position $\delta_{T,CMD}$ as well as the commanded control surface deflections η_{CMD}, ξ_{CMD} and ζ_{CMD}.

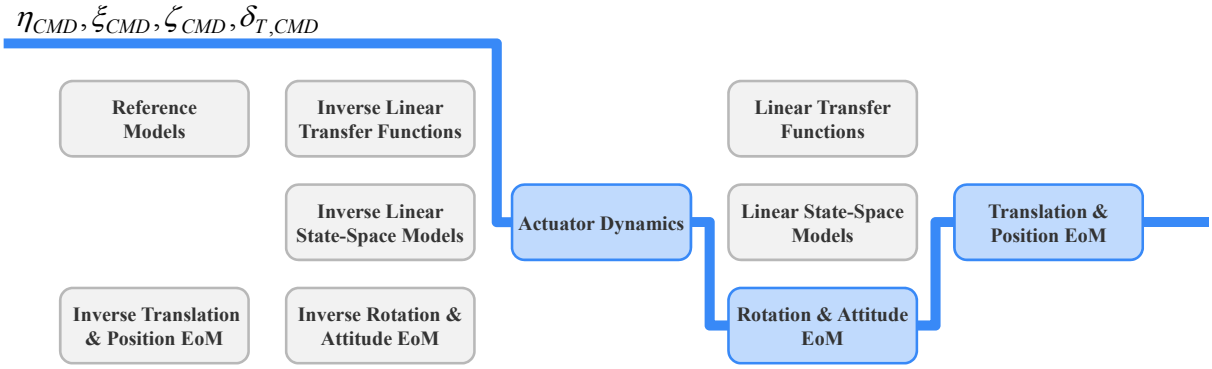

Figure 27. Full, Non-Linear Simulation Mode

$$\mathbf{x} = \left[x, y, z, V_K, \alpha_K, \beta_K, q_0, q_1, q_2, q_3, \vec{\omega}_K^{IB}, \delta_T, \eta, \dot{\eta}, \xi, \dot{\xi}, \zeta, \dot{\zeta}\right]^T \tag{3.378}$$

$$\mathbf{u} = \left[\eta_{CMD}, \xi_{CMD}, \zeta_{CMD}, \delta_{T,CMD}\right]^T \tag{3.379}$$

3.6.5 Hybrid Simulation Mode with State-Space Models, Inner Loop Inversion and Reference Models

This simulation mode expands the hybrid simulation mode with state-space models introduced in chapter 3.6.3 by the inversion of the linearized state-space models and reference models for the controls that are now the aerodynamic angle of attack α_A, the aerodynamic sideslip angle β_A and the first order time derivative of the aerodynamic flight-path bank angle $\dot{\mu}_A$. If only the inversion controller for the inner loop is used, the hybrid simulation model with linear state-space models can be simulated utilizing the same controls as the pure non-linear point-mass model without inner loop that is to say the angle of attack α_A, the sideslip angle β_A, the first order time derivative of the aerodynamic flight-path bank angle $\dot{\mu}_A$ and the thrust lever position δ_T.

Additionally to the states of the hybrid simulation mode with state-space models listed in chapter 3.6.3, the simulation model state vector is augmented by the reference models states

that are the reference values for the aerodynamic attitude angles $\alpha_{A,REF}$, $\beta_{A,REF}$ and $\mu_{A,REF}$ plus their first order derivatives with respect to time $\dot{\alpha}_{A,REF}$, $\dot{\beta}_{A,REF}$ and $\dot{\mu}_{A,REF}$.

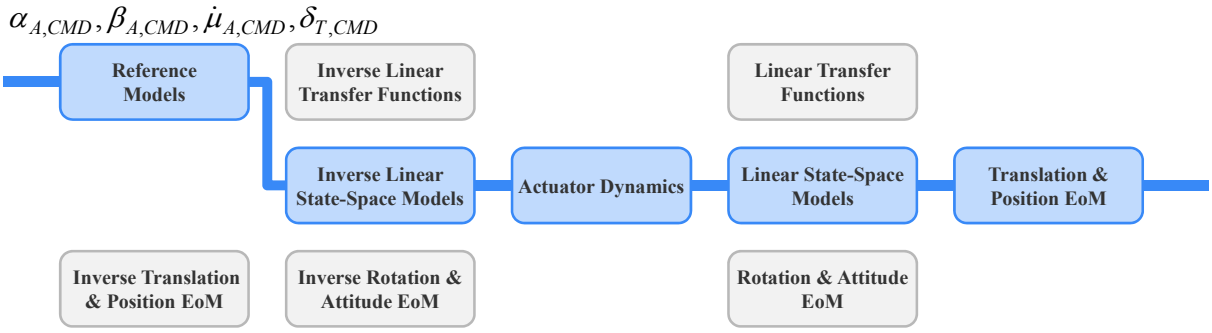

Figure 28. Hybrid Simulation Mode with State-Space Model, Inner Loop Inversion and Reference Models

$$\mathbf{x} = \left[x, y, z, V_K, \alpha_K, \beta_K, q_0, q_1, q_2, q_3, p_K^*, q_K^*, r_K^*, \delta_T, \eta, \dot{\eta}, \xi, \dot{\xi}, \zeta, \dot{\zeta}, \alpha_A, \dot{\alpha}_A, \beta_A, \dot{\beta}_A, \mu_A\right]^T \tag{3.380}$$

$$\mathbf{u} = \left[\alpha_{A,CMD}, \beta_{A,CMD}, \dot{\mu}_{A,CMD}, \delta_{T,CMD}\right]^T \tag{3.381}$$

3.6.6 Full, Non-Linear Simulation Mode with Inner Loop Inversion and Reference Models

Analogous to the hybrid simulation mode depicted in chapter 3.6.5 with linear state-space models and the corresponding inner loop inversion and reference models, this simulation mode (Fig. 29) augments the full non-linear simulation mode illustrated in chapter 3.6.4 by the dynamic inversion subsystems of the non-linear inner loop and the respective reference models for the command signals that again are the aerodynamic angle of attack α_A, the aerodynamic sideslip angle β_A and the first order time derivative of the aerodynamic flight-path bank angle $\dot{\mu}_A$. In addition to the states incorporated in the full, non-linear simulation mode of chapter 3.6.4, six extra states in association with the reference models are added that are the aerodynamic attitude angles $\alpha_{A,REF}$, $\beta_{A,REF}$ and $\mu_{A,REF}$ as well as their first order time derivatives $\dot{\alpha}_{A,REF}$, $\dot{\beta}_{A,REF}$ and $\dot{\mu}_{A,REF}$.

By this simulation mode where only the inversion controller for the non-linear inner loop is taken into account, one can simulate the full, non-linear 6-Degree-of-Freedom simulation model using the same controls as for the non-linear point-mass model without any inner loop respectively the hybrid simulation model depicted in chapter 3.6.5 with linear state-space models and the corresponding inversion controllers for the state-space models.

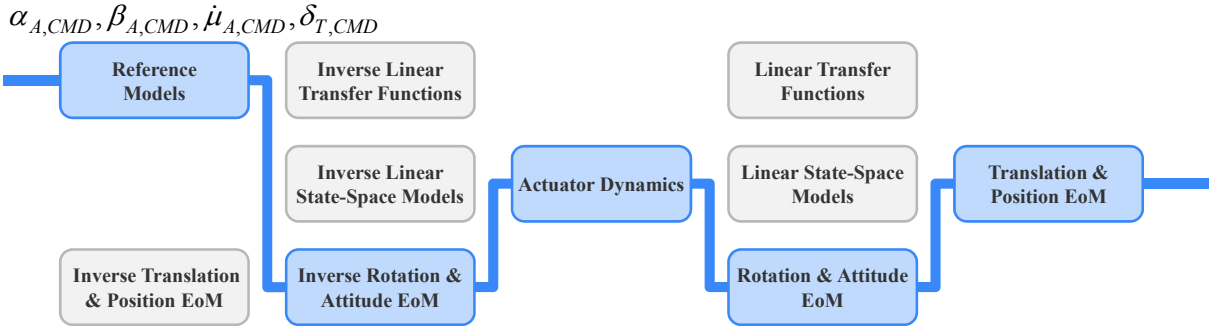

Figure 29. Non-Linear Simulation Mode with Inversion and Reference Models

$$\mathbf{x} = \left[x, y, z, V_K, \alpha_K, \beta_K, q_0, q_1, q_2, q_3, \vec{\omega}_K^{IB}, \delta_T, \eta, \dot{\eta}, \xi, \dot{\xi}, \zeta, \dot{\zeta}, \alpha_A, \dot{\alpha}_A, \beta_A, \dot{\beta}_A, \mu_A\right]^T \tag{3.382}$$

$$\mathbf{u} = \left[\alpha_{A,CMD}, \beta_{A,CMD}, \dot{\mu}_{A,CMD}, \delta_{T,CMD}\right]^T \tag{3.383}$$

3.6.7 Trajectory Following Mode with Linear Transfer Functions

The trajectory following mode with linear transfer functions depicted in Fig. 30 comprises the point-mass simulation model with linear transfer functions for the inner loop, the actuator dynamics models and the inverse simulation models for the point-mass model and the linear transfer functions. Thus, the state vector consists of the point-mass model states plus five states caused by the linear transfer functions that are the load factors n_y and n_z, their first order time derivatives $\dot{n}_y$ and $\dot{n}_z$ and the roll rate p_K. The reference trajectory has to be given in parameterized vector form and has to be at least four times continuously differentiable.

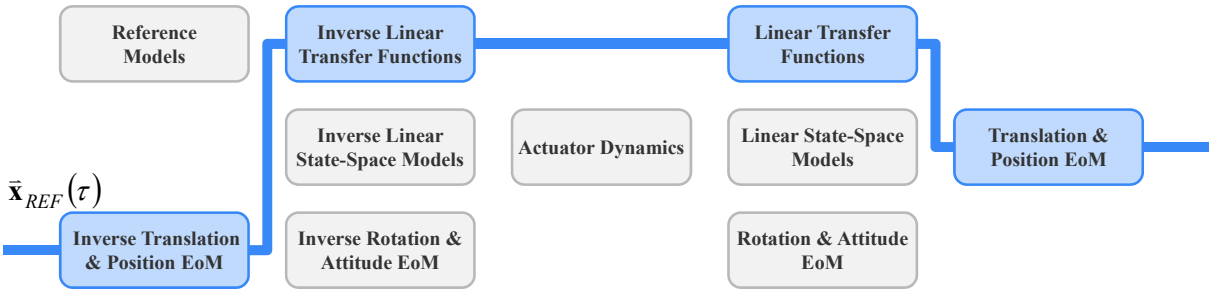

Figure 30. Trajectory Following Mode with Linear Transfer Functions

$$\mathbf{x} = \left[x, y, z, V_K, q_0, q_1, q_2, q_3, n_{y,A}, \dot{n}_{y,A}, n_{z,A}, \dot{n}_{z,A}, p_K^*, \delta_T\right]^T \tag{3.384}$$

$$\mathbf{u} = \vec{\mathbf{x}}_{REF}(\tau) \tag{3.385}$$

3.6.8 Trajectory Following Mode with State-Space Models

The trajectory following mode with state-space models (Fig. 31) is the same as the trajectory following mode with linear transfer functions except for the inner loop and the inverted inner loop that are now simulated by the linearized state-space models respectively their dynamic inversion subsystems. Thus, instead of the states associated with the linear transfer functions, the state vector consists of the states for the linear state-space models that are the kinematic angle of attack α_K, the kinematic sideslip angle β_K and the pseudo-roll rates p_K^*, q_K^* and r_K^* plus six states brought by the incorporated actuator dynamics, namely the control surface deflections η, ξ, ζ and their first order time derivatives $\dot{\xi}$, $\dot{\eta}$ and $\dot{\zeta}$. This means that a more realistic reproduction of the rotational and attitude dynamics is taken into account by this trajectory following mode than it is the case for the trajectory following mode described in chapter 3.6.7.

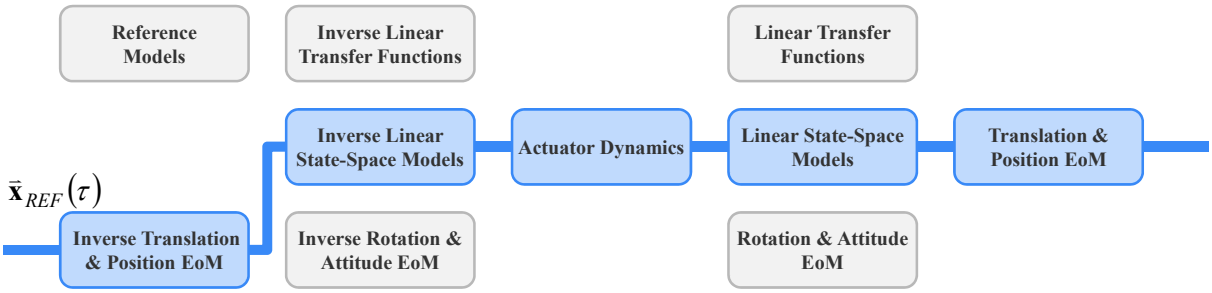

Figure 31. Trajectory Following Mode with State-Space Models

$$\mathbf{x} = \left[x, y, z, V_K, \alpha_K, \beta_K, q_0, q_1, q_2, q_3, p_K^*, q_K^*, r_K^*, \delta_T, \eta, \dot{\eta}, \xi, \dot{\xi}, \zeta, \dot{\zeta}\right]^T \tag{3.386}$$

$$\mathbf{u} = \bar{\mathbf{x}}_{REF}(\tau) \tag{3.387}$$

3.6.9 Trajectory Following Mode with Full, Non-Linear Inner Loop

The most accurate and realistic trajectory following mode is the mode with the full, non-linear inner loop shown in Fig. 32. Instead of linear transfer functions or linear state-space models as in the preceding two trajectory following modes this mode utilizes the full non-linear inner loop and the respective inverse simulation model equations and thus allows for a representation of the rotational and attitude dynamics in the most realist manner. Additionally to the states of the point-mass simulation model and the states of the actuator dynamics, this mode adds the states of the full non-linear inner loop that are the kinematic angle of attack α_K, the kinematic sideslip angle β_K, the roll rate p_K, the pitch rate q_K and the yaw rate r_K to the simulation model state vector.

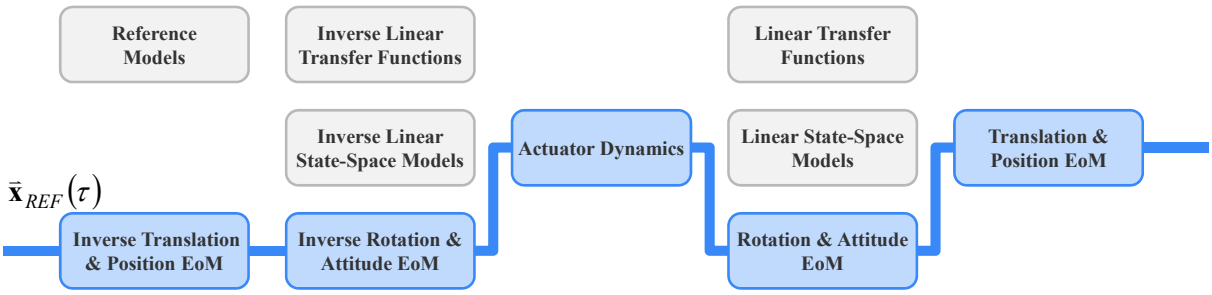

Figure 32. Trajectory Following Mode with Non-Linear Inner Loop

$$\mathbf{x} = \left[x, y, z, V_K, \alpha_K, \beta_K, q_0, q_1, q_2, q_3, \bar{\boldsymbol{\omega}}_K^{IB}, \delta_T, \eta, \dot{\eta}, \xi, \dot{\xi}, \zeta, \dot{\zeta}\right]^T \tag{3.388}$$

$$\mathbf{u} = \bar{\mathbf{x}}_{REF}(\tau) \tag{3.389}$$

4

Optimization Algorithm

4.1 Algorithm Abstract

In this chapter, a newly developed optimization algorithm is introduced that allows for the generation of robust and suitable initial guesses for the optimization of aircraft trajectories with underlying simulation models of varying depths of modeling fidelity. The optimization algorithm is based on the scalable, multi-fidelity simulation model presented in detail in chapter 3 as well as the various simulations modes given therein. In the following, the optimization algorithm is outlined and for the various optimization steps of the algorithm, the general optimization problem stated in chapter 2 is rendered more precisely. Furthermore, a slightly modified algorithm is illustrated, resulting in the same final result as the original algorithm that is an optimal trajectory for a full, non-linear 6-DoF simulation model. The various steps performed during the optimization procedure are depicted in Fig. 33.

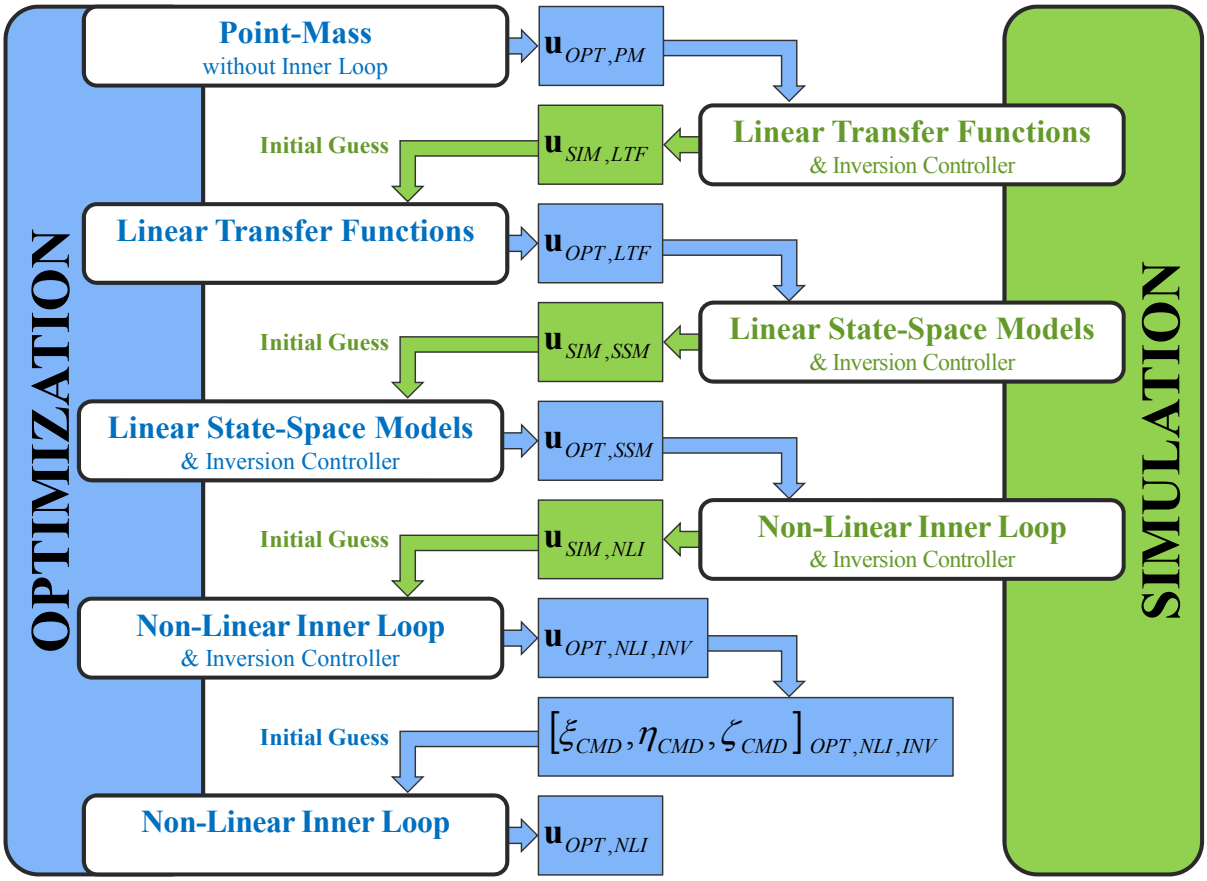

Figure 33. Optimization Algorithm I

The algorithm is initialized with a given trajectory that is feasible for the point-mass simulation model, i.e. the point-mass simulation mode given in chapter 3.6.1 is utilized first. A procedure for generating feasible trajectories that are already close to the optimal trajectories when multiple waypoints have to be passed is depicted in detail in chapter 4.2.2. Utilizing the feasible trajectory as initial guess, the optimal trajectory for the point-mass simulation model is computed that is step **1A** of the optimization algorithm. Therefore, the state vector respectively the control vector of the optimization problem stated in chapter 2.2 are:

$$\mathbf{x} = \left[x, y, z, V_K, \chi_K, \gamma_K, \mu_K, \delta_T\right]^T \tag{4.1}$$

$$\mathbf{u} = \left[\alpha_{A,CMD}, \beta_{A,CMD}, \dot{\mu}_{A,CMD}, \delta_{T,CMD}\right]^T \tag{4.2}$$

Once the optimal trajectory for the point-mass model has been found, the modeling complexity of the simulation model is increased by incorporating the linear inner loop with linear transfer functions for the load factors and the roll rate, thus making use of the simulation mode given in chapter 3.6.2. As can be seen from Eq. (3.375) in chapter 3.6.2, the same command inputs as for the point-mass simulation model are required, thus the trajectory can easily be simulated utilizing the hybrid simulation mode with linear transfer functions where the controls are now the input signals to the linear transfer functions in the inner loop. Within the optimization algorithm, this simulation of the aircraft trajectory by the hybrid simulation mode with linear transfer functions constitutes step **1B**.

By simulating the trajectory utilizing the command inputs that have been optimal for the point-mass model without any inner loop, the resulting simulated trajectory will deviate from the trajectory that has been optimal utilizing the point-mass simulation mode because of the increased dynamic order of the simulation model. Without any inner loop, step inputs for the aerodynamic angle of attack α_A and the aerodynamic sideslip angle β_A or the first order time derivative of the aerodynamic bank angle $\dot{\mu}_A$ are directly forwarded to the point-mass simulation model, while now with an inner loop featuring linear transfer functions the respective step inputs are delayed by first or second order linear transfer functions. Thus, the reaction of the point-mass model to changes in the control inputs is also delayed. The simulated flight system incorporating linear transfer functions will not be able to follow the trajectory that has been computed for the point-mass model if the same control time histories are fed to the simulation model. Hence, for the simulation task the inversion controllers for the outer loop (chapter 3.3) together with the error feedbacks outlined in chapter 3.5 are utilized to force the simulated flight system with linear transfer functions onto the optimal trajectory found for the model without any inner loop, resulting in the control inputs $\mathbf{u}_{SIM,LTF}$ for the hybrid simulation model with linear transfer functions (see Fig. 33). At this, the reference trajectory $\vec{\mathbf{r}}_{REF} = \vec{\mathbf{r}}_{OPT,PM}$ is only given up to its second order time derivative since the control inputs of the point-mass simulation model in step **1A** are approximated linearly. Thus, the point-mass simulation model allows only for the computation of smooth second order time derivatives for the position vector. The first order time derivatives of the position vector are obtained automatically by the position propagation equations. Consequently, the inversion controllers and the error feedbacks can also only be utilized up to the respective derivative order, i.e. the second order time derivatives in the trajectory control loop and the first order time derivatives in the kinematic flight-path control loop.

Alternatively, the trajectory following mode with linear transfer functions of chapter 3.6.7 can be used to generate the commands $\mathbf{u}_{SIM,LTF}$. Then, a smooth and continuously differentiable reference trajectory up to its fourth order time derivative has to be given that is usually not the case regarding the trajectory that results from the optimization using the point-mass simulation model. Thus, an approach to generate a reference trajectory with smooth fourth order time derivatives from the reference trajectory that is only twice continuously differentiable has to be established. One possibility is to fit the given reference trajectory by *B*-splines of sufficient order. Then, the newly established reference trajectory is sufficiently smooth and the required derivatives can easily be calculated from the *B*-spline curve fit. Another possibility to generate the required reference trajectory up to its fourth order time derivative is provided in chapter 4.2.3. There, a substitute optimization problem is established to smoothen the given reference trajectory. Besides the reference trajectory itself, also the reference time history for the kinematic flight-path bank angle μ_K (that is set equal to the aerodynamic flight-path bank angle μ_A for the point-mass model) has to be considered. The time history for the bank angle that is output by the point-mass simulation model is only continuously differentiable once, while for the error feedbacks stated in chapter 3.5 a reference time history is required that is at least twice continuously differentiable. Thus, the reference time history for the flight-path bank angle μ_K has also to be fitted by *B*-splines. Alternatively, it can also be treated by the substitute optimization problem given in chapter 4.2.3.

Of course, the control histories $\mathbf{u}_{SIM,LTF}$ resulting from the simulation step **1B** are unlikely to be optimal since the cost function has not been minimized by optimization, but an initial guess for the optimization task utilizing the hybrid simulation mode with linear transfer functions has been generated that fulfills all given boundary conditions to a wide extent and that might already come close to an optimal solution based on this hybrid simulation model. This optimization task represents the next step in the optimization algorithm (step **2A**), where the corresponding state and control vector within the general optimization problem are:

$$\mathbf{x} = \left[x, y, z, V_K, \chi_K, \gamma_K, \mu_K, n_{y,A}, \dot{n}_{y,A}, n_{z,A}, \dot{n}_{z,A}, p_K^*, \delta_T\right]^T \tag{4.3}$$

$$\mathbf{u} = \left[\alpha_{A,CMD}, \beta_{A,CMD}, \dot{\mu}_{A,CMD}, \delta_{T,CMD}\right]^T \tag{4.4}$$

When the optimal solution for a trajectory utilizing the simulation model with the attitude and rotational dynamics represented by linear transfer functions has been computed, the obtained time histories for the position vector $\vec{\mathbf{r}}_{OPT,LTF}$ can then be used as reference trajectory $\vec{\mathbf{r}}_{REF}$ for simulating the trajectory utilizing the simulation mode of chapter 3.6.8. At this, a simulation model is taken into account that features an inner loop with linear state-space models augmented by the appropriate inversion controllers. This simulation task represents step **2B** of the optimization algorithm depicted in Fig. 33. In step **2A** of the optimization algorithm the point-mass simulation model has been augmented by linear transfer functions, while the control inputs to the augmented simulation models are linearly interpolated. Thus, due to the increased dynamic order of the simulation model in step **2A,** a reference trajectory $\vec{\mathbf{r}}_{REF} = \vec{\mathbf{r}}_{OPT,LTF}$ can be computed that features smooth time derivatives up to the fourth order. Again, the simulated trajectory is likely to deviate from the optimal trajectory found for the model with linear transfer functions since now linear state-space models are incorporated in the simulation model. Thus, the modeling fidelity is increased and in contrast to the optimization based on linear transfer functions in the inner loop, the coupling between the states in the

longitudinal motion respectively the coupling between the states in the lateral motion is now incorporated in the simulation model, causing the simulated trajectory to deviate from the trajectory that has been optimal for the hybrid simulation mode incorporating linear transfer functions. Due to this reason, the error feedbacks given in chapter 3.5 are used to force the aircraft model with linear state-space models in the inner loop back onto the trajectory that has been optimal for the model with linear transfer functions in the inner loop. To sum up, in step **2B** the hybrid simulation model with linear state-space models in the inner loop tries to follow the trajectory that has been obtained by the optimization using the hybrid simulation model with linear transfer functions. Finally, the control time histories $\mathbf{u}_{SIM,SSM}$ that are the input to the next step of the optimization algorithm have to be restored by appropriate inverse reference models (see chapter 4.2.1).

Here again it has to be stated that the control time histories $\mathbf{u}_{SIM,SSM}$ are not assumed to be optimal since they were generated by simulation and not by optimization, but they provide a well suitable initial guess for the next step in the optimization algorithm that is step **3A**. Within this step, the time histories $\mathbf{u}_{SIM,SSM}$ found by simulating the trajectory using the model with linear state-space models, inversion controller and error feedbacks in turn are used as initial guess for the optimization of the aircraft trajectory utilizing the simulation mode with linear state-space models supplemented by the appropriate inversion controller (see chapter 3.6.5) so that the control and state vector with regard to the general optimization problem read:

$$\begin{aligned}\mathbf{x} = \big[&x, y, z, V_K, \alpha_K, \beta_K, \chi_K, \gamma_K, \mu_K, p_K^*, q_K^*, r_K^*, \delta_T, \\ &\eta, \dot{\eta}, \xi, \dot{\xi}, \zeta, \dot{\zeta}, \alpha_A, \dot{\alpha}_A, \beta_A, \dot{\beta}_A, \mu_A, \dot{\mu}_A\big]^T\end{aligned} \tag{4.5}$$

$$\mathbf{u} = \left[\alpha_{A,CMD}, \beta_{A,CMD}, \dot{\mu}_{A,CMD}, \delta_{T,CMD}\right]^T \tag{4.6}$$

Stepping forward in the optimization procedure, in step **3B** the position vector $\vec{\mathbf{r}}_{OPT,SSM}$ being optimal for the simulation model with the state-space models and the inversion controller is then used as reference trajectory $\vec{\mathbf{r}}_{REF}$ for a simulation based on the model with the non-linear inner loop combined with the respective inversion controllers (see chapter 3.6.9). Due to its dynamic order, the hybrid simulation model with linear state-space models used in step 3A of the optimization algorithm allows for a direct computation of a reference trajectory $\vec{\mathbf{r}}_{REF} = \vec{\mathbf{r}}_{OPT,SSM}$ up to the fourth order time derivative. As before, due to the increased depth of modeling, deviations from the trajectory that has been optimal for the simulation model with state-space models and inversion controller in the inner loop will occur. These deviations are corrected by the appropriate error feedbacks, see chapter 3.5. Thus, in step **3B** the simulation model with the nonlinear inner loop attempts to follow the trajectory that is the outcome of the optimization with the hybrid simulation model with linear state-space models in the inner loop (step **3A**). Again, the controls $\mathbf{u}_{SIM,NLI}$ have to be restored utilizing appropriate inverse reference models (chapter 4.2.1). In the sequel, the controls $\mathbf{u}_{SIM,NLI}$ constitute the input to step **4A** of the optimization algorithm.

As before, the controls $\mathbf{u}_{SIM,NLI}$ might not be optimal for the increased modeling fidelity but result in a sub-optimal trajectory that obeys all boundary conditions to a great extent. Thus, the controls $\mathbf{u}_{SIM,NLI}$ represent a good and well-suited initial guess for the next step in the optimization algorithm that is step **4A**. Here, the optimization task is accomplished based on the simulation mode with the full, non-linear rotation and attitude dynamics inclusively

inversion controller given in chapter 3.6.6. The control respectively the state vector of the general optimization problem stated in the preceding chapter are then modified to:

$$\mathbf{x} = \left[x, y, z, V_K, \alpha_K, \beta_K, \chi_K, \gamma_K, \mu_K, \bar{\boldsymbol{\omega}}_K^{IB}, \delta_T, \eta, \dot{\eta}, \xi, \dot{\xi}, \zeta, \dot{\zeta}, \alpha_A, \dot{\alpha}_A, \beta_A, \dot{\beta}_A, \mu_A, \dot{\mu}_A\right]^T \tag{4.7}$$

$$\mathbf{u} = \left[\alpha_{A,CMD}, \beta_{A,CMD}, \dot{\mu}_{A,CMD}, \delta_{T,CMD}\right]^T \tag{4.8}$$

As can be seen from Eq. (4.7), computing the optimal trajectory respectively the optimal controls $\mathbf{u}_{OPT,NLI,INV}$ for this level of the simulation model in turn yields time histories for the control surface deflections η, ξ and ζ. Finally, in the last step **5A** of the optimization algorithm these time histories for the control surface deflections are utilized as quite good initial guesses for the optimization of an aircraft trajectory based on a full 6-degree of freedom simulation model with non-linear attitude and rotational dynamics but without any inversion controller so that the control surface deflections are the directly commanded control inputs, where the simulation mode dedicated to this optimization task can be found in chapter 3.6.4. Regarding the general optimization problem, the control vector and the state vector for this last step of the optimization algorithm comprise the following quantities:

$$\mathbf{x} = \left[x, y, z, V_K, \alpha_K, \beta_K, \chi_K, \gamma_K, \mu_K, \bar{\boldsymbol{\omega}}_K^{IB}, \delta_T, \eta, \dot{\eta}, \xi, \dot{\xi}, \zeta, \dot{\zeta}\right]^T \tag{4.9}$$

$$\mathbf{u} = \left[\eta_{CMD}, \xi_{CMD}, \zeta_{CMD}, \delta_{T,CMD}\right]^T \tag{4.10}$$

Thus, the final output of the optimization algorithm are the controls $\mathbf{u}_{OPT,NLI}$ that give the optimal aircraft trajectory based on a full, non-linear 6-degree of freedom simulation model. In Table 2, a summary of the various steps that are performed during the optimization algorithm described so far is given.

ALGORITHM I	
1A	**Optimization** using point-mass simulation mode
1B	**Simulation** using hybrid simulation mode with linear transfer functions
2A	**Optimization** using hybrid simulation mode with linear transfer functions
2B	**Simulation** using hybrid simulation mode with linear state-space models
3A	**Optimization** using hybrid simulation mode with linear state-space models, inversion controller and reference models
3B	**Simulation** using full simulation mode with non-linear inner loop
4A	**Optimization** using full 6-DoF simulation mode with non-linear inner loop, inversion controller and reference models
5A	**Optimization** using full 6-DoF simulation mode with non-linear inner loop

Table 2. Optimization Algorithm I

Furthermore, an alternative optimization algorithm is developed that is a slightly modified version of algorithm **I**. A summary of the steps performed therein can be found in table 3, the optimization algorithm itself is illustrated in Fig. 34. Up to step **3A**, the alternative

optimization algorithm proceeds in exactly the same way as algorithm **I**. As can be noticed from Eq. (4.5), performing the optimization task in step **3A** also produces time histories for the control surface deflections η, ξ and ζ. Instead of simulating the aircraft trajectory utilizing the full, 6-DoF simulation mode with the non-linear inner loop, the appropriate inversion controller and reference models that would have been step **3B** of algorithm **I**, the alternative algorithm **II** proceeds by optimizing the aircraft trajectory making use of the simulation mode given in chapter 3.6.3. This simulation mode is a hybrid simulation mode with linear state-space models in the inner loop, but without any inversion controller or reference models. In the optimization task of step **4A** of the modified algorithm, the state time histories for the control surface deflections η, ξ and ζ being the outcome of step **3A** are used as initial guess, and the control and state vector of the general optimization problem are set to:

$$\mathbf{x} = \left[x, y, z, V_K, \alpha_K, \beta_K, \chi_K, \gamma_K, \mu_K, p_K^*, q_K^*, r_K^*, \delta_T, \eta, \dot{\eta}, \xi, \dot{\xi}, \zeta, \dot{\zeta}\right]^T \tag{4.11}$$

$$\mathbf{u} = \left[\eta_{CMD}, \xi_{CMD}, \zeta_{CMD}, \delta_{T,CMD}\right]^T \tag{4.12}$$

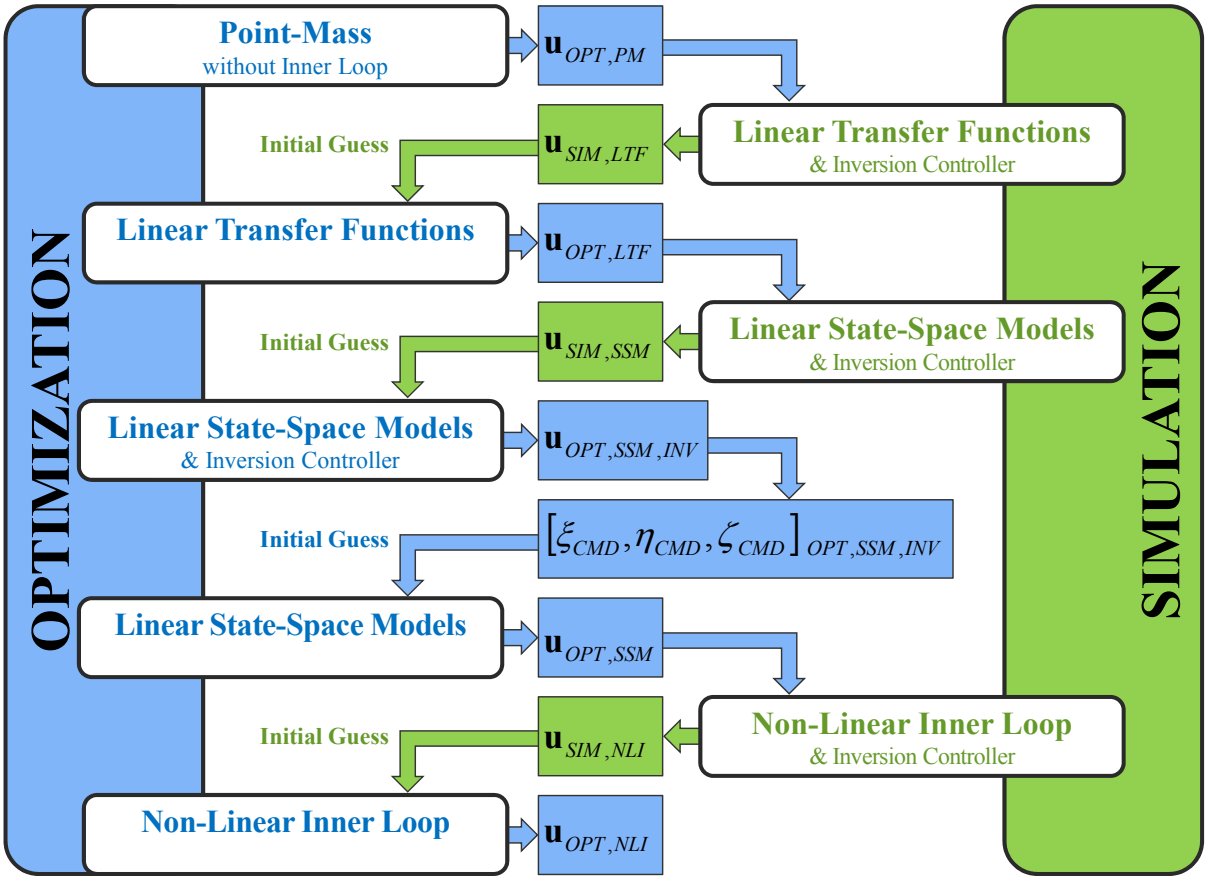

Figure 34. Optimization Algorithm II

Then, in step **4B** of optimization algorithm **II**, the trajectory following mode of chapter 3.6.9 is utilized that involves the 6-Degree-of-Freedom simulation model with the full, non-linear inner loop. The simulation model must follow the trajectory that has been obtained by the optimization in the preceding step **4A** of the optimization algorithm. As depicted above, step **4A** is based on the hybrid simulation model with linear state-space models in the inner loop. Here again the statement holds that due to the increased depth of modeling deviations from the aircraft trajectory that has been optimal for the optimization based on the simulation mode with the linear state-space models will occur, so that the error feedbacks given in

chapter 3.5 are incorporated in the simulation model to cancel out these deviations. By the simulation in step **4B** of optimization algorithm **II**, the control time histories $\mathbf{u}_{SIM,NLI}$ result (see Fig. 34). The control vector **u** is given by the control surface deflections η, ξ and ζ and any deviation from the reference trajectory is directly fed back to these control quantities. Finally, the time histories $\mathbf{u}_{SIM,NLI}$ for the controls provide a suitable initial guess to perform the last step of the optimization algorithm **II**, step **5A**, that again is identical to the last step of algorithm **I** and that optimizes the aircraft trajectory based on the full, non-linear 6-DoF simulation model of chapter 3.6.4.

ALGORITHM II	
1A	**Optimization** using point-mass simulation mode
1B	**Simulation** using hybrid simulation mode with linear transfer functions
2A	**Optimization** using hybrid simulation mode with linear transfer functions
2B	**Simulation** using hybrid simulation mode with linear state-space models
3A	**Optimization** using hybrid simulation mode with linear state-space models, inversion controller and reference models
4A	**Optimization** using hybrid simulation mode with linear state-space models
4B	**Simulation** using full 6-DoF simulation mode with non-linear inner loop
5A	**Optimization** using full 6-DoF simulation mode with non-linear inner loop

Table 3. Optimization Algorithm II

4.2 Implementation Details

4.2.1 Inverse Reference Models

Within the optimization algorithms of chapter 4.1, corrected reference values up to their second order time derivatives for the aerodynamic attitude angles that are the angle of attack α_A, the angle of sideslip β_A and the bank angle μ_A are obtained after simulating the respective trajectory in step **1B**, **2B** or **3B** of algorithm **I** or in step **1B** or **2B** of algorithm **II**. In the successive optimization tasks, corrected command values for the attitude angles are required as control inputs to the respective reference models, producing just the same reference values for the attitude angles. The command time histories can be restored by making use of the appropriate inverted reference models. For example, the inversion of the second order reference model for the aerodynamic angle of attack α_A reads:

$$\begin{aligned} \alpha^G_{A,CMD} &= \frac{1}{\omega_0^2}\left(s^2 + 2\cdot\zeta\cdot\omega_0\cdot s + \omega_0^2\right)\cdot\alpha^G_{A,REF} \\ &= \frac{1}{\omega_0^2}\left(\ddot{\alpha}^G_{A,REF} + 2\cdot\zeta\cdot\omega_0\cdot\dot{\alpha}^G_{A,REF} + \omega_0^2\cdot\alpha^G_{A,REF}\right) \end{aligned} \tag{4.13}$$

generating the required command time history $\alpha_{A,CMD}$ that in turn is used as initial guess for the optimization task in the subsequent step of the optimization algorithm. The computation

of the first and second order time derivatives of the angle of attack α_A is given by Eqs. (3.258) and (3.259).

4.2.2 Initial Guess Generation

In this chapter, an algorithm is illustrated that allows for an automatic generation of initial guesses respectively feasible trajectories for the optimization algorithms outlined in the preceding chapter when multiple waypoints have to be passed at prescribed kinematic course angles. In addition, besides the position coordinates and the course angle, further quantities can be prescribed at the respective waypoints like e.g. speed values, bank angles, etc. In the following, the initial guess generation is illustrated for a point-mass simulation model, but the extension of the proposed algorithm to simulation models of higher fidelity is straightforward.

First of all, the trajectory optimization problem is split up into multiple phases where each single phase is defined as the flight path segment between two succeeding waypoints. Thus, the initial and final boundary conditions for each phase are defined by the position coordinates $\vec{\mathbf{r}} = [x, y, z]^T$ of the respective waypoints, the initial and final course angles χ_K and the supplementary quantities specified at the considered waypoints. The algorithm then starts by computing the optimal solution for the first phase, using a homotopy procedure that is illustrated in Fig. 35. At first, an auxiliary waypoint is introduced and the optimization problem is solved for a flight from the initial waypoint of the first phase to the newly defined auxiliary waypoint. To initialize the homotopy procedure, the auxiliary waypoint is specified as a waypoint at the same altitude as the initial waypoint with a certain distance d from the initial waypoint in the direction of the initial course angle:

$$x_{AUX,ini} = x_{WPI} + d \cdot \cos \chi_{K,WPI} \tag{4.14}$$

$$y_{AUX,ini} = y_{WPI} + d \cdot \sin \chi_{K,WPI} \tag{4.15}$$

$$z_{AUX,ini} = z_{WPI} \tag{4.16}$$

where the subscript WPI denotes the initial waypoint and the subscript AUX the auxiliary waypoint. The course angle χ_K and the kinematic velocity V_K at the auxiliary waypoint are set to the same values that are prescribed at the initial waypoint:

$$\chi_{K,AUX,ini} = \chi_{K,WPI} \tag{4.17}$$

$$V_{K,AUX,ini} = V_{K,WPI} \tag{4.18}$$

If no value for the kinematic velocity V_K at the initial waypoint of the first phase is specified at all, it is initially set to the intermediate value of the lower and the upper bound for the kinematic velocity. In this case, the initial boundary constraint for the kinematic velocity V_K is freed and the determination of the kinematic velocity $V_{K,WPI}$ at the initial waypoint is left to the optimization. Additionally, the values for the kinematic flight-path bank angle μ_K and the kinematic flight-path inclination angle γ_K at the initial waypoint and at the auxiliary waypoint are initially set to zero:

$$\gamma_{K,AUX,ini} = \gamma_{K,WPI,ini} = 0° \tag{4.19}$$

$$\mu_{K,AUX,ini} = \mu_{K,WPI,ini} = 0° \tag{4.20}$$

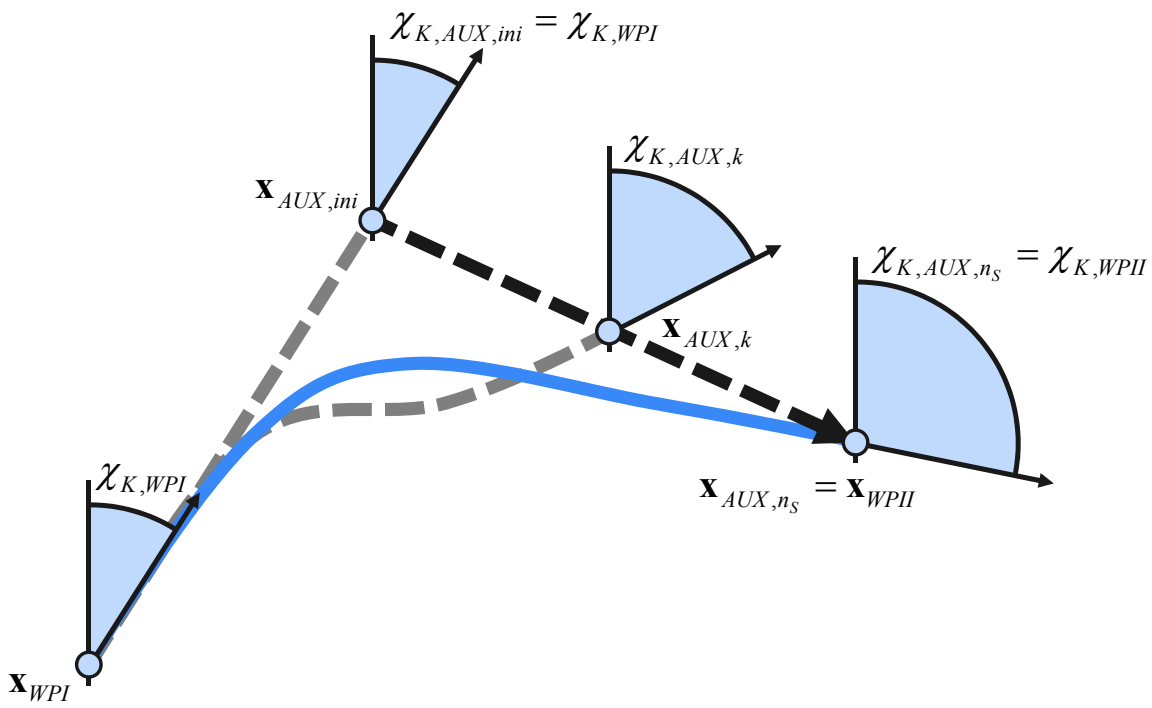

Figure 35. Homotopy Procedure

This means that the initial waypoint of the first phase and the auxiliary waypoint are positioned in line at the same level and that the aircraft can pass the two waypoints flying straight and level. Moreover, the position of the auxiliary waypoint is chosen such that the distance from the initial waypoint to the auxiliary waypoint is just the same as the distance from the initial waypoint to the given terminal waypoint of this phase:

$$d = \left\| \begin{pmatrix} x_{WPII} - x_{WPI} \\ y_{WPII} - y_{WPI} \\ z_{WPII} - z_{WPI} \end{pmatrix} \right\|_2 \tag{4.21}$$

Here, the subscript *WPII* denotes the terminal waypoint. Finally, the initial control inputs $\mathbf{u}_{ini,0}(t)$ that are required to solve the initial optimization problem for a straight and level flight between the initial waypoint and the auxiliary waypoint can e.g. be calculated using a trim routine that forces the aircraft to fly straight and level. Therefore, the trim routine has to determine the control time histories $\mathbf{u}_{ini,0}(t)$ in such a way that the first order time derivatives of the following state variables equal zero:

$$\dot{z}(t) = \dot{\chi}_K(t) = \dot{\gamma}_K(t) = \dot{\mu}_K(t) = \dot{V}_K(t) = 0 \tag{4.22}$$

Then the homotopy procedure starts whereat the state vector $\mathbf{x}_{AUX}$ of the auxiliary waypoint is gradually changed so that the auxiliary waypoint $\mathbf{x}_{AUX}$ approaches the given terminal waypoint $\mathbf{x}_{WPII}$ of the first phase:

$$\mathbf{x}_{AUX,k} = \mathbf{x}_{AUX,ini} + \frac{k \cdot \left(\mathbf{x}_{WPII} - \mathbf{x}_{AUX,ini}\right)}{n_S} \qquad k = 1, \ldots, n_S \tag{4.23}$$

in which k is the actual iteration number and n_S the total number of iteration steps. In doing so, for each iteration step k the corresponding optimization problem that determines the

optimal flight path between the initial waypoint $\mathbf{x}_{WPI}$ and the actual auxiliary waypoint $\mathbf{x}_{AUX,k}$ is solved.

If the values for the kinematic flight-path bank angle $\mu_{K,WPI}$ and the kinematic inclination angle $\gamma_{K,WPI}$ at the initial waypoint of the first phase are prescribed, the bank angle μ_K and the inclination angle γ_K are alternated by applying the same iterative scheme as for the auxiliary waypoint $\mathbf{x}_{AUX}$ so that the initial boundary constraints for the bank angle μ_K and the inclination angle γ_K are finally met:

$$\gamma_{K,WPI,k} = \gamma_{K,WPI,ini} + \frac{k \cdot \left(\gamma_{K,WPI} - \gamma_{K,WPI,ini}\right)}{n_S} \qquad k = 1, \ldots, n_S \tag{4.24}$$

$$\mu_{K,WPI,k} = \mu_{K,WPI,ini} + \frac{k \cdot \left(\mu_{K,WPI} - \mu_{K,WPI,ini}\right)}{n_S} \qquad k = 1, \ldots, n_S \tag{4.25}$$

where $\gamma_{K,WPI,ini}$ and $\mu_{K,WPI,ini}$ are specified by Eq. (4.19) respectively Eq. (4.20). On the other hand, if no values for the kinematic flight-path bank angle $\mu_{K,WPI}$ and the kinematic inclination angle $\gamma_{K,WPI}$ at the initial waypoint are specified, the corresponding initial boundary constraints are freed and the determination of those values is left to the optimization. The same holds for the controls, i.e. if no control values $\mathbf{u}_{WPI}$ at the initial waypoint of the first phase are specified, the determination of those values is also left to the optimization. Otherwise, initial boundary constraints for the controls are enforced and the control values are alternated within each iteration by:

$$\mathbf{u}_{WPI,k} = \mathbf{u}_{WPI,ini} + \frac{k \cdot \left(\mathbf{u}_{WPI} - \mathbf{u}_{WPI,ini}\right)}{n_S} \qquad k = 1, \ldots, n_S \tag{4.26}$$

where the controls $\mathbf{u}_{WPI,ini}$ are set equal to the controls $\mathbf{u}_{ini,0}(t)$ determined by the trim routine:

$$\mathbf{u}_{WPI,ini} = \mathbf{u}_{ini,0}(t) \tag{4.27}$$

In each iteration step k, the resulting slightly altered optimization problem is then solved utilizing the optimal solution $\mathbf{u}_{opt}(t)$ from the previous optimization run as initial guess $\mathbf{u}_{ini}(t)$:

$$\mathbf{u}_{ini,k}(t) = \mathbf{u}_{opt,k\text{-}1}(t) \qquad k = 2, \ldots, n_S \tag{4.28}$$

This procedure is repeated until the state vector of the original terminal waypoint of the first phase is met, so that the auxiliary waypoint $\mathbf{x}_{AUX}$ and the terminal waypoint $\mathbf{x}_{WPII}$ of the first phase coincidence:

$$\mathbf{x}_{AUX,n_S} = \mathbf{x}_{WPII} \tag{4.29}$$

The number of iterations n_S that is taken to shift the auxiliary waypoint $\mathbf{x}_{AUX}$ from its initial to its final position depends on the distance d_0 between the initial position of the auxiliary waypoint and the position of the prescribed terminal waypoint of the first phase:

$$d_0 = \left\| \begin{pmatrix} x_{WPII} - x_{AUX,ini} \\ y_{WPII} - y_{AUX,ini} \\ z_{WPII} - z_{AUX,ini} \end{pmatrix} \right\|_2 \tag{4.30}$$

The iteration number n_S is derived such that a maximum displacement d_{max} of the auxiliary waypoint within one iteration is not exceeded:

$$n_S = \text{ceil}\left(\frac{d_0}{d_{\max}}\right) \tag{4.31}$$

The value for the maximum displacement can be set within specified bounds, and it allows for a trade-off between

- increased stability of the initial guess generation if a low value is specified, or
- decreased computational time for generating the initial guess if the maximum displacement is set to a high value.

Furthermore, to speed up the initial guess generation the optimization tolerance that has to be achieved by the optimization within each iteration of the homotopy procedure can be set to a relatively low value, before the optimization tolerance is set to a higher value for the last iteration of the homotopy procedure. Additionally, the initial guess generation incorporates a special treatment of waypoint settings where the given terminal waypoint is very close or even identical to the initial waypoint and where the change in the course angle is very large between the initial and the final waypoint. An example maneuver would be a 270° turn where the aircraft has to return to its initial position in the shortest possible time. Before the homotopy procedure starts, the waypoint positions are checked and if a setting as described above is identified, an intermediate auxiliary waypoint is introduced so that the auxiliary waypoint is not shifted directly towards the terminal waypoint. Thus, infeasible waypoint positions and settings for the course angle are avoided since this might result in severe computational problems, causing the optimization and thus the homotopy procedure to fail.

Once the homotopy procedure is completed for the first phase, it is repeated in the same way for the remaining phases. At this, the determination of the initial boundary values that are eventually not specified at the initial waypoints of the distinctive phases, that could be the bank angle μ_K, the inclination angle γ_K, the kinematic velocity V_K and/or the controls $\mathbf{u}$, is not left to the optimization as it would have been the case in the first phase, but initial boundary constraints are enforced that require the respective quantities to take the final values of the preceding phases once the homotopy procedure for the respective phase has been finished:

$$\mathbf{x}\left(t_{0,i+1}\right) = \mathbf{x}\left(t_{f,i}\right) \quad i = 1,\ldots,p-1 \tag{4.32}$$

$$\mathbf{u}\left(t_{0,i+1}\right) = \mathbf{u}\left(t_{f,i}\right) \quad i = 1,\ldots,p-1 \tag{4.33}$$

with p being the number of phases. Regarding the path inclination angle γ_K, the bank angle μ_K and the controls $\mathbf{u}$, this is achieved by setting the initial starting values due to Eqs. (4.19), (4.20) and (4.27) and then alternating the initial boundary constraints according to Eqs. (4.24) to (4.26) during the homotopy procedure. Thus, the continuity of the time histories for the states as well as the controls is guaranteed across all phase boundaries.

Finally, the computed states and controls of all the phases are put together, producing a quite good initial guess for the optimization of the whole aircraft trajectory. This initial guess can be termed sub-optimal since the aircraft trajectory has only been optimized between the successive waypoints but it has not been optimized as a whole.

Table 4 summarizes the algorithm for the automatic generation of the initial guess for the entire trajectory optimization problem.

ALGORITHM III		
1		Split the optimization problem into p phases due to given waypoints
2		Solve the optimization problem for each single phase $i = 1, \ldots, p$
	2.A	Introduce auxiliary waypoint $\mathbf{x}_{AUX,i}$ according to (4.14) to (4.20)
	2.B	Determine initial controls $\mathbf{u}_{ini,0,i}(t)$ by a trim routine such that (4.22) holds
	2.C	Determine the number of iteration steps n_S using (4.30) and (4.31)
	2.D	Apply the homotopy procedure defined by Eqs. (4.23) to (4.26) and (4.28)
3		Combine $\mathbf{u}_{opt,i}(t)$, $i = 1, \ldots, p$ to give $\mathbf{u}_{sub\text{-}optimal}(t)$

Table 4. Automatic Generation of Initial Guesses

4.2.3 Substitute Optimization Problem

In this chapter, a substitute optimization problem is established to smoothen a given reference trajectory and to generate a reference trajectory up to its fourth order time derivative for the position vector that is required in step **1B** of the trajectory algorithm outlined in chapter 4.1. Therefore, a slightly modified version of the point-mass simulation model is set up. In this simulation model, only the kinematics given in chapter 3.2.2 and 3.2.3, but no aerodynamics are taken into account. Thus, the inputs to the modified point-mass simulation model would be made up by the load factors $\vec{\mathbf{n}}_{\bar{K}}$ in the Intermediate Kinematic Flight-Path Frame $\bar{K}$ respectively the first order time derivative of the kinematic flight-path bank angle μ_K. Instead of the load factors or the first order time derivative of the bank angle, their second order time derivatives are chosen as control inputs to the point-mass simulation model and the following ordinary differential equations are added to the simulation model:

$$\ddot{\mu}_K = \ddot{\mu}_{K,CMD} \tag{4.34}$$

$$\frac{d}{dt}\begin{pmatrix} n_{z,\bar{K}} \\ \dot{n}_{z,\bar{K}} \end{pmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}\begin{pmatrix} n_{z,\bar{K}} \\ \dot{n}_{z,\bar{K}} \end{pmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix}\ddot{n}_{z,\bar{K},CMD} \tag{4.35}$$

The same equations hold for the load factors $n_{x,\bar{K}}$ and $n_{y,\bar{K}}$ in the x- respectively the y-direction of the Intermediate Kinematic Flight-Path Frame $\bar{K}$. In doing so, the load factors $\vec{\mathbf{n}}_{\bar{K}}$ and the flight-path bank angle μ_K are delayed and the total dynamic order of the point-mass simulation model is artificially increased. Furthermore, now the time histories for the position vector as well as for the flight-path bank angle μ_K that are produced by the simulation model are sufficiently smooth.

Then, the following substitute trajectory optimization problem is solved: Determine the optimal control history

$$\mathbf{u}_{opt}(t) \in \mathbb{R}^m \tag{4.36}$$

and the corresponding optimal state trajectory

$$\mathbf{x}_{opt}(t) \in \mathbb{R}^n \tag{4.37}$$

that minimize the Lagrange cost functional

$$J = \int_{t_0}^{t_f} \left(\Delta\vec{\mathbf{r}}(t)\right)^2 dt = \int_{t_0}^{t_f} \left(\vec{\mathbf{r}}(t) - \vec{\mathbf{r}}_{PM}(t)\right)^2 dt \tag{4.38}$$

subject to the state dynamics given by Eqs. (4.34) and (4.35) as well as the position propagation equations and translation equations of motion of chapter 3.2. The corresponding control and state vector are:

$$\mathbf{u} = \left[\ddot{n}_{x,\bar{K},CMD}, \ddot{n}_{y,\bar{K},CMD}, \ddot{n}_{z,\bar{K},CMD}, \ddot{\mu}_{K,CMD}\right]^T \tag{4.39}$$

$$\mathbf{x} = \left[x, y, z, V_K, q_0, q_1, q_2, q_3, \dot{\mu}_K, n_{x,\bar{K}}, \dot{n}_{x,\bar{K}}, n_{y,\bar{K}}, \dot{n}_{y,\bar{K}}, n_{z,\bar{K}}, \dot{n}_{z,\bar{K}}\right]^T \tag{4.40}$$

The least-square criterion J in Eq. (4.38) minimizes deviations between the actual position vector $\vec{\mathbf{r}} = [x, y, z]^T$ and the reference position vector $\vec{\mathbf{r}}_{PM}$ that has been produced by the optimization task in step **1A** of the optimization algorithm of chapter 4.1, utilizing the unmodified point-mass simulation model.

5

Bilevel Optimal Control

The following chapters are concerned with the statement and the effective solution of a special class of bilevel optimal control problems, where an optimal solution of the upper level *parameter optimization problem* depends on the optimal solutions of one or more lower level *optimal control problems*. Although the bilevel programming problem is a combination of an upper level optimization problem and a certain number of lower level optimal control problems, it is termed bilevel optimal control problem since the solution of the entire bilevel problem is clearly dominated by the solution of the lower level optimal control problems. After the statement of the bilevel optimal control problem in chapter 5.1, chapter 5.2 introduces the sensitivity analysis for the lower level optimal control problems that forms the basis for the solution algorithm given in chapter 5.3.

5.1 Statement of the Bilevel Optimal Control Problem

Bilevel optimal control problems where an optimal solution of the upper level *parameter optimization problem* depends on the optimal solutions of one or more lower level *optimal control problems* can be stated as follows:

$$\min J\left(\mathbf{x}_i\left(\mathbf{z}_i(\mathbf{p}),\mathbf{p},t\right),\mathbf{y}_i\left(\mathbf{z}_i(\mathbf{p}),\mathbf{p},t\right),\mathbf{z}_i(\mathbf{p}),\mathbf{p}\right) \qquad i=1,\ldots,n \tag{5.1}$$

subject to

$$\mathbf{G}\left(\mathbf{x}_i\left(\mathbf{z}_i(\mathbf{p}),\mathbf{p},t\right),\mathbf{y}_i\left(\mathbf{z}_i(\mathbf{p}),\mathbf{p},t\right),\mathbf{z}_i(\mathbf{p}),\mathbf{p}\right)\le\mathbf{0} \qquad i=1,\ldots,n \tag{5.2}$$

where $\mathbf{x}_i(\mathbf{p})$ are the state functions, $\mathbf{y}_i(\mathbf{p})$ the output functions and $\mathbf{z}_i(\mathbf{p})$ the parameter vectors with respect to optimal solutions of optimal control problems as stated in chapter 2.2. Here, n indicates the number of the lower level *optimal control problems*. J denotes the objective of the upper level *parameter optimization problem* (5.1), $\mathbf{G}$ is the corresponding constraint vector and $\mathbf{p}$ the parameter vector of the upper level optimization problem. The parameter vector $\mathbf{p}$ is also involved in the solution of the lower level optimal control problems. Fig. 36 depicts the structure of the stated bilevel optimal control problems.

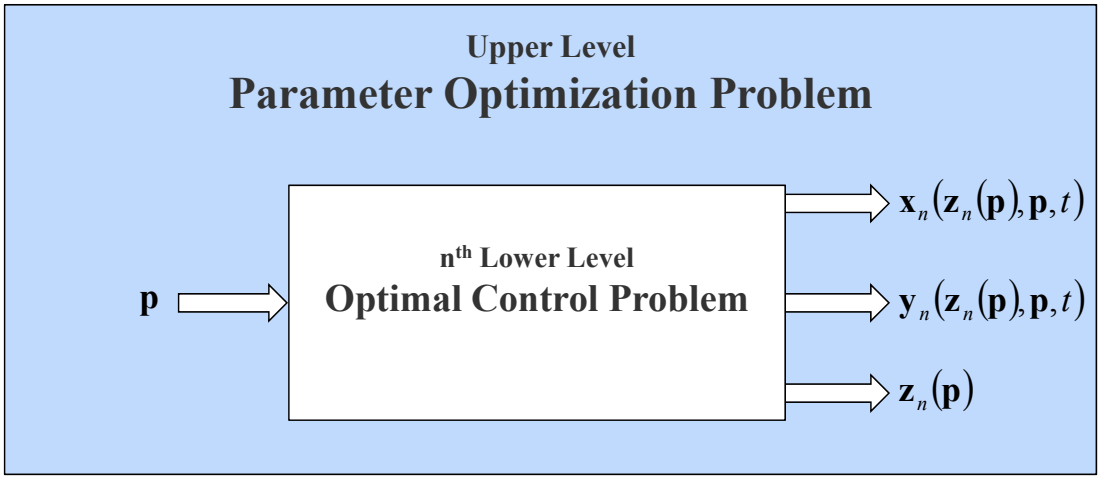

Figure 36. Structure of Bilevel Optimal Control Problem

If for the solution of such bilevel programming problems the gradient of the objective of the upper level optimization problem with respect to the parameter vector **p** was evaluated by means of numerical methods, this would result in an extreme computational effort. Utilizing central differences for the evaluation of the gradient of the upper level optimization problem, each lower level optimal control problem would have to be solved twice to compute the central difference with respect to one single parameter. Thus, an efficient way for the solution of such bilevel problems is needed that avoids the time consuming evaluation of the gradient of the upper level optimization problem. The basis for this solution algorithm is a sensitivity analysis for the lower level optimal control problems that is given in the following chapter.

5.2 Sensitivity Analysis for Lower Level Optimal Control Problems

In general, the goal of a sensitivity analysis is to determine how the solution of an optimal control problem changes when certain parameters within the optimal control problem are altered. At this, the parameters under consideration can either be subject to optimization or not. The sensitivity analysis can be utilized to compute a suboptimal solution of the optimal control problem in fairly short time or it can be applied if parameters with uncertainty are present in the optimal control problem. Theoretical fundamentals on the theory of sensitivity analysis can be found e.g. in Refs. [Fiacco, 1976] and [Fiacco, 1983]. In Refs. [Büskens, 1998] and [Büskens, 2000], Büskens gives a technique that is based on the work of Fiacco and that allows for the computation of sensitivity differentials and suboptimal solutions for a discretized optimal control problem. In the following, basic results for the computation of sensitivity information utilizing the technique of Büskens are given. These equations for the computation of sensitivity information form the basis for the efficient solution of the bilevel optimal control problem stated in chapter 5.1 by the algorithm outlined in the chapter 5.3.

For the nonlinear programming problem resulting from the discretization of the original optimal control problem, the augmented Lagrange cost function $L(\mathbf{z}, \boldsymbol{\mu}, \mathbf{p})$ together with the equality constraints $\mathbf{G}_a(\mathbf{z}, \mathbf{p})$ constitute the following set of equations:

$$L(\mathbf{z}, \boldsymbol{\mu}, \mathbf{p}) = J(\mathbf{z}, \mathbf{p}) + \boldsymbol{\mu}_a^T \mathbf{G}_a(\mathbf{z}, \mathbf{p}) \tag{5.3}$$

$$\mathbf{G}_a(\mathbf{z}, \mathbf{p}) = \mathbf{0} \tag{5.4}$$

where $\mathbf{z}$ is the parameter vector and $\mathbf{p}$ a vector of additional parameters associated with the optimal control problem (see chapter 2.2). $J(\mathbf{z}, \mathbf{p})$ denotes the final cost function. $\mathbf{G}_a(\mathbf{z}, \mathbf{p})$ is the vector of equality and active inequality constraints and $\boldsymbol{\mu}_a$ the vector of the corresponding Lagrange multipliers. For any solution of the nonlinear programming problem to be an optimal solution, the necessary conditions have to be fulfilled. This means, the gradient of the Lagrange function $L(\mathbf{z}, \boldsymbol{\mu}, \mathbf{p})$ with respect to the parameter vector $\mathbf{z}$ has to equal zero in order to ensure that the computed solution is at least a local minima or maxima:

$$\mathbf{L}_\mathbf{z}(\mathbf{z},\boldsymbol{\mu},\mathbf{p}) = \frac{\partial J(\mathbf{z},\mathbf{p})}{\partial \mathbf{z}} + \boldsymbol{\mu}_a^T \frac{\partial \mathbf{G}_a(\mathbf{z},\mathbf{p})}{\partial \mathbf{z}} = \mathbf{J}_\mathbf{z}(\mathbf{z},\mathbf{p}) + \boldsymbol{\mu}_a^T \mathbf{G}_{a,\mathbf{z}}(\mathbf{z},\mathbf{p}) = \mathbf{0} \tag{5.5}$$

In the following, Eq. (5.5) and Eq. (5.4) are joint together to give the vector of constraints $\mathbf{F}$:

$$\mathbf{F} = \begin{pmatrix} \mathbf{L}_\mathbf{z}(\mathbf{z},\boldsymbol{\mu},\mathbf{p}) \\ \mathbf{G}_a(\mathbf{z},\mathbf{p}) \end{pmatrix} = \begin{pmatrix} \mathbf{J}_\mathbf{z}(\mathbf{z},\mathbf{p}) + \boldsymbol{\mu}_a^T \mathbf{G}_{a,\mathbf{z}}(\mathbf{z},\mathbf{p}) \\ \mathbf{G}_a(\mathbf{z},\mathbf{p}) \end{pmatrix} = \mathbf{0} \tag{5.6}$$

Then, the Jacobian of the constraint vector $\mathbf{F}$ with respect to the parameter vector $\mathbf{z}$ and the vector of Lagrange multipliers $\boldsymbol{\mu}$ is:

$$\mathbf{F}_{(\mathbf{z},\boldsymbol{\mu}_a)} = \begin{pmatrix} \frac{\partial \mathbf{F}}{\partial \mathbf{z}} & \frac{\partial \mathbf{F}}{\partial \boldsymbol{\mu}_a} \end{pmatrix} = \begin{pmatrix} \mathbf{L}_{\mathbf{zz}}(\mathbf{z},\boldsymbol{\mu},\mathbf{p}) & \mathbf{G}_{a,\mathbf{z}}^T(\mathbf{z},\mathbf{p}) \\ \mathbf{G}_{a,\mathbf{z}}(\mathbf{z},\mathbf{p}) & \mathbf{0} \end{pmatrix} = \mathbf{0} \tag{5.7}$$

where the matrix $\mathbf{L}_{\mathbf{zz}}$ evaluates to:

$$\mathbf{L}_{\mathbf{zz}}(\mathbf{z},\boldsymbol{\mu},\mathbf{p}) = \frac{\partial \mathbf{J}_\mathbf{z}(\mathbf{z},\mathbf{p})}{\partial \mathbf{z}} + \boldsymbol{\mu}_a^T \frac{\partial \mathbf{G}_{a,\mathbf{z}}(\mathbf{z},\mathbf{p})}{\partial \mathbf{z}} = \mathbf{J}_{\mathbf{zz}}(\mathbf{z},\mathbf{p}) + \boldsymbol{\mu}_a^T \mathbf{G}_{a,\mathbf{zz}}(\mathbf{z},\mathbf{p}) \tag{5.8}$$

Since it is the goal to compute a solution of the nonlinear programming problem that is optimal or at least sub-optimal for small perturbations in the parameter vector $\mathbf{p}$, it is required that the necessary optimality conditions (5.6) hold in the vicinity of the parameter vector $\mathbf{p}$. This can be achieved by requiring that the gradient of the optimality conditions (5.6) with respect to the parameter vector $\mathbf{p}$ equals zero:

$$\mathbf{F}_\mathbf{p} = \mathbf{F}_{(\mathbf{z},\boldsymbol{\mu}_a)} \cdot \begin{pmatrix} \frac{d\mathbf{z}_0}{d\mathbf{p}} \\ \frac{d\boldsymbol{\mu}_{a,0}}{d\mathbf{p}} \end{pmatrix} + \begin{pmatrix} \mathbf{L}_{\mathbf{zp}}(\mathbf{z}_0,\boldsymbol{\mu}_0,\mathbf{p}_0) \\ \mathbf{G}_{a,\mathbf{p}}(\mathbf{z}_0,\mathbf{p}_0) \end{pmatrix} \overset{!}{=} \mathbf{0} \tag{5.9}$$

$$\mathbf{F}_\mathbf{p} = \begin{pmatrix} \mathbf{L}_{\mathbf{zz}}(\mathbf{z}_0,\boldsymbol{\mu}_0,\mathbf{p}_0) & \mathbf{G}_{a,\mathbf{z}}^T(\mathbf{z}_0,\mathbf{p}_0) \\ \mathbf{G}_{a,\mathbf{z}}(\mathbf{z}_0,\mathbf{p}_0) & \mathbf{0} \end{pmatrix} \begin{pmatrix} \frac{d\mathbf{z}_0}{d\mathbf{p}} \\ \frac{d\boldsymbol{\mu}_{a,0}}{d\mathbf{p}} \end{pmatrix} + \begin{pmatrix} \mathbf{L}_{\mathbf{zp}}(\mathbf{z}_0,\boldsymbol{\mu}_0,\mathbf{p}_0) \\ \mathbf{G}_{a,\mathbf{p}}(\mathbf{z}_0,\mathbf{p}_0) \end{pmatrix} \overset{!}{=} \mathbf{0} \tag{5.10}$$

where $\mathbf{p}_0$, $\mathbf{z}_0$ and $\boldsymbol{\mu}_0$ indicate the values of the parameter vector $\mathbf{z}$, the vector of Lagrange multipliers $\boldsymbol{\mu}$ respectively the parameter vector $\mathbf{p}$ of the optimal solution of the nonlinear programming problem. The matrix $\mathbf{L}_{\mathbf{zp}}$ is given by:

$$\mathbf{L}_{\mathbf{zp}}(\mathbf{z},\boldsymbol{\mu},\mathbf{p}) = \frac{\partial \mathbf{J}_\mathbf{z}(\mathbf{z},\mathbf{p})}{\partial \mathbf{p}} + \boldsymbol{\mu}_a^T \frac{\partial \mathbf{G}_{a,\mathbf{z}}(\mathbf{z},\mathbf{p})}{\partial \mathbf{p}} = \mathbf{J}_{\mathbf{zp}}(\mathbf{z},\mathbf{p}) + \boldsymbol{\mu}_a^T \mathbf{G}_{a,\mathbf{zp}}(\mathbf{z},\mathbf{p}) \tag{5.11}$$

From Eq. (5.10) the sensitivity matrix $d\mathbf{z}_0/d\mathbf{p}$ and also the sensitivity matrix $d\boldsymbol{\mu}_{a,0}/d\mathbf{p}$ are then obtained by:

$$\begin{pmatrix} \dfrac{d\mathbf{z}_0}{d\mathbf{p}} \\ \dfrac{d\boldsymbol{\mu}_{a,0}}{d\mathbf{p}} \end{pmatrix} = -\begin{pmatrix} \mathbf{L}_{\mathbf{zz}}(\mathbf{z}_0, \boldsymbol{\mu}_0, \mathbf{p}_0) & \mathbf{G}_{a,\mathbf{z}}^T(\mathbf{z}_0, \mathbf{p}_0) \\ \mathbf{G}_{a,\mathbf{z}}(\mathbf{z}_0, \mathbf{p}_0) & \mathbf{0} \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{L}_{\mathbf{zp}}(\mathbf{z}_0, \boldsymbol{\mu}_0, \mathbf{p}_0) \\ \mathbf{G}_{a,\mathbf{p}}(\mathbf{z}_0, \mathbf{p}_0) \end{pmatrix} \tag{5.12}$$

Here, the dimension of the matrix $d\mathbf{z}_0/d\mathbf{p}$ is $n_{var} \times n_{par}$, where n_{var} is the length of the parameter vector $\mathbf{z}$ and where n_{par} is the number of parameters, i.e. the length of the parameter vector $\mathbf{p}$. The matrix $d\boldsymbol{\mu}_{a,0}/d\mathbf{p}$ is a $n_{act} \times n_{par}$-matrix, where n_{act} is the number of active constraints. The Hessian $\mathbf{L}_{\mathbf{zz}}$ is a $n_{var} \times n_{var}$-matrix, while the matrix $\mathbf{G}_{a,\mathbf{z}}$ has the dimension $n_{act} \times n_{var}$. The dimension of the matrix $\mathbf{L}_{\mathbf{zp}}$ is $n_{var} \times n_{par}$ and the dimension of the matrix $\mathbf{G}_{a,\mathbf{p}}$ is $n_{act} \times n_{par}$.

At this, the sensitivity matrix $d\mathbf{z}_0/d\mathbf{p}$ indicates how the parameter vector $\mathbf{z}_0$ has to be changed in order to obtain a sub-optimal solution if any parameter in the parameter vector $\mathbf{p}$ is subject to perturbation. From the sensitivity matrix given by Eq. (5.12), the sensitivity of the cost function, the sensitivity of the constraints as well as the sensitivity of the state functions with respect to the parameter vector $\mathbf{p}$ can be obtained. The results are given in the following.

The sensitivity of any constraint function $g_i(\mathbf{z}, \mathbf{p})$ with respect to the parameter vector $\mathbf{p}$ evaluates to:

$$\frac{dg_i(\mathbf{z}_0, \mathbf{p}_0)}{d\mathbf{p}} = \begin{bmatrix} \dfrac{\partial g_i(\mathbf{z}_0, \mathbf{p}_0)}{\partial \mathbf{z}} & \dfrac{\partial g_i(\mathbf{z}_0, \mathbf{p}_0)}{\partial \boldsymbol{\mu}_a} \end{bmatrix} \begin{pmatrix} \dfrac{d\mathbf{z}_0}{d\mathbf{p}} \\ \dfrac{d\boldsymbol{\mu}_{a,0}}{d\mathbf{p}} \end{pmatrix} + \frac{\partial g_i(\mathbf{z}_0, \mathbf{p}_0)}{\partial \mathbf{p}} \tag{5.13}$$

$$\frac{dg_i(\mathbf{z}_0, \mathbf{p}_0)}{d\mathbf{p}} = \mathbf{g}_{i(\mathbf{z},\boldsymbol{\mu}_a)}(\mathbf{z}_0, \mathbf{p}_0) \begin{pmatrix} \dfrac{d\mathbf{z}_0}{d\mathbf{p}} \\ \dfrac{d\boldsymbol{\mu}_{a,0}}{d\mathbf{p}} \end{pmatrix} + \mathbf{g}_{i,\mathbf{p}}(\mathbf{z}_0, \mathbf{p}_0) \tag{5.14}$$

Since the constraints $g_i(\mathbf{z}, \mathbf{p})$ are only a function of the parameter vectors $\mathbf{z}$ and $\mathbf{p}$ but not of the Lagrange multipliers $\boldsymbol{\mu}$, Eq. (5.14) can be written as:

$$\frac{dg_i(\mathbf{z}_0, \mathbf{p}_0)}{d\mathbf{p}} = \begin{bmatrix} \mathbf{g}_{i,\mathbf{z}}(\mathbf{z}_0, \mathbf{p}_0) & \mathbf{0} \end{bmatrix} \begin{pmatrix} \dfrac{d\mathbf{z}_0}{d\mathbf{p}} \\ \dfrac{d\boldsymbol{\mu}_{a,0}}{d\mathbf{p}} \end{pmatrix} + \mathbf{g}_{i,\mathbf{p}}(\mathbf{z}_0, \mathbf{p}_0) \tag{5.15}$$

For the sensitivity of the cost function $J(\mathbf{z}, \mathbf{p})$ with respect to the parameter vector $\mathbf{p}$ the following relationships result:

$$\frac{dJ(\mathbf{z}_0,\mathbf{p}_0)}{d\mathbf{p}} = \left[\frac{\partial J(\mathbf{z}_0,\mathbf{p}_0)}{\partial \mathbf{z}} \quad \frac{\partial J(\mathbf{z}_0,\mathbf{p}_0)}{\partial \boldsymbol{\mu}_a}\right] \begin{pmatrix} \dfrac{d\mathbf{z}_0}{d\mathbf{p}} \\ \dfrac{d\boldsymbol{\mu}_{a,0}}{d\mathbf{p}} \end{pmatrix} + \frac{\partial J(\mathbf{z}_0,\mathbf{p}_0)}{\partial \mathbf{p}} \tag{5.16}$$

$$\frac{dJ(\mathbf{z}_0,\mathbf{p}_0)}{d\mathbf{p}} = \mathbf{J}_{(\mathbf{z},\boldsymbol{\mu}_a)}(\mathbf{z}_0,\mathbf{p}_0) \begin{pmatrix} \dfrac{d\mathbf{z}_0}{d\mathbf{p}} \\ \dfrac{d\boldsymbol{\mu}_{a,0}}{d\mathbf{p}} \end{pmatrix} + \mathbf{J}_{\mathbf{p}}(\mathbf{z}_0,\mathbf{p}_0) \tag{5.17}$$

With the cost function $J(\mathbf{z}, \mathbf{p})$ being solely a function of the parameter vectors $\mathbf{z}$ and $\mathbf{p}$ but not of the Lagrange multiplier vector $\boldsymbol{\mu}$, it follows:

$$\frac{dJ(\mathbf{z}_0,\mathbf{p}_0)}{d\mathbf{p}} = \left[\mathbf{J}_{\mathbf{z}}(\mathbf{z}_0,\mathbf{p}_0) \quad \mathbf{0}\right] \begin{pmatrix} \dfrac{d\mathbf{z}_0}{d\mathbf{p}} \\ \dfrac{d\boldsymbol{\mu}_{a,0}}{d\mathbf{p}} \end{pmatrix} + \mathbf{J}_{\mathbf{p}}(\mathbf{z}_0,\mathbf{p}_0) \tag{5.18}$$

Using Eq. (5.5), the sensitivity of the cost function becomes:

$$\frac{dJ(\mathbf{z}_0,\mathbf{p}_0)}{d\mathbf{p}} = \left[-\boldsymbol{\mu}_{a,0}^T \mathbf{G}_{a,\mathbf{z}}(\mathbf{z}_0,\mathbf{p}_0) \quad \mathbf{0}\right] \begin{pmatrix} \dfrac{d\mathbf{z}_0}{d\mathbf{p}} \\ \dfrac{d\boldsymbol{\mu}_{a,0}}{d\mathbf{p}} \end{pmatrix} + \mathbf{J}_{\mathbf{p}}(\mathbf{z}_0,\mathbf{p}_0) \tag{5.19}$$

From the second line of Eq. (5.10), one has for the active constraint functions $\mathbf{G}_a(\mathbf{z}, \mathbf{p})$ in the vicinity of the parameter vector $\mathbf{p}_0$:

$$\frac{d\mathbf{G}_a(\mathbf{z}_0,\mathbf{p}_0)}{d\mathbf{p}} = \left[\mathbf{G}_{a,\mathbf{z}}(\mathbf{z}_0,\mathbf{p}_0) \quad \mathbf{0}\right] \begin{pmatrix} \dfrac{d\mathbf{z}_0}{d\mathbf{p}} \\ \dfrac{d\boldsymbol{\mu}_{a,0}}{d\mathbf{p}} \end{pmatrix} + \mathbf{G}_{a,\mathbf{p}}(\mathbf{z}_0,\mathbf{p}_0) \overset{!}{=} 0. \tag{5.20}$$

Then, the derivative of the active constraint functions $\mathbf{G}_a(\mathbf{z}, \mathbf{p})$ with respect to the parameter vector $\mathbf{p}$ can be written as:

$$\mathbf{G}_{a,\mathbf{p}}(\mathbf{z}_0,\mathbf{p}_0) = -\left[\mathbf{G}_{a,\mathbf{z}}(\mathbf{z}_0,\mathbf{p}_0) \quad \mathbf{0}\right] \begin{pmatrix} \dfrac{d\mathbf{z}_0}{d\mathbf{p}} \\ \dfrac{d\boldsymbol{\mu}_{a,0}}{d\mathbf{p}} \end{pmatrix} \tag{5.21}$$

Utilizing Eq. (5.21) together with Eq. (5.19), it finally results that the sensitivity of the cost function $J(\mathbf{z}, \mathbf{p})$ with respect to the parameter vector $\mathbf{p}$ equals the sensitivity of the Lagrange cost $L(\mathbf{z}, \boldsymbol{\mu}, \mathbf{p})$ with respect to the parameter vector $\mathbf{p}$:

$$\frac{dJ(\mathbf{z}_0, \mathbf{p}_0)}{d\mathbf{p}} = \mathbf{J}_{\mathbf{p}}(\mathbf{z}_0, \mathbf{p}_0) + \boldsymbol{\mu}_{a,0}^T \mathbf{G}_{a,\mathbf{p}}(\mathbf{z}_0, \mathbf{p}_0) = \mathbf{L}_{\mathbf{p}}(\mathbf{z}_0, \boldsymbol{\mu}_0, \mathbf{p}_0) \tag{5.22}$$

The sensitivity of the state function $\mathbf{x}(t)$ with respect to the parameter vector $\mathbf{p}$ is given by:

$$\frac{d\mathbf{x}(\mathbf{z}_0, \mathbf{p}_0, t)}{d\mathbf{p}} = \frac{\partial \mathbf{x}(\mathbf{z}_0, \mathbf{p}_0, t)}{\partial \mathbf{z}} \cdot \frac{d\mathbf{z}_0}{d\mathbf{p}} + \frac{\partial \mathbf{x}(\mathbf{z}_0, \mathbf{p}_0, t)}{\partial \mathbf{p}} \tag{5.23}$$

Accordingly, the sensitivity of any output function $\mathbf{y}(t)$ with respect to the parameter vector $\mathbf{p}$ may be evaluated by:

$$\frac{d\mathbf{y}(\mathbf{z}_0, \mathbf{p}_0, t)}{d\mathbf{p}} = \frac{\partial \mathbf{y}(\mathbf{z}_0, \mathbf{p}_0, t)}{\partial \mathbf{z}} \cdot \frac{d\mathbf{z}_0}{d\mathbf{p}} + \frac{\partial \mathbf{y}(\mathbf{z}_0, \mathbf{p}_0, t)}{\partial \mathbf{p}} \tag{5.24}$$

For an optimal solution ($\mathbf{z}_0$, $\boldsymbol{\mu}_0$, $\mathbf{p}_0$) of the nonlinear programming problem, a sub-optimal solution with regard to a perturbed parameter vector $\mathbf{p}$ can be obtained by a first order approximation using the above sensitivity results:

$$\mathbf{x}^*(\mathbf{z}, \mathbf{p}, t) \approx \mathbf{x}(\mathbf{z}_0, \mathbf{p}_0, t) + \frac{d\mathbf{x}(\mathbf{z}_0, \mathbf{p}_0, t)}{d\mathbf{p}} (\mathbf{p} - \mathbf{p}_0) \tag{5.25}$$

$$g_i^*(\mathbf{z}, \mathbf{p}) \approx g_i(\mathbf{z}_0, \mathbf{p}_0) + \frac{dg_i(\mathbf{z}_0, \mathbf{p}_0)}{d\mathbf{p}} (\mathbf{p} - \mathbf{p}_0) \tag{5.26}$$

$$J^*(\mathbf{z}, \mathbf{p}) \approx J(\mathbf{z}_0, \mathbf{p}_0) + \frac{dJ(\mathbf{z}_0, \mathbf{p}_0)}{d\mathbf{p}} (\mathbf{p} - \mathbf{p}_0) \tag{5.27}$$

where it is mentioned that the perturbed parameter vector $\mathbf{p}$ has to lie in a certain vicinity of the parameter vector $\mathbf{p}_0$ associated to the optimal solution. If the perturbation is too large, i.e. the perturbed parameter vector is not within a valid vicinity of the parameter vector $\mathbf{p}_0$, the sensitivity results are not valid any more. Possibilities for the estimation of the size of the valid vicinity are given in Ref. [Bueskens, 1998]. According to Ref. [Bueskens, 1998], the validity of the sensitivity results is usually limited by changes in the set of active constraints. Thus, one possibility to estimate the valid perturbation $\Delta\mathbf{p}$ of the parameter vector $\mathbf{p}$ is to consider the inactive constraints. An inactive constraint becomes active if its value goes to zero, i.e.

$$g_i^*(\mathbf{z}, \mathbf{p}) \approx g_i(\mathbf{z}_0, \mathbf{p}_0) + \frac{dg_i(\mathbf{z}_0, \mathbf{p}_0)}{d\mathbf{p}} (\mathbf{p} - \mathbf{p}_0) = 0 \tag{5.28}$$

Consequently, the valid perturbation $\Delta\mathbf{p}$ considering the i-th constraint can be determined as:

$$\Delta\mathbf{p}_i = \mathbf{p} - \mathbf{p}_0 \approx -\frac{g_i(\mathbf{z}_0, \mathbf{p}_0)}{\left(\frac{dg_i(\mathbf{z}_0, \mathbf{p}_0)}{d\mathbf{p}} \right)} \tag{5.29}$$

This implies that if any perturbation is larger than the perturbation determined by Eq. (5.29), the respective inactive constraint of the sub-optimal solution would be violated, resulting in a non-valid sub-optimal solution.

5.3 Solution Algorithm

According to Ref. [Falk, 1995], the solution algorithms that can been applied to solve bilevel optimization problems can be divided into three categories. Algorithms of the first category add the necessary optimality conditions for the lower level optimization problem (i.e. the Karush-Kuhn-Tucker conditions) as constraints to the upper level optimization problem. At this, the bilevel programming problem is turned into a nonconvex, standard single level optimization problem. Algorithms that utilize double-penalty functions to approximate the upper and lower level optimization problems, thus transforming them into a sequence of unconstrained optimization problems, form the second category. In the third category, there are algorithms that utilize gradient information with respect to the lower level optimization problem in order to solve the upper level optimization method by a descent method where usually all constraints are associated to the lower level optimization problem.

The solution algorithm that is proposed in this chapter for the solution of the bilevel optimal control problem stated in chapter 5.1 falls into the third category. At this, the sensitivity analysis outlined in chapter 5.2 is not utilized in order to compute a suboptimal solution of an optimal control problem but to determine the gradient of the objective of the upper level optimization problem with respect to selected parameters of the lower level optimal control problems. Furthermore, if there are any equality or inequality constraints present in the upper level optimization problem, the sensitivity analysis is also utilized to compute the gradient of the constraint vector of the upper level optimization problem with respect to the selected parameters of the lower level optimal control problems.

The sensitivity matrix $d\mathbf{z}/d\mathbf{p}$ that is a basic result of the sensitivity analysis is used to directly compute the gradient of the objective of the upper level optimization problem with respect to the parameter vector $\mathbf{p}$. Here, $\mathbf{z}$ is the parameter vector of a lower level optimal control problem and $\mathbf{p}$ is the parameter vector of the upper level optimization problem, where the parameter vector $\mathbf{p}$ is also involved in the solution of the lower level optimal control problems. In Table 5, the various steps that have to be taken in every iteration step k of the upper level optimization problem are listed, where n denotes the number of lower level optimal control problems. The sub-optimal solutions $\tilde{\mathbf{z}}_{i,k}$ for the parameter vectors $\mathbf{z}_{i,k}$ are obtained by a first order approximation of the parameter vectors $\mathbf{z}_{i,k\text{-}1}$ of the preceding iteration step:

$$\tilde{\mathbf{z}}_{i,k} = \mathbf{z}_{i,k-1} + \left.\frac{d\mathbf{z}_i}{d\mathbf{p}^T}\right|_{k-1} \left(\mathbf{p}_k - \mathbf{p}_{k-1}\right) \qquad i = 1,\ldots,n \tag{5.30}$$

Here, $\mathbf{z}_{i,k}$ is the optimal solution of the i-th lower level optimal control problem at the k-th iteration step of the upper level optimization problem. Then, the optimal solutions $\mathbf{z}_{i,k}$, $i = 1, \ldots, n$ can either be computed using the sub-optimal solutions $\tilde{\mathbf{z}}_{i,k}$ or the optimal solutions $\mathbf{z}_{i,k\text{-}1}$ from the previous iteration step as initial guesses. The optimal control problems are solved by applying the direct multiple shooting method given in chapter 2.2, converting the optimal control problems into parameter optimization problems by means of discretization of the control time histories. Furthermore, the objective J_k and the constraint vector $\mathbf{G}_k$ of the upper level optimization problem have to be evaluated. After carrying out the sensitivity analysis outlined in chapter 5.2 in order to obtain the sensitivity matrices $d\mathbf{z}_{i,k}/d\mathbf{p}_k$,, the gradient $dJ_k/d\mathbf{p}_k$ of the objective and the Jacobian $d\mathbf{G}_k/d\mathbf{p}_k$ of the constraint vector with respect to the parameter vector $\mathbf{p}_k$ can be computed straight forward:

Given the objective J_k,

$$J_k(\mathbf{p}_k) = J_k\left(\mathbf{x}_i\left(\mathbf{z}_{i,k}(\mathbf{p}_k), \mathbf{p}_k, t\right), \mathbf{y}_i\left(\mathbf{z}_{i,k}(\mathbf{p}_k), \mathbf{p}_k, t\right), \mathbf{z}_{i,k}(\mathbf{p}_k), \mathbf{p}_k\right) \qquad i = 1, \ldots, n \tag{5.31}$$

and the constraint vector $\mathbf{G}_k$,

$$\mathbf{G}_k(\mathbf{p}_k) = \mathbf{G}_k\left(\mathbf{x}_i\left(\mathbf{z}_{i,k}(\mathbf{p}_k), \mathbf{p}_k, t\right), \mathbf{y}_i\left(\mathbf{z}_{i,k}(\mathbf{p}_k), \mathbf{p}_k, t\right), \mathbf{z}_{i,k}(\mathbf{p}_k), \mathbf{p}_k\right) \qquad i = 1, \ldots, n \tag{5.32}$$

the gradient $dJ_k/d\mathbf{p}_k$ of the objective with respect to the parameter vector $\mathbf{p}_k$ evaluates to

$$\frac{dJ_k}{d\mathbf{p}_k} = \sum_{i=1}^{n} \frac{dJ_k}{d\mathbf{x}_i} \cdot \frac{d\mathbf{x}_i}{d\mathbf{p}_k} + \sum_{i=1}^{n} \frac{dJ_k}{d\mathbf{y}_i} \cdot \frac{d\mathbf{y}_i}{d\mathbf{p}_k} + \sum_{i=1}^{n} \frac{dJ_k}{d\mathbf{z}_i} \cdot \frac{d\mathbf{z}_{i,k}}{d\mathbf{p}_k} + \frac{dJ_k}{d\mathbf{p}_k} \tag{5.33}$$

while the Jacobian $d\mathbf{G}_k/d\mathbf{p}_k$ of the constraint vector with respect to the parameter vector $\mathbf{p}_k$ is

$$\frac{d\mathbf{G}_k}{d\mathbf{p}_k} = \sum_{i=1}^{n} \frac{d\mathbf{G}_k}{d\mathbf{x}_i} \cdot \frac{d\mathbf{x}_i}{d\mathbf{p}_k} + \sum_{i=1}^{n} \frac{d\mathbf{G}_k}{d\mathbf{y}_i} \cdot \frac{d\mathbf{y}_i}{d\mathbf{p}_k} + \sum_{i=1}^{n} \frac{d\mathbf{G}_k}{d\mathbf{z}_i} \cdot \frac{d\mathbf{z}_{i,k}}{d\mathbf{p}_k} + \frac{d\mathbf{G}_k}{d\mathbf{p}_k} \tag{5.34}$$

where $d\mathbf{x}_i/d\mathbf{p}_k$ and $d\mathbf{y}_i/d\mathbf{p}_k$ are given by Eq. (5.23) respectively Eq. (5.24).

ALGORITHM I

1	Given $\mathbf{p}_k$, $\mathbf{p}_{k-1}$, $\mathbf{z}_{i,k-1}$, $i = 1, \ldots, n$ and $d\mathbf{z}_{i,k-1}/d\mathbf{p}_{k-1}$
1A	Compute the sub-optimal solutions $\tilde{\mathbf{z}}_{i,k}$, $i = 1, \ldots, n$ by Eq. (5.30)
1B	Find the optimal solutions $\mathbf{z}_{i,k}$, $i = 1, \ldots, n$ using the sub-optimal solutions $\tilde{\mathbf{z}}_{i,k}$, $i = 1, \ldots, n$ or the optimal solutions $\mathbf{z}_{i,k-1}$ as initial guesses
1C	Get the sensitivity matrices $d\mathbf{z}_{i,k}/d\mathbf{p}_k$, $d\mathbf{x}_i/d\mathbf{p}_k$ and $d\mathbf{y}_i/d\mathbf{p}_k$ utilizing the sensitivity analysis of chapter 5.2 (Eqs. (5.12), (5.23) respectively (5.24))
2A	Evaluate the objective J_k of the upper level optimization problem
2B	Compute the gradient $dJ_k/d\mathbf{p}_k$ using $d\mathbf{x}_i/d\mathbf{p}_k$, $d\mathbf{y}_i/d\mathbf{p}_k$ and $d\mathbf{z}_{i,k}/d\mathbf{p}_k$ (Eq. (5.33))
3A	Evaluate the constraint vector $\mathbf{G}_k$ of the upper level optimization problem
3B	Compute the Jacobian $d\mathbf{G}_k/d\mathbf{p}_k$ using $d\mathbf{x}_i/d\mathbf{p}_k$, $d\mathbf{y}_i/d\mathbf{p}_k$ and $d\mathbf{z}_{i,k}/d\mathbf{p}_k$ (Eq. (5.34))

i … index of lower level optimal control problem, $i = 1, \ldots, n$
k … iteration step of upper level parameter optimization problem

Table 5. Algorithm Abstract for the Solution of the Bilevel Programming Problem

The proposed algorithm allows for a direct computation of the gradient of the upper level optimization problem at each iteration step. Thus, the time consuming evaluation of the gradient of the upper level optimization problem by numerical techniques can be avoided and an efficient way for the solution of the bilevel programming problem is implemented.

6

Applications and Results

In the field of optimal control, the optimization of air race trajectories constitutes a very challenging application. The aerobatic aircraft taking part in the air races are extremely agile and the setup of the race tracks causes the pilots to fully exploit the dynamics of the aircraft. Thus, the optimization of race trajectories as well as the optimization of race track layouts are well suited to show the performance of the solution algorithms outlined in chapter 4 and 5. In the following chapter 6.1, the optimization of an air race trajectory is formulated as benchmark problem and in chapter 6.2, the time-optimal race trajectory is provided for the highest fidelity level of the various simulation models that is the 6-DoF simulation model. In chapter 6.3 and 6.4, the solutions of bilevel optimal control problems are depicted where it has been the goal to optimize the layout of the race track such that certain safety criteria or the fairness of the race track become optimal.

6.1 Air Race as Benchmark Problem

The basic procedure of the regarded air races is as follows: after passing a starting point, which can be defined by a significant landmark like e.g. the chain bridge in Budapest, the aircraft have to fly a course defined by inflatable pylons at minimum time. The pylons form gates which either are to be passed wings level (level gate, Fig. 37), or at 90° bank angle (knife edge gate, Fig. 38) which can be differentiated by their color.

Figure 37. Level Gate (blue)

Figure 38. Knife Edge Gate (red)

Other features are the “Quadro” (Fig. 39), consisting of two pylon pairs that have to be passed from perpendicular directions or the “Chicane” (Fig. 41) which is a sequence of single pylons in a straight line requiring rapid changes in turn direction. Furthermore, re-alignment and aerobatic maneuvers like vertical rolls or Half Cuban Eights (Fig. 40) are included to re-position the aircraft with respect to the track.

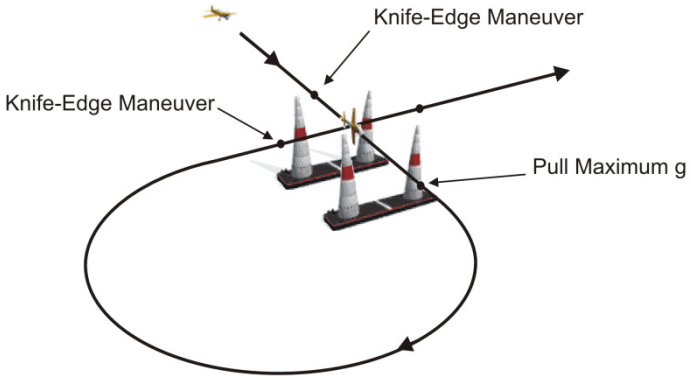

Figure 39. Quadro

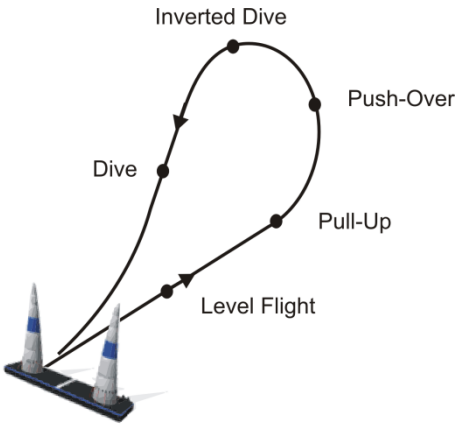

Figure 40. Half Cuban Eight

The race ends by passing a finish gate which in many cases is equal to the start gate or again a significant landmark. The air races take place right in the hearts of large cities like Budapest, San Diego, Rio de Janeiro, New York or Berlin and are often located on rivers or on the waterside.

Figure 41. Chicane

For such air races, the pilots are flying different types of aerobatic aircraft like e.g. the Zivko Edge 540 (Fig. 42) or the MXS-R. Those aerobatic aircraft are very agile, featuring e.g. roll rates up to 420°/s. Table 6 gives an overview of the technical specifications of such an aerobatic aircraft.

Figure 42. Zivko Edge 540

Aircraft Specifications		
mass	m [kg]	693.0
wing area	S [m²]	8.928
wing span	b [m]	7.5
half wing span	s [m]	3.75
chord length	$\bar{c}$ [m]	1.44
reference speed	V_{ref} [m/s]	30.0
maximum thrust	T_{ref} [N]	$0.8mg$
moment of inertia	I_{xx} [kg·m²]	420.30356820
	I_{yy} [kg·m²]	726.71842759
	I_{zz} [kg·m²]	919.24457818

Table 6. Aircraft Specifications

In order to win such an air race competition, the pilot has to find the fastest possible flight course through the gates, i.e. he tries to finish the race course in the minimum possible flight time. Thus, for the air race trajectory optimization problem, the Bolza cost functional given by Eq. (2.4) reduces to a Mayer functional since the only objective of the trajectory optimization problem is to minimize the final time:

$$J = t_f \tag{6.1}$$

The initial boundary conditions of Eq. (2.6) for the optimization problem are given by the position of the start gate, whereas the final boundary conditions (Eq. (2.7)) are determined by

the location of the finishing gate and the direction the finishing gate has to be passed by the aircraft:

$$\vec{\mathbf{r}}(t_0) - \vec{\mathbf{r}}_{StartGate} = \vec{\mathbf{0}} \tag{6.2}$$

$$\vec{\mathbf{r}}(t_f) - \vec{\mathbf{r}}_{FinalGate} = \vec{\mathbf{0}} \tag{6.3}$$

where $\vec{\mathbf{r}}$ denotes the position vector. The requirement that the pilot has to fly through certain gates in a certain direction and at given bank angles imposes interior point conditions (Eq. (2.8)) to the trajectory optimization problem. Basically, there are two different types of gates, level gates and knife edge gates. Level gates have to be passed wings level, i.e. with the kinematic bank angle Φ_K equal to zero whereas knife edge gates have to be flown through with a bank angle $\Phi_K = \pm 90°$. The resulting conditions read:

$$\Phi_K\left(\vec{\mathbf{r}}_{KnifeEdgeGate}\right) \pm 90° = 0° \tag{6.4}$$

$$\Phi_K\left(\vec{\mathbf{r}}_{LevelGate}\right) = 0° \tag{6.5}$$

Furthermore, the direction in which the various air race gates have to be passed is enforced by the following relationship for the heading angle ψ_K:

$$\psi_K\left(\vec{\mathbf{r}}_{Gate}\right) - \psi_i = 0° \tag{6.6}$$

At the chicane gates, there are only the final boundary conditions for the position vector $\vec{\mathbf{r}}$, but no final boundary conditions for the kinematic bank angle Φ_K or the heading angle ψ_K.

By separating the entire race trajectory into multiple phases from gate to gate, these interior point conditions are transformed into final boundary conditions for each phase. The phases then have to be connected to the preceding phases to guarantee the continuity of the state and the control time histories:

$$\mathbf{x}_{i-1}\left(t_{f,i-1}\right) - \mathbf{x}_i\left(t_{0,i}\right) = \mathbf{0} \qquad i = 2,...,n \tag{6.7}$$

$$\mathbf{u}_{i-1}\left(t_{f,i-1}\right) - \mathbf{u}_i\left(t_{0,i}\right) = \mathbf{0} \qquad i = 2,...,n \tag{6.8}$$

where n denotes the number of phases, $t_{f,i}$ the final time of the i-th phase and $t_{0,i}$ the initial time of the i-th phase. Additionally, path constraints have to be fulfilled along the flight path for an air race. While no equality path constraints are present, inequality path constraints arise from safety regulations or from aircraft performance limits. First of all, of course a certain ground clearance has to be respected by the pilots:

$$z_{\min} - z(t) \leq 0 \tag{6.9}$$

Furthermore, the safety regulations require that an upper limit and a lower limit of the load factor $n_{z,B}$ in the direction of the z-axis of the Body Fixed Reference Frame B is never exceeded:

$$n_{Z,\min} - n_{Z,B}(t) \leq 0 \tag{6.10}$$

$$n_{Z,B}(t) - n_{Z,\max} \leq 0 \tag{6.11}$$

Besides the kinematic never-exceed speed $V_{K,max}$ given by the safety regulations, the aerodynamic velocity V_A of the aircraft must not go below the stall speed $V_{A,stall}$ of the aircraft:

$$V_{A,stall} - V_A(t) \leq 0 \tag{6.12}$$

$$V_A(t) - V_{K,\max} \leq 0 \tag{6.13}$$

Additional inequality path constraints are due to aircraft performance limitations with respect to the minimum and maximum angle of attack α_A

$$\alpha_{A,\min} - \alpha_A(t) \leq 0 \tag{6.14}$$

$$\alpha_A(t) - \alpha_{A,\max} \leq 0 \tag{6.15}$$

as well as the minimum and maximum roll rate p_K

$$p_{K,\min} - p_K(t) \leq 0 \tag{6.16}$$

$$p_K(t) - p_{K,\max} \leq 0 \tag{6.17}$$

Furthermore, in order to avoid dangerous flying that can lead to disqualification path constraints with respect to the flight path bank angle μ_K can be introduced. At this, large flight path bank angles and especially inverted flight close to the ground are avoided. The corresponding values for the various bounds of the inequality path constraints are listed in Table 7.

Path Constraint Specifications			due to
altitude	$z_{\min}$ [m]	7.5	ground clearance
load factor	$n_{z,\min}$ [-]	-2.0	race regulations
	$n_{z,\max}$ [-]	12.0	race regulations
velocity	$V_{A,stall}$ [m/s]	25.0	aircraft performance
	$V_{K,\max}$ [m/s]/[kts]	102.9/200.0	race regulations
angle of attack	$\alpha_{A,\min}$ [rad]/[°]	-0.35/-20.05	aircraft performance
	$\alpha_{A,\max}$ [rad] /[°]	0.35/20.05	aircraft performance
roll rate	$p_{K,\min}$ [rad/s] /[°/s]	-7.33/-420.0	aircraft performance
	$p_{K,\max}$ [rad/s] /[°/s]	7.33/+420.0	aircraft performance

Table 7. Path Constraints Specifications

At this, the stall velocity or the stall speed $V_{A,stall}$ is the minimum required aerodynamic velocity to sustain the aircraft weight in a 1g, steady-state level flight at sea level. With respect to the aircraft simulation model that is utilized for the trajectory optimization, various parameters have to be specified especially for the aerodynamic properties. For the full non-linear 6-DoF aircraft simulation model, the aerodynamic force coefficient equations are:

$$C_D = C_{D0} + k_L \cdot \left(C_L - C_{L,C_{D0}}\right)^2 + k_Y \cdot \left(C_Y\right)^2 \tag{6.18}$$

$$C_Y = C_{Y\beta} \cdot \beta + C_{Yp} \cdot \tilde{p}_A + C_{Yr} \cdot \tilde{r}_A + C_{Y\varsigma} \cdot \varsigma + C_{Y\xi} \cdot \xi \tag{6.19}$$

$$C_L = C_{L0} + C_{L\alpha} \cdot \alpha + C_{Lq} \cdot \tilde{q}_A + C_{L\eta} \cdot \eta \tag{6.20}$$

Aerodynamic Force Coefficients					
C_{D0}	0.0761	$C_{Y\beta}$	-0.589355	C_{L0}	0.055
k_Y	1.69677	C_{Yp}	0.042480	$C_{L\alpha}$	4.75
k_L	0.05134	C_{Yr}	0.048340	C_{Lq}	-3.479492
		$C_{Y\zeta}$	-0.195313	$C_{L\eta}$	-0.073242
		$C_{Y\xi}$	0.195313		

Table 8. Aerodynamic Force Coefficients

The aerodynamic moment coefficients are calculated from:

$$C_l = C_{l\xi} \cdot \xi + C_{l\varsigma} \cdot \varsigma + C_{lp} \cdot \tilde{p}_A + C_{lr} \cdot \tilde{r}_A + C_{l\beta} \cdot \beta \tag{6.21}$$

$$C_m = C_{m0} + C_{m\alpha} \cdot \alpha + C_{m\eta} \cdot \eta + C_{mq} \cdot \tilde{q}_A \tag{6.22}$$

$$C_n = C_{n\xi} \cdot \xi + C_{n\varsigma} \cdot \varsigma + C_{np} \cdot \tilde{p}_A + C_{nr} \cdot \tilde{r}_A + C_{n\beta} \cdot \beta \tag{6.23}$$

Aerodynamic Moment Coefficients					
$C_{l\beta}$	0.024902	C_{m0}	-0.004883	$C_{n\beta}$	0.149902
C_{lp}	-0.583008	$C_{m\alpha}$	-0.145406	C_{np}	0.014648
C_{lr}	-0.087891	C_{mq}	-16.930176	C_{nr}	-0.157715
$C_{l\zeta}$	0.001	$C_{m\eta}$	-0.634766	$C_{n\zeta}$	0.170898
$C_{l\xi}$	-0.303711			$C_{n\xi}$	-0.014648

Table 9. Aerodynamic Moment Coefficients

The implemented aerodynamic derivatives are listed in Table 8 respectively Table 9 and shall represent a generic aerobatic aircraft. The corresponding technical parameters can be found in Table 6 where it is mentioned that the remaining moments of inertia are set to zero.

For the computation of the thrust force T, the following equation is utilized:

$$T = T_{ref} \cdot \left(\frac{\rho}{\rho_{ref}} \right)^{n_\rho} \cdot \left(\frac{V_A^G}{V_{ref}} \right)^{n_V} \cdot \delta_{T,CMD} \tag{6.24}$$

where T_{ref} is the engine's reference thrust, V_{ref} the reference velocity, ρ_{ref} the reference air density and n_ρ the density exponent. The exponent n_V gives the dependency of the thrust on the aerodynamic velocity.

For the air race trajectory optimization problem, a flat, non-rotating Earth has been assumed due to the very limited spatial extent of the air races. Furthermore, the density has been set constant because the maximum change in the aircraft's altitude is very small and thus the influence of the static atmosphere model is negligible.

6.2 Minimum Time Air Race Trajectories

In the preceding chapter, the air race trajectory optimization problem has been stated in detail. In this chapter, the results for the 3-DoF point-mass simulation model and the full non-linear 6-DoF simulation model for the race track layout shown in chapter 6.3.3.1 will be given that have been obtained using the multiple shooting method of chapter 2.2. Out of comparison reasons, the same control discretization grid has been chosen for the point-mass simulation model and the 6-DoF simulation model. Therefore, the race trajectory has been split up into eight phases according to the race gates and a certain number of equally distributed grid points has been defined for each single phase according to the approximate phase durations (Table 10).

Phase Number	Number of Grid Points *n*
1	40
2	20
3	20
4	40
5	40
6	100
7	80
8	80

Table 10. Control Discretization

For the point-mass simulation model, the final race time evaluates to 44.74s. Fig. 43 shows the optimized time histories for the controls of the point-mass simulation model that are the angle of attack $\alpha_{A,CMD}$, the sideslip angle $\beta_{A,CMD}$, the first order time derivative of the bank angle $\dot{\mu}_{A,CMD}$ and the thrust lever position $\delta_{T,CMD}$. Furthermore, the upper and lower bounds for the controls are drawn. Here it can be seen that the full roll rate capability is utilized for the roll maneuvers at the various air race gates. The sideslip angle $\beta_{A,CMD}$ has been limited to ±15° assuming that for a sideslip angle of ±15° the linearly approximated sideforce curve equals roughly the maximum achievable sideforce before the vertical tail of the aerobatic aircraft stalls. The thrust lever position is at its upper boundary all the time.

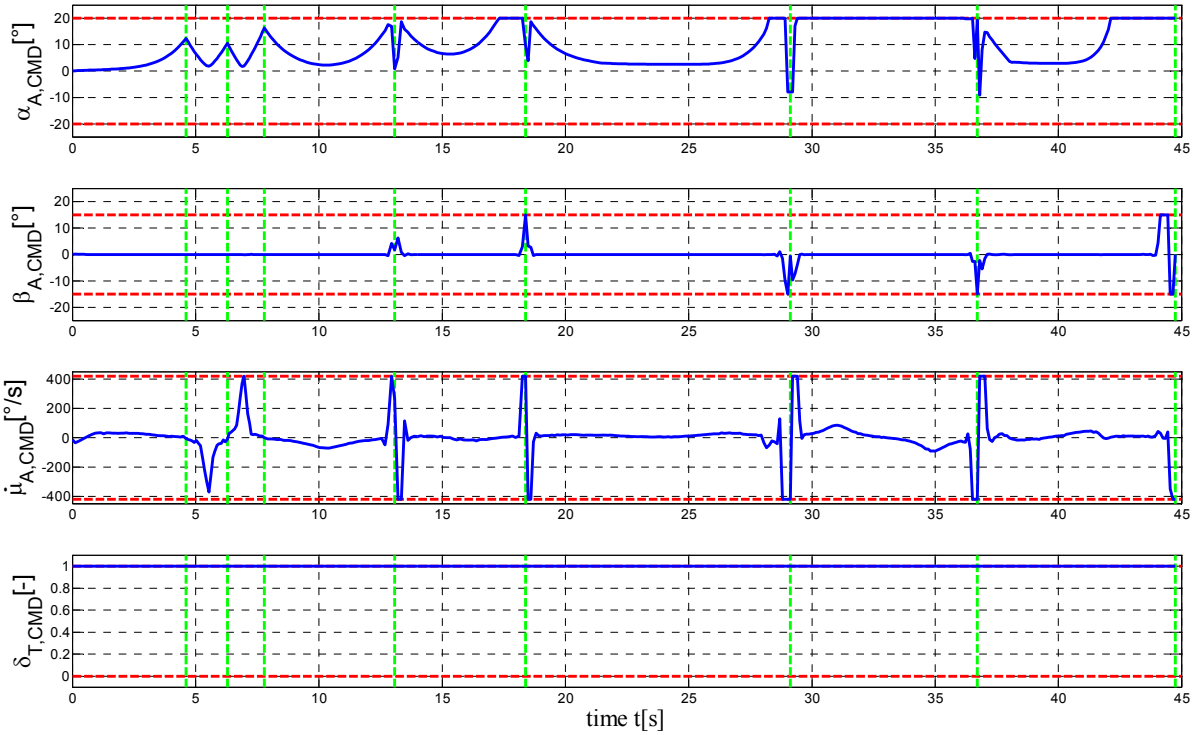

Figure 43. Controls (Point-Mass Simulation Model)

In Fig. 44 and Fig. 45, the according time histories for the translational states and the position states as well as the implemented path constraints are shown. Finally, in Fig. 46 the load factor in the direction of the *z*-axis of the Body-Fixed Frame *B* together with its lower and its upper bound are depicted.

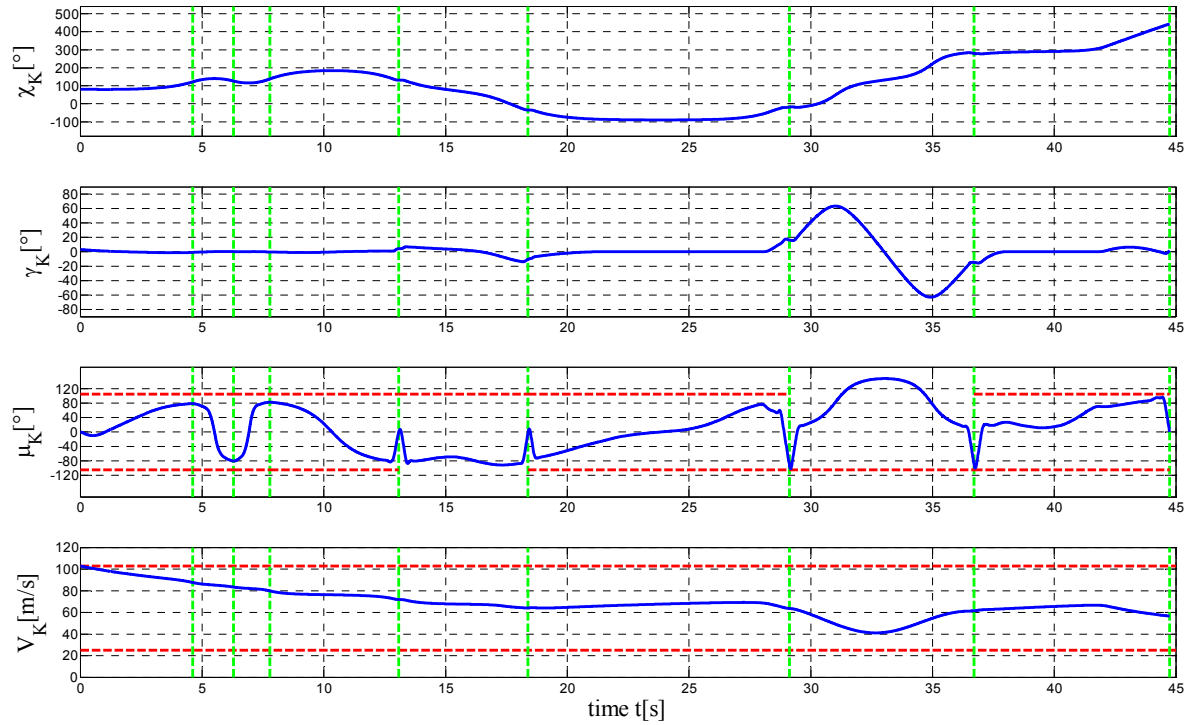

Figure 44. Translational States (Point-Mass Simulation Model)

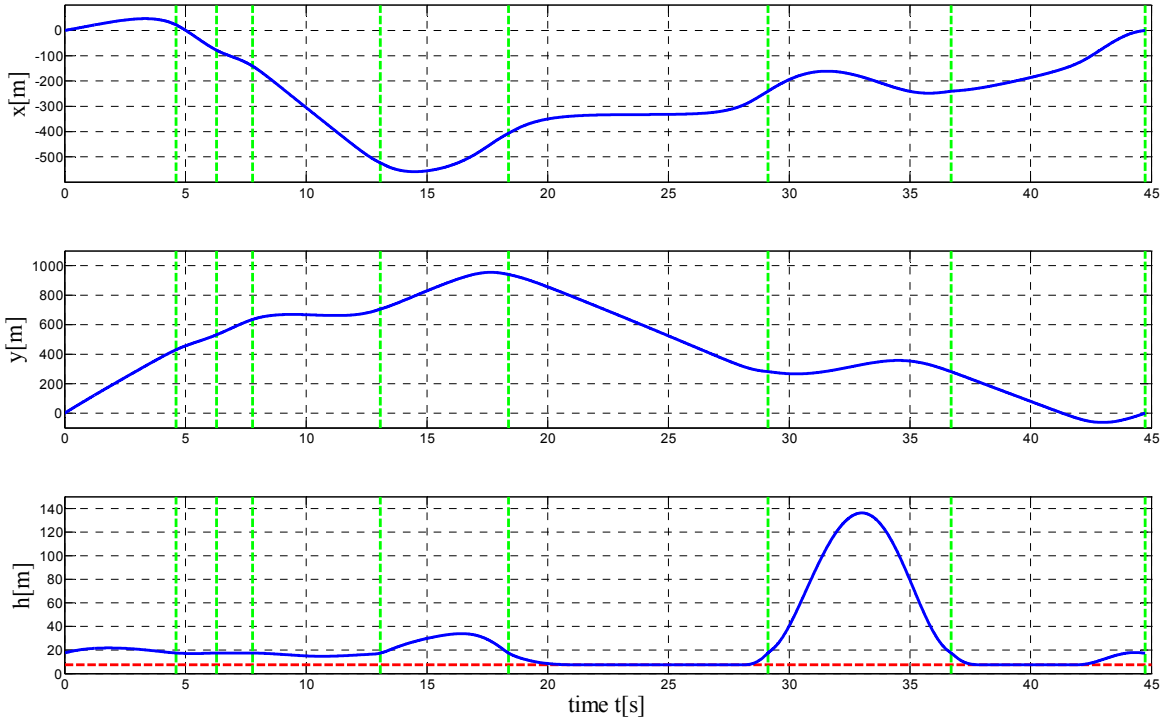

Figure 45. Position States (Point-Mass Simulation Model)

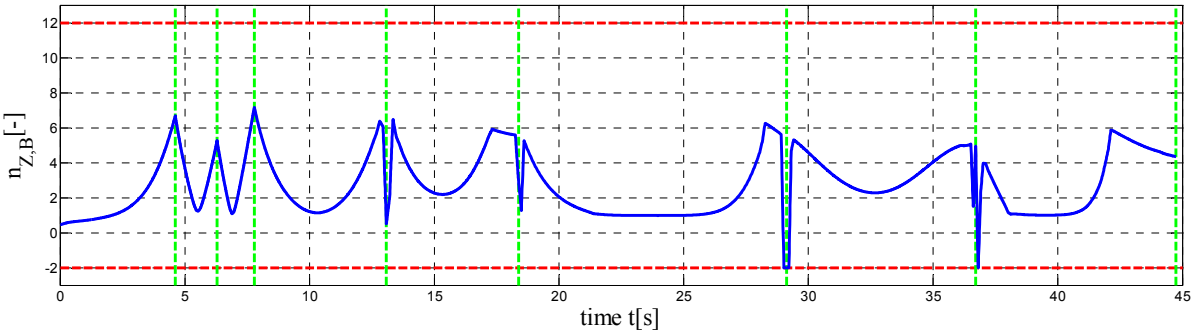

Figure 46. Load Factor $n_{Z,B}$ (Point-Mass Simulation Model)

In Fig. 47, the three-dimensional optimal air race trajectory for the full non-linear 6-DoF simulation model is shown that has been obtained by applying the optimization algorithm outlined in chapter 4. Here it is mentioned that the various optimization tasks throughout the optimization algorithm have been accomplished utilizing Lagrange cost functions w.r.t. the squared control derivatives in order to avoid possible control oscillations and to reduce the inherent highly non-linear dynamics of the resulting trajectories and to allow for a successful completion of the trajectory simulation tasks subsequent to the optimization tasks.

Figure 47. Time-Optimal Race Trajectory for the 6-DoF Simulation Model

Although the race track is quasi two-dimensional, i.e. the race gates are all about on the same level, the resulting optimal trajectory is three-dimensional. This is especially true for the 270°-turn that is required for flying through the "Quadro": here, the aircraft pulls up in order to shorten the flight time for this maneuver. The final minimum race time for flying one round of the described race course equals 45.82s. Here, a Lagrange cost with respect to the second order time derivatives of the control surface deflections (see Eq. (6.25)) has been introduced to avoid undesirable oscillations of the control surface deflections.

$$L = k_L \cdot \int_0^{t_f} \left(k_\xi \cdot \ddot{\xi}^2 + k_\eta \cdot \ddot{\eta}^2 + k_\varsigma \cdot \ddot{\varsigma}^2 \right) dt \tag{6.25}$$

In Fig. 48, the time histories of the commanded control surface deflections together with the real control surface deflections are given and their lower and upper bounds are depicted. As can be seen, the whole trajectory is flown at full thrust.

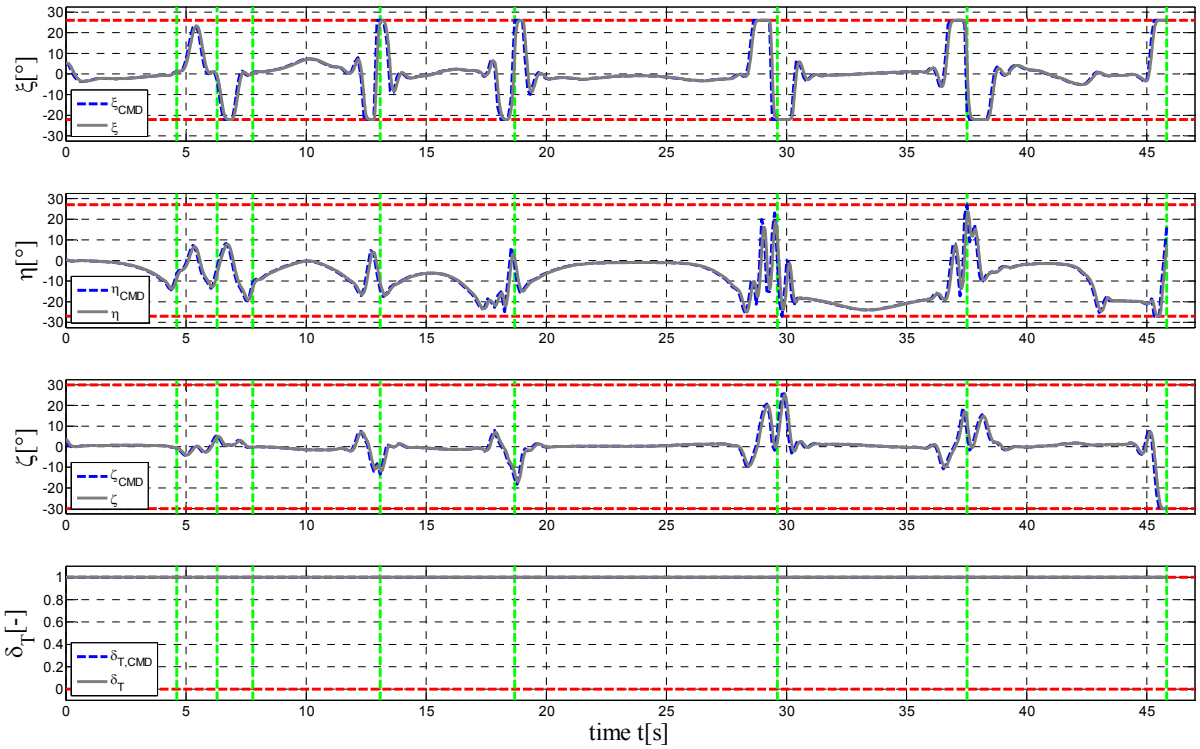

Figure 48. Commanded and real Control Values (6-DoF Simulation Model)

In Fig. 49, the rotational states for the optimized air race trajectory are shown. With regard to the rotational states, only the roll rate has been limited. In contrary to the 3-DoF point-mass simulation model where the bounds with respect to the first order time derivative of the bank angle are active multiple times the 6-DoF simulation model does not reach the roll rate limits.

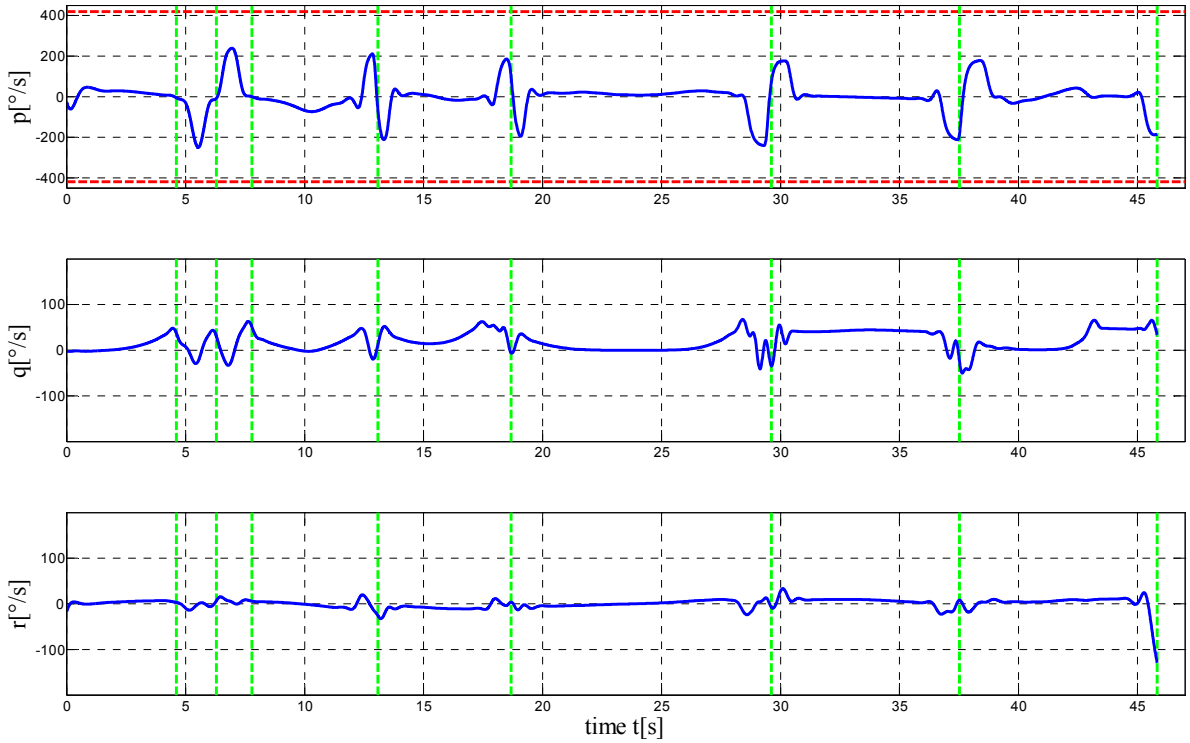

Figure 49. Rotational States (6-DoF Simulation Model)

In Fig. 50, the time history of the aircraft's kinematic velocity together with the angle of attack and the sideslip angle are shown. Starting with the maximum allowed kinematic speed that is 200.0kts or 102.9m/s, the velocity of the aircraft decreases although the optimal race trajectory is completely flown at full thrust, i.e. the thrust lever position is permanently set to one as can be seen from Fig. 48.

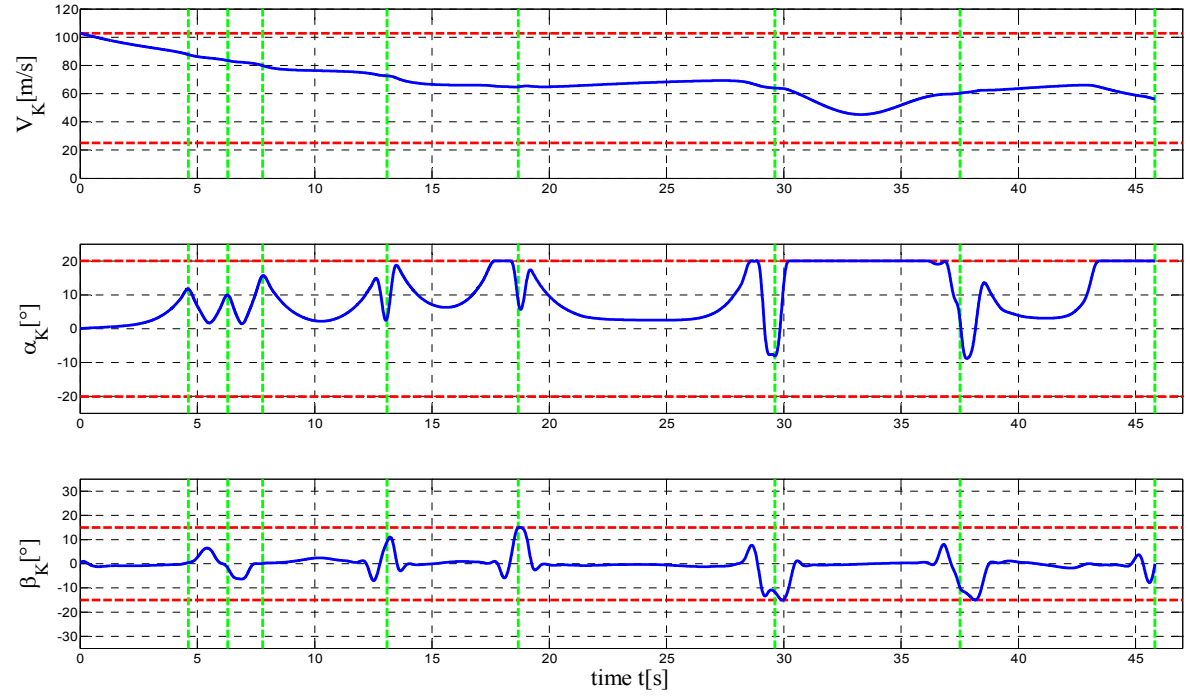

Figure 50. Kinematic Velocity and Attitude States (6-DoF Simulation Model)

In Fig. 51, the states for the flight-path angles are given. Here, the bank angle limitation is reached at a single point in the 8th phase, while for the 3-DoF simulation model the bank angle limitation does not become active at all.

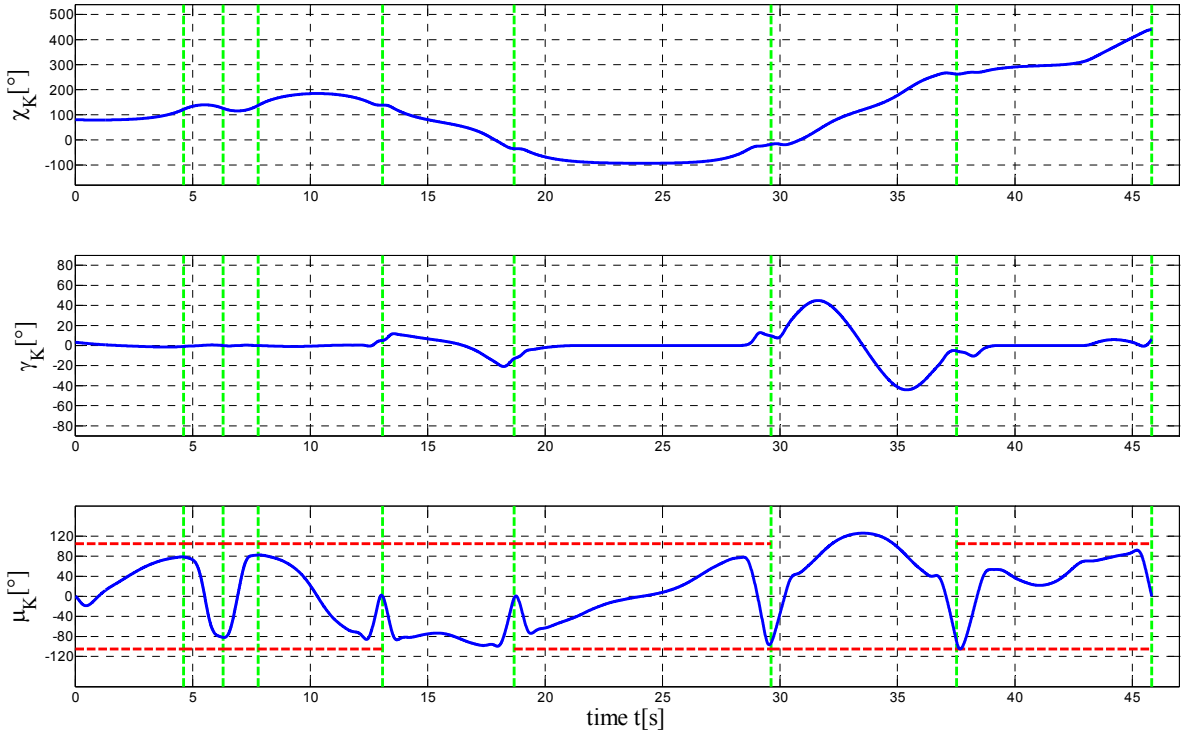

Figure 51. Flight-Path Variables (6-DoF Simulation Model)

Finally, in Fig. 52 the states of the position variables are shown and in Fig 53 the time history for the load factor in the direction of the *z*-axis of the Body-Fixed Reference Frame *B* is drawn. Here again, as for the point-mass simulation model, only the lower bound with respect to the load factor becomes active.

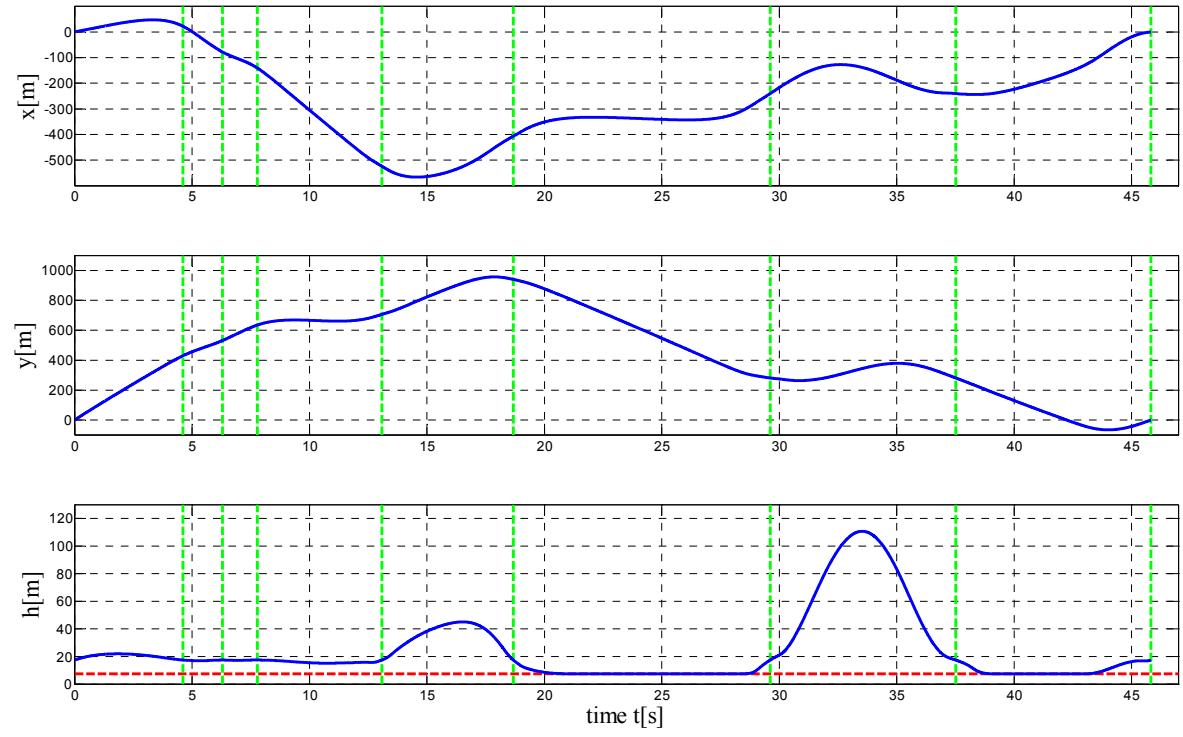

Figure 52. Position States (6-DoF Simulation Model)

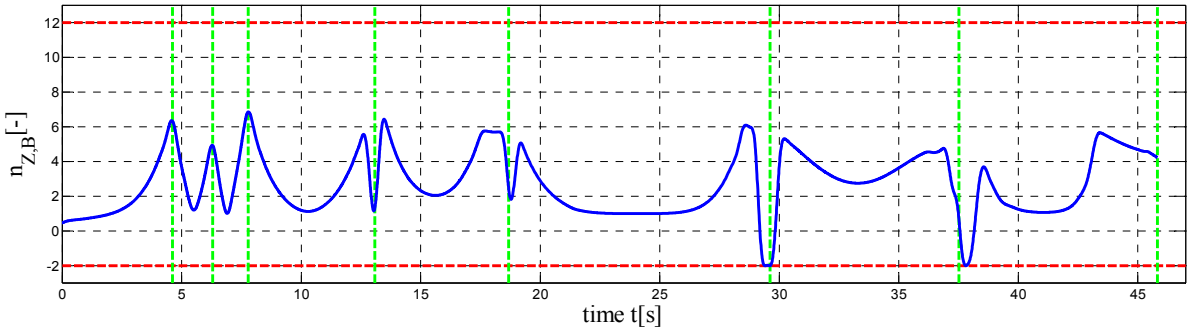

Figure 53. Load Factor $n_{Z,B}$ (6-DoF Simulation Model)

6.3 Increasing the Safety of Air Races

For air races, the aspect that has to be considered more than anything else is safety. First of all it has to be the self-set requirement of the organizer that safety is of paramount interest and thus the most important criterion overruling everything else. Furthermore local public and aviation authorities must be convinced of the safety concept to grant permission for the event on a local basis. If one wants to design the layout of the race track such that a certain safety criterion is maximized (or minimized), a bilevel optimal control problem as described in chapter 5 arises. The upper level optimization problem is a parameter optimization problem and ought to place the race gates such that the respective safety criterion is maximized or minimized. At each iteration of the upper level optimization problem, its objective depends on the solution of a lower level optimal control problem that gives the minimum possible race time for fixed positions of the race gates. Fig. 54 depicts the basic principle of the bilevel optimal control problem that has to be solved in order to achieve a maximum level of safety for the respective race track.

The goal of the upper level optimization problem is to position the air race gates such that a certain safety criterion is maximized or minimized. Besides the northward and eastward positions x and y of the gates also the azimuth angles ψ of the gates are considered as optimization parameters so that the parameter vector $\mathbf{p}$ is given by

$$\mathbf{p} = \left[x_i, y_i, \psi_i\right] \quad i = 1,\ldots,s \tag{6.26}$$

where s is the number of race gates.

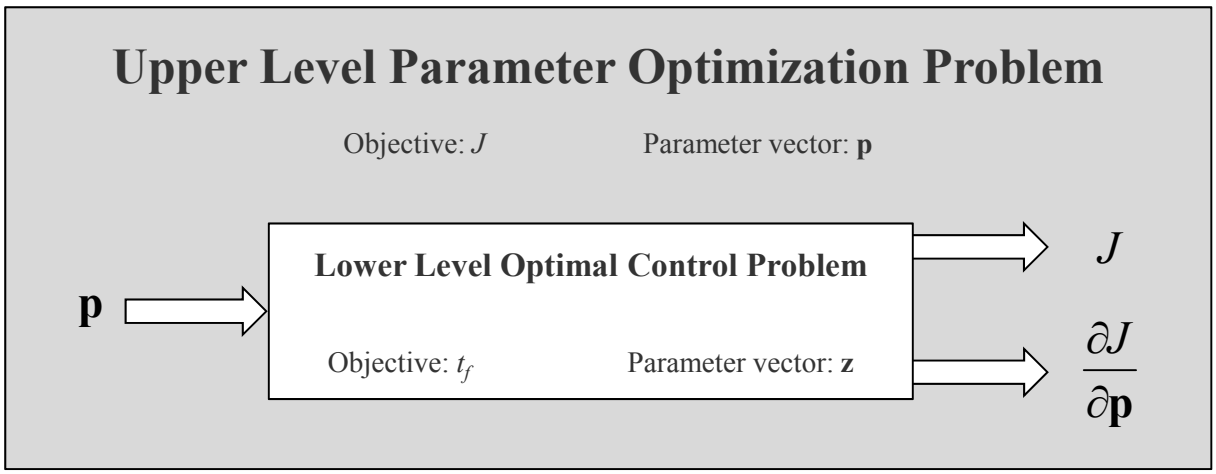

Figure 54. Safety Bilevel Optimal Control Problem

If the race track layout contains a “Chicane” (this means a slalom), the y-axis of the local Navigation Frame N is aligned with the direction of the „Chicane“. Then, additional constraints are added to the upper level optimal control problem that enforce equal distances between the various pylons of the „Chicane“ as well as the placement of the pylons in a straight line. If the „Chicane“ consists of three single pylons, the corresponding constraints reads:

$$x_2 - x_1 = \Delta x \tag{6.27}$$

$$x_3 - x_1 = 0 \tag{6.28}$$

$$y_3 - 2 \cdot y_2 + y_1 = 0 \tag{6.29}$$

where x_i, $i = 1, 2, 3$ and y_i, $i = 1, 2, 3$ are the positions of the race gates that have to be passed by the aircraft (see Fig. 55). Eqs. (6.27) and (6.28) enforce an S-curve while constraint (6.29) positions the second gate in the middle of the first and the third gate. At this it is ensured that the principal layout of the „Chicane“ is kept throughout the solution of the bilevel optimal control problem. Furthermore, the upper level parameter optimization problem has to be augmented by constraints that enforce the “Quadro” respectively the identical position for the first and the final race gate:

$$x_2 - x_1 = 0 \tag{6.30}$$

$$y_2 - y_1 = 0 \tag{6.31}$$

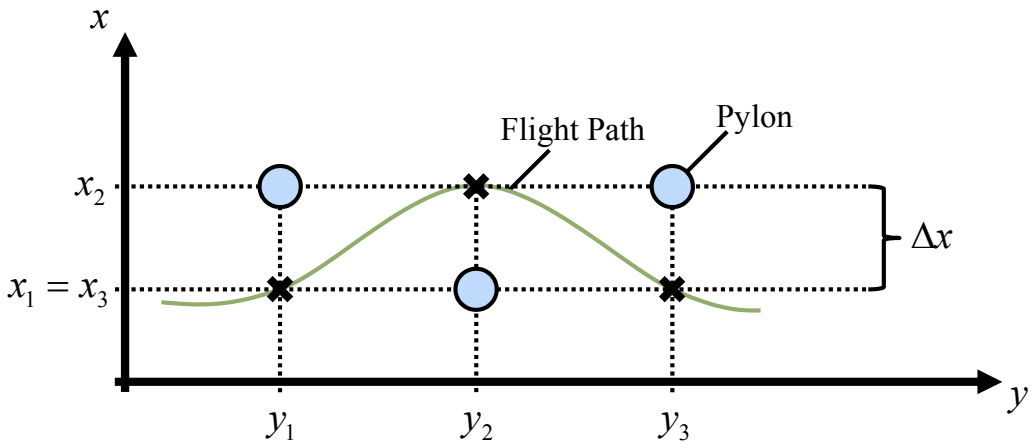

Figure 55. Layout for the Implementation of the Constraints for the Chicane

With respect to the “Quadro” that is in fact a 270°-turn, the corresponding constraint for the heading angle ψ is:

$$\psi_2 - \psi_1 = \frac{3}{2}\pi \tag{6.32}$$

while for the start and the finish gate the heading angle ψ has to obey the following relationship:

$$\psi_2 - \psi_1 = t \cdot \pi \tag{6.33}$$

where t is the number of turns the pilot has to accomplish during the race.

The safety criteria that are considered in the following lead to *minimax* problems given by

$$\min_{\mathbf{p}}\left(\max S(t)\right) \tag{6.34}$$

respectively *maximin* problems of the following form

$$\max_{\mathbf{p}}\left(\min S(t)\right) \tag{6.35}$$

where $S(t)$ denotes the measure for the safety criterion that is to be maximized or minimized. Those *minimax* respectively *maximin* problems can be transformed into standard optimization problems resulting in

$$\min_{\mathbf{p}}\left(S_{\max}\right) \tag{6.36}$$

subject to

$$S_{\max} - S(t) \geq 0 \tag{6.37}$$

for the *minimax* problem respectively

$$\min_{\mathbf{p}}\left(-S_{\min}\right) \tag{6.38}$$

subject to

$$S(t) - S_{\min} \geq 0 \tag{6.39}$$

for the *maximin* problem, where $S_{\max}$ and $S_{\min}$ denote the upper bound respectively the lower bound with respect to the selected safety criterion S. Supposing that that the safety criterion $S(t)$ is a function of the state history $x(t)$, an output time history $y(t)$ or the parameter vector $\mathbf{z}$ of the lower level optimal control problem, the Jacobian of the constraints (6.37) respectively (6.39) can be directly computed using the sensitivity results of chapter 5.2:

$$\frac{\partial\left(S_{\max} - S(t)\right)}{\partial \mathbf{p}^T} = -\frac{\partial S(t)}{\partial \mathbf{p}^T} = -\frac{\partial S(t)}{\partial x(t)} \cdot \frac{\partial x(t)}{\partial \mathbf{p}^T} - \frac{\partial S(t)}{\partial y(t)} \cdot \frac{\partial y(t)}{\partial \mathbf{p}^T} - \frac{\partial S(t)}{\partial \mathbf{z}^T} \cdot \frac{\partial \mathbf{z}}{\partial \mathbf{p}^T} \tag{6.40}$$

$$\frac{\partial\left(S(t) - S_{\min}\right)}{\partial \mathbf{p}^T} = \frac{\partial S(t)}{\partial \mathbf{p}^T} = \frac{\partial S(t)}{\partial x(t)} \cdot \frac{\partial x(t)}{\partial \mathbf{p}^T} + \frac{\partial S(t)}{\partial y(t)} \cdot \frac{\partial y(t)}{\partial \mathbf{p}^T} + \frac{\partial S(t)}{\partial \mathbf{z}^T} \cdot \frac{\partial \mathbf{z}}{\partial \mathbf{p}^T} \tag{6.41}$$

As for the lower level optimal control problems, SNOPT (Ref. [Gill, 2007]) has been used for the solution of the upper level optimization problem.

6.3.1 Computation of Sensitivity Information for the Air Race Bilevel Optimal Control Problem

Since for the lower level optimal control problem the objective is the final race time t_f, the matrices $\mathbf{J_{zz}}$ and $\mathbf{J_{zp}}$ of Eq. (5.8) respectively Eq. (5.11) in chapter 5.2 evaluate to zero. Furthermore, in Eq. (5.11) the tensor $\mathbf{G_{zp}}$ only contains zeros. This is due to the fact that the parameters $\mathbf{p}$ which are the positions of the air race gates are only involved in the initial and final boundary constraints as well as the interior point constraints (see Eqs. (6.2), (6.3) and (6.7)). Differentiating the constraints with respect to the parameter vector $\mathbf{z}$ of the lower level optimal control problem cancels out the parameters $\mathbf{p}$ so that the parameters $\mathbf{p}$ are not incorporated in the Jacobian $\mathbf{G_z}$ any more, hence the tensor $\mathbf{G_{zp}}$ being a tensor of zeros. Thus, the matrix $\mathbf{L_{zp}}$ of Eq. (5.11) is a matrix of zeros, too. The matrix $\mathbf{G_p}$ also contains solely zeros except for the derivatives of the initial boundary conditions, the final boundary conditions and

the interior points conditions where the parameter vector **p** is involved linearly as can be seen from Eqs. (6.2), (6.3) and (6.7). Thus, the respective entries of the matrix $\mathbf{G_p}$ equal -1 when differentiating any of the initial boundary conditions, the final boundary conditions or the interior point conditions with respect to the parameter vector **p** that comprises the northward positions x, the eastward positions y and the headings ψ of the race gates, e.g.

$$\frac{\partial\left(x(t_0)-x_{StartGate}\right)}{\partial \mathbf{p}^T}=\frac{\partial\left(x(t_0)-x_1\right)}{\partial \mathbf{p}^T}=\begin{bmatrix}-1 & 0 & \dots & 0\end{bmatrix} \tag{6.42}$$

if x_1 was the first element of the parameter vector **p**.

6.3.2 Safety Criteria for Air Races

In the subsequent chapters, various safety criteria and the computation of their gradients are explained. In chapter 6.3.3, optimal air race tracks for these safety criteria are given.

6.3.2.1. Minimum Distance to Crowd

A very crucial safety criterion is the distance to the spectators. During the air race, the pilots have to keep a prescribed minimum distance to the spectators if they do not want to be disqualified. For the computation of the distance to the crowd, the position of the aircraft is projected into the horizontal plane and the spectator areas are wrapped by polygons in the horizontal plane that represent the foremost line of the spectators. Every polygon consists of multiple piecewise linear curves and is called *crowd line*. For at least one point of all the polygons, the horizontal distance to the actual position of the aircraft is minimal, referred to as the minimum distance to crowd $d_C(t)$. The minimum distance to the i-th segment of the k-th polygon is computed by

$$d_{C,k,i}(t)=\sqrt{\left(x_{FP,k,i}(t)-x(t)\right)^2+\left(y_{FP,k,i}(t)-y(t)\right)^2} \tag{6.43}$$

where x_{FP} and y_{FP} denote the northward respectively the eastward position of the perpendicular footpoint on the respective segment (see Fig. 56). The position of the footpoint is given by

$$x_{FP,k,i}(t)=x_{1,k,i}+\lambda_{k,i}(t)\cdot\left(x_{2,k,i}-x_{1,k,i}\right) \tag{6.44}$$

$$y_{FP,k,i}(t)=y_{1,k,i}+\lambda_{k,i}(t)\cdot\left(y_{2,k,i}-y_{1,k,i}\right) \tag{6.45}$$

where x_1, y_1, x_2 and y_2 are the coordinates of the end points of segment (k, i). $\lambda(t)$ evaluates to

$$\lambda_{k,i}(t)=\frac{\left(x(t)-x_{1,k,i}\right)\cdot\left(x_{2,k,i}-x_{1,k,i}\right)+\left(y(t)-y_{1,k,i}\right)\cdot\left(y_{2,k,i}-y_{1,k,i}\right)}{\left(x_{2,k,i}-x_{1,k,i}\right)^2+\left(y_{2,k,i}-y_{1,k,i}\right)^2} \tag{6.46}$$

and only if $\lambda(t) \in [0, 1]$ a perpendicular footpoint on the respective segments exists. Otherwise, if $\lambda(t) < 0$ the footpoint is set to the first endpoint

$$x_{FP,k,i}(t)=x_{1,k,i} \tag{6.47}$$

$$y_{FP,k,i}(t)=y_{1,k,i} \tag{6.48}$$

and if $\lambda(t) > 0$, the footpoint is set to the second endpoint:

$$x_{FP,k,i}(t)=x_{2,k,i} \tag{6.49}$$

$$y_{FP,k,i}(t) = y_{2,k,i} \tag{6.50}$$

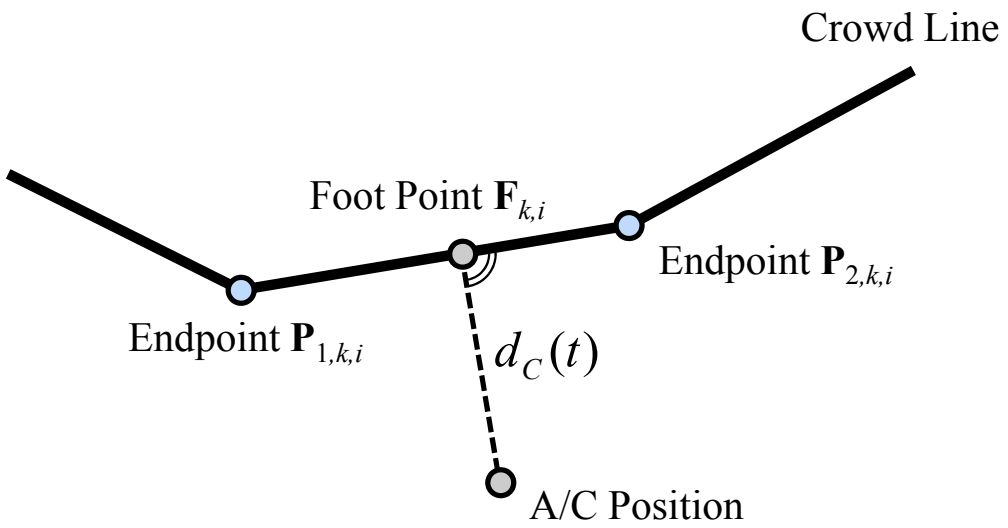

Figure 56. Calculation of Footpoint

Then, the overall minimum distance $d_C(t)$ with respect to all segments is found by

$$d_C(t) = \min d_{C,k,i}(t) \qquad k = 1,...,p \qquad i = 1,...,s_k \tag{6.51}$$

where p is the number of polygons and s_k the number of segments of the k-th polygon. Table 11 states the algorithm for the computation of the minimum distance to the crowd lines for a given position of the aircraft.

ALGORITHM 6.I

1 For $k = 1, \ldots, p$
  For $i = 1, \ldots, s_k$
    Compute $\lambda_{k,i}(t)$ by Eq. (6.46)
    If $\lambda_{k,i}(t) < 0$
      Set actual footpoint to first endpoint of segment (k, i)
    Else if $\lambda_{k,i}(t) > 1$
      Set actual footpoint to second endpoint of segment (k, i)
    Else if $\lambda_{k,i}(t) \in [0, 1]$
      Calculate actual footpoint by Eqs. (6.44) and (6.45)
    Compute distance to crowd $d_{C,k,i}(t)$ for segment (k, i) by Eq. (6.43)
2 Select minimum distance to crowd $d_C(t)$ from all $d_{C,k,i}(t)$

Table 11. Computation of Minimum Distance to Crowd

The gradient of the distance to crowd $d_C(t)$ with respect to the parameter vector **p** of the upper level parameter optimization problem is:

$$\frac{\partial d_C(t)}{\partial \mathbf{p}^T} = \frac{\partial d_C(t)}{\partial x(t)} \cdot \frac{\partial x(t)}{\partial \mathbf{p}^T} + \frac{\partial d_C(t)}{\partial y(t)} \cdot \frac{\partial y(t)}{\partial \mathbf{p}^T} \tag{6.52}$$

where the gradients $\partial x(t)/\partial \mathbf{p}$ and $\partial y(t)/\partial \mathbf{p}$ are obtained by the sensitivity analysis explained in chapter 5.2. The derivatives $\partial d_C(t)/\partial x(t)$ and $\partial d_C(t)/\partial y(t)$ can be calculated by differentiating Eq. (6.43) with respect to $x(t)$ respectively $y(t)$.

6.3.2.2. Minimum Time to Crowd

The time to crowd is defined as the time between the first deviation from the race track and the arrival at the spectators if the actual translation flight states are kept unchanged, i.e. if the aircraft continues on a straight trajectory. This safety criterion can be seen as a further development of the distance to crowd since it takes into account the current direction of motion of the aircraft while the preceding safety criterion only evaluates the current aircraft position. Thus, a small distance to the crowd line is acceptable if the aircraft is flying parallel to the crowd line while a large distance to the crowd line is required if the aircraft is heading directly towards the spectators.

Again, the spectator areas are specified by crowd lines as in chapter 6.3.2.1 and the actual velocity vector is projected into the horizontal plane. For the actual position of the aircraft, the time to crowd has to be computed with respect to all segments of all polygons. Therefore, it is checked first if the aircraft is flying parallel to the i-th segment of the k-th polygon by:

$$\dot{x}(t) \cdot \left(y_{2,k,i} - y_{1,k,i}\right) - \dot{y}(t) \cdot \left(x_{2,k,i} - x_{1,k,i}\right) = 0 \tag{6.53}$$

If so, the aircraft will never reach the respective segment and the minimum time to crowd is set to infinity. Otherwise, it is checked if the projection of the actual velocity vector into the horizontal plane points towards segment (k, i) by the following relationship:

$$\lambda_{k,i}(t) = \frac{\dot{x}(t) \cdot \left(y(t) - y_{1,k,i}\right) - \dot{y}(t) \cdot \left(x(t) - x_{1,k,i}\right)}{\dot{x}(t) \cdot \left(y_{2,k,i} - y_{1,k,i}\right) - \dot{y}(t) \cdot \left(x_{2,k,i} - x_{1,k,i}\right)} \tag{6.54}$$

where $\lambda(t)$ is obtained from Eq. (6.55) that gives the condition for the computation of the time to the crowd line $t_C(t)$ if the aircraft continues on a straight trajectory with constant velocity:

$$\begin{pmatrix} x(t) \\ y(t) \end{pmatrix} + \begin{pmatrix} \dot{x}(t) \\ \dot{y}(t) \end{pmatrix} \cdot t_{C,k,i}(t) = \begin{pmatrix} x_{1,k,i} \\ x_{2,k,i} \end{pmatrix} + \lambda_{k,i}(t) \cdot \begin{pmatrix} x_{2,k,i} - x_{1,k,i} \\ y_{2,k,i} - y_{1,k,i} \end{pmatrix} \tag{6.55}$$

Only if $\lambda(t) \in [0, 1]$, the prolongation of the actual velocity vector crosses the specific segment. In this case, the time until the segment is reached if the actual flight states are kept is computed by:

$$t_{C,k,i}(t) = \frac{\left(y(t) - y_{1,k,i}\right) \cdot \left(x_{2,k,i} - x_{1,k,i}\right) - \left(x(t) - x_{1,k,i}\right) \cdot \left(y_{2,k,i} - y_{1,k,i}\right)}{\dot{x}(t) \cdot \left(y_{2,k,i} - y_{1,k,i}\right) - \dot{y}(t) \cdot \left(x_{2,k,i} - x_{1,k,i}\right)} \tag{6.56}$$

If the resulting time value is negative, the aircraft is veering away from the segment and it does not cross this specific part of the crowd lines at any time. For all the situations where the aircraft does not cross a specific segment (k, i), the time to crowd with respect to this segment is set to infinity:

$$t_{C,k,i}(t) \to \infty \tag{6.57}$$

Then, the overall minimum time $t_C(t)$ with respect to all segments results from:

$$t_C(t) = \min t_{C,k,i}(t) \qquad k = 1,...,p \qquad i = 1,...,s_k \tag{6.58}$$

where p is the number of polygons and s_k the number of segments of the k-th polygon.

ALGORITHM 6.II

```
1   For k = 1, …, p
        For i = 1, …, s_k
            If Eq. (6.53) holds
                Set t_C,k,i(t) to infinity
            Else
                Compute λ_k,i(t) by Eq. (6.54)
                If λ_k,i(t) ∈ [0, 1]
                    Calculate minimum time to crowd t_C,k,i(t)
                    for segment (k, i) by Eq. (6.56)
                    If t_C,k,i(t) < 0
                        Set t_C,k,i(t) to infinity
                Else set t_C,k,i(t) to infinity
2   Select minimum time to crowd t_C(t) from all t_C,k,i(t)
```

Table 12. Computation of Minimum Time to Crowd

In Table 12, the algorithm for the computation of the minimum time to the crowd lines for the actual aircraft position is given.

The evaluation of the gradient of the time to crowd with respect to the parameter vector **p** is done by:

$$\frac{\partial t_C(t)}{\partial \mathbf{p}^T} = \frac{\partial t_C(t)}{\partial x(t)} \cdot \frac{\partial x(t)}{\partial \mathbf{p}^T} + \frac{\partial t_C(t)}{\partial y(t)} \cdot \frac{\partial y(t)}{\partial \mathbf{p}^T} + \frac{\partial t_C(t)}{\partial \dot{x}(t)} \cdot \frac{\partial \dot{x}(t)}{\partial \mathbf{p}^T} + \frac{\partial t_C(t)}{\partial \dot{y}(t)} \cdot \frac{\partial \dot{y}(t)}{\partial \mathbf{p}^T} \tag{6.59}$$

where the gradients $\partial x(t)/\partial \mathbf{p}$, $\partial y(t)/\partial \mathbf{p}$, $\partial \dot{x}(t)/\partial \mathbf{p}$ and $\partial \dot{y}(t)/\partial \mathbf{p}$ are obtained by the sensitivity analysis explained in chapter 5.2. The derivatives $\partial t_C(t)/\partial x(t)$, $\partial t_C(t)/\partial y(t)$, $\partial t_C(t)/\partial \dot{x}(t)$ and $\partial t_C(t)/\partial \dot{y}(t)$ can be evaluated by differentiating Eq. (6.56).

6.3.2.3. Minimum Time to Crowd based on the Normal Velocity Component

The time to crowd safety criterion can be modified so that the computation of the minimum time to crowd is based on the component of the velocity that is normal to a certain segment of the crowd line. Therefore, it first has to be checked if the aircraft is flying parallel to the i-th segment of the k-th polygon by:

$$\dot{x}(t) \cdot \left(y_{2,k,i} - y_{1,k,i}\right) - \dot{y}(t) \cdot \left(x_{2,k,i} - x_{1,k,i}\right) = 0 \tag{6.60}$$

If so, the normal velocity component with respect to this segment equals zero and the minimum time to crowd is set to infinity:

$$t_{C,k,i}(t) \to \infty \tag{6.61}$$

Next, it is verified by the following relationship if a perpendicular footpoint on the respective segment exists:

$$\lambda_{k,i}(t) = \frac{\left(x(t) - x_{1,k,i}\right)\cdot\left(x_{2,k,i} - x_{1,k,i}\right) + \left(y(t) - y_{1,k,i}\right)\cdot\left(y_{2,k,i} - y_{1,k,i}\right)}{\left(x_{2,k,i} - x_{1,k,i}\right)^2 + \left(y_{2,k,i} - y_{1,k,i}\right)^2} \tag{6.62}$$

If $\lambda(t) \notin [0, 1]$, no perpendicular footpoint and thus no normal velocity component can be computed for the respective segment. Consequently, the minimum time to crowd based on the normal velocity is set to infinity, see Eq. (6.61). Otherwise, a perpendicular footpoint on the particular segment is existent and the position of the footpoint evaluates to:

$$x_{FP,k,i}(t) = x_{1,k,i} + \lambda_{k,i}(t)\cdot\left(x_{2,k,i} - x_{1,k,i}\right) \tag{6.63}$$

$$y_{FP,k,i}(t) = y_{1,k,i} + \lambda_{k,i}(t)\cdot\left(y_{2,k,i} - y_{1,k,i}\right) \tag{6.64}$$

Then, the velocity component that is normal to the respective segment is given by:

$$V_{\perp,k,i}(t) = \frac{\mathbf{d}_{C,k,i}(t) \circ \mathbf{V}_{hor}(t)}{d_{C,k,i}(t)} = \frac{1}{d_{C,k,i}(t)} \begin{pmatrix} x_{FP,k,i}(t) - x(t) \\ y_{FP,k,i}(t) - y(t) \end{pmatrix} \circ \begin{pmatrix} \dot{x}(t) \\ \dot{y}(t) \end{pmatrix} \tag{6.65}$$

where the minimum distance $d_{C,k,i}(t)$ is given by Eq. (6.43). Next, the time to crowd $t_{C,k,i}(t)$ based on the normal velocity component is obtained by the following equation:

$$t_{C,k,i}(t) = \frac{d_{C,k,i}(t)}{V_{\perp,k,i}(t)} = \frac{\left(x_{FP,k,i}(t) - x(t)\right)^2 + \left(y_{FP,k,i}(t) - y(t)\right)^2}{\left[\left(x_{FP,k,i}(t) - x(t)\right)\cdot\dot{x}(t) + \left(y_{FP,k,i}(t) - y(t)\right)\cdot\dot{y}(t)\right]} \tag{6.66}$$

Here, if the time to crowd $t_{C,k,i}(t)$ takes a negative value the aircraft is veering away from the respective segment and the time to crowd is set to infinity since in this case there is no threat for this segment.

Finally, the overall minimum time to crowd $t_C(t)$ with respect to all segments is found by:

$$t_C(t) = \min t_{C,k,i}(t) \qquad k = 1, ..., p \qquad i = 1, ..., s_k \tag{6.67}$$

where p is the number of polygons and s_k the number of segments of the k-th polygon.

Table 13 states the algorithm for the computation of the minimum time to the crowd lines based on the normal velocity component for a given position of the aircraft.

The gradient of the time to crowd $t_C(t)$ with respect to the parameter vector $\mathbf{p}$ of the upper level parameter optimization problem evaluates to:

$$\frac{\partial t_C(t)}{\partial \mathbf{p}^T} = \frac{\partial t_C(t)}{\partial x(t)}\cdot\frac{\partial x(t)}{\partial \mathbf{p}^T} + \frac{\partial t_C(t)}{\partial y(t)}\cdot\frac{\partial y(t)}{\partial \mathbf{p}^T} + \frac{\partial t_C(t)}{\partial \dot{x}(t)}\cdot\frac{\partial \dot{x}(t)}{\partial \mathbf{p}^T} + \frac{\partial t_C(t)}{\partial \dot{y}(t)}\cdot\frac{\partial \dot{y}(t)}{\partial \mathbf{p}^T} \tag{6.68}$$

where the gradients $\partial x(t)/\partial \mathbf{p}$, $\partial y(t)/\partial \mathbf{p}$, $\partial \dot{x}(t)/\partial \mathbf{p}$ and $\partial \dot{y}(t)/\partial \mathbf{p}$ are obtained by the sensitivity analysis explained in chapter 5.2. The derivatives $\partial t_C(t)/\partial x(t)$, $\partial t_C(t)/\partial y(t)$, $\partial t_C(t)/\partial \dot{x}(t)$ and $\partial t_C(t)/\partial \dot{y}(t)$ are obtained by differentiating Eq. (6.66).

ALGORITHM 6.III

1 For $k = 1, \ldots, p$

For $i = 1, \ldots, s_k$

If Eq. (6.60) holds

Set $t_{C,k,i}(t)$ to infinity

Else

Compute $\lambda_{k,i}(t)$ by Eq. (6.62)

If $\lambda_{k,i}(t) \in [0, 1]$

Calculate minimum time to crowd $t_{C,k,i}(t)$ for segment (k, i) by Eqs. (6.63) to (6.66)

If $t_{C,k,i}(t) < 0$

Set $t_{C,k,i}(t)$ to infinity

Else set $t_{C,k,i}(t)$ to infinity

2 Select minimum time to crowd $t_C(t)$ from all $t_{C,k,i}(t)$

Table 13. Computation of the Minimum Time to Crowd based on the Normal Velocity Component

6.3.2.4. Minimum *Maximum Directed Energy*-equivalent Time

The energy that is directed towards the crowd is regarded as another important safety criterion: The higher the maximum directed energy is, the less safe is the race track. This safety criterion is even an advancement of the time to crowd since it involves also the absolute value of the kinematic velocity besides the current position and the actual direction of motion of the respective airplane. At this, it is taken into account that an aircraft heading directly towards the crowd line at a high kinematic velocity poses the less danger to the spectators the larger the distance of the aircraft to the crowd line is. Thus, situations where the aircraft is heading directly towards the spectators at a high total energy are rated highly critical if the aircraft is close to the crowd line.

The computation of the maximum directed energy is based upon the total velocity $V_{total}(t)$ that is derived from the actual total energy of the aircraft:

$$E_{total}(t) = \frac{1}{2} m V_{total}^2(t) = E_{kin}(t) + E_{pot}(t) = \frac{1}{2} m V^2(t) + mg(-z(t)) \tag{6.69}$$

where $E_{kin}(t)$ is the kinetic energy and $E_{pot}(t)$ the actual potential energy of the aircraft. The total velocity vector $\mathbf{V}_{total}(t)$ then equals:

$$\mathbf{V}_{total}(t) = \frac{\mathbf{V}(t)}{V(t)} V_{total}(t) = \mathbf{V}(t) \sqrt{1 - 2g \frac{z(t)}{V^2(t)}} \tag{6.70}$$

Next, the total velocity vector $\mathbf{V}_{total}(t)$ is projected into the horizontal plane:

$$\mathbf{V}_{total,hor}(t) = \begin{pmatrix} \dot{x}(t) \\ \dot{y}(t) \end{pmatrix} \sqrt{1 - 2g \frac{z(t)}{V^2(t)}} \tag{6.71}$$

The portion of the total velocity vector $\mathbf{V}_{total}(t)$ that is directed towards a specific point on the crowd lines is given by:

$$\mathbf{V}_{C,k,i}(t) = \frac{\mathbf{d}_{C,hor,k,i}(t)}{d_{C,hor,k,i}(t)} \cdot \cos\kappa(t) \cdot V_{total,hor}(t) \tag{6.72}$$

where $V_{total,hor}(t)$ is the absolute value of the total velocity vector $\mathbf{V}_{total,hor}(t)$. $\kappa(t)$ represents the angle between the total velocity vector $\mathbf{V}_{total,hor}(t)$ and the direction vector $\mathbf{d}_{C,hor,k,i}(t)$ in the horizontal plane (see Eq. (6.76) and Fig. 57). The direction vector $\mathbf{d}_{C,hor,k,i}(t)$ is the vector between the actual aircraft position and a specific point on the i-th segment of the k-th polygon of the crowd lines:

$$\mathbf{d}_{C,hor,k,i}(t) = \begin{pmatrix} x_{FP,k,i}(t) - x(t) \\ y_{FP,k,i}(t) - y(t) \end{pmatrix} \tag{6.73}$$

where $x_{FP,k,i}$ and $y_{FP,k,i}$ are the northward and the eastward position of the respective point on the crowd line:

$$x_{FP,k,i}(t) = x_{1,k,i} + \lambda \cdot \left(x_{2,k,i} - x_{1,k,i}\right) \tag{6.74}$$

$$y_{FP,k,i}(t) = y_{1,k,i} + \lambda \cdot \left(y_{2,k,i} - y_{1,k,i}\right) \tag{6.75}$$

Accordingly, $d_{C,hor,k,i}(t)$ is the absolute value of the direction vector $\mathbf{d}_{C,hor,k,i}(t)$. Furthermore, $\kappa(t)$ in Eq. (6.72) represents the angle between the total velocity vector $\mathbf{V}_{total,hor}(t)$ and the direction vector $\mathbf{d}_{C,hor,k,i}(t)$ in the horizontal plane (see Fig. 57):

$$\kappa(t) = \arccos\left(\frac{\mathbf{d}_{C,hor,k,i}(t) \circ \mathbf{V}_{total,hor}(t)}{d_{C,hor,k,i}(t) \cdot V_{total,hor}(t)}\right) \tag{6.76}$$

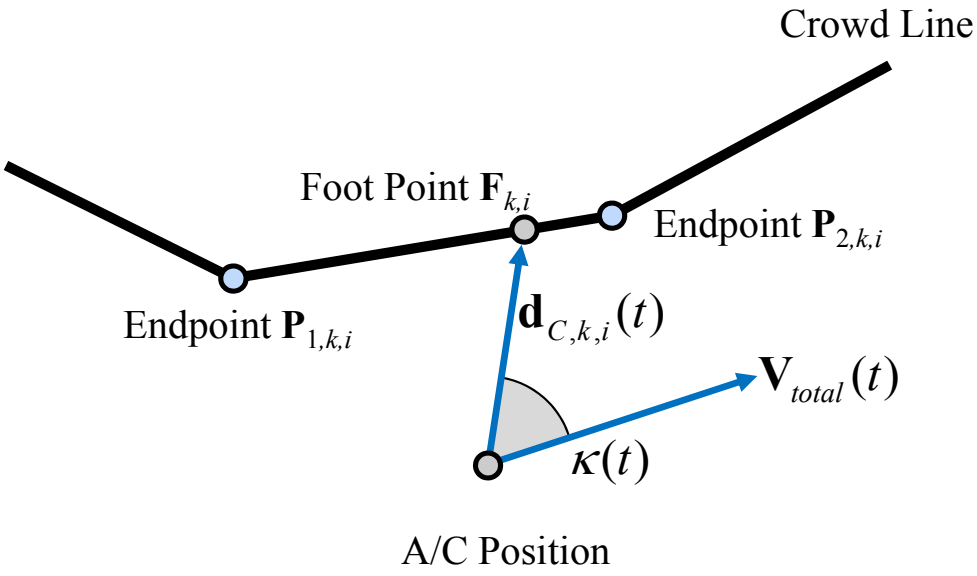

Figure 57. Angle $\kappa(t)$

Here, if $\kappa(t)$ evaluates to 90°, the velocity vector is perpendicular on the current direction vector. Thus, the energy that is directed towards the specific point on the crowd line is zero.

Then, the absolute value of the velocity vector $\mathbf{V}_{C,k,i}(t)$ (Eq. (6.72)) pointing towards a specific point on the crowd lines evaluates to:

$$\begin{aligned} V_{C,k,i}(t) &= \frac{\mathbf{d}_{C,hor,k,i}(t) \circ \mathbf{V}_{total,hor}(t)}{d_{C,hor,k,i}(t)} \\ &= \frac{1}{d_{C,hor,k,i}(t)} \begin{pmatrix} x_{FP,k,i}(t) - x(t) \\ y_{FP,k,i}(t) - y(t) \end{pmatrix} \circ \begin{pmatrix} \dot{x}(t) \\ \dot{y}(t) \end{pmatrix} \sqrt{1 - 2g \frac{z(t)}{V^2(t)}} \end{aligned} \tag{6.77}$$

Theoretically, the safety criterion with respect to the maximum direct energy has to be computed for every point on the crowd lines before the largest value is selected. Since this would imply an infinite number of points on the crowd line, a specific step size $\Delta\lambda$ is chosen and the value of λ is increased by this step size in the interval [0,1] for the computation of the safety criterion on all segments i of all polygons k. The velocity $V_C(t)$ is a measure for the energy that is directed towards a specific point on the crowd lines. In order to obtain a safety criterion with a reasonable meaning, the absolute distance to the crowd $d_{C,hor}(t)$ has also to be involved since a high directed energy is worse if the aircraft is close to the crowd line. Thus, the distance to the crowd $d_{C,hor}(t)$ is divided by the velocity $V_C(t)$ to give an equivalent time $t_C(t)$ that measures the energy directed towards the crowd line:

$$\begin{aligned} t_{C,k,i}(t) &= \frac{d_{C,hor,k,i}(t)}{V_{C,k,i}(t)} \\ &= \frac{\left(x_{FP,k,i}(t) - x(t)\right)^2 + \left(y_{FP,k,i}(t) - y(t)\right)^2}{\left[\dot{x}(t) \cdot \left(x_{FP,k,i}(t) - x(t)\right) + \dot{y}(t) \cdot \left(y_{FP,k,i}(t) - y(t)\right)\right] \cdot \sqrt{1 - 2g \frac{z(t)}{V^2(t)}}} \end{aligned} \tag{6.78}$$

The time $t_C(t)$ is called the *maximum directed energy*-equivalent (*MDE*-equivalent) time $t_C(t)$. If $t_C(t)$ is negative, the directed energy does not point towards the specific point on the crowd line and thus $t_C(t)$ is set to infinity. Then, the overall minimum *MDE*-equivalent time $t_C(t)$ with respect to all segments results from:

$$t_C(t) = \min t_{C,k,i}(t) \qquad k = 1, \ldots, p \qquad i = 1, \ldots, s_k \tag{6.79}$$

where p is the number of polygons and s_k the number of segments of the k-th polygon. In Table 14, the algorithm for the computation of the minimum *MDE*-equivalent time for the actual aircraft position is given.

The evaluation of the gradient of the safety criterion given by Eq. (6.79) with respect to the parameter vector $\mathbf{p}$ reads:

$$\begin{aligned} \frac{\partial t_C(t)}{\partial \mathbf{p}^T} = &\frac{\partial t_C(t)}{\partial x(t)} \cdot \frac{\partial x(t)}{\partial \mathbf{p}^T} + \frac{\partial t_C(t)}{\partial y(t)} \cdot \frac{\partial y(t)}{\partial \mathbf{p}^T} + \frac{\partial t_C(t)}{\partial z(t)} \cdot \frac{\partial z(t)}{\partial \mathbf{p}^T} + \\ &+ \frac{\partial t_C(t)}{\partial \dot{x}(t)} \cdot \frac{\partial \dot{x}(t)}{\partial \mathbf{p}^T} + \frac{\partial t_C(t)}{\partial \dot{y}(t)} \cdot \frac{\partial \dot{y}(t)}{\partial \mathbf{p}^T} + \frac{\partial t_C(t)}{\partial V(t)} \cdot \frac{\partial V(t)}{\partial \mathbf{p}^T} \end{aligned} \tag{6.80}$$

where the gradients $\partial x(t)/\partial \mathbf{p}$, $\partial y(t)/\partial \mathbf{p}$, $\partial z(t)/\partial \mathbf{p}$, $\partial \dot{x}(t)/\partial \mathbf{p}$, $\partial \dot{y}(t)/\partial \mathbf{p}$ and $\partial V(t)/\partial \mathbf{p}$ are obtained by applying the sensitivity analysis of chapter 5.2. The derivatives $\partial t_C(t)/\partial x(t)$, $\partial t_C(t)/\partial y(t)$, $\partial t_C(t)/\partial z(t)$, $\partial t_C(t)/\partial \dot{x}(t)$, $\partial t_C(t)/\partial \dot{y}(t)$ and $\partial t_C(t)/\partial V(t)$ result from differentiation of Eq. (6.78).

ALGORITHM 6.IV

1 For $k = 1, \ldots, p$
  For $i = 1, \ldots, s_k$
    Set $t_{C,k,i}(t)$ to infinity
    Increase λ by $\Delta\lambda$ from $\lambda = 0.0$ to $\lambda = 1.0$
      Compute $\kappa(t)$ by Eq. (6.76)
      If $\kappa(t)$ equals 90°
        Set $t_{C,k,i,\lambda}(t)$ to infinity
      Else
        Compute $t_{C,k,i,\lambda}(t)$ by Eqs. (6.69) to (6.78)
        If $t_{C,k,i,\lambda}(t) < 0$
          Set $t_{C,k,i,\lambda}(t)$ to infinity
      If $t_{C,k,i,\lambda}(t) < t_{C,k,i}(t)$
        Set $t_{C,k,i}(t)$ to $t_{C,k,i,\lambda}(t)$
2 Select minimum *maximum directed energy*-equivalent time $t_C(t)$ from all $t_{C,k,i}(t)$

Table 14. Computation of Minimum *Maximum Directed Energy*-equivalent Time

6.3.2.5. Ballistic Extrapolation

The safety criterion *ballistic extrapolation* shall provide a virtual measure for the assessment of the current value of the directed energy of the aircraft. Therefore, the aircraft is regarded as a mass part and its ballistic trajectory in a vacuum is computed. For the computation of the flight path of such a mass part, the initial conditions of the mass part are set equal to the current direction of motion and the current kinematic velocity of the aircraft. It is assumed that the only force acting is the gravitational force. Then, the impact zone, i.e. the zone where the mass parts will probably reach the ground level, is regarded as an indicator for the relative comparison of various race track layouts with respect to safety: a race track layout is regarded the more safe the larger the distance between the spectator areas and the impact zone is.

For the computation of the ballistic trajectory in a vacuum, the equations of motion of the mass part are:

$$x(\tau) = x(t) + \dot{x}(t) \cdot \tau \tag{6.81}$$

$$y(\tau) = y(t) + \dot{y}(t) \cdot \tau \tag{6.82}$$

$$\begin{aligned} z(\tau) &= z(t) + \dot{z}(t) \cdot \tau + \frac{1}{2} \cdot g \cdot \tau^2 \\ &= z(t) - V_K(t) \cdot \sin\gamma_K(t) \cdot \tau + \frac{1}{2} \cdot g \cdot \tau^2 \end{aligned} \tag{6.83}$$

where t is the time point when the extrapolation starts. Setting Eq. (6.83) to zero, the time point $t_B(t)$ when the mass part will approximately hit the ground is:

$$t_B(t) = \frac{1}{g}\left(V_K(t)\sin\gamma_K(t) + \sqrt{V_K^2(t)\sin^2\gamma_K(t) - 2\cdot g\cdot z(t)}\right) \tag{6.84}$$

Inserting $t_B(t)$ into Eqs. (6.81) and (6.82), one obtains the northward and the eastward position where the mass part falls onto the ground.

Evaluating the gradient of the ballistic extrapolation time $t_B(t)$ of Eq. (6.84) with respect to the parameter vector **p** results in:

$$\frac{\partial t_B(t)}{\partial \mathbf{p}^T} = \frac{\partial t_B(t)}{\partial z(t)}\cdot\frac{\partial z(t)}{\partial \mathbf{p}^T} + \frac{\partial t_B(t)}{\partial V_K(t)}\cdot\frac{\partial V_K(t)}{\partial \mathbf{p}^T} + \frac{\partial t_B(t)}{\partial \gamma_K(t)}\cdot\frac{\partial \gamma_K(t)}{\partial \mathbf{p}^T} \tag{6.85}$$

where the gradients $\partial z(t)/\partial\mathbf{p}$, $\partial V_K(t)/\partial\mathbf{p}$ and $\partial\gamma_K(t)/\partial\mathbf{p}$ are obtained by the sensitivity analysis given in chapter 5.2. For the computation of the derivatives $\partial t_B(t)/\partial z(t)$, $\partial t_B(t)/\partial V_K(t)$ and $\partial t_C(t)/\partial\gamma_K(t)$, Eq. (6.84) has to be differentiated accordingly.

6.3.2.6. Pilot Blinding

Another relevant situation arises if the pilot gets blinded by the Sun and thus cannot recognize possible obstacles directly in front of the aircraft. In order to avoid such situations, a safety criterion is established that measures the current angle between the direction towards the Sun and the kinematic velocity vector of the actual flight state. In doing so it is assumed that the pilot approximately peers into the direction the aircraft is currently flying. The smaller the angle towards the Sun is, the higher is the risk for the pilot to be blinded by the Sun. Thus, the angle between the Sun direction and the kinematic velocity vector has to be as large as possible.

First, the current position of the Sun respectively the direction towards the Sun in the local Navigation Frame *N* has to be established. Given the actual day of the year with January 1st of a leap year featuring the day number $N = 1$, the solar declination angle δ can be approximated by (Ref. [Winter, 1991]):

$$\delta = 0.409105\cdot\sin\left(p\cdot\left[N - 82.3 + 1.93\cdot\sin\left(p\cdot(N-2.4)\right)\right]\right) \tag{6.86}$$

with the period p being:

$$p = \frac{2\pi}{365.25} \tag{6.87}$$

Here a simple leap-year cycle is assumed, this means that the Earth cycles around the Sun once in 365.25 days. Furthermore, the Equation of Time *EOT* has to be computed that corrects the assumption of a constant circular motion of the Earth around the Sun (Ref. [Duffie, 1980]):

$$\begin{aligned} EOT = \frac{229.2}{60}\cdot\ & \left(0.000075 + \right. \\ & + 0.001868\cdot\cos\left[(N-1)\cdot p\right] - 0.014615\cdot\cos\left[2\cdot(N-1)\cdot p\right] \\ & \left. - 0.032077\cdot\sin\left[(N-1)\cdot p\right] - 0.040890\cdot\sin\left[2\cdot(N-1)\cdot p\right]\right) \end{aligned} \tag{6.88}$$

Then, the local solar time $t_{sol,loc}$ is (Ref. [Winter 1991])

$$t_{sol,loc}(t) = t_{loc} - 12.0 + EOT + \frac{24.0}{360.0} \cdot (\lambda_0 - \lambda(t)) \tag{6.89}$$

where $\lambda(t)$ represents the aircraft's current longitude and λ_0 the geographic longitude referring to the standard local time zone t_{loc}. Next, the hour angle ω evaluates to:

$$\omega(t) = t_{sol,loc}(t) \cdot \frac{2\pi}{24.0} \tag{6.90}$$

finally resulting in the current direction vector $\vec{\mathbf{n}}_{sol}$ towards the Sun given in the local Navigation Frame N:

$$(\vec{\mathbf{n}}_{sol})_N = \mathbf{M}_{NO} \cdot \begin{pmatrix} -\sin\mu(t) \cdot \cos\delta \cdot \cos\omega(t) + \cos\mu(t) \cdot \sin\delta \\ -\cos\delta \cdot \sin\omega(t) \\ -\cos\mu(t) \cdot \cos\delta \cdot \cos\omega(t) - \sin\mu(t) \cdot \sin\delta \end{pmatrix}_O \tag{6.91}$$

The Sun angle $\sigma(t)$ between the aircraft's current velocity vector and the direction vector towards the Sun is obtained by:

$$\sigma(t) = \arccos\left[(\vec{\mathbf{n}}_{sol})_N \circ \begin{pmatrix} \cos\chi(t)\cos\gamma(t) \\ \sin\chi(t)\cos\gamma(t) \\ -\sin\gamma(t) \end{pmatrix}_N \right] \tag{6.92}$$

Evaluating the gradient of the Sun angle $\sigma(t)$ given by Eq. (6.92) with respect to the parameter vector $\mathbf{p}$ reads:

$$\begin{aligned} \frac{\partial\sigma(t)}{\partial\mathbf{p}^T} = \frac{\partial\sigma(t)}{\partial\lambda(t)} \cdot \frac{\partial\lambda(t)}{\partial\mathbf{p}^T} + \frac{\partial\sigma(t)}{\partial\mu(t)} \cdot \frac{\partial\mu(t)}{\partial\mathbf{p}^T} + \\ + \frac{\partial\sigma(t)}{\partial\chi(t)} \cdot \frac{\partial\chi(t)}{\partial\mathbf{p}^T} + \frac{\partial\sigma(t)}{\partial\gamma(t)} \cdot \frac{\partial\gamma(t)}{\partial\mathbf{p}^T} \end{aligned} \tag{6.93}$$

As before, the gradients $\partial\lambda(t)/\partial\mathbf{p}$, $\partial\mu(t)/\partial\mathbf{p}$, $\partial\chi(t)/\partial\mathbf{p}$ and $\partial\gamma(t)/\partial\mathbf{p}$ result by applying the sensitivity analysis of chapter 5.2. The derivatives $\partial\sigma(t)/\partial\lambda(t)$, $\partial\sigma(t)/\partial\mu(t)$, $\partial\sigma(t)/\partial\chi(t)$ and $\partial\sigma(t)/\partial\gamma(t)$ are obtained by differentiation of Eq. (6.92).

6.3.2.7. Load Factor Fatigue Index

The actual load factor value as well as the preceding load factor time history greatly influence the current capabilities of the pilot. In the worst case, the pilot could lose consciousness due to too high load factors what is called *G-LOC* which stands for G-induced loss of consciousness. Thus, the load factor time history of the optimal race trajectory plays an important role with respect to the safety of the respective race track. A safety index with respect to the load factor has to take into account

- the actual load factor value,
- the load factor onset rate
- and an integral term over the preceding load factor time history.

While too high load factors as well as too high load factor onset rates can directly cause unconsciousness of the pilot, the integral term takes into account the stress of the pilot due to

long-duration high load factor levels that can also lead to *G-LOC*. Therefore, for the load factor fatigue index $E(t)$ the following transfer function is defined:

$$E(t) = \left(k_1 \frac{T_1 s}{T_1 s + 1} + k_2 \frac{1}{T_2 s + 1} + k_3 \right) \cdot \Delta n_z(t) \tag{6.94}$$

where $\Delta n_z(t)$ is the deviation from the load factor $n_z = 1.0$ required for horizontal flight:

$$\Delta n_z(t) = n_z(t) - 1.0 \tag{6.95}$$

The P-element involves the actual load factor deviation into the load factor fatigue index. The DT_1-element measures the load factor onset rate and the PT_1-element represents the integral term where the time constant T_2 associated with the PT_1-element is set to a much higher value than the time constant T_1 of the DT_1-element. The factors k_1 to k_3 allow for a weighting of the influence of the various elements onto the load factor fatigue index. Written in second-order state-space form, the following differential equations for the load factor fatigue index $E(t)$ result:

$$\begin{aligned} \frac{d}{dt}\begin{pmatrix} E(t) \\ \dot{E}(t) \end{pmatrix} &= \begin{bmatrix} 0 & 1 \\ -\frac{1}{T_1 T_2} & -\frac{T_1 + T_2}{T_1 T_2} \end{bmatrix} \begin{pmatrix} E(t) \\ \dot{E}(t) \end{pmatrix} + \\ &+ \begin{bmatrix} 0 & 0 & 0 \\ \frac{(k_2 + k_3)}{T_1 T_2} & \frac{(k_1 + k_2 + k_3) T_1 + k_3 T_2}{T_1 T_2} & (k_1 + k_3) \end{bmatrix} \begin{pmatrix} \Delta n_z(t) \\ \Delta \dot{n}_z(t) \\ \Delta \ddot{n}_z(t) \end{pmatrix} \end{aligned} \tag{6.96}$$

Fig. 58 depicts the load factor fatigue index $E(t)$ as well as the contributions of the P-element, the PT_1-element and DT_1-element to the load factor fatigue index for a generic load factor time history $\Delta n_z(t)$.

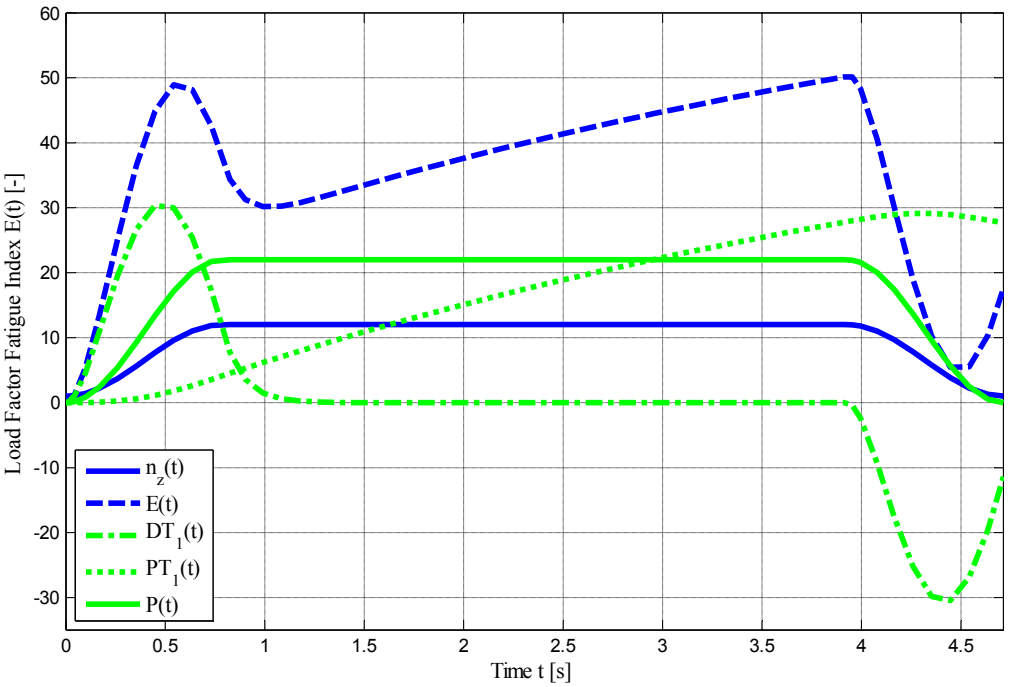

Figure 58. Load Factor Fatigue Index

The gradient of the 2nd order time derivative of the load factor fatigue index $E(t)$ with respect to the parameter vector $\mathbf{p}$ of the upper level parameter optimization problem is:

$$\frac{\partial \ddot{E}(t)}{\partial \mathbf{p}^T} = \left(\frac{1}{T_1 T_2} \left[(k_1 + k_3) T_1 T_2 \cdot \frac{\partial \Delta \ddot{n}_z(t)}{\partial \mathbf{p}^T} + \left((k_1 + k_2 + k_3) T_1 + k_3 T_2 \right) \cdot \frac{\partial \Delta \dot{n}_z(t)}{\partial \mathbf{p}^T} + \right. \right. \\ \left. \left. + (k_2 + k_3) \cdot \frac{\partial \Delta n_z(t)}{\partial \mathbf{p}^T} - (T_1 + T_2) \cdot \frac{\partial \dot{E}(t)}{\partial \mathbf{p}^T} - \frac{\partial E(t)}{\partial \mathbf{p}^T} \right] \right) \tag{6.97}$$

where the gradients $\partial \Delta \ddot{n}_z(t)/\partial \mathbf{p}$, $\partial \Delta \dot{n}_z(t)/\partial \mathbf{p}$ and $\partial \Delta n_z(t)/\partial \mathbf{p}$ are obtained by the sensitivity analysis explained in chapter 5.2. Eq. (6.97) then has to be integrated twice to give the required gradient of the load factor fatigue index $E(t)$ w.r.t. the parameter vector $\mathbf{p}$, $\partial E(t)/\partial \mathbf{p}$.

6.3.3 Optimized Air Race Tracks for Selected Safety Criteria

First, in this chapter the initial layout of the race track that is to be optimized is described. Then, race tracks are depicted that are optimized with respect to the safety criteria that have been defined in the preceding chapter. For the solution of the lower level optimal control problems, a slightly modified point-mass simulation model has been utilized in order to reduce the dimensionality of the lower level optimal control problem and thus the computational time that is required for the solution of the entire bilevel optimal control problem. For the modified simulation model, the controls w.r.t. the aerodynamic sideslip angle β_A as well as the thrust lever position δ_T have been set to zero respectively one since it has been observed that all the time-optimal trajectories are mostly flown at full thrust with nearly no sideslip angle.

6.3.3.1. Initial Race Track Layout

At the outset of the solution of the bilevel programming problem, the basic layout of the air race track is given (Fig. 59). The initial setup is similar to the air race that took place in San Diego in 2009. Table 15 gives an overview of the approximate positions x and y of the air race gates in the local Navigation Frame N. The local Navigation Frame N is derived from a *NED*-Frame with its origin located at the position of the first gate.

Race Gate	Position x	Position y
1/7	0.00m	0.00m
2a	22.33m	429.75m
2b	-77.76m	531.33m
2c	-140.93m	635.51m
3	-523.67m	705.86m
4	-406.59m	940.27m
5/6	-240.28m	281.30m

Table 15. Approximate Positions of the Air Race Gates in the Local Navigation Frame *N*

The position of the first gate in geodetic coordinates with longitude λ and latitude μ is −117°10’29.9” respectively 32°42’18.5”. The “Chicane” is defined by three separate gates (Race Gates 2a-2b) and for each gate the position that has to be passed by the aircraft is given.

While the positions and the directions of the race gates are subject to optimization, the basic layout shall remain unchanged throughout the optimization procedure. Especially race track elements like the “Chicane” and the “Quadro” are to be preserved throughout the optimization. Furthermore, for the computation of the safety criteria two appropriate spectator areas that are wrapped by crowd lines are defined (Fig. 59). The crowd lines are given in form of piecewise linear polygons:

$$\mathbf{P}_1 = \begin{pmatrix} -100 & -150 & -280 & -720 & -1000 \\ -750 & -530 & -530 & 800 & 900 \end{pmatrix}_N \tag{6.98}$$

$$\mathbf{P}_2 = \begin{pmatrix} 680 & 400 & 440 & 360 & 350 & 220 & -250 & -280 & -490 & -720 \\ -100 & 210 & 250 & 290 & 420 & 420 & 1030 & 1280 & 1250 & 1540 \end{pmatrix}_N \tag{6.99}$$

Here again, N denotes a local Navigation Frame with its origin located at the position of the first gate and the x-, y- and z-axis of the Navigation Frame N pointing into northward, eastward and downward direction.

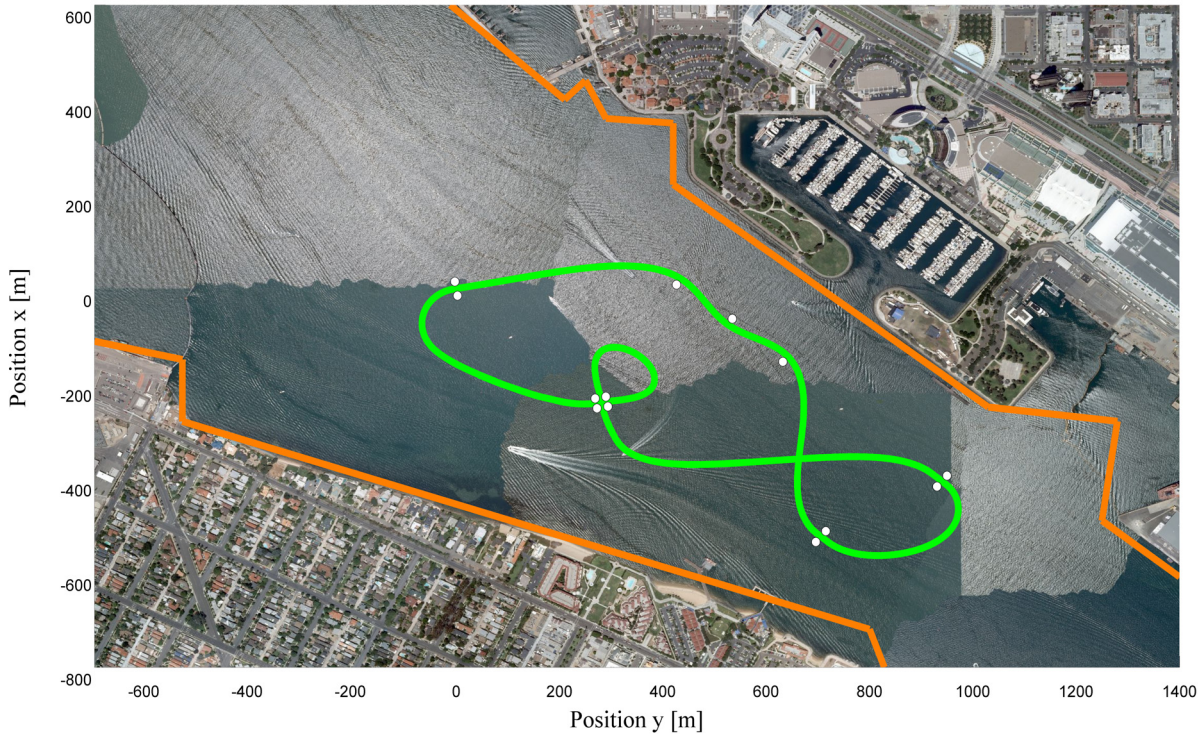

Figure 59. Initial Track Layout and Crowd Lines

Within the optimization problems, the upper and lower bounds with respect to the gate positions can be imposed either directly in the Navigation Frame N or alternatively in Gate Frames G. At this, the Gate Frame for the i-th gate is derived from the Navigation Frame by rotating the Navigation Frame around its z-axis by an arbitrary gate rotation angle $\alpha_{G,i}$. Then, the upper and lower bounds on the i-th gate’s forward position x and sideward position y can be imposed in the i-th Gate Frame:

$$(x)_{G,i,LB} \leq (x)_{G,i} \leq (x)_{G,i,UB} \tag{6.100}$$

$$(y)_{G,i,LB} \le (y)_{G,i} \le (y)_{G,i,UB} \tag{6.101}$$

Thus, individual gate boxes for each gate can be defined and the edges of the gate boxes do not have to be aligned with the *x*-axis respectively the *y*-axis of the Navigation Frame *N*. For the optimization task, the gate positions given in the Gate Frame have to be transformed into the Navigation Frame:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix}_{N,i} = \mathbf{M}_{NG,i} \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix}_{G,i} \tag{6.102}$$

where $\mathbf{M}_{NG,i}$ denotes the transformation matrix between the *i*-th Gate Frame and the Navigation Frame:

$$\mathbf{M}_{NG,i} = \begin{pmatrix} \cos\alpha_{G,i} & -\sin\alpha_{G,i} & 0 \\ \sin\alpha_{G,i} & \cos\alpha_{G,i} & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{6.103}$$

Furthermore, the parameter sensitivities with respect to the respective gate position parameters have to be transformed back into the *i*-th Gate Frame before the gradient for the upper level parameter optimization problem can be computed. For example, the following transformation has to be applied to the parameter sensitivities of the final time t_f with respect to the gate position parameters:

$$\begin{pmatrix} \frac{\partial t_f}{\partial (x)_{G,i}} \\ \frac{\partial t_f}{\partial (y)_{G,i}} \\ \frac{\partial t_f}{\partial (z)_{G,i}} \end{pmatrix} = \mathbf{M}_{NG,i}^T \cdot \begin{pmatrix} \frac{\partial t_f}{\partial (x)_{N,i}} \\ \frac{\partial t_f}{\partial (y)_{N,i}} \\ \frac{\partial t_f}{\partial (z)_{N,i}} \end{pmatrix} \tag{6.104}$$

Here it is mentioned that on all the following figures the filled dots mark the optimized positions of the race gates while the dots that are not filled indicate the initial race gate positions. The colorbars provide a visualization of the respective safety criterion for the optimized race track layouts.

6.3.3.2. Minimum Distance to Crowd

With regard to the minimum distance to crowd, the *maximin* upper level parameter optimization problem of the bilevel optimal control problem is:

$$\max_{\mathbf{p}} \left(\min d_C(t) \right) \tag{6.105}$$

where $d_C(t)$ is the distance to the crowd and $\mathbf{p}$ the parameter vector of the upper level parameter optimization problem. The *maximin* problem can be transformed into a standard parameter optimization problem:

$$\min_{\mathbf{p}} \left(-d_{C,\min} \right) \quad \text{s.t.} \quad d_C(t) - d_{C,\min} \ge 0 \tag{6.106}$$

Fig. 60 depicts the optimized race track layout and the corresponding flight trajectory together with the initial race track layout. While for the initial race track layout the minimum distance to crowd is $d_{C,min}$ = 142.36m, the minimum distance to crowd for the optimized race track evaluates to $d_{C,min}$ = 193.26m. Thus, the minimum distance to crowd $d_{C,min}$ is increased by 35.8%. The minimum distance to crowd is visualized by the dashed, red line in Fig. 60.

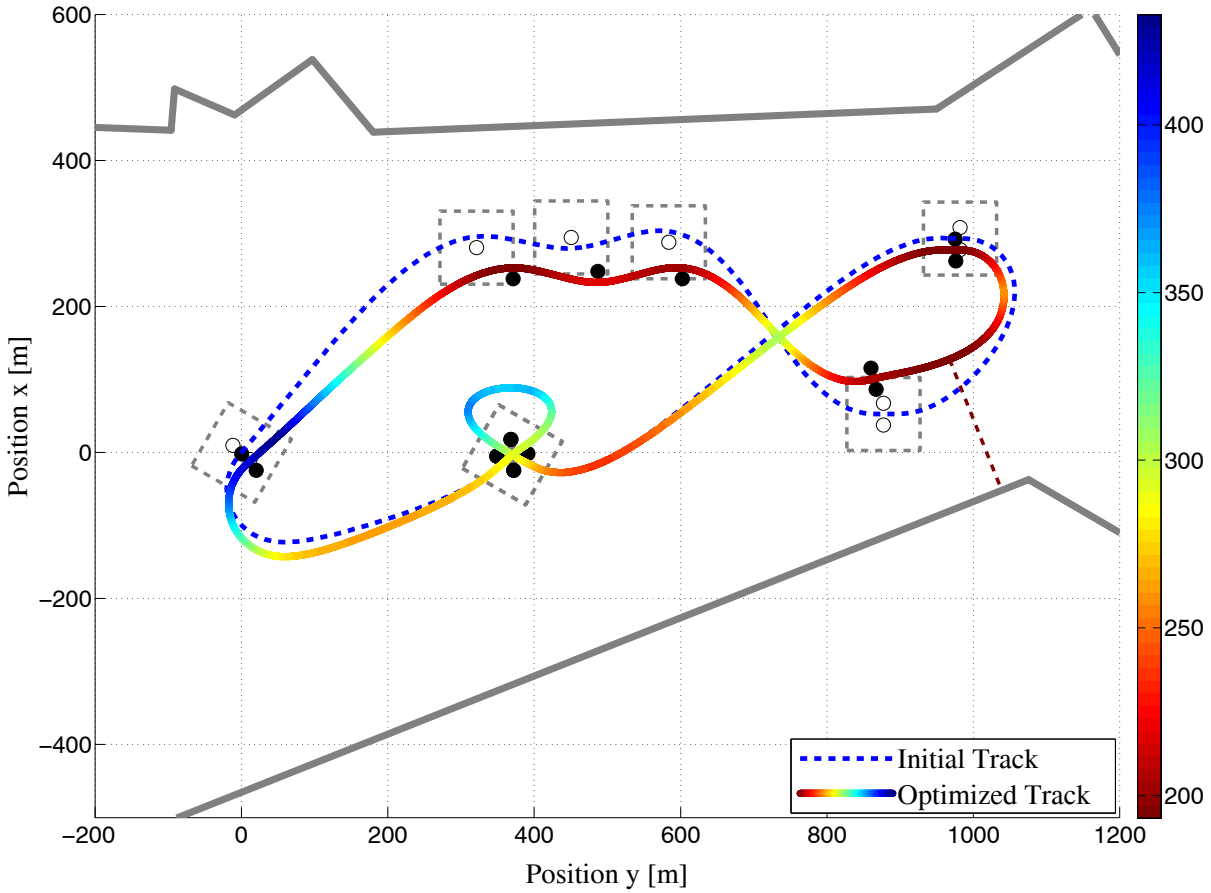

Figure 60. Race Track optimized w.r.t. Distance to Crowd [m]

6.3.3.3. Minimum Time to Crowd

For the minimum time to crowd, a *maximin* upper level parameter optimization problem can be stated as:

$$\max_{\mathbf{p}} \left(\min t_C(t) \right) \tag{6.107}$$

where $t_C(t)$ is the distance to the crowd and $\mathbf{p}$ the parameter vector of the upper level parameter optimization problem. As before, the *maximin* problem is transformed into the following minimization problem:

$$\min_{\mathbf{p}} \left(-t_{C,\min} \right) \quad \text{s.t.} \quad t_C(t) - t_{C,\min} \geq 0 \tag{6.108}$$

The race trajectory and the optimized race track layout can be seen in Fig. 61. Here, the race track layout is optimized such that finally the minimum time to crowd is $t_{C,min}$ = 4.57s at every point of the race trajectory. With the minimum time to crowd being $t_{C,min}$ = 3.24s for the initial race track layout, this corresponds to an increase by 41.0%. In Fig. 61, the point of the trajectory where the minimum time to crowd occurs is marked with a red circle.

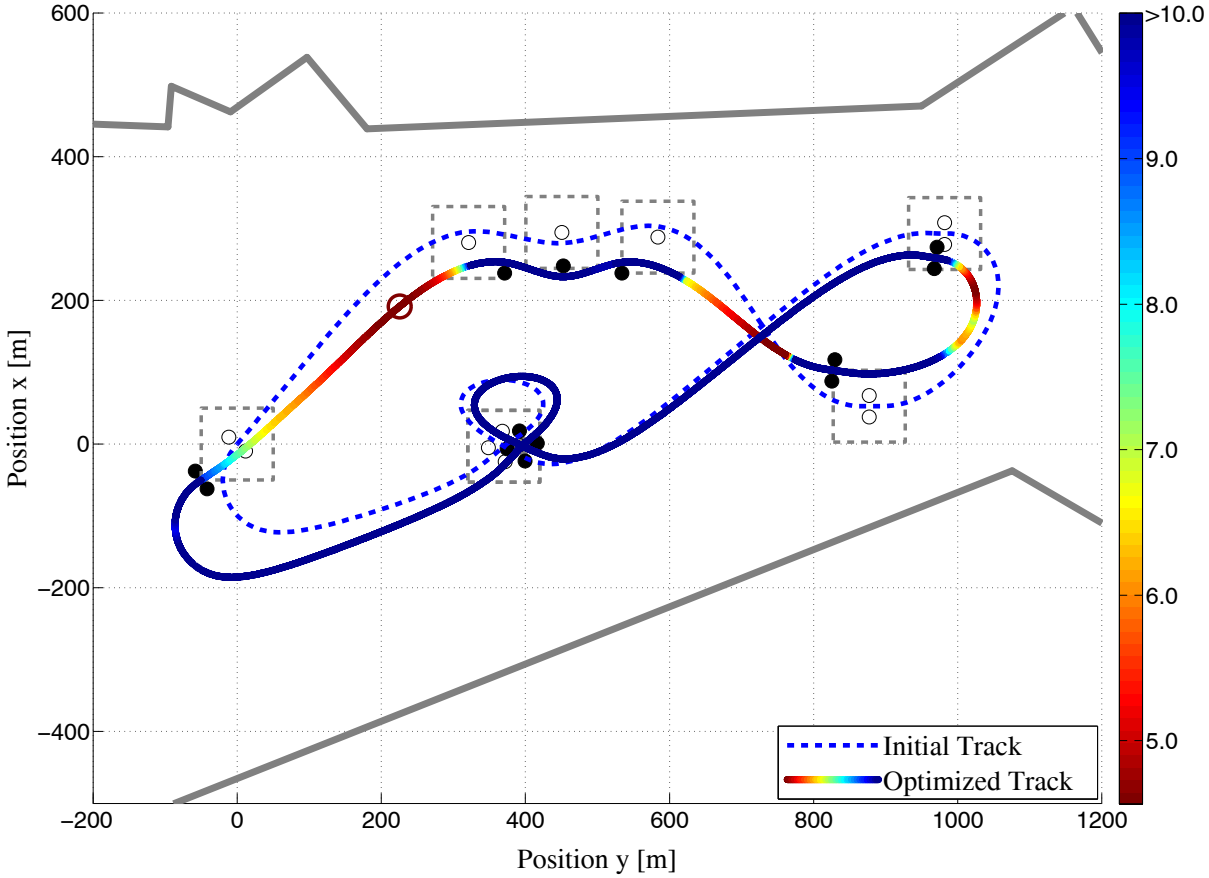

Figure 61. Race Track optimized w.r.t. Time to Crowd [s]

6.3.3.4. Minimum Time to Crowd based on the Normal Velocity Component

For the minimum time to crowd based on the normal velocity component, the same upper level parameter optimization problem as stated by Eqs. (6.107) and (6.108) in the preceding chapter results. The resulting race trajectory and the corresponding time history of the time to crowd $t_C(t)$ are depicted in Fig. 62. For the optimized race track, the minimum time to crowd amounts to $t_{C,min}$ = 4.63s at every time point t of the race trajectory. Hence, the initial value for the minimum time to crowd that is $t_{C,min}$ = 3.24s is increased by 42.8%. In Fig. 62, the point of the trajectory corresponding to the minimum time to crowd is highlighted by a red circle.

6.3.3.5. Minimum *Maximum Directed Energy*-equivalent Time

With respect to the *maximum directed energy*-equivalent time, the upper level parameter optimization problem features the following *maximin* problem:

$$\max_{\mathbf{p}}\left(\min t_C(t)\right) \tag{6.109}$$

Here, $t_C(t)$ is the *maximum directed energy*-equivalent time and **p** the parameter vector of the upper level parameter optimization problem. The transformation of the *maximin* problem gives:

$$\min_{\mathbf{p}}\left(-t_{C,\min}\right) \quad \text{s.t.} \quad t_C(t) - t_{C,\min} \geq 0 \tag{6.110}$$

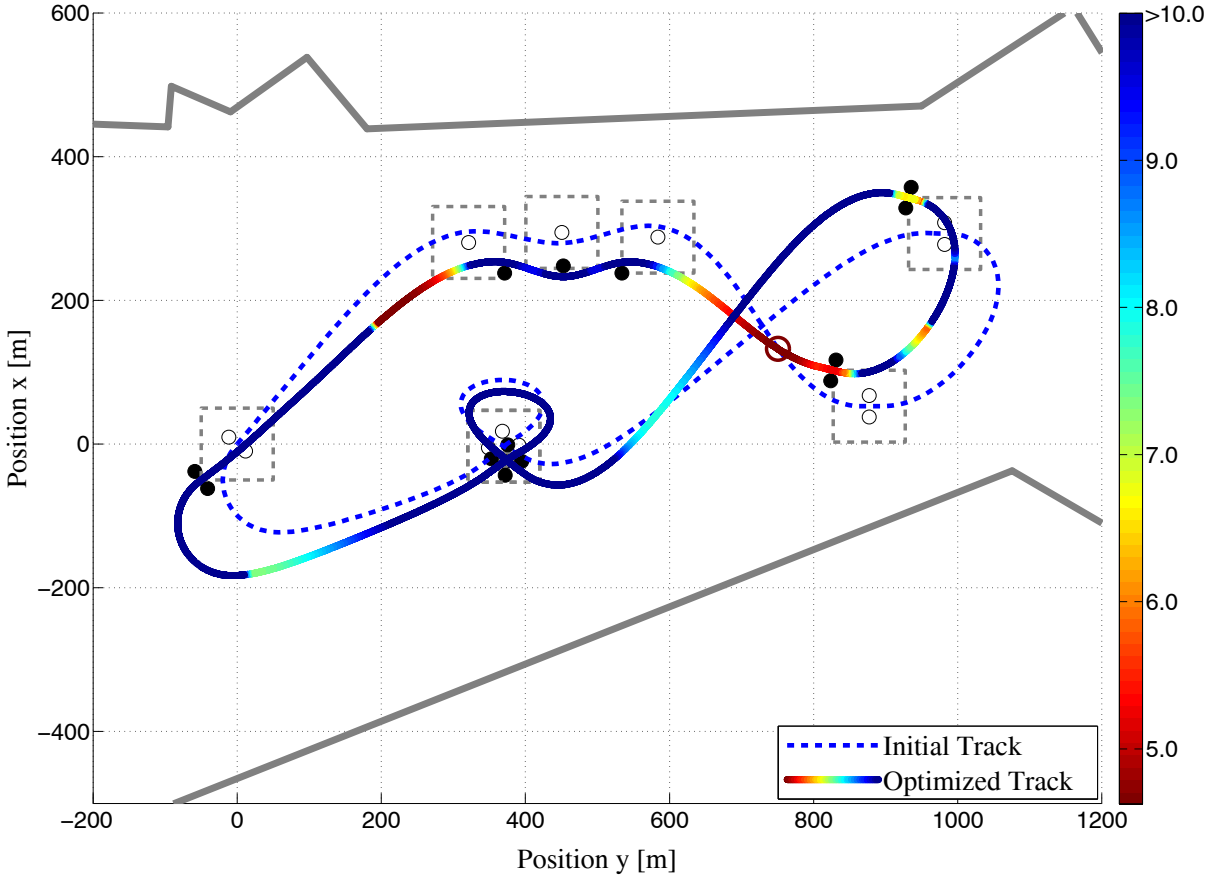

Figure 62. Race Track optimized w.r.t. Time to Crowd based on the Normal Velocity Component [s]

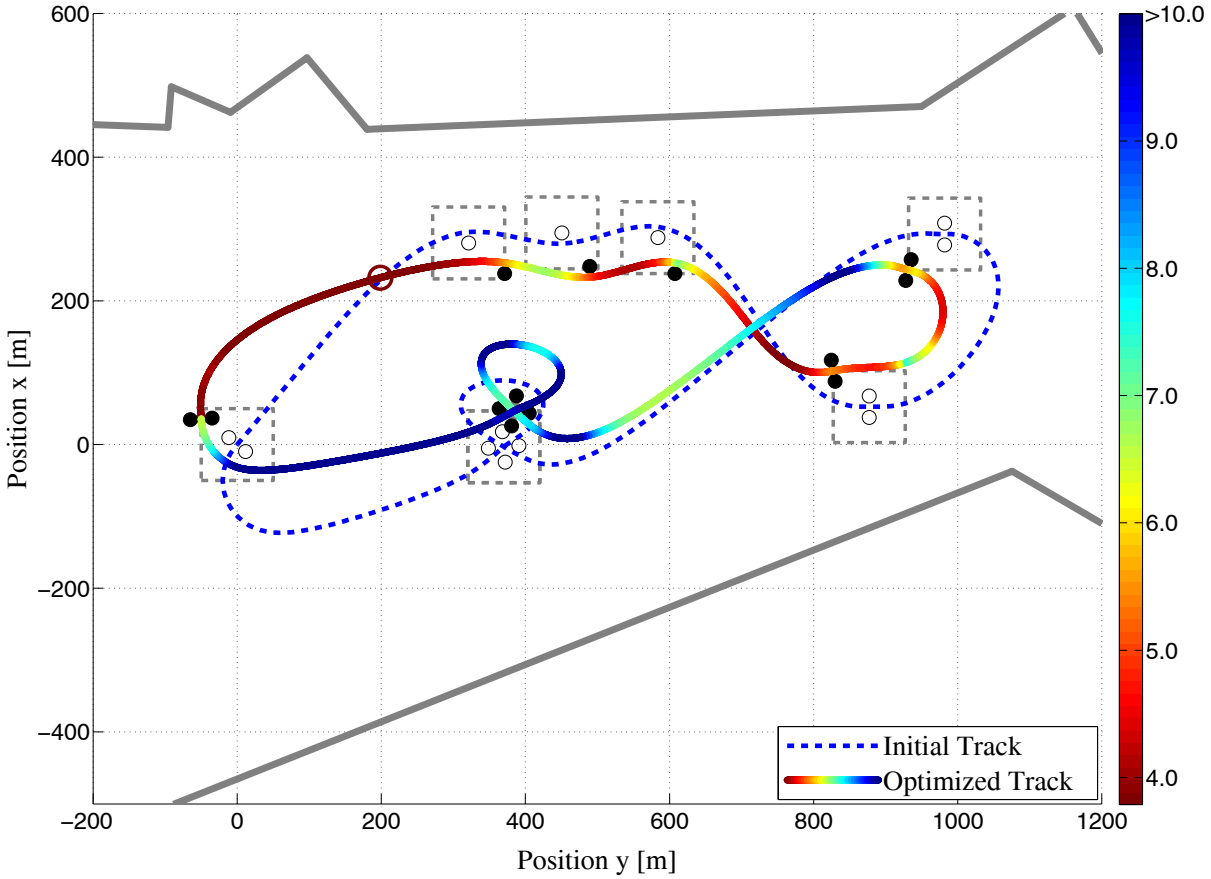

Figure 63. Race Track optimized w.r.t. Directed Energy [s]

In Fig. 63, an optimized layout of the race track is presented. For the optimized race track layout, the minimum *MDE*-equivalent time $t_{C,min}$ is $t_{C,min}$ = 3.79s. Compared to the minimum *MDE*-equivalent time for the initial race track layout that is $t_{C,min}$ = 2.48s, this constitutes an increase by 52.5%. The point of the trajectory that features the minimum *MDE*-equivalent time is labeled by a red circle.

6.3.3.6. Ballistic Extrapolation

Regarding the mass part flight time $t_B(t)$ calculated by the ballistic extrapolation in chapter 6.3.2.5, it holds that if at time t the flight time $t_B(t)$ until the ground level is reached is smaller than the time to crowd $t_C(t)$, the mass part will not reach any of the spectator areas if a ballistic flight path of the mass part in a vacuum is taken as basis. This is due to the fact that for the calculation of both time measures the same horizontal velocity vector is utilized. Thus, if the inequality

$$\Delta t(t) = t_C(t) - t_B(t) \geq 0 \tag{6.111}$$

holds for all times t of the race trajectory, none of the mass parts will reach the spectator areas if the flight path of the mass part is computed as a ballistic trajectory in a vacuum. As mentioned above, the ballistic extrapolation concept only provides a virtual measure for the qualitative comparison of various race track layouts with respect to safety. Furthermore, with respect to Eq. (6.111) an additional safety margin might be taken into account so that the time difference $\Delta t(t)$ is restricted to lie above a certain time margin $\Delta t_{\min}$:

$$\Delta t(t) = t_C(t) - t_B(t) \geq \Delta t_{\min} \tag{6.112}$$

Then, the following *maximum* upper level parameter optimization problem can be stated to layout the race track more safe with respect to the ballistic extrapolation safety criterion:

$$\max_{\mathbf{p}} \left(\Delta t_{\min} \right) \tag{6.113}$$

i.e. the larger the final time margin $\Delta t_{\min}$ is, the more safe is the race track. Rewriting the maximization problem into a minimization problem gives:

$$\min_{\mathbf{p}} \left(-\Delta t_{\min} \right) \quad \text{s.t.} \quad \Delta t(t) - \Delta t_{\min} \geq 0 \tag{6.114}$$

In Fig. 64, the race track that has been optimized with respect to the ballistic extrapolation safety criterion is depicted. The final race track layout has been improved such that the time margin $\Delta t_{\min}$ for the time-optimal race trajectory equals $\Delta t_{\min}$ = 2.85s. Thus, the initial time margin $\Delta t_{\min}$ = 1.31s is increased by 117.2%. The point corresponding to the minimum time margin $\Delta t_{\min}$ is shown by a red circle in Fig. 64.

Fig. 65 depicts the ballistic extrapolation zone, i.e. the zone where the mass parts would hit the ground if a ballistic flight path in a vacuum was assumed.

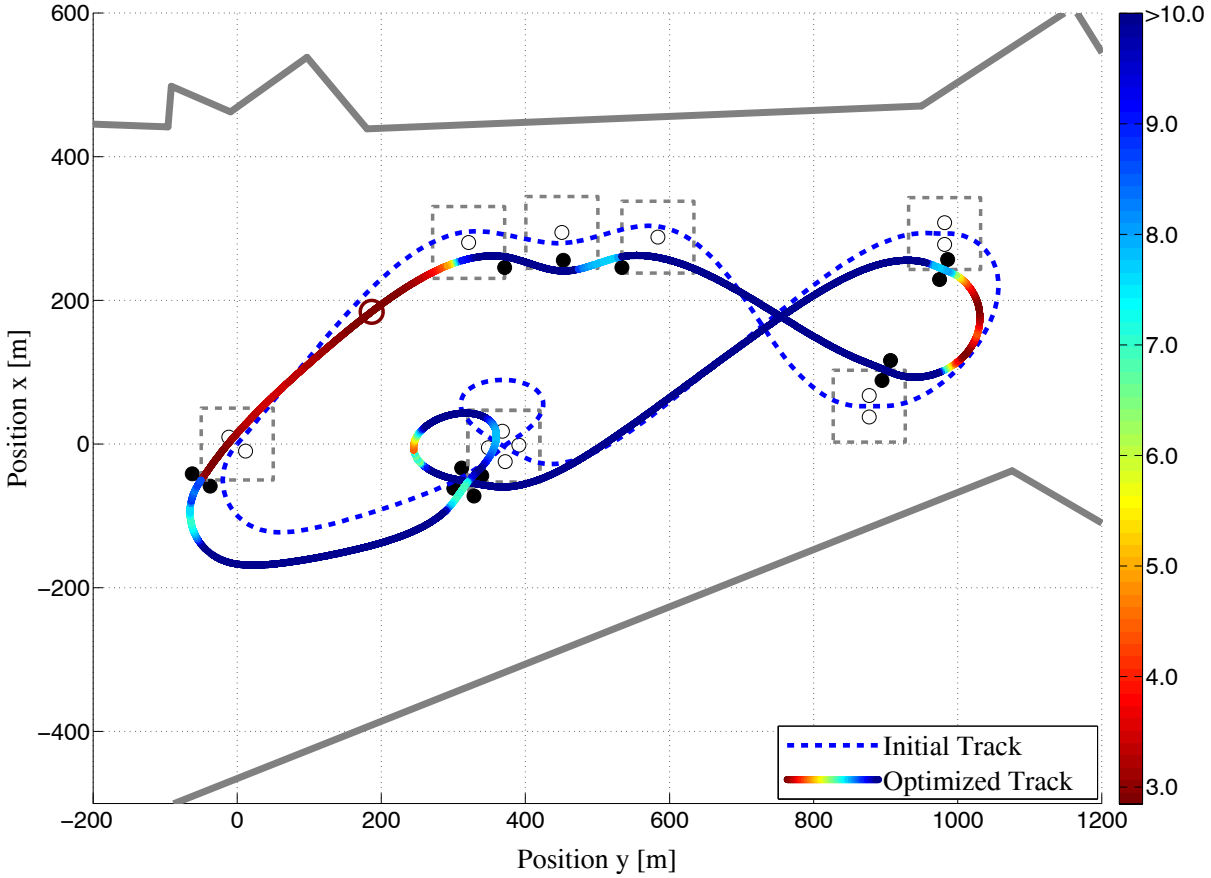

Figure 64. Race Track optimized w.r.t. Ballistic Extrapolation [s]

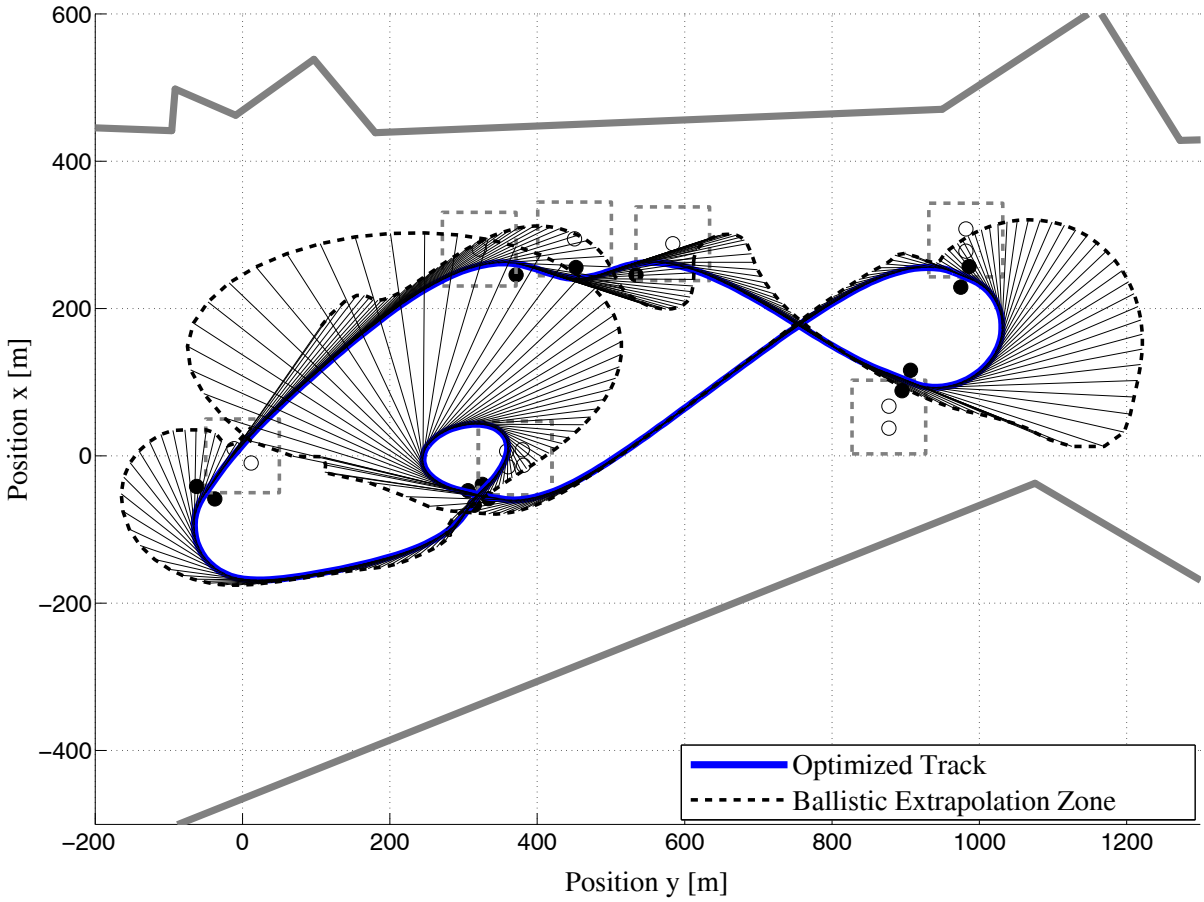

Figure 65. Ballistic Extrapolation Zone

6.3.3.7. Pilot Blinding

For the safety criterion that has been set up to avoid any blinding of the pilot by the Sun, the *maximin* upper level parameter optimization problem of the bilevel optimal control problem reads:

$$\max_{\mathbf{p}}\left(\min \sigma(t)\right) \tag{6.115}$$

where $\sigma(t)$ is the angle between the aircraft's current velocity vector and the direction vector towards the Sun and $\mathbf{p}$ the parameter vector of the upper level parameter optimization problem. Then, the *maximin* problem has to be transformed into a standard minimization problem:

$$\min_{\mathbf{p}}\left(-\sigma_{\min}\right) \quad \text{s.t.} \quad \sigma(t)-\sigma_{\min} \geq 0 \tag{6.116}$$

where $\sigma_{\min}$ represents a lower threshold for the Sun angle. Fig. 66 depicts the race trajectories for the initial race track layout as well as for the race track layout optimized with respect to the Sun angle $\sigma(t)$. For the optimized race track layout, the minimum Sun angle is 0.930*rad* (=53.24°) while the minimum Sun angle for the initial race track layout is 0.594*rad* (=34.04°). Thus, the minimum Sun angle is increased by 56.4%. In Fig. 66, the point where the minimum Sun angle occurs is pointed out with a red circle.

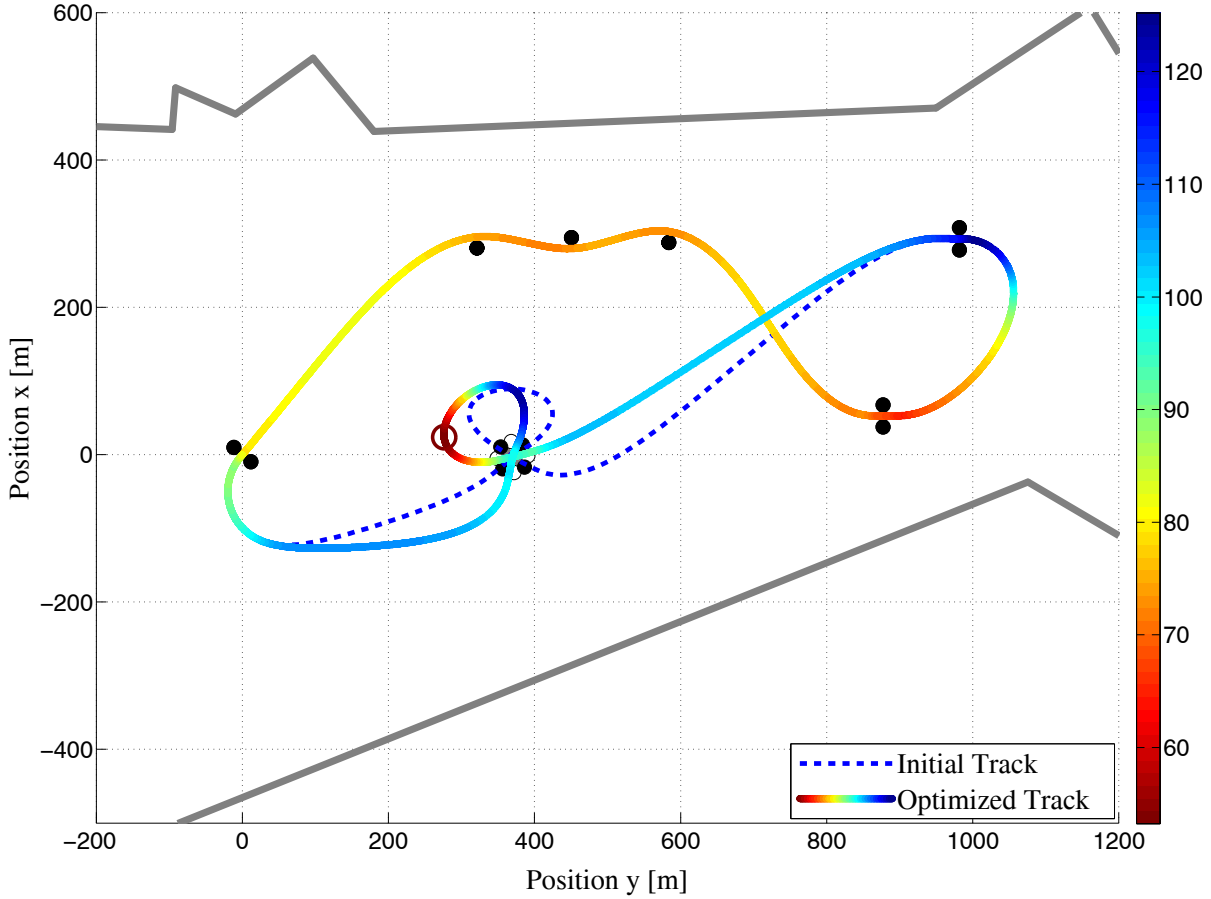

Figure 66. Race Track optimized w.r.t. Pilot Blinding [°]

6.3.3.8. Load Factor Fatigue Index

With respect to the load factor fatigue index $E(t)$, the upper level parameter optimization problem is represented by the following *minimax* problem:

$$\min_{\mathbf{p}}\left(\max E(t)\right) \tag{6.117}$$

Here again, **p** is the parameter vector of the upper level parameter optimization problem. By the transformation of the *minimax* problem, one obtains:

$$\min_{\mathbf{p}}\left(E_{\max}\right) \quad \text{s.t.} \quad E_{\max} - E(t) \geq 0 \tag{6.118}$$

In Fig. 67, the layout of the race track optimized with respect to the load factor fatigue index $E(t)$ is presented. For the optimized race track layout, the peak of the load factor fatigue index $E(t)$ is reduced to $E_{\max} = 17.24$, a reduction of 13.4% in comparison to the maximum initial load factor fatigue index value $E_{\max} = 19.90$. The point of the trajectory with the maximum value of the load factor fatigue index is marked by a red circle.

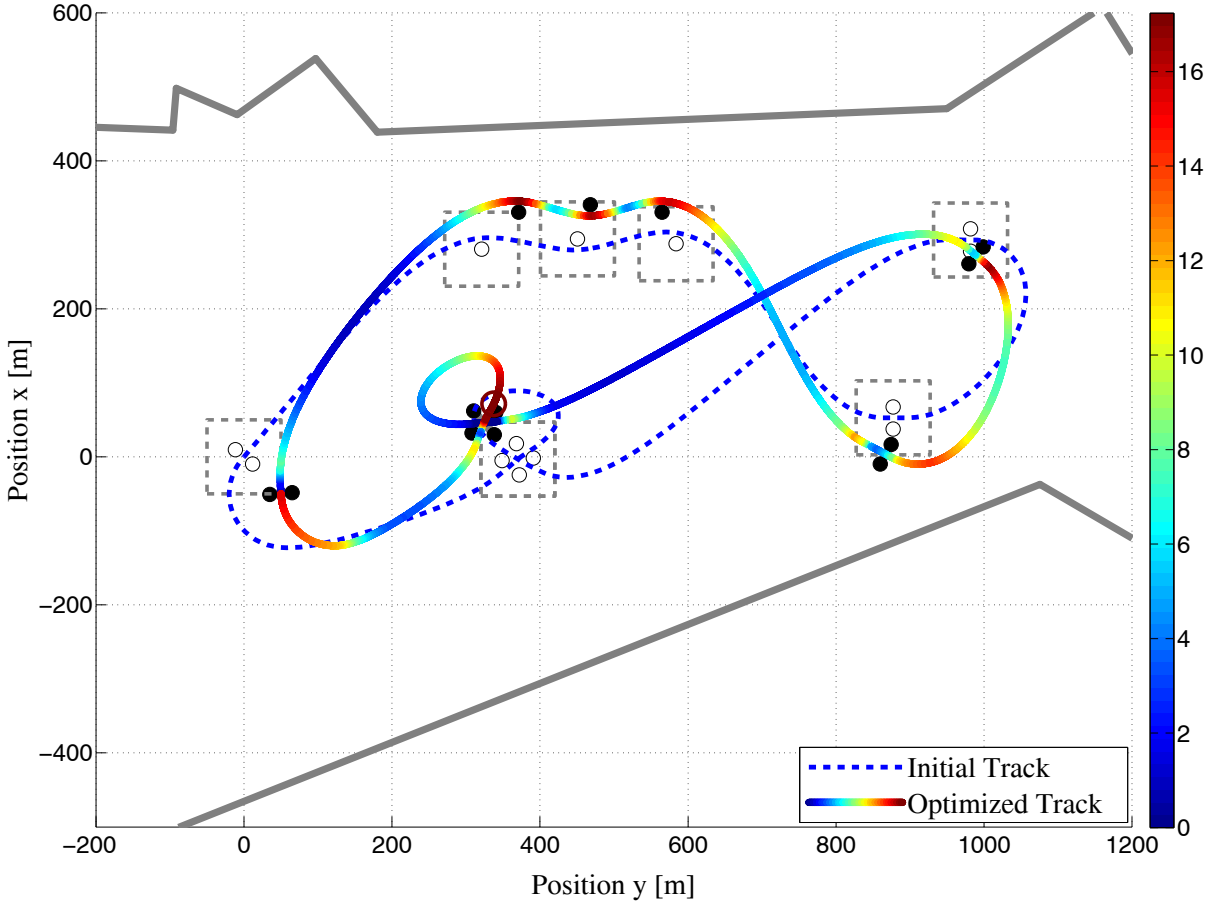

Figure 67. Race Track optimized w.r.t. Load Factor Fatigue Index [-]

6.4 Increasing the Fairness in Air Races

6.4.1 Increasing the Fairness for Two Competing Aircraft

For the air races described in chapter 6.1, the pilots are flying different types of aerobatic aircraft like e.g. the Zivko Edge 540 or the MXS-R. While one type might be able to reach higher velocities, the other type might be more agile, having an advantage over the faster type

if the race track layout is more winding. In order to reach a high level of fairness for the pilots participating, the track planners have to position the gates such that different aircraft have the same chance of winning, which means that for the designed race track the minimum possible flight times have to be identical regardless of the respective aircraft in use.

If one wants to determine such a race track layout mathematically, this will result in a bilevel optimal control problem, because different optimization tasks with distinct optimization goals are to be solved. The upper level problem is a nonlinear parameter computation problem and ought to place the race gates such that the difference in the minimum race times of different aircraft becomes zero. At each iteration of the upper level optimization problem, its objective depends on the solution of at least two lower level optimal control problems that give the minimum possible race times for the different aircraft types for fixed positions of the gates. Fig. 68 depicts the basic principle of the bilevel optimal control problem that has to be solved in order to achieve equal chances for the participating pilots.

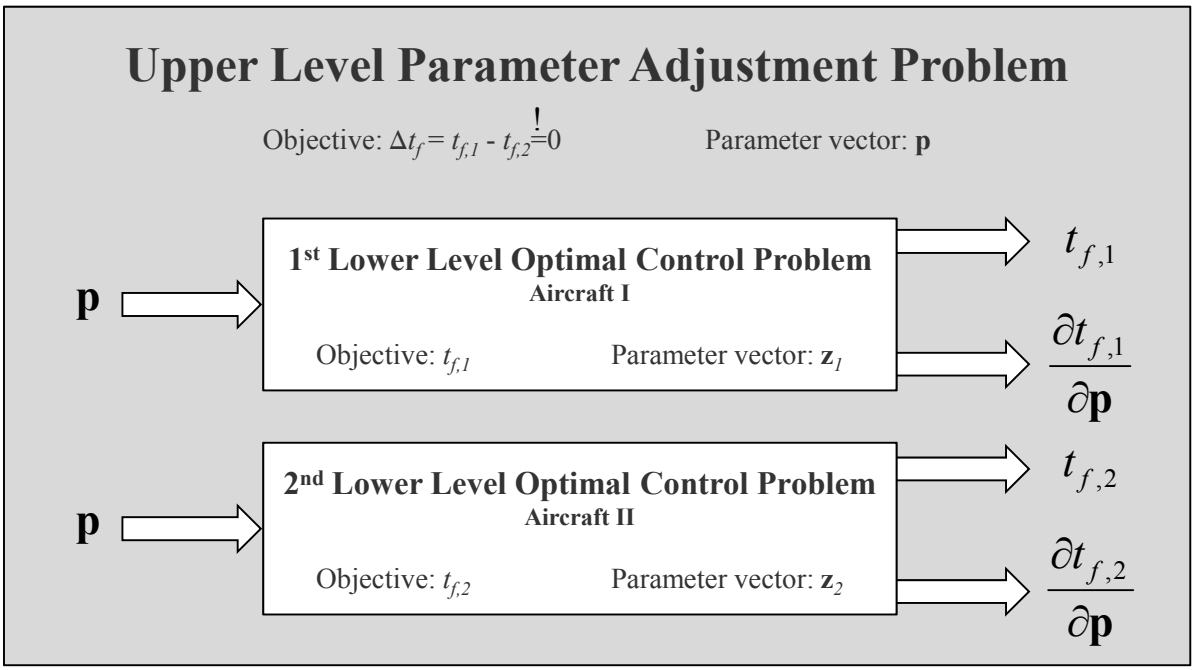

Figure 68. Fairness Bilevel Optimal Control Problem

The goal of the upper level optimization problem is to position the air race gates such that two different aircraft require the same minimum race time t_f . Besides the forward and sideward positions x and y of the gates also the azimuth angles ψ of the gates are considered as free parameters so that the parameter vector **p** is given by

$$\mathbf{p} = \left[x_i, y_i, \psi_i\right] \quad i = 1,...,s \tag{6.119}$$

where s is the number of race gates.

In order to enforce race track elements like e.g. the “Quadro” or the “Chicane”, the upper level optimization problem is augmented by the same constraints as the bilevel optimal control problem concerned with the safety of air races that is described in chapter 6.3.

Table 16 gives an overview of the parameter values that have been utilized for the point-mass simulation model of the two aircraft. For the second aircraft, the maximum thrust T_{max} is set to $0.7mg$, slowing down the aircraft. At the same time, the parameter k associated with the drag polar of the aircraft is reduced. By this, the drag increase due to an increase of the lift is

reduced so that higher angles of attack α_A come along with a smaller drag coefficient C_D than in case of the first aircraft. This might be beneficial for the second aircraft when flying at high bank angles μ_K, allowing for sharper turns without losing too much speed and thus making the second aircraft more agile.

Aircraft Specifications		I	II
mass	m [kg]	693.0	693.0
wing area	S [m²]	8.928	8.928
wing span	b [m]	7.5	7.5
reference speed	V_{ref} [m/s]	30.0	30.0
maximum thrust	T_{max} [N]	$0.8mg$	$0.7mg$
aerodynamic coefficients	C_{D0} [-]	0.0761	0.0761
	k [-]	0.051340000	0.024065625
	$C_{Q\beta}$ [-]	-0.589355	-0.589355
	$C_{L\alpha}$ [-]	4.75	4.75
	C_{L0} [-]	0.055	0.055

Table 16. Aircraft Specifications

The solution of the race fairness problem is accomplished in two different ways: first, the upper level fairness problem is formulated as a parameter optimization problem and then solved using a gradient-based optimization method. Second, the fairness problem is stated as a system of equations that can then be solved using any root-finding algorithm like e.g. Newton's method.

Formulation of the Fairness Problem as an Optimization Problem

The upper level fairness optimization problem can be stated as follows: Find a parameter vector $\mathbf{p}$ restricted by its lower bound $\mathbf{p}_{LB}$ and its upper bound $\mathbf{p}_{UB}$,

$$\mathbf{p}_{LB} \leq \mathbf{p} \leq \mathbf{p}_{UB} \tag{6.120}$$

such that the objective

$$J = \left(t_{f,1}(\mathbf{p}) - t_{f,2}(\mathbf{p})\right)^2 \tag{6.121}$$

is minimized. Here, $t_{f,1}$ is the minimum possible race time of the first aircraft type and $t_{f,2}$ the minimum possible race time of the second aircraft type. For this objective the square of the time difference has been chosen so that it is ensured that the objective function is positive and continuously differentiable.

The gradient of the objective J with respect to the parameter $\mathbf{p}$ is

$$J_{\mathbf{p}} = 2 \cdot \left(t_{f,1}(\mathbf{p}) - t_{f,2}(\mathbf{p}) \right) \cdot \left(\frac{\partial t_{f,1}(\mathbf{p})}{\partial \mathbf{p}^T} - \frac{\partial t_{f,2}(\mathbf{p})}{\partial \mathbf{p}^T} \right) \tag{6.122}$$

At this, the sensitivity results obtained by Eq. (5.12) in chapter 5.2 can be utilized directly to compute the gradient $J_{\mathbf{p}}$ so that the bilevel optimal control problem can be solved very efficiently. As for the lower level optimal control problems, SNOPT (Ref. [Gill, 2007]) has been utilized to solve the upper level optimization problem. Furthermore, since the two lower level optimal control problems do not depend on each other, the optimization of the optimal control problems has been parallelized in order to further shorten the time required for the solution of the entire bilevel programming problem.

Fig. 69 depicts the final result for the fair race track layout where the final time difference equals −0.85ms. The initial time difference between the minimum race times of the two aircraft has been 1.057s.

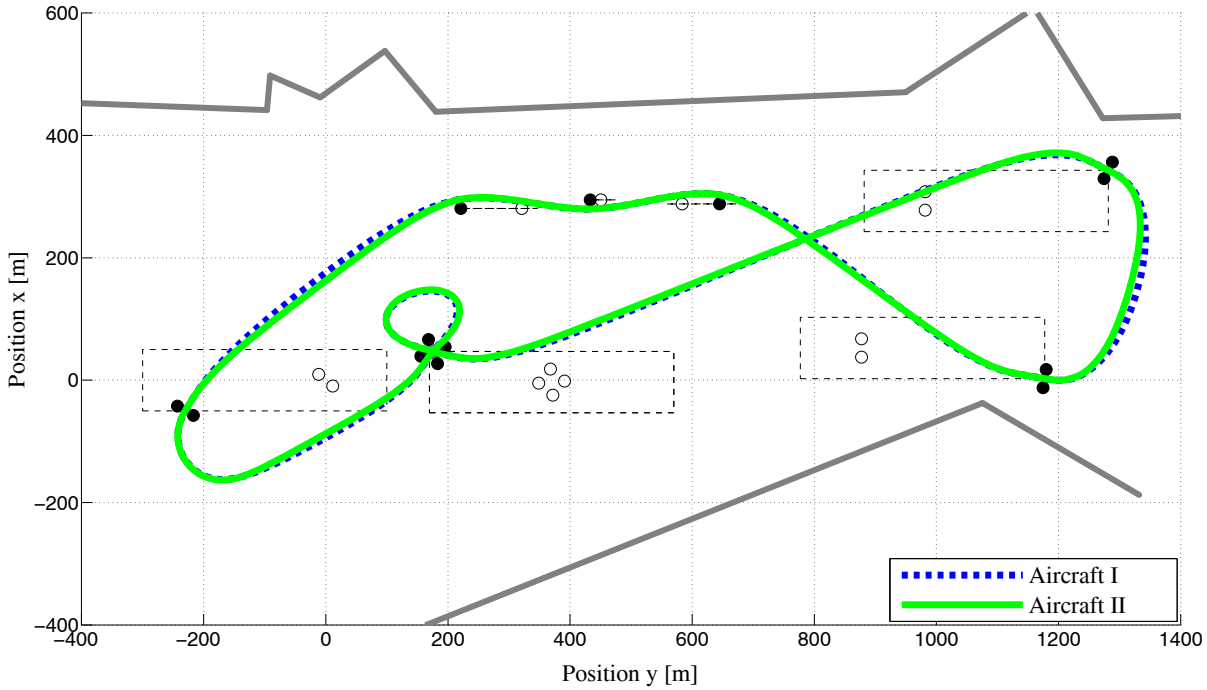

Figure 69. Race Track optimized w.r.t. to Fairness using SNOPT

Solution of the Fairness Problem using Newton's Method

The solution of the upper level fairness problem can also be regarded as the solution of a system of equations: for a race track layout to be a fair race track, the final race times of the two competing aircraft have to be the same, this means the difference between the final race times has to equal zero:

$$\Delta t_f = t_{f,1}(\mathbf{p}) - t_{f,2}(\mathbf{p}) \overset{!}{=} 0 \tag{6.123}$$

Together with the constraints that have to be imposed to maintain basic race track elements like e.g. the "Chicane" or the "Quadro", a system of equations results that can then be solved using any appropriate root finding algorithm:

$$\mathbf{f} = \begin{pmatrix} \Delta t_f \\ \mathbf{C} \end{pmatrix} \overset{!}{=} \mathbf{0} \tag{6.124}$$

Here, **C** denotes the constraint vector due to the race track elements that are to be maintained. In the following, Newton's method is applied for finding the roots of the above system of equations. At this, Eq. (6.124) is rewritten into a first-order Taylor series expansion:

$$\mathbf{f} = \mathbf{f}(\mathbf{p}_0) + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{p}}\right|_0 \cdot (\mathbf{p} - \mathbf{p}_0) = \mathbf{f}(\mathbf{p}_0) + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{p}}\right|_0 \cdot \Delta\mathbf{p} \overset{!}{=} \mathbf{0} \tag{6.125}$$

Eq. (6.125) can then be solved for $\Delta\mathbf{p}$:

$$\Delta\mathbf{p} = -\left[\left.\frac{\partial \mathbf{f}}{\partial \mathbf{p}}\right|_0\right]^+ \cdot \mathbf{f}(\mathbf{p}_0) \tag{6.126}$$

where $^+$ denotes the pseudo-inverse since the number of equations does not necessarily equal the number of optimization parameters. Here again, the sensitivity results obtained by Eq. (5.12) in chapter 5.2 can be used directly to compute the gradient matrix $\partial\mathbf{f}/\partial\mathbf{p}$. Finally, the following iteration step has to be done to drive the system of equations (6.124) iteratively to zero:

$$\mathbf{p} = \mathbf{p}_0 + \lambda \cdot \Delta\mathbf{p} \tag{6.127}$$

where λ can be utilized to adjust the step size of the Newton step. So far, no upper or lower bounds with respect to the parameter vector **p** have been taken into account. Therefore, it is first checked if any elements of the parameter vector **p** violate their upper or lower boundaries. If this is the case, the values of the respective elements are reset to their boundaries. Then, an additional step $\Delta\mathbf{p}_{red}$ in the parameter vector **p** is computed that takes into account only those elements of the parameter vector **p** that are not at their boundaries. Accordingly, only the corresponding columns of the gradient matrix $\partial\mathbf{f}/\partial\mathbf{p}$ are taken into account, giving the reduced gradient matrix $\partial\mathbf{f}/\partial\mathbf{p}_{red}$. The additional step $\Delta\mathbf{p}_{red}$ is computed by:

$$\Delta\mathbf{p}_{red} = -\left[\left.\frac{\partial \mathbf{f}}{\partial \mathbf{p}}\right|_{0,red}\right]^+ \cdot \left[\mathbf{f}(\mathbf{p}_0) + \left.\frac{\partial \mathbf{f}}{\partial \mathbf{p}}\right|_0 \cdot \Delta\mathbf{p}_{bounded}\right] \tag{6.128}$$

Here, $\Delta\mathbf{p}_{bounded}$ is the step $\Delta\mathbf{p}$ given by Eq. (6.126) once the upper and lower boundaries have been imposed. Finally, the new parameter vector **p** evaluates to:

$$\mathbf{p} = \mathbf{p}_0 + \Delta\mathbf{p}_{bounded} + \Delta\mathbf{p}_{red} \tag{6.129}$$

The computation steps given by Eq. (6.128) and Eq. (6.129) can then be repeated until the newly obtained additional step $\Delta\mathbf{p}_{red}$ does not violate any of the boundaries of the parameter vector **p**. At this, the upper and lower boundaries with respect to the parameter vector **p** are taken into account while the remaining degrees of freedom are utilized to drive the constraint vector **f** to zero.

In Fig. 70, the fair race track layout for the above stated fairness problem is depicted that has been obtained by applying the outlined method where the final time difference is −0.99ms.

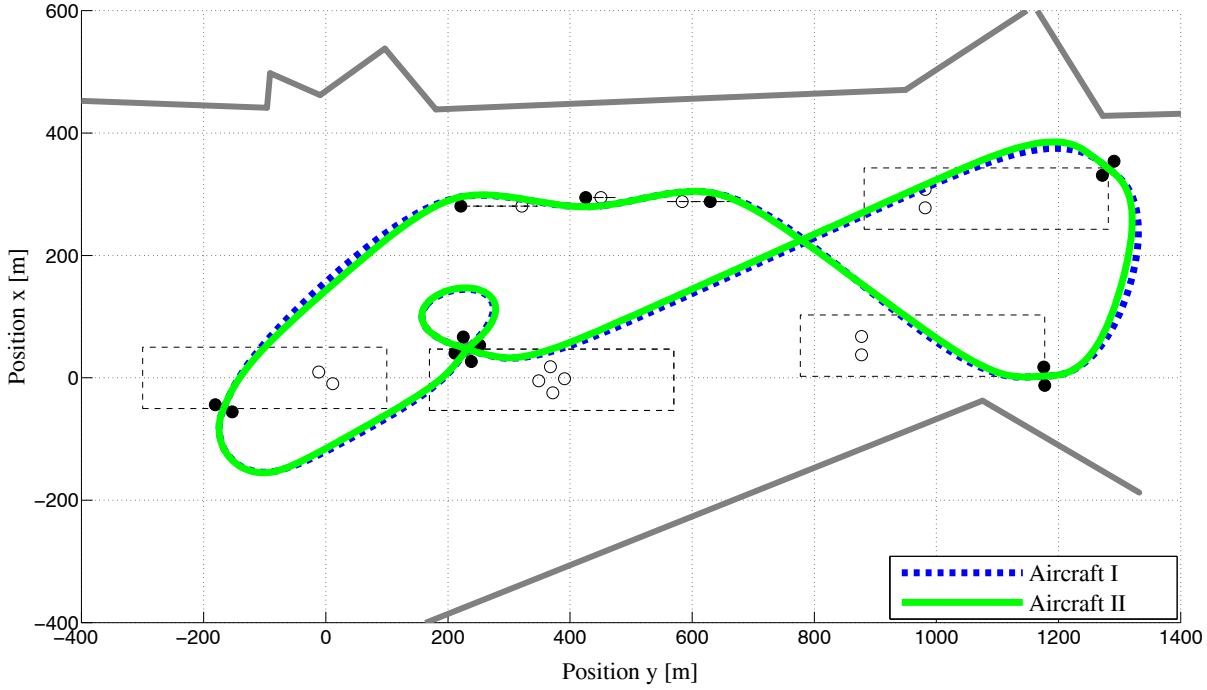

Figure 70. Race Track optimized w.r.t. Fairness using Newton's Method

6.4.2 Increasing the Fairness for Three Competing Aircraft

In the following, a race fairness problem for three competing aircraft is posed. For the third aircraft the maximum thrust is set to $0.75mg$ while the drag polar parameter k is reduced to 0.038505. The upper level fairness problem can then be stated as follows: Find a parameter vector $\mathbf{p}$ such that the objective

$$\begin{aligned} J &= \left(t_{f,1}(\mathbf{p}) - t_{f,2}(\mathbf{p})\right)^2 + \left(t_{f,1}(\mathbf{p}) - t_{f,3}(\mathbf{p})\right)^2 + \left(t_{f,2}(\mathbf{p}) - t_{f,3}(\mathbf{p})\right)^2 \\ &= \left(\Delta t_{f1,2}\right)^2 + \left(\Delta t_{f1,3}\right)^2 + \left(\Delta t_{f2,3}\right)^2 \end{aligned} \tag{6.130}$$

is minimized where $t_{f,1}$ is the minimum possible race time of the first aircraft type, $t_{f,2}$ the minimum possible race time of the second aircraft type and $t_{f,3}$ the minimum possible race time of the third aircraft type for the given parameter vector $\mathbf{p}$. As before, upper and lower bounds with respect to the parameter vector $\mathbf{p}$ have to be taken into account:

$$\mathbf{p}_{LB} \le \mathbf{p} \le \mathbf{p}_{UB} \tag{6.131}$$

Then, the gradient of the objective J with respect to the parameter $\mathbf{p}$ evaluates to:

$$\begin{aligned} J_{\mathbf{p}} = {} & 2 \cdot \left(t_{f,1}(\mathbf{p}) - t_{f,2}(\mathbf{p})\right) \cdot \left(\frac{\partial t_{f,1}(\mathbf{p})}{\partial \mathbf{p}^T} - \frac{\partial t_{f,2}(\mathbf{p})}{\partial \mathbf{p}^T} \right) + \\ & 2 \cdot \left(t_{f,1}(\mathbf{p}) - t_{f,3}(\mathbf{p})\right) \cdot \left(\frac{\partial t_{f,1}(\mathbf{p})}{\partial \mathbf{p}^T} - \frac{\partial t_{f,3}(\mathbf{p})}{\partial \mathbf{p}^T} \right) + \\ & 2 \cdot \left(t_{f,2}(\mathbf{p}) - t_{f,3}(\mathbf{p})\right) \cdot \left(\frac{\partial t_{f,2}(\mathbf{p})}{\partial \mathbf{p}^T} - \frac{\partial t_{f,3}(\mathbf{p})}{\partial \mathbf{p}^T} \right) \end{aligned} \tag{6.132}$$

Again, the sensitivity results given by Eq. (5.12) in chapter 5.2 are utilized to compute the gradient $J_{\mathbf{p}}$. As for the fairness problem with two competing aircraft, SNOPT (Ref. [Gill, 2007]) has been utilized to solve the upper level optimization problem. Furthermore, the

solution of the three lower level optimal control problems can again been parallelized in order to shorten the time required for the solution of the entire bilevel programming problem.

Fig. 71 depicts the final result for the race track layout. While the initial time differences evaluated to $\Delta t_{f1,2} = 1.057\text{s}$, $\Delta t_{f1,3} = 0.508\text{s}$ and $\Delta t_{f2,3} = -0.549\text{s}$, the time differences finally could be reduced to $\Delta t_{f1,2} = 0.0021\text{s}$, $\Delta t_{f1,3} = -0.0017\text{s}$ and $\Delta t_{f2,3} = -0.0038\text{s}$.

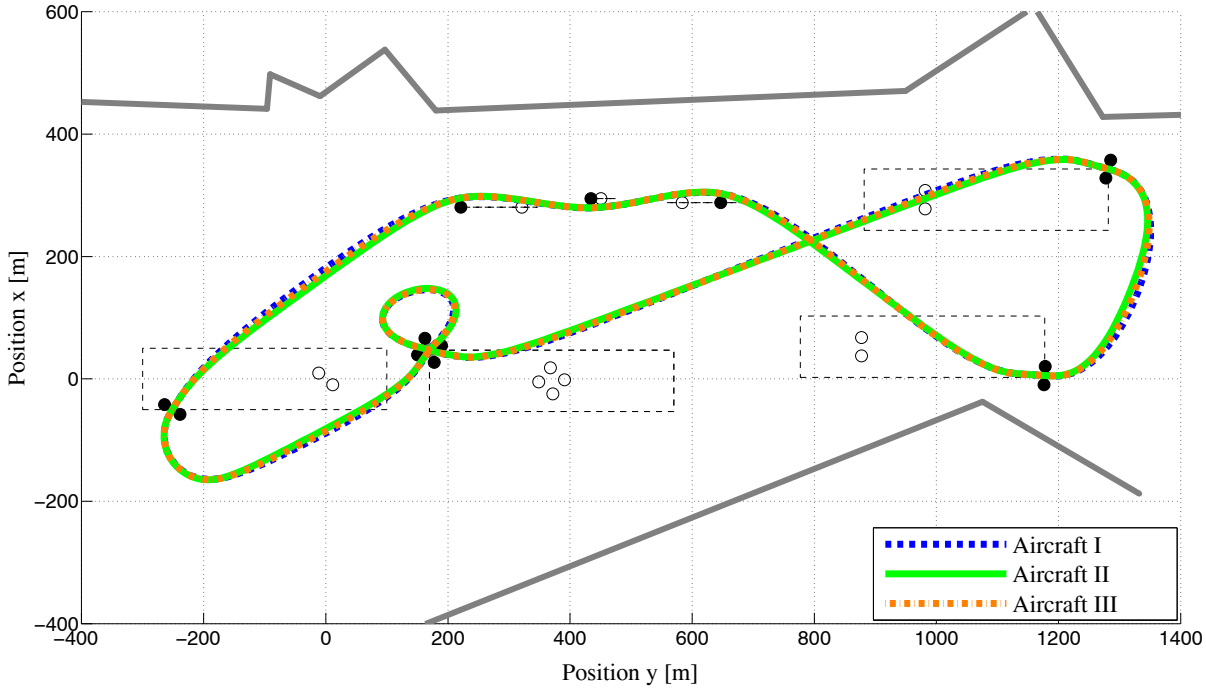

Figure 71. Race Track optimized w.r.t. Fairness for 3 Competing Aircraft

6.4.3 Increasing the Fairness and the Safety of Air Races

As one might notice from Fig. 71, the time optimal trajectories for the fair race track layout get very close to the crowd line. Therefore, it is straight forward to combine the aspect of fairness with one or more of the above defined safety criteria. In the following, a race track layout is sought where a maximum level of fairness for three competing aircraft is achieved and where the time optimal trajectories keep a certain minimum distance to the crowd. Thus, the following upper level parameter problem taking into account the fairness and the safety of the race track layout is stated: Find a parameter vector $\mathbf{p}$ such that the objective

$$\begin{aligned} J &= \left(t_{f,1}(\mathbf{p}) - t_{f,2}(\mathbf{p})\right)^2 + \left(t_{f,1}(\mathbf{p}) - t_{f,3}(\mathbf{p})\right)^2 + \left(t_{f,2}(\mathbf{p}) - t_{f,3}(\mathbf{p})\right)^2 \\ &= \left(\Delta t_{f1,2}\right)^2 + \left(\Delta t_{f1,3}\right)^2 + \left(\Delta t_{f2,3}\right)^2 \end{aligned} \tag{6.133}$$

is minimized subject to

$$d_{C,1}(\mathbf{p},t) - d_{C,\min} \geq 0 \tag{6.134}$$

$$d_{C,2}(\mathbf{p},t) - d_{C,\min} \geq 0 \tag{6.135}$$

$$d_{C,3}(\mathbf{p},t) - d_{C,\min} \geq 0 \tag{6.136}$$

where the distance to crowd $d_{C,t}$ is computed as explained in chapter 6.3.2.1. The sensitivity results given by Eq. (5.12) in chapter 5.2 are utilized to compute the gradient $J_{\mathbf{p}}$ of the objective as well as the Jacobian of the constraints. It is mentioned that the upper level

parameter optimization problem can be augmented by any of the above stated safety criteria. Moreover, multiple safety criteria might be added to the fairness problem. In order to keep the dimension of the upper level parameter optimization problem as small as possible, the path constraints can be taken into account only in areas of the flight trajectories where it is likely that the safety criterion will be violated.

In Fig. 72, the final result for the fair and safe race track layout is shown. As can be seen from Fig. 73, the minimum distance to crowd $d_{C,t}$ for the final race track layout lies above the threshold of 100.0 m for all three competing aircraft while the final time differences equal zero.

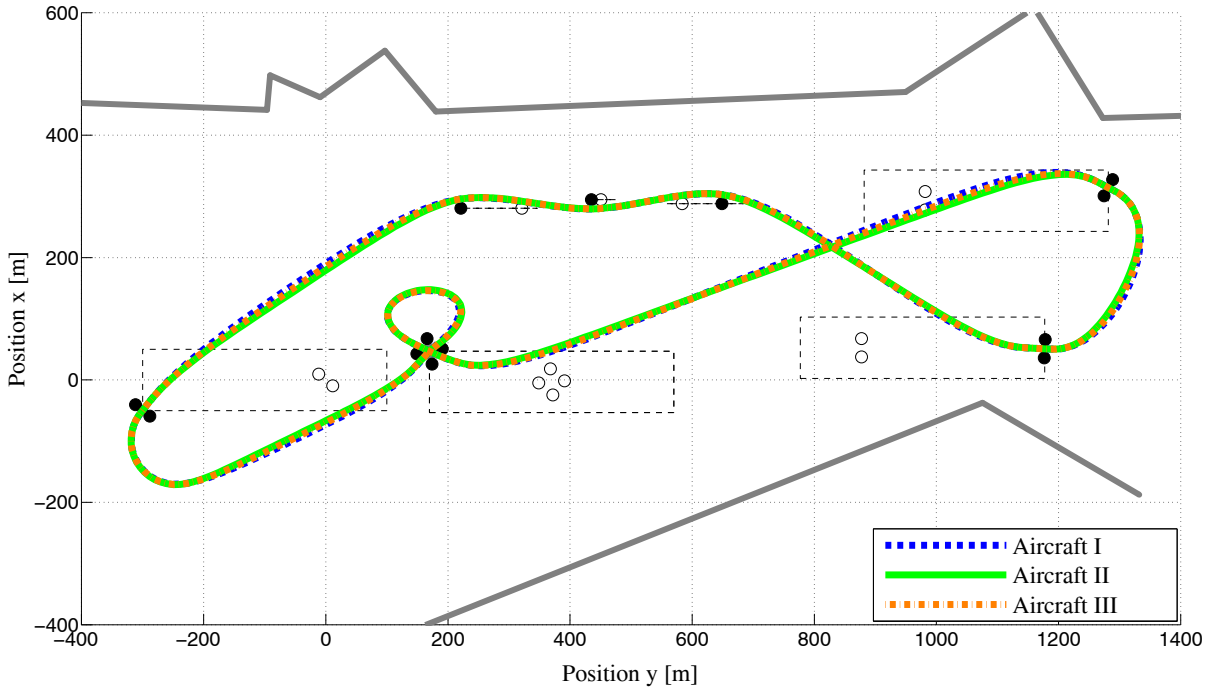

Figure 72. Race Track optimized w.r.t. Fairness and Safety

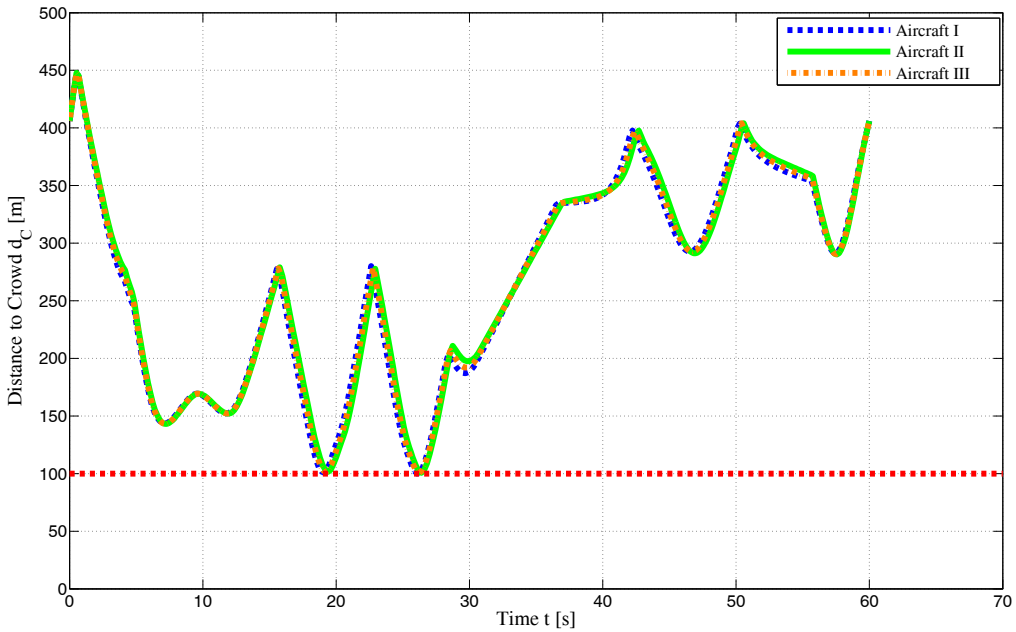

Figure 73. Distance to Crowd in Combination with Race Fairness

7

Summary and Perspectives

7.1 Summary

Within the thesis at hand, a framework is established that allows for the efficient and robust solution of highly complex trajectory optimization problems as well as a special type of bilevel optimal control problems. In the following, the main topics of the thesis are recapitulated.

In chapter 2, a general framework for the treatment and the solution of optimal control problems is depicted that is based on the multiple shooting method. After the statement of the general optimal control problem, the infinite dimensional continuous time optimal control problem is transformed into a finite dimensional parameter optimization problem by discretization of the control variables. The Jacobian and the Hessian of the transformed optimal control problem are derived analytically using the gradient of the dynamic system with respect to the control vector and the state vector in conjunction with the appropriate sensitivity equations. For the analytical evaluation of the gradient of the dynamic system, a method is depicted that follows the block structure of the implemented simulation model and that is very flexible to modifications with regard to the simulation model. The transformed optimal control problem is scaled to improve the stability and the convergence for the computation of the optimal solution. A mesh refinement algorithm with respect to the control grids is introduced in order to reproduce the continuous time controls by the discretized controls as closely as possible. At the same time, unnecessary control discretization points are dropped to reduce the size of the parameter optimization problem. Furthermore, path constraint violations in between the current path constraint grid points are detected and additional grid points are inserted to cancel out the violations.

The structure and the various subsystems of a scalable, multi-fidelity simulation model that is specifically tailored for optimization tasks are illustrated in chapter 3. The simulation model features a special sequential structure that follows the causal dynamic chain of flight systems and that is the non-linear point-mass simulation model in the outer loop followed by various representations for the rotational and attitude dynamics in the inner loop. The depth of modeling for the inner loop is scalable from load factor transfer functions via state-space models to the full, non-linear rotation and attitude equations of motion. The simulation model takes into account environmental influences like static and convective wind fields and comprises additional subsystems like e.g. actuator dynamics in order to achieve optimal trajectories that are as realistic as possible. Based on the principle of dynamic inversion, a controller is implemented that features the same sequential structure as the system dynamics itself and that also accounts for the environmental influences. The simulation model is

augmented by reference models to produce the required derivatives for the command inputs fed into the inversion controller. Error feedbacks are implemented to cancel out deviations between the inversion controller and the states of the simulation plant due to possible non-minimum phase effects and numerical computation impreciseness.

Based on the optimization simulation model, an algorithm for the solution of highly complex trajectory optimization problems without the necessity for the user to provide any initial guess is outlined in chapter 4. A homotopy procedure is introduced that allows for the generation of an initial guess for the optimization of highly non-linear trajectories using the point-mass simulation model. Starting with the optimal solution for the point-mass simulation model, the algorithm increases the modeling fidelity of the simulation model step by step, ending up with the optimal trajectory for the full, non-linear 6-DoF simulation model. In order to perform the step from the pure point-mass simulation model to the point-mass simulation model augmented by the linear transfer functions within the optimization algorithm, a substitute optimization problem is formulated.

In chapter 5, a bilevel optimal control problem is stated that comprises a parameter optimization problem at the upper level and one or more optimal control problems at the lower level. Details for the sensitivity analysis with regard to the discretized lower level optimal control problems are given. The sensitivity analysis allows for an analytical evaluation of the gradient of the upper level parameter optimization problem, thus avoiding the time-consuming evaluation of the gradient by numerical methods. Thus, an efficient method for the solution of the stated bilevel optimal control problem is developed. The given gradient evaluation method can also be utilized within a multidisciplinary design optimization framework that involves the discipline of trajectory optimization for an efficient and accurate evaluation of the gradient of the upper level optimization problem.

The proposed solution methods for highly complex trajectory optimization problems as well as for the special type of bilevel optimal control problems have been applied to the very challenging task of optimizing the race trajectories respectively the race track layouts for a Red Bull Air Race. In chapter 6.1, the Air Race problem is formulated as a benchmark. A minimum time air race trajectory for the full, non-linear 6-Degree-of-Freedom point-mass simulation model is depicted. Bilevel optimal control problems as stated in chapter 5 result if one wants to increase the safety respectively the fairness of the race track since the pilots will not take care of these aspects as long as no penalties are imposed. The pilots are focused on flying the race track such that the minimum possible race time results, and the designer of the race track layout has to arrange the gates such that certain safety criteria or fairness are optimal if the pilots fly the minimum-time race trajectory. Various safety criteria that are the distance to crowd, the time to crowd, the maximum directed energy, the ballistic extrapolation, the pilot blinding and a load factor fatigue index are derived and the corresponding results are shown. Finally, for two different aircraft with one aircraft being more agile and the other aircraft being faster a fair layout of the race track is found where both aircraft require the same minimum race time.

7.2 Perspectives

Within the framework for the solution of the trajectory optimization problem, the multiple shooting method that is a direct method for the solution of optimal control problems has been implemented. For the multiple shooting method, only the controls are discretized, while for

the states only a limited number of multiple shooting nodes is introduced. The state time histories are then obtained by integration (or "shooting") of the dynamic system from one multiple shooting node to the next. Other direct methods are the collocations methods, where the states are discretized together with the controls. Especially the so-called pseudospectral methods have gained much attention during the last years, and a lot of research has been carried out on this topic. With the collocation methods, there is no integration of the dynamic system but the equations of motion are added as supplementary constraints to the discretized optimal control problem that have to be fulfilled at the optimal solution. At this, the number of parameters as well as the numbers of constraints is increased significantly within the discrete parameter optimization problem in comparison to the multiple shooting method, especially if a large number of states and controls is present. But nowadays, efficient and powerful algorithms for the solution of large, nonlinear parameter optimization problems with up to thousands of constraints and variables are available, especially for the optimization of sparse problems, i.e. optimization problems where many of the elements of the Jacobian are zero. At the same time, the cost of integrating dynamic systems with a large number of states in combination with an enormous number of first and second order sensitivity equations for the computation of the Jacobian respectively the Hessian can be avoided with the collocation methods. Thus, collocation methods could be investigated for the solution of highly complex trajectory optimization problems in order to compare their performance to that of the multiple shooting method.

With regard to the bilevel optimal control problem for the layout of air race tracks, further safety criteria have to be developed and implemented like e.g. the time to ground, the maximum wing root bending moment, the engine gyroscopic moment or various load factor consciousness, illusion or fatigue indices. Furthermore, the existing safety criteria might also be developed further: for example, the prediction algorithm for the calculation of the safety criterion time to crowd might be extended taking into account the current flight path bank angle of the aircraft since at high bank angles a curved propagation of the aircraft trajectory might be more realistic than a straight extrapolation. For the fairness aspect, not only the positions of the air race gates might be seen as parameters of the upper level optimization problem but also aircraft parameters like e.g. the aircraft mass. This would imply that superior aircraft have to take on board additional penalty mass to level out the chances between the participating aircraft.

Another crucial aspect that may be subject to further research is related to the numerous aircraft parameters that are implemented in the simulation models. So far, most of the aircraft parameters are obtained by estimation. More realistic and feasible aircraft parameters have to be derived e.g. by parameter estimation methods in order to provide reliable results and statements concerning the fairness and the safety of the considered race tracks.

In summary, by the thesis at hand a framework has been established that allows for the solution of two very challenging optimization tasks, namely the optimization of flight trajectories using high-fidelity simulation models as well as the efficient solution of complex bilevel optimal control problems. Especially the generation of suitable initial guesses for the trajectory optimization based on high-fidelity simulation models is seen as main contribution of this thesis.

A

Appendix

A.1 Second Order State Sensitivity Equations

The full second order state sensitivity equations are:

$$\begin{aligned}
\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \dot{\mathbf{x}}}{\partial \mathbf{z}^T}\right)\right) = &\left[\mathbf{I}_{n_{\text{var}}} \otimes \frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\right]\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T}\right)\right) + \left[\left(\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T}\right)^T \otimes \mathbf{I}_{n_S}\right]\cdot \\
&\cdot\left\{\frac{\partial}{\partial \mathbf{x}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\right)\right)\cdot\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T} + \frac{\partial}{\partial \mathbf{u}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\right)\right)\cdot\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T} + \right. \\
&\left. + \frac{\partial}{\partial t_f}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\right)\right)\cdot\frac{\partial t_f}{\partial \mathbf{z}^T} + \frac{\partial}{\partial \mathbf{p}}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}^T}\right)\right)\cdot\frac{\partial \mathbf{p}}{\partial \mathbf{z}^T}\right\} + \\
&+\left[\mathbf{I}_{n_{\text{var}}} \otimes \frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\right]\frac{\partial}{\partial \mathbf{z}^T}\left(vec\left(\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T}\right)\right) + \left[\left(\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T}\right)^T \otimes \mathbf{I}_{n_S}\right]\cdot \\
&+\left\{\frac{\partial}{\partial \mathbf{x}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\right)\right)\cdot\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T} + \frac{\partial}{\partial \mathbf{u}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\right)\right)\cdot\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T}\right. \\
&\left. + \frac{\partial}{\partial t_f}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\right)\right)\cdot\frac{\partial t_f}{\partial \mathbf{z}^T} + \frac{\partial}{\partial \mathbf{p}}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}^T}\right)\right)\cdot\frac{\partial \mathbf{p}}{\partial \mathbf{z}^T}\right\} + \\
&+\left[\left(\frac{\partial t_f}{\partial \mathbf{z}^T}\right)^T \otimes \mathbf{I}_{n_S}\right]\cdot \\
&\cdot\left\{\frac{\partial}{\partial \mathbf{x}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial t_f}\right)\right)\cdot\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T} + \frac{\partial}{\partial \mathbf{u}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial t_f}\right)\right)\cdot\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T} + \right. \\
&\left. + \frac{\partial}{\partial t_f}\left(vec\left(\frac{\partial \mathbf{f}}{\partial t_f}\right)\right)\cdot\frac{\partial t_f}{\partial \mathbf{z}^T} + \frac{\partial}{\partial \mathbf{p}}\left(vec\left(\frac{\partial \mathbf{f}}{\partial t_f}\right)\right)\cdot\frac{\partial \mathbf{p}}{\partial \mathbf{z}^T}\right\} + \\
&+\left[\left(\frac{\partial \mathbf{p}}{\partial \mathbf{z}^T}\right)^T \otimes \mathbf{I}_{n_S}\right]\cdot \\
&\cdot\left\{\frac{\partial}{\partial \mathbf{x}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}^T}\right)\right)\cdot\frac{\partial \mathbf{x}(t)}{\partial \mathbf{z}^T} + \frac{\partial}{\partial \mathbf{u}^T}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}^T}\right)\right)\cdot\frac{\partial \mathbf{u}(t)}{\partial \mathbf{z}^T} + \right. \\
&\left. + \frac{\partial}{\partial t_f}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}^T}\right)\right)\cdot\frac{\partial t_f}{\partial \mathbf{z}^T} + \frac{\partial}{\partial \mathbf{p}}\left(vec\left(\frac{\partial \mathbf{f}}{\partial \mathbf{p}^T}\right)\right)\cdot\frac{\partial \mathbf{p}}{\partial \mathbf{z}^T}\right\} +
\end{aligned} \tag{A.1}$$

A.2 Frames

In the following, the various coordinate systems and their characteristics are described according to Ref. [Holzapfel, 2009a].

A.2.1 Earth-Centered Inertial (*ECI*) Frame *I*

Index:	*I*
Role:	Notation frame for Newtonian Inertial Physics (i.e. valid Euclidean Frame)
Origin:	Center of Earth
Translation:	Around the Sun with solar system
Rotation:	None
x-axis:	In equatorial plane, pointing towards vernal equinox
y-axis:	In equatorial plane to form a right-hand system
z-axis:	Rotation axis of Earth

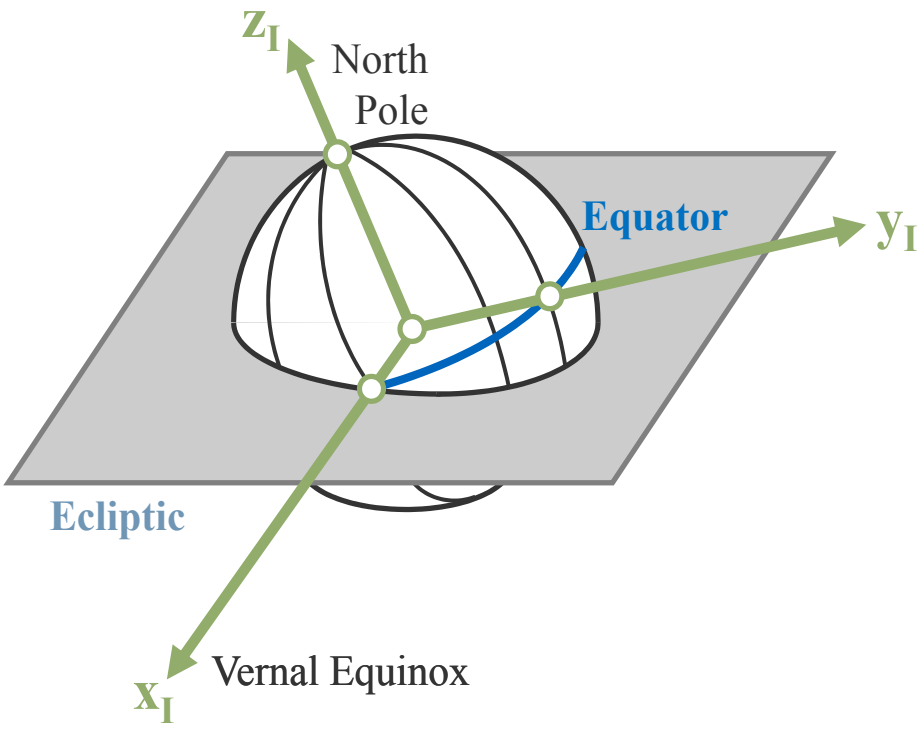

Figure 74. *ECI*-Frame *I*

A.2.2 Earth-Centered-Earth-Fixed (*ECEF*) Frame *E*

Index:	*E*
Role:	Notation frame for positioning and navigation
Origin:	Center of Earth
Translation:	Moves with *ECI*-Frame
Rotation:	Earth rotation about *z*-axis with Earth angular rate, i.e. approximately 2π/24h
x-axis:	In equatorial plane, pointing through Greenwich meridian
y-axis:	In equatorial plane to form a right-hand system
z-axis:	Rotation axis of Earth

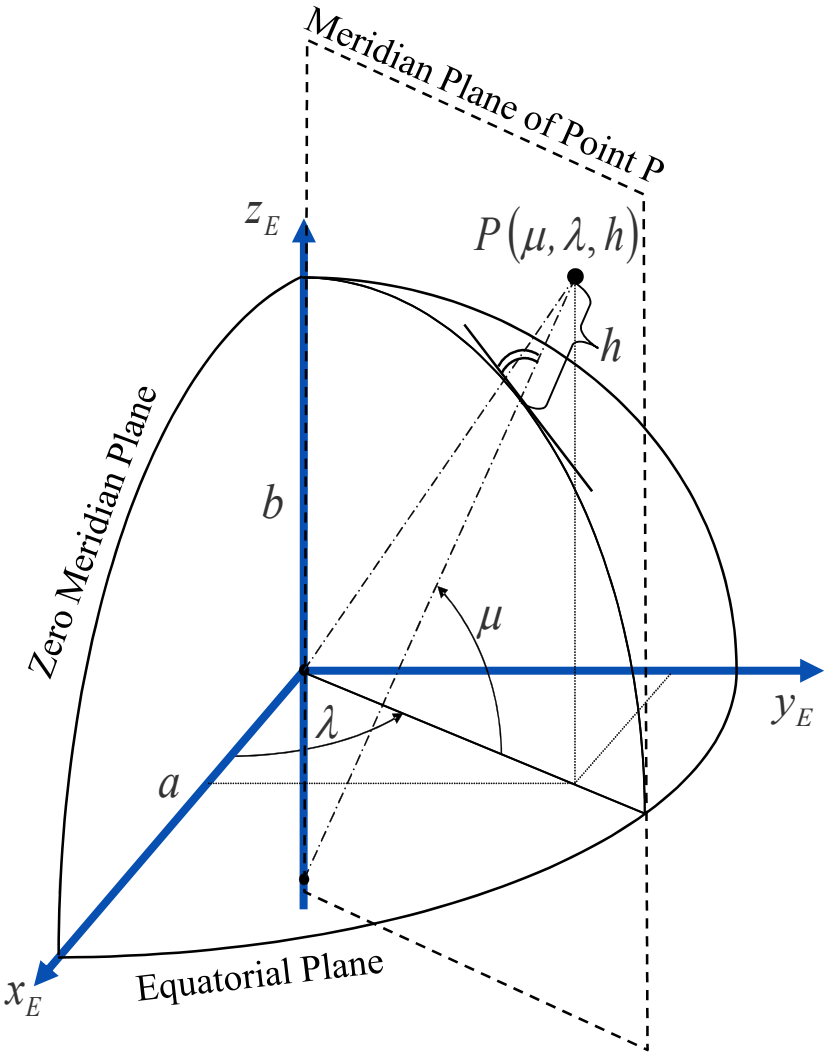

Figure 75. *ECEF*-Frame *E*

A.2.3 North-East-Down (*NED*) Frame *O*

Index:	*O*
Role:	Notation frame for velocity and orientation
Origin:	Reference point of aircraft
Translation:	Moves with aircraft reference point
Rotation:	Rotates with transport rate to keep the *NED*-alignment
x-axis:	Parallel to local geoid surface, pointing to geographic north pole
y-axis:	Parallel to local geoid surface, pointing east to form a right hand system
z-axis:	Pointing downwards, perpendicular to local geoid surface

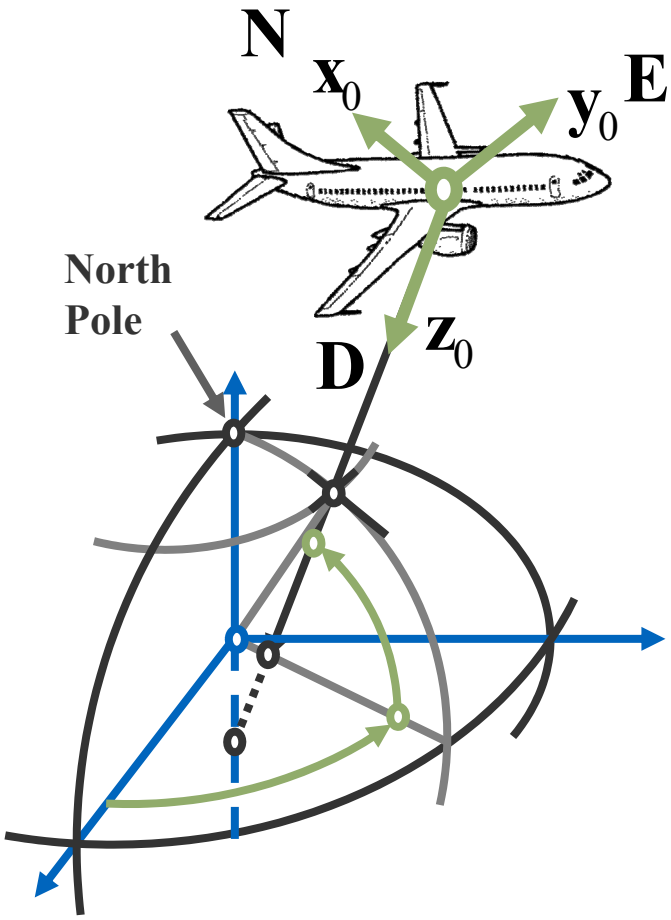

Figure 76. North-East-Down (*NED*) Frame *O*

A.2.4 Body-Fixed Frame *B*

Index:	*B*
Role:	Notation frame
Origin:	Reference point of aircraft
Translation:	Moves with aircraft reference point
Rotation:	Rotates with rigid body aircraft

x-axis:	Pointing towards aircraft nose in symmetry plane
y-axis:	Pointing to right (starboard) wing to form an orthogonal right-hand system
z-axis:	Pointing downwards in symmetry plane, perpendicular to *x*- and *y*-axes

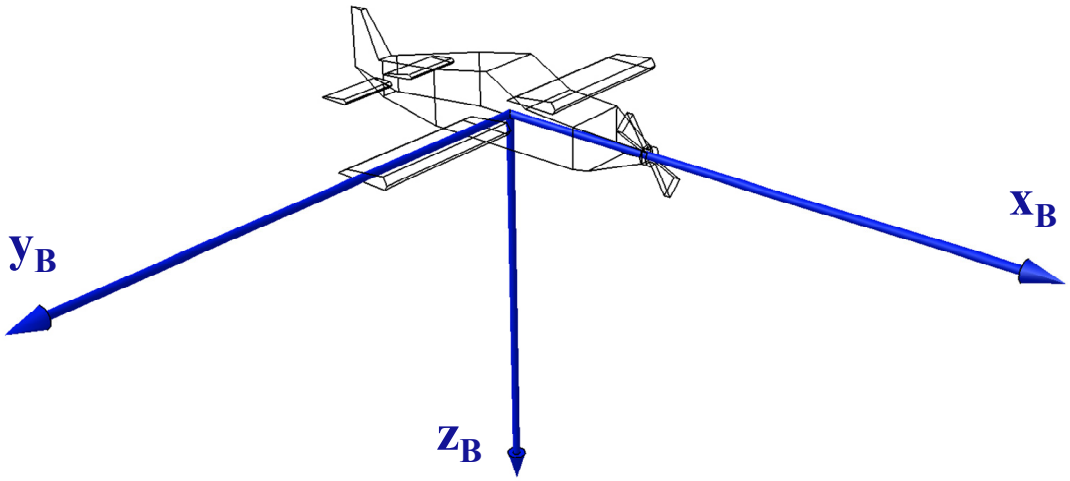

Figure 77. Body-Fixed Frame *B*

A.2.5 Kinematic-Frame *K*

Index:	*K*
Role:	Notation frame for flight path
Origin:	Reference point of aircraft
Translation:	Moves with aircraft reference point
Rotation:	Rotates with direction of kinematic aircraft motion

x-axis:	Aligned with the kinematic velocity, pointing into the direction of the kinematic velocity
y-axis:	Pointing to right, perpendicular to the *x*- und *z*- axes
z-axis:	Pointing downwards, parallel to the projection of the local surface normal of the WGS-84 ellipsoid into a plane perpendicular to the *x*-axis (i.e. perpendicular to the kinematic velocity)

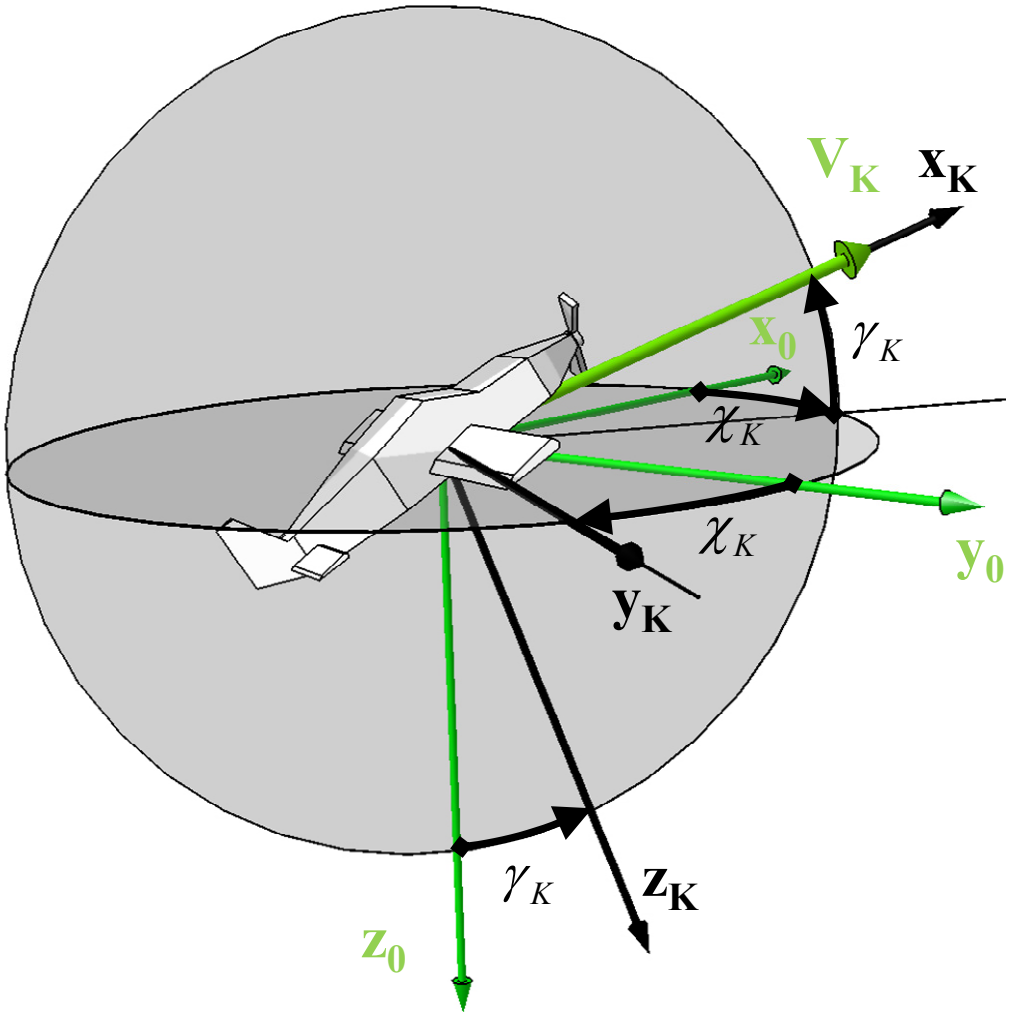

Figure 78. Kinematic-Frame *K*

A.2.6 Intermediate Kinematic Frame $\bar{K}$

Index:	$\bar{K}$
Role:	Notation frame
Origin:	Reference point of aircraft
Translation:	Moves with aircraft reference point
Rotation:	Rotates with direction of kinematic aircraft motion
x-axis:	Aligned with the kinematic velocity, pointing into the direction of the kinematic velocity
y-axis:	Pointing to right, perpendicular to the *x*- und *z*- axes
z-axis:	*z*-axis of the Kinematic Frame *K* rotated clockwise by the kinematic flight-path bank angle μ_K

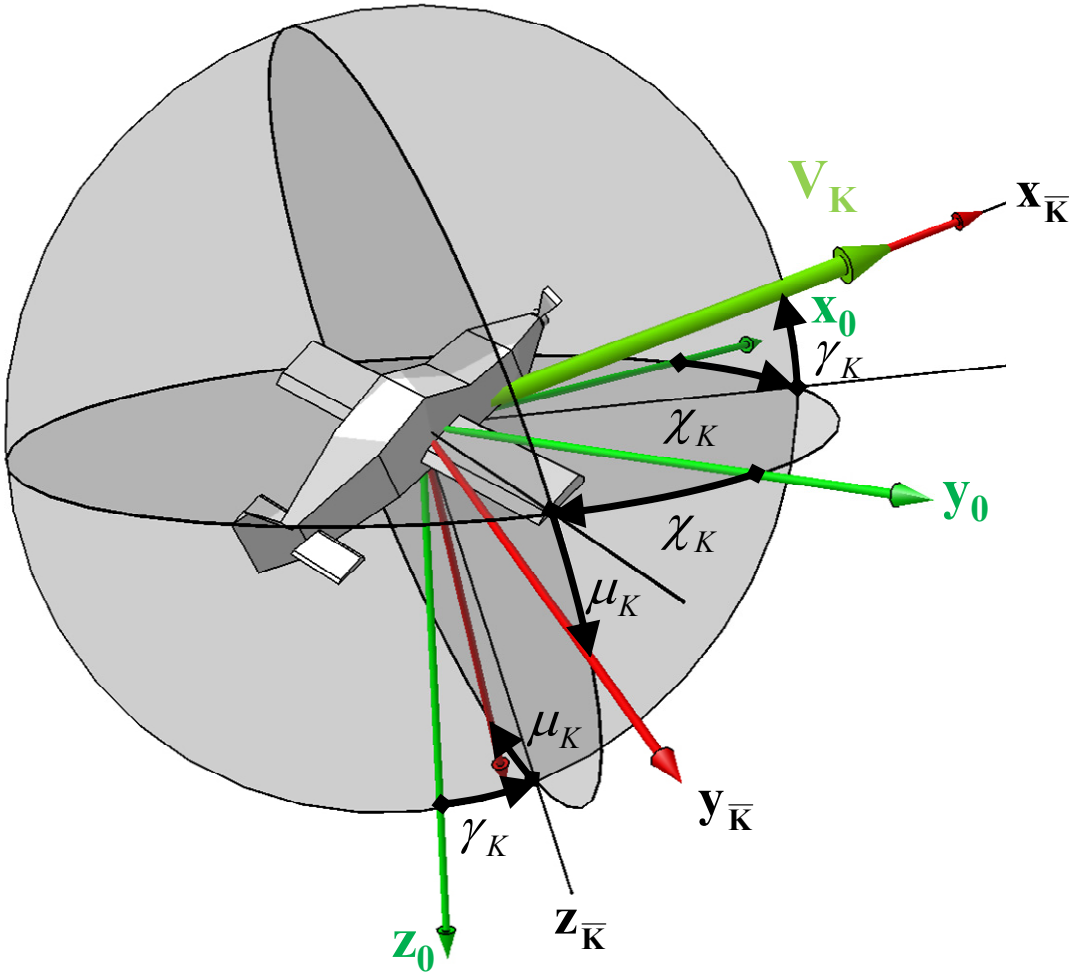

Figure 79. Intermediate Kinematic-Frame $\bar{K}$

A.2.7 Aerodynamic Frame *A*

Index:	*A*
Role:	Notation frame for aerodynamic flow
Origin:	Aerodynamic reference point of aircraft
Translation:	Moves with aircraft reference point
Rotation:	Rotates with direction of airflow

x-axis:	Aligned with aerodynamic velocity, pointing into the direction of the aerodynamic velocity
y-axis:	Pointing to right, perpendicular to the *x*- und *z*- axes
z-axis:	Pointing downwards in the symmetry plane of the aircraft, perpendicular to the *x*-axis

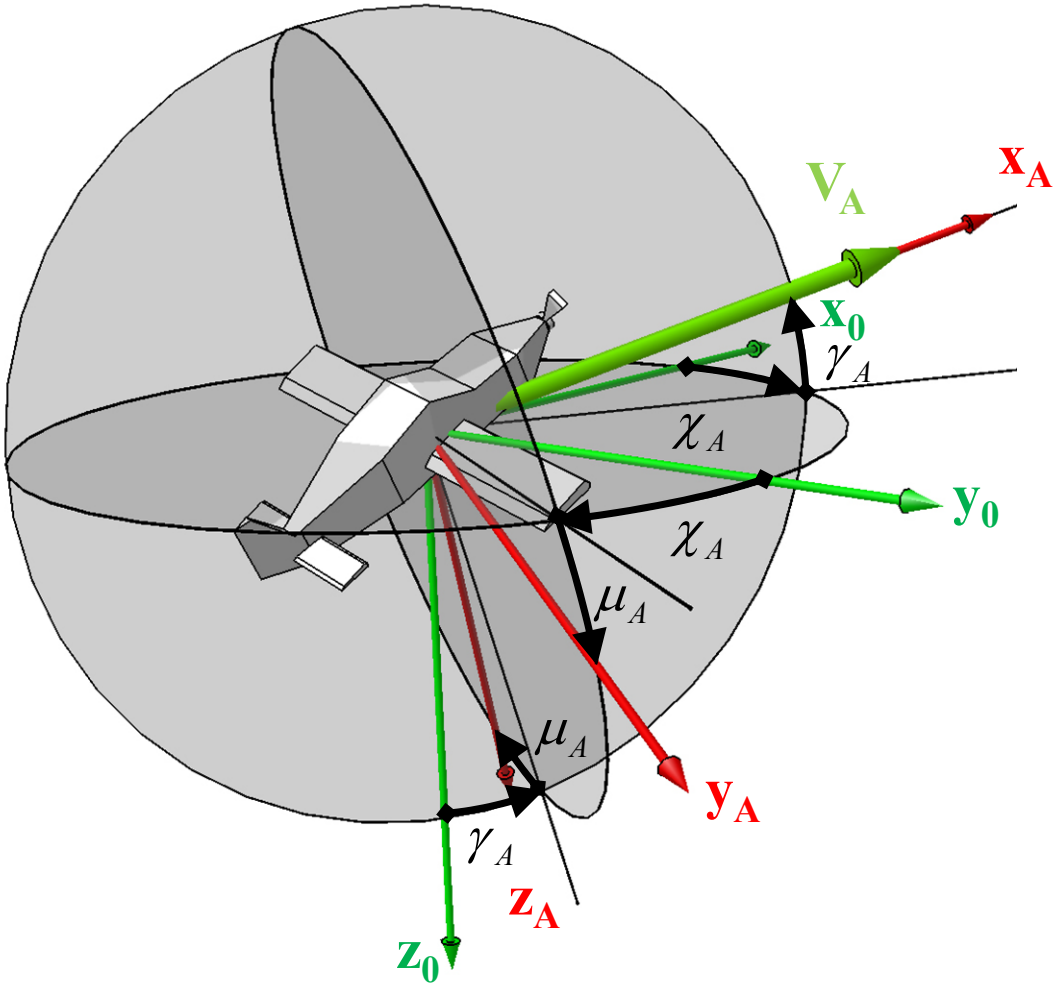

Figure 80. Aerodynamic Frame *A*

A.2.8 Intermediate Aerodynamic Frame $\bar{A}$

Index:	$\bar{A}$
Role:	Notation frame
Origin:	Aerodynamic reference point of aircraft
Translation:	Moves with aircraft reference point
Rotation:	Rotates with direction of airflow
x-axis:	Aligned with aerodynamic velocity, pointing into the direction of the aerodynamic velocity
y-axis:	Pointing to right, perpendicular to the *x*- und *z*- axes
z-axis:	*z*-axis of the Aerodynamic Frame *A* rotated counterclockwise by the aerodynamic flight-path bank angle μ_A

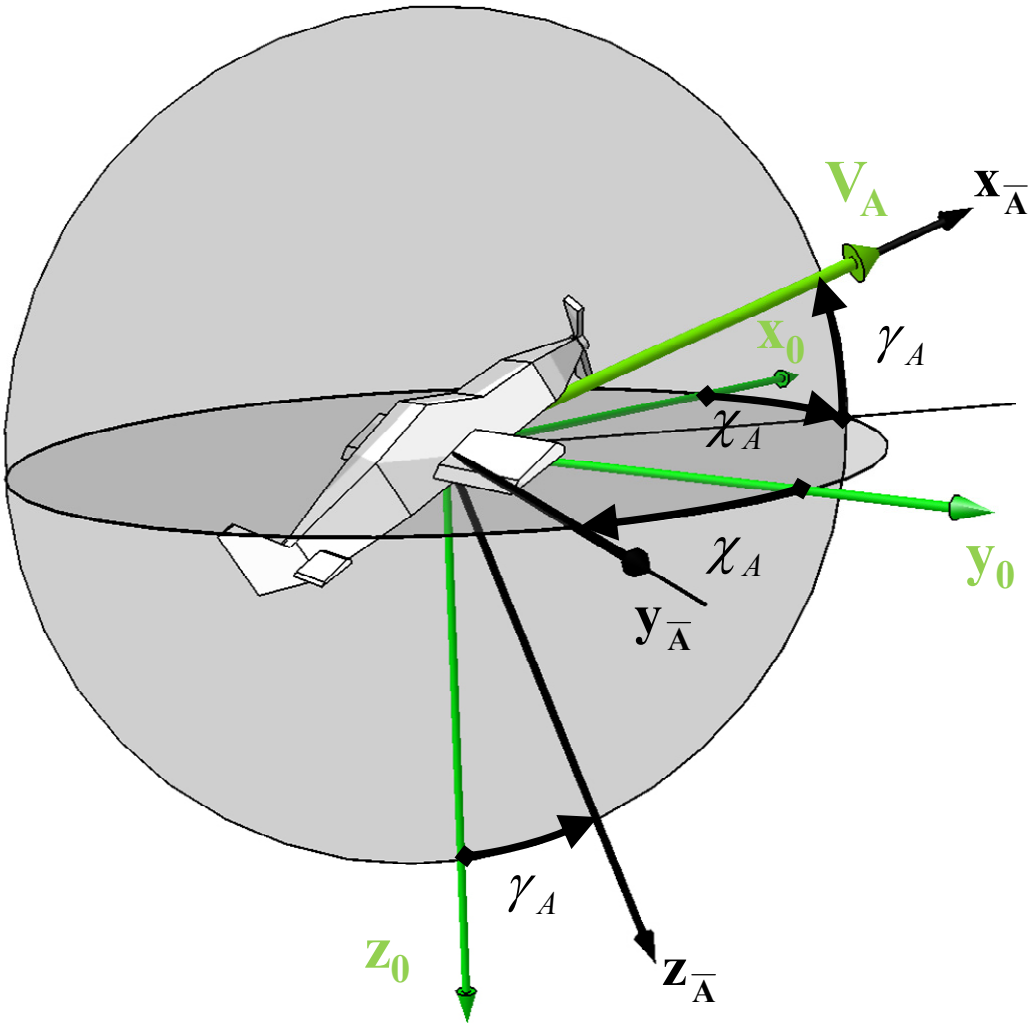

Figure 81. Intermediate Aerodynamic Frame $\bar{A}$

A.3 Transformation Matrices and Angular Rates

In the following chapters, the sequences of rotation, transformation matrices and angular velocities between the coordinate systems specified in the preceding chapter are given (Ref. [Holzapfel, 2009a].

A.3.1 *ECEF*-Frame *E* – *NED*-Frame *O*

Angles: Geodetic Longitude λ

Geodetic Latitude μ

Sequence of rotation: $\lambda \rightarrow \mu$

Transformation matrix:

$$\mathbf{M}_{OE} = \begin{bmatrix} -\sin\mu\cos\lambda & -\sin\mu\sin\lambda & \cos\mu \\ -\sin\lambda & \cos\lambda & 0 \\ -\cos\mu\cos\lambda & -\cos\mu\sin\lambda & -\sin\mu \end{bmatrix} \tag{A.2}$$

Angular velocity:

$$\left(\bar{\boldsymbol{\omega}}^{E0}\right)_E = \begin{bmatrix} \dot{\mu}\cdot\sin\lambda \\ -\dot{\mu}\cdot\cos\lambda \\ \dot{\lambda} \end{bmatrix}_E \tag{A.3}$$

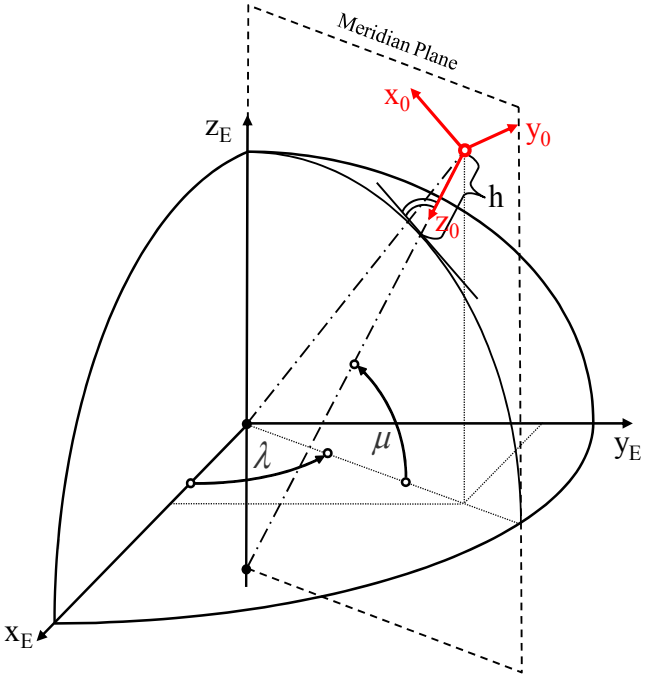

Figure 82. *ECEF*-Frame *E* – *NED*-Frame *O*

A.3.2 *NED*-Frame *O* – Body-Fixed Frame *B*

Angles: Azimuth Angle Ψ
Inclination Angle Θ
Bank Angle Φ

Sequence of rotation: $\Psi \rightarrow \Theta \rightarrow \Phi$

Transformation matrix:

$$\mathbf{M}_{BO} = \begin{bmatrix} \cos\Psi\cdot\cos\Theta & \sin\Psi\cdot\cos\Theta & -\sin\Theta \\ \cos\Psi\cdot\sin\Theta\cdot\sin\Phi-\sin\Psi\cdot\cos\Phi & \sin\Psi\cdot\sin\Theta\cdot\sin\Phi+\cos\Psi\cdot\cos\Phi & \cos\Theta\cdot\sin\Phi \\ \cos\Psi\cdot\sin\Theta\cdot\cos\Phi+\sin\Psi\cdot\sin\Phi & \sin\Psi\cdot\sin\Theta\cdot\cos\Phi-\cos\Psi\cdot\sin\Phi & \cos\Theta\cdot\cos\Phi \end{bmatrix} \tag{A.4}$$

Angular velocity:

$$\left(\vec{\omega}^{OB}\right)_O = \begin{bmatrix} \dot{\Phi}\cdot\cos\Theta\cdot\cos\Psi-\dot{\Theta}\cdot\sin\Psi \\ \dot{\Phi}\cdot\cos\Theta\cdot\sin\Psi+\dot{\Theta}\cdot\cos\Psi \\ \dot{\Psi}-\dot{\Phi}\cdot\sin\Theta \end{bmatrix}_O \tag{A.5}$$

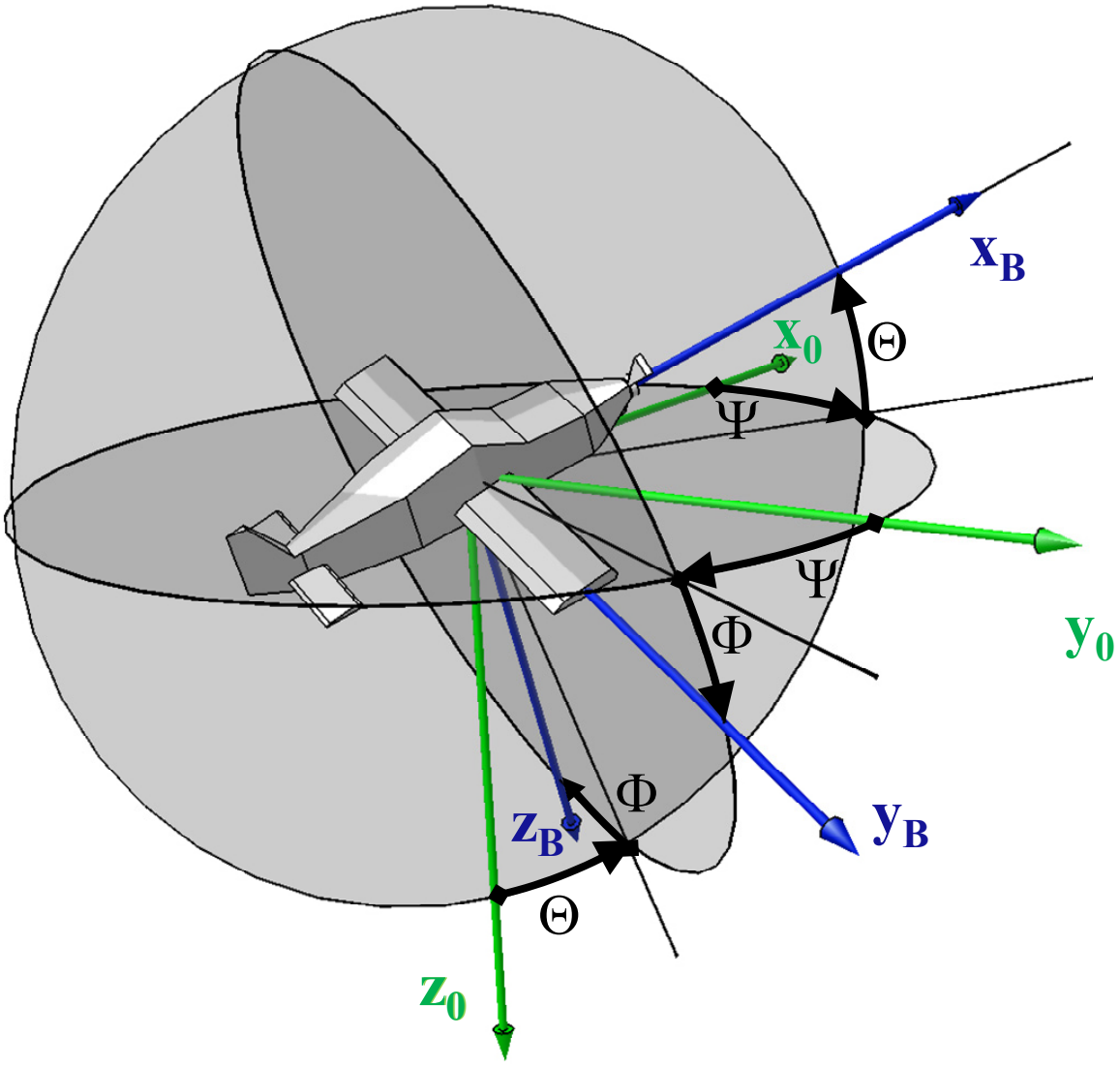

Figure 83. *NED*-Frame *O* – Body-Fixed Frame *B*

A.3.3 *NED*-Frame *O* – Kinematic Flight-Path Frame *K*

Angles: Flight-Path Azimuth Angle χ_K

Flight-Path Inclination Angle γ_K

Sequence of rotation: $\chi_K \rightarrow \gamma_K$

Transformation matrix:

$$\mathbf{M}_{KO} = \begin{bmatrix} \cos\chi_K \cdot \cos\gamma_K & \sin\chi_K \cdot \cos\gamma_K & -\sin\gamma_K \\ -\sin\chi_K & \cos\chi_K & 0 \\ \cos\chi_K \cdot \sin\gamma_K & \sin\chi_K \cdot \sin\gamma_K & \cos\gamma_K \end{bmatrix} \tag{A.6}$$

Angular velocity:

$$\left(\bar{\boldsymbol{\omega}}^{OK}\right)_O = \begin{bmatrix} -\dot{\gamma}_K \cdot \sin\chi_K \\ \dot{\gamma}_K \cdot \cos\chi_K \\ \dot{\chi}_K \end{bmatrix}_O \tag{A.7}$$

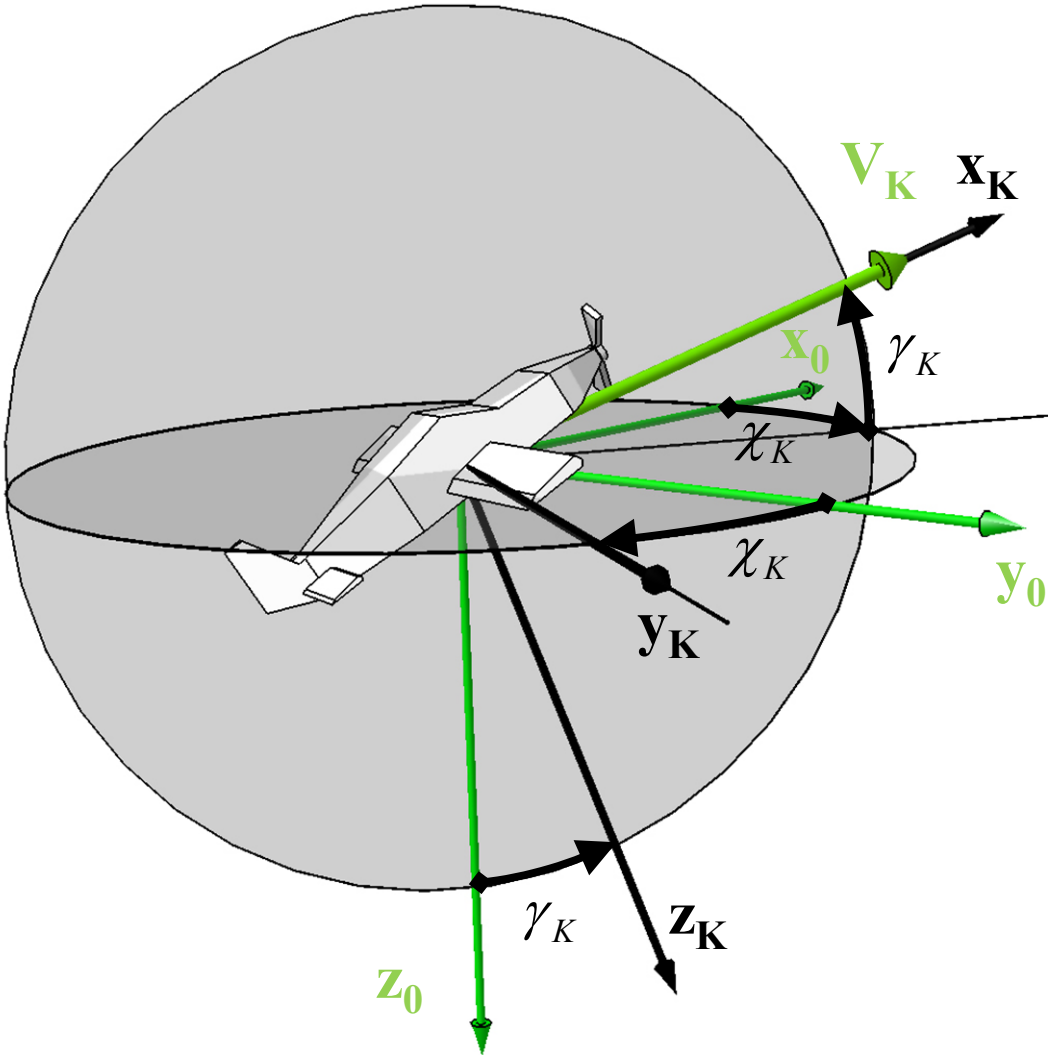

Figure 84. *NED*-Frame *O* – Kinematic Flight-Path Frame *K*

A.3.4 *NED*-Frame *O* – Intermediate Kinematic Flight-Path Frame $\bar{K}$

Angles: Kinematic Flight-Path Azimuth Angle χ_K
Kinematic Flight-Path Inclination Angle γ_K
Kinematic Flight-Path Bank Angle μ_K

Sequence of rotation: $\chi_K \rightarrow \gamma_K \rightarrow \mu_K$

Transformation matrix:

$$\mathbf{M}_{\bar{K}O} = \begin{bmatrix} \cos\chi_K \cos\gamma_K & \sin\chi_K \cos\gamma_K & -\sin\gamma_K \\ \cos\chi_K \sin\gamma_K \sin\mu_K - \sin\chi_K \cos\mu_K & \sin\chi_K \sin\gamma_K \sin\mu_K + \cos\chi_K \cos\mu_K & \cos\gamma_K \sin\mu_K \\ \cos\chi_K \sin\gamma_K \cos\mu_K + \sin\chi_K \sin\mu_K & \sin\chi_K \sin\gamma_K \cos\mu_K - \cos\chi_K \sin\mu_K & \cos\gamma_K \cos\mu_K \end{bmatrix} \quad \text{(A.8)}$$

Angular velocity:

$$\left(\vec{\boldsymbol{\omega}}^{O\bar{K}}\right)_O = \begin{bmatrix} \dot{\mu}_K \cdot \cos\chi_K \cdot \cos\gamma_K - \dot{\gamma}_K \cdot \sin\chi_K \\ \dot{\mu}_K \cdot \sin\chi_K \cdot \cos\gamma_K + \dot{\gamma}_K \cdot \cos\chi_K \\ \dot{\chi}_K - \dot{\mu}_K \cdot \sin\gamma_K \end{bmatrix}_O \quad \text{(A.9)}$$

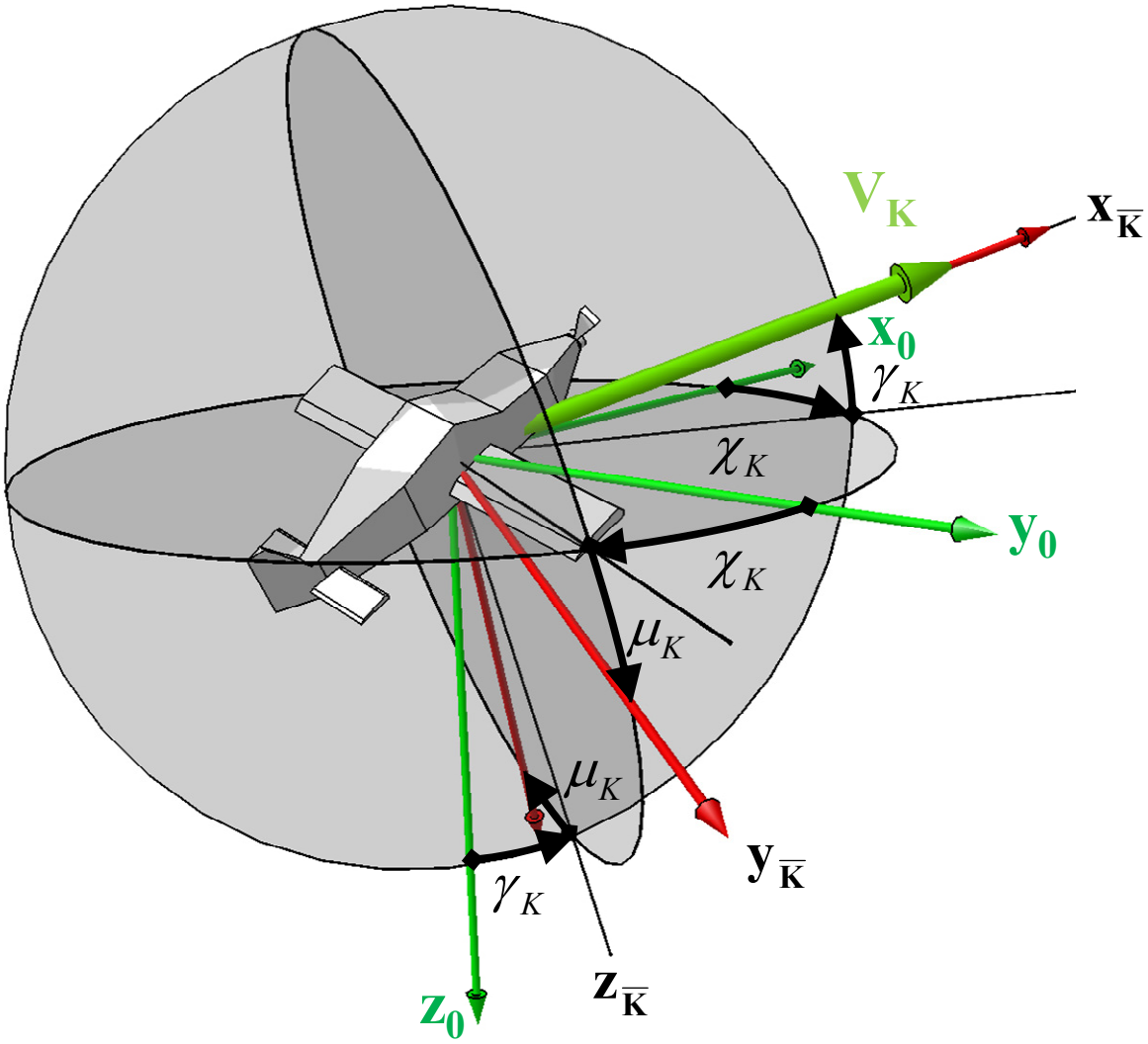

Figure 85. *NED*-Frame *O* – Intermediate Kinematic Flight-Path Frame $\bar{K}$

A.3.5 *NED*-Frame *O* – Aerodynamic Frame *A*

Angles: Aerodynamic Flight-Path Course Angle χ_A
Aerodynamic Flight-Path Inclination Angle γ_A
Aerodynamic Flight-Path Bank Angle μ_A

Sequence of rotation: $\chi_A \rightarrow \gamma_A \rightarrow \mu_A$

Transformation matrix:

$$\mathbf{M}_{AO} = \begin{bmatrix} \cos\chi_A \cos\gamma_A & \sin\chi_A \cos\gamma_A & -\sin\gamma_A \\ \cos\chi_A \sin\gamma_A \sin\mu_A - \sin\chi_A \cos\mu_A & \sin\chi_A \sin\gamma_A \sin\mu_A + \cos\chi_A \cos\mu_A & \cos\gamma_A \sin\mu_A \\ \cos\chi_A \sin\gamma_A \cos\mu_A + \sin\chi_A \sin\mu_A & \sin\chi_A \sin\gamma_A \cos\mu_A - \cos\chi_A \sin\mu_A & \cos\gamma_A \cos\mu_A \end{bmatrix} \quad \text{(A.10)}$$

Angular velocity:

$$\left(\vec{\boldsymbol{\omega}}^{OA}\right)_O = \begin{bmatrix} \dot{\mu}_A \cdot \cos\chi_A \cdot \cos\gamma_A - \dot{\gamma}_A \cdot \sin\chi_A \\ \dot{\mu}_A \cdot \sin\chi_A \cdot \cos\gamma_A + \dot{\gamma}_A \cdot \cos\chi_A \\ \dot{\chi}_A - \dot{\mu}_A \cdot \sin\gamma_A \end{bmatrix}_O \quad \text{(A.11)}$$

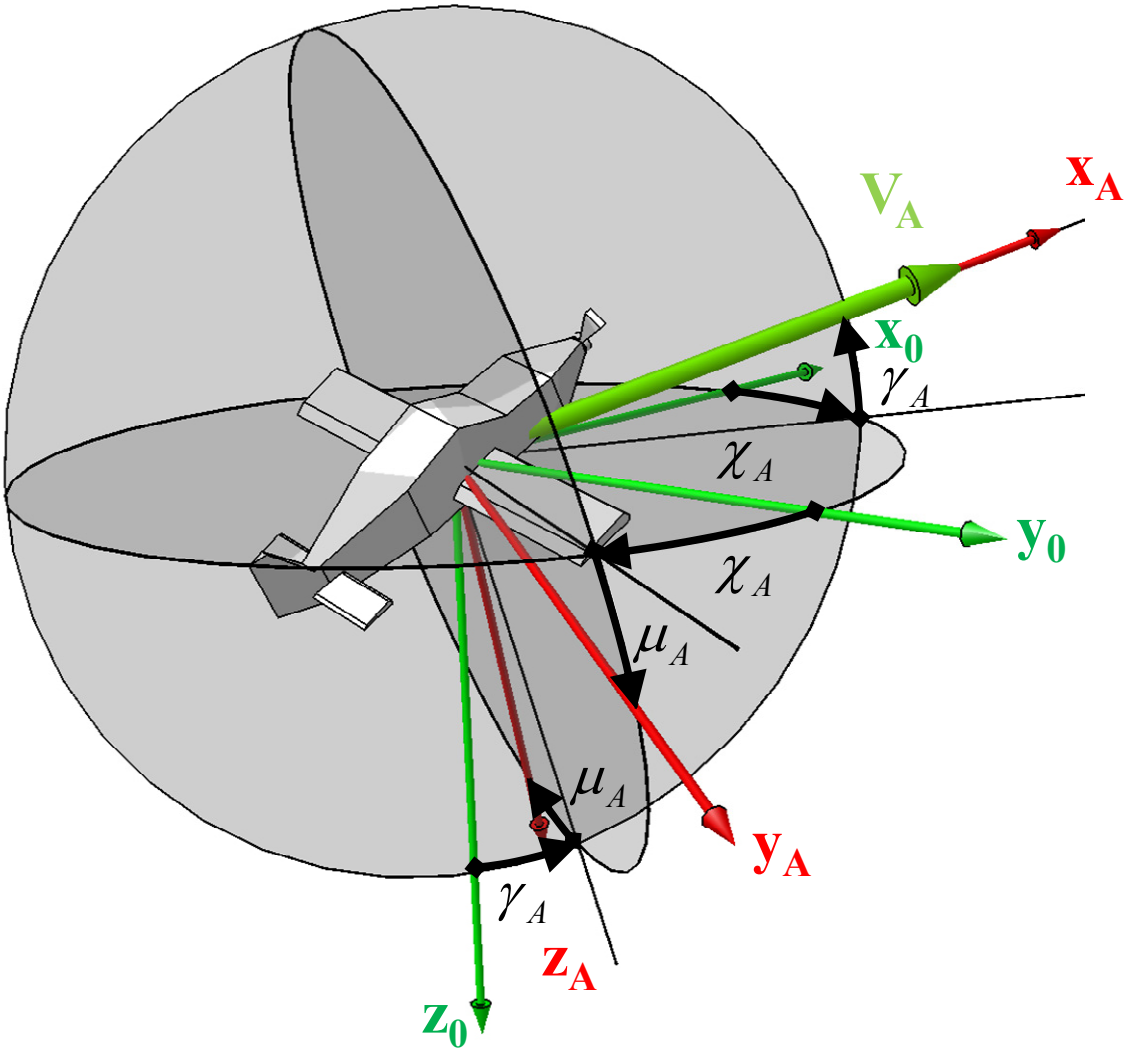

Figure 86. *NED*-Frame *O* – Aerodynamic Frame *A*

A.3.6 *NED*-Frame *O* – Intermediate Aerodynamic Frame $\bar{A}$

Angles: Aerodynamic Flight-Path Azimuth Angle χ_A
Aerodynamic Flight-Path Inclination Angle γ_A

Sequence of rotation: $\chi_A \rightarrow \gamma_A$

Transformation matrix:

$$\mathbf{M}_{\bar{A}O} = \begin{bmatrix} \cos\chi_A \cdot \cos\gamma_A & \sin\chi_A \cdot \cos\gamma_A & -\sin\gamma_A \\ -\sin\chi_A & \cos\chi_A & 0 \\ \cos\chi_A \cdot \sin\gamma_A & \sin\chi_A \cdot \sin\gamma_A & \cos\gamma_A \end{bmatrix} \tag{A.12}$$

Angular velocity:

$$\left(\bar{\boldsymbol{\omega}}^{O\bar{A}}\right)_O = \begin{bmatrix} -\dot{\gamma}_A \cdot \sin\chi_A \\ \dot{\gamma}_A \cdot \cos\chi_A \\ \dot{\chi}_A \end{bmatrix}_O \tag{A.13}$$

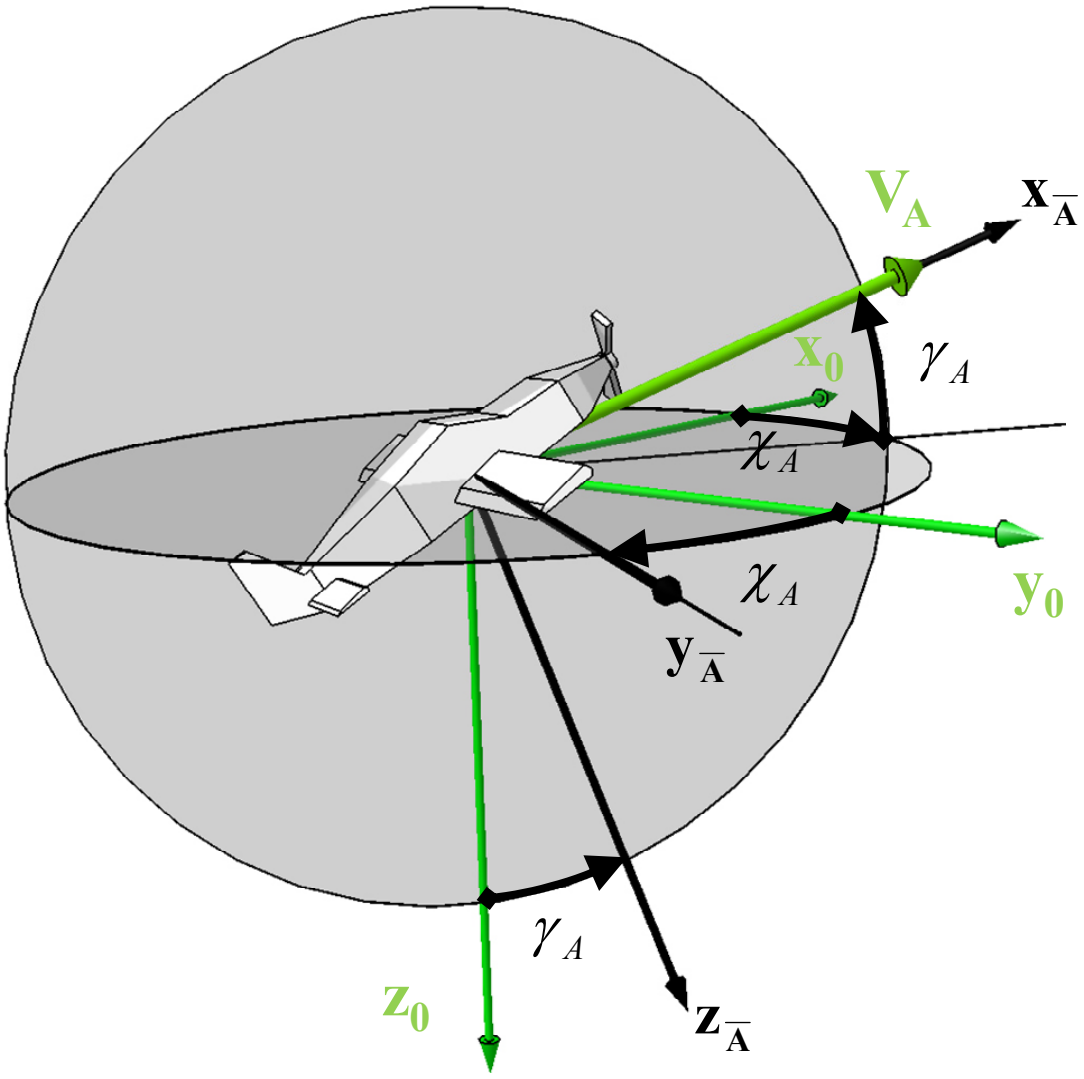

Figure 87. *NED*-Frame *O* – Intermediate Aerodynamic Frame $\bar{A}$

A.3.7 Kinematic Flight-Path Frame *K* – Body-Fixed Frame *B*

Angles: Kinematic Flight-Path Bank Angle μ_K
Kinematic Sideslip Angle β_K
Kinematic Angle of Attack α_K

Sequence of rotation: $\mu_K \rightarrow -\beta_K \rightarrow \alpha_K$

Transformation matrix:

$$\mathbf{M}_{KB} = \begin{bmatrix} \cos\alpha_K \cos\beta_K & \sin\beta_K & \sin\alpha_K \cos\beta_K \\ -\cos\alpha_K \sin\beta_K \cos\mu_K + \sin\alpha_K \sin\mu_K & \cos\beta_K \cos\mu_K & -\sin\alpha_K \sin\beta_K \cos\mu_K - \cos\alpha_K \sin\mu_K \\ -\cos\alpha_K \sin\beta_K \sin\mu_K - \sin\alpha_K \cos\mu_K & \cos\beta_K \sin\mu_K & -\sin\alpha_K \sin\beta_K \sin\mu_K + \cos\alpha_K \cos\mu_K \end{bmatrix} \tag{A.14}$$

Angular velocity:

$$\left(\vec{\omega}^{KB}\right)_B = \begin{bmatrix} \dot{\mu}_K \cdot \cos\alpha_K \cdot \cos\beta_K + \dot{\beta}_K \cdot \sin\alpha_K \\ \dot{\alpha}_K + \dot{\mu}_K \cdot \sin\beta_K \\ \dot{\mu}_K \cdot \sin\alpha_K \cdot \cos\beta_K - \dot{\beta}_K \cdot \cos\alpha_K \end{bmatrix}_B \tag{A.15}$$

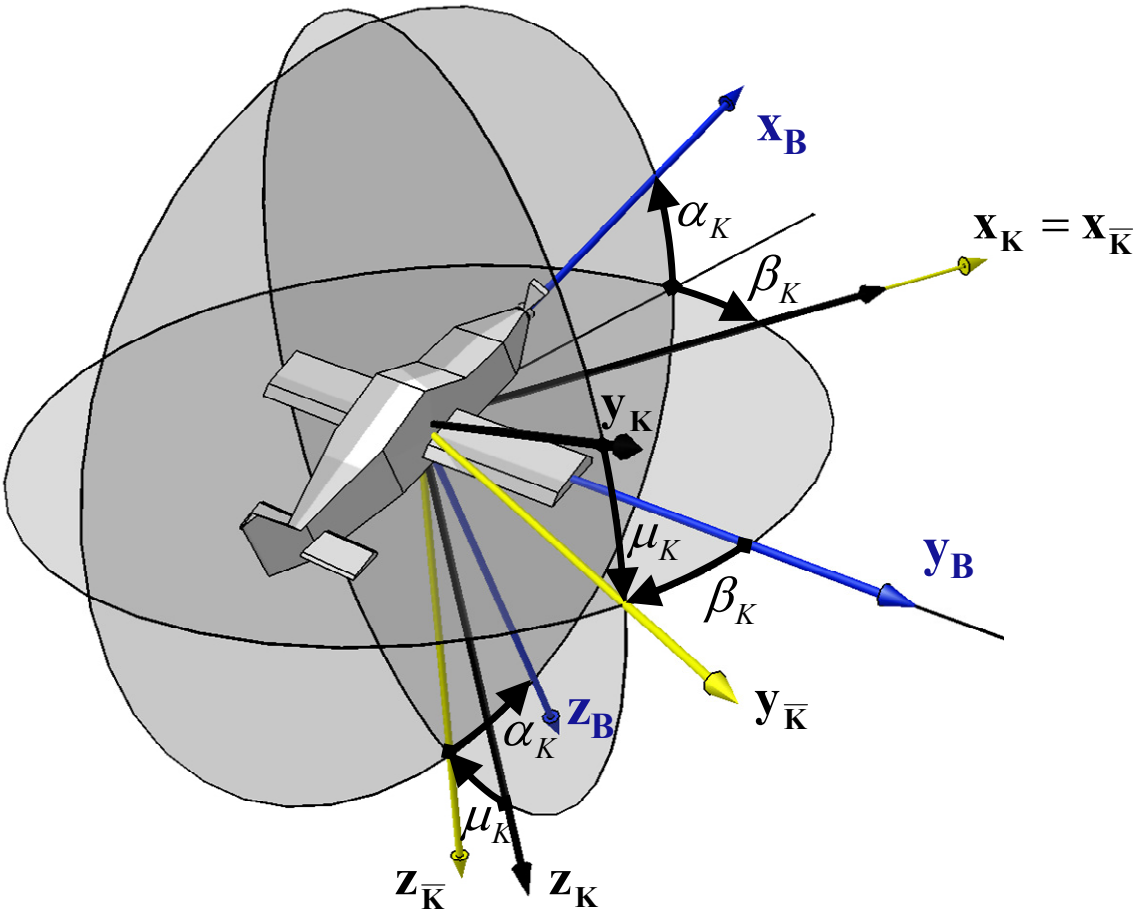

Figure 88. Kinematic Flight-Path Frame *K* – Body-Fixed Frame *B*

A.3.8 Intermediate Kinematic Flight-Path Frame $\bar{K}$ – Body-Fixed Frame *B*

Angles: Kinematic Sideslip Angle β_K

Kinematic Angle of Attack α_K

Sequence of rotation: $-\beta_K \rightarrow \alpha_K$

Transformation matrix:

$$\mathbf{M}_{\bar{K}B} = \begin{bmatrix} \cos\alpha_K \cos\beta_K & \sin\beta_K & \sin\alpha_K \cos\beta_K \\ -\cos\alpha_K \sin\beta_K & \cos\beta_K & -\sin\alpha_K \sin\beta_K \\ -\sin\alpha_K & 0 & \cos\alpha_K \end{bmatrix} \tag{A.16}$$

Angular velocity:

$$\left(\bar{\boldsymbol{\omega}}^{\bar{K}B}\right)_B = \begin{bmatrix} \dot{\beta}_K \cdot \sin\alpha_K \\ \dot{\alpha}_K \\ -\dot{\beta}_K \cdot \cos\alpha_K \end{bmatrix}_B \tag{A.17}$$

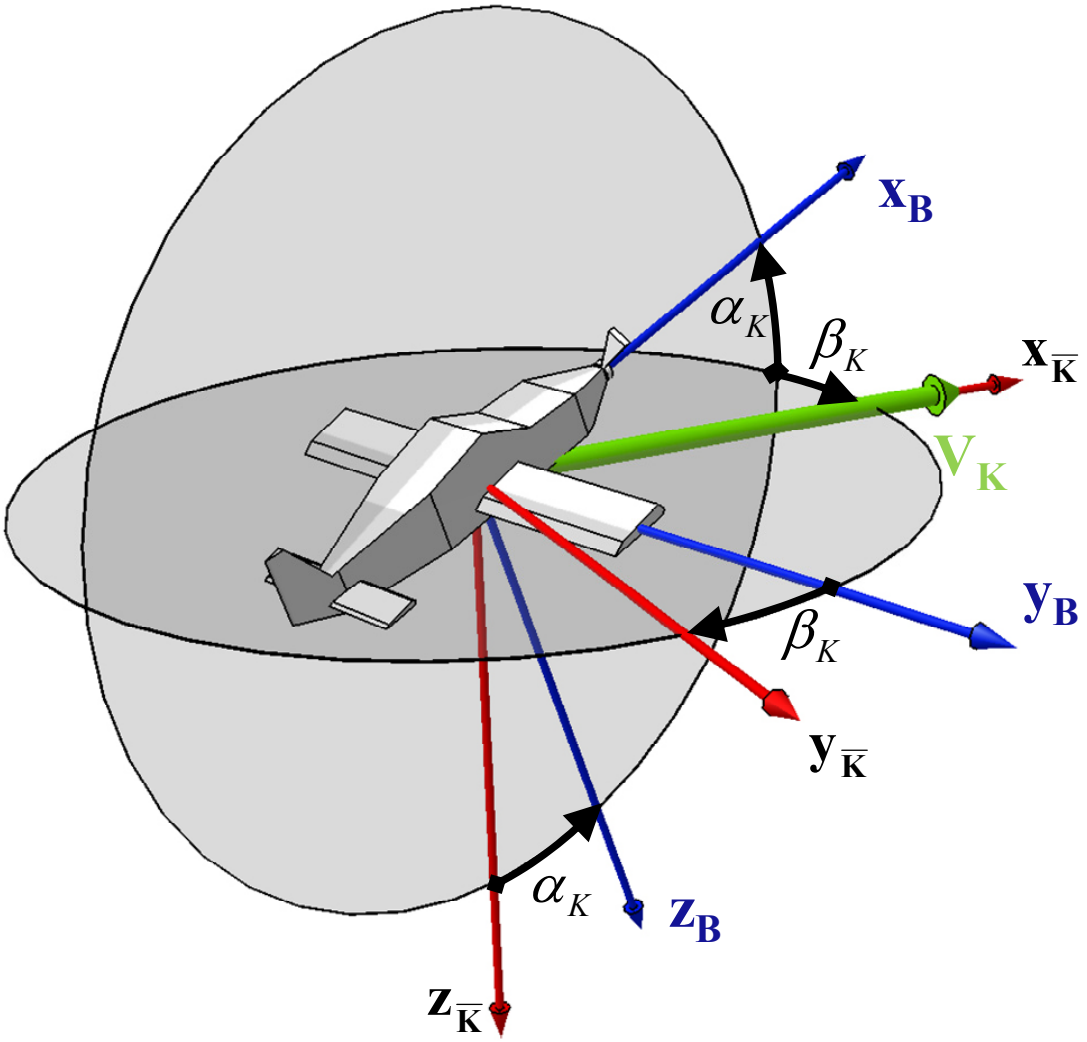

Figure 89. Intermediate Kinematic Flight-Path Frame $\bar{K}$ – Body-Fixed Frame *B*

A.3.9 Aerodynamic Frame *A* – Body-Fixed Frame *B*

Angles: Kinematic Sideslip Angle β_A

Kinematic Angle of Attack α_A

Sequence of rotation: $-\beta_A \rightarrow \alpha_A$

Transformation matrix:

$$\mathbf{M}_{AB} = \begin{bmatrix} \cos\alpha_A \cos\beta_A & \sin\beta_A & \sin\alpha_A \cos\beta_A \\ -\cos\alpha_A \sin\beta_A & \cos\beta_A & -\sin\alpha_A \sin\beta_A \\ -\sin\alpha_A & 0 & \cos\alpha_A \end{bmatrix} \tag{A.18}$$

Angular velocity:

$$\left(\bar{\boldsymbol{\omega}}^{AB}\right)_B = \begin{bmatrix} \dot{\beta}_A \cdot \sin\alpha_A \\ \dot{\alpha}_A \\ -\dot{\beta}_A \cdot \cos\alpha_A \end{bmatrix}_B \tag{A.19}$$

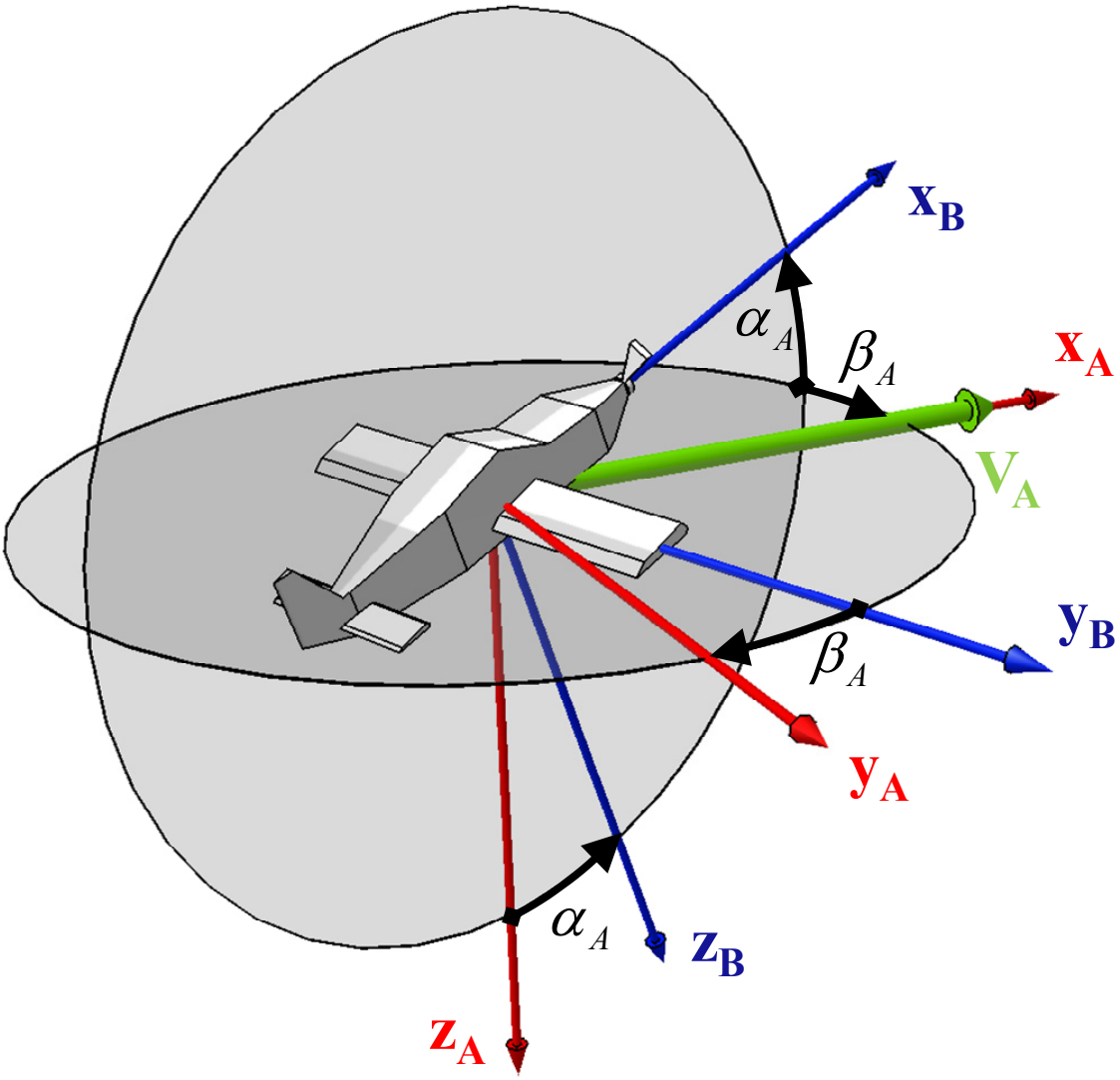

Figure 90. Aerodynamic Frame *A* – Body-Fixed Frame *B*

A.3.10 Intermediate Aerodynamic Frame $\bar{A}$ – Body-Fixed Frame B

Angles: Aerodynamic Flight-Path Bank Angle μ_A
Aerodynamic Sideslip Angle β_A
Aerodynamic Angle of Attack α_A

Sequence of rotation: $\mu_A \rightarrow -\beta_A \rightarrow \alpha_A$

Transformation matrix:

$$\mathbf{M}_{\bar{A}B} = \begin{bmatrix} \cos\alpha_A \cos\beta_A & \sin\beta_A & \sin\alpha_A \cos\beta_K \\ -\cos\alpha_A \sin\beta_A \cos\mu_A + \sin\alpha_A \sin\mu_A & \cos\beta_A \cos\mu_A & -\sin\alpha_A \sin\beta_A \cos\mu_A - \cos\alpha_A \sin\mu_A \\ -\cos\alpha_A \sin\beta_A \sin\mu_A - \sin\alpha_A \cos\mu_A & \cos\beta_A \sin\mu_A & -\sin\alpha_A \sin\beta_A \sin\mu_A + \cos\alpha_A \cos\mu_A \end{bmatrix} \quad \text{(A.20)}$$

Angular velocity:

$$\left(\bar{\boldsymbol{\omega}}^{\bar{A}B}\right)_B = \begin{bmatrix} \dot{\mu}_A \cdot \cos\alpha_A \cdot \cos\beta_A + \dot{\beta}_A \cdot \sin\alpha_A \\ \dot{\alpha}_A + \dot{\mu}_A \cdot \sin\beta_A \\ \dot{\mu}_A \cdot \sin\alpha_A \cdot \cos\beta_A - \dot{\beta}_A \cdot \cos\alpha_A \end{bmatrix}_B \quad \text{(A.21)}$$

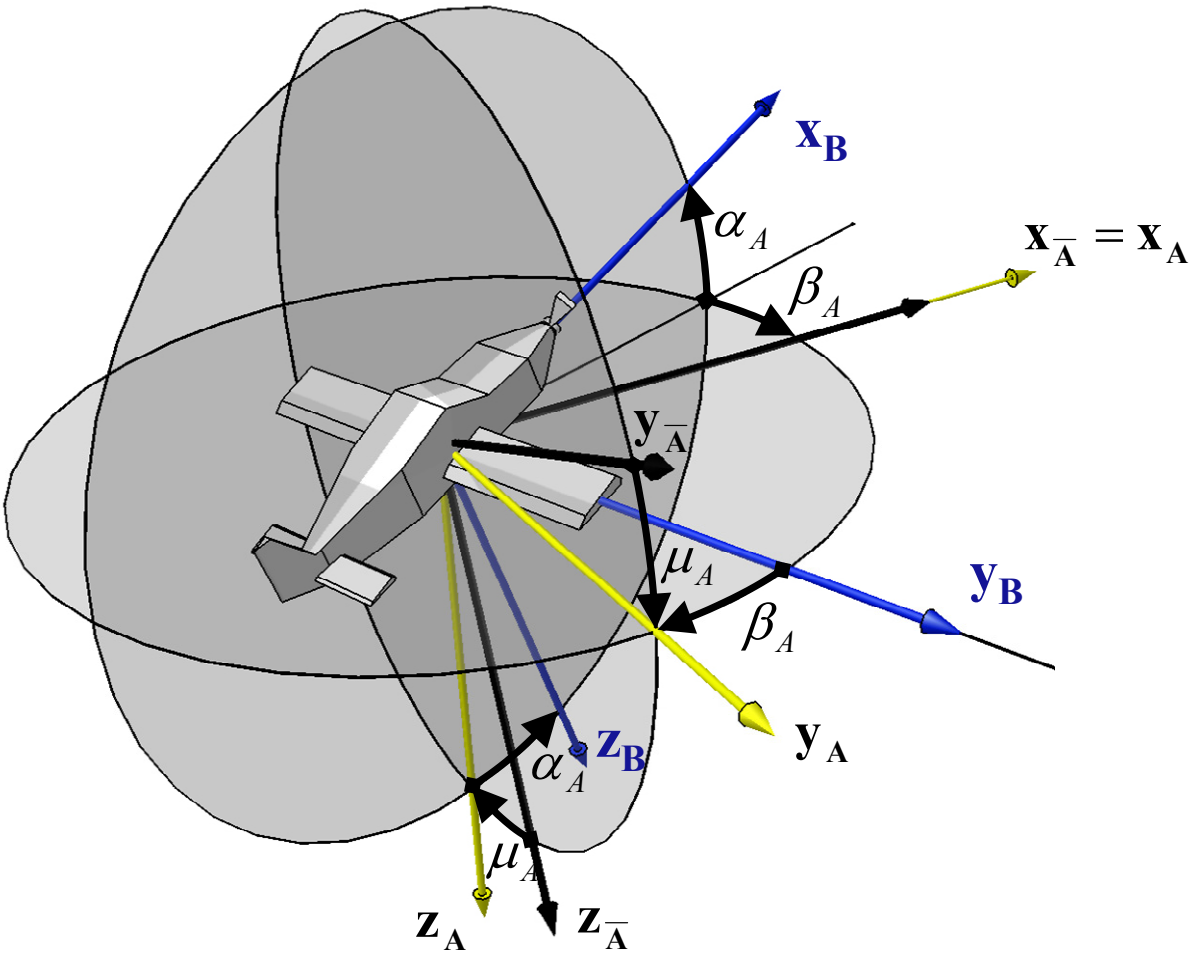

Figure 91. Intermediate Aerodynamic Frame $\bar{A}$ – Body-Fixed Frame B

A.4 Linearized State-Space Models

A.4.1 Longitudinal State-Space Models

For the longitudinal motion, the linearized state-space model is:

$$\begin{bmatrix} \dot{V} \\ \dot{\gamma} \\ \dot{\alpha} \\ \dot{q} \\ \dot{h} \\ \dot{x} \end{bmatrix} = \begin{bmatrix} X_V & X_\gamma & X_\alpha & X_q & 0 & 0 \\ -Z_V & Z_\gamma & -Z_\alpha & -Z_q & 0 & 0 \\ Z_V & -Z_\gamma & Z_\alpha & Z_q+1 & 0 & 0 \\ M_V & 0 & M_\alpha & M_q & 0 & 0 \\ \sin\gamma_0 & V_0\cos\gamma_0 & 0 & 0 & 0 & 0 \\ \cos\gamma_0 & -V_0\sin\gamma_0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} V \\ \gamma \\ \alpha \\ q \\ h \\ x \end{bmatrix} + \begin{bmatrix} X_\eta & X_{\delta_T} \\ -Z_\eta & -Z_{\delta_T} \\ Z_\eta & Z_{\delta_T} \\ M_\eta & M_{\delta_T} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \eta \\ \delta_T \end{bmatrix} \tag{A.22}$$

with the corresponding force and moment coefficients:

$$X_V \approx -\frac{\bar{q}_0 \cdot S}{m V_0} \cdot \left[M_0 \cdot \left.\frac{\partial C_D}{\partial M}\right|_0 + 2C_D\big|_0 \right] \tag{A.23}$$

$$X_\gamma = -g\cos\gamma_0 \tag{A.24}$$

$$X_\alpha \approx \frac{\bar{q}_0 \cdot S}{m} \cdot \left[C_L\big|_0 - C_{D\alpha} \right] \tag{A.25}$$

$$X_q = -\frac{\bar{q}_0 \cdot S}{m} \cdot \frac{\bar{c}}{2V_0} \cdot C_{Dq} \tag{A.26}$$

$$X_\eta = -\frac{\bar{q}_0 \cdot S}{m} \cdot C_{D\eta} \tag{A.27}$$

$$X_{\delta_T} = \frac{1}{m} \cdot \left[\left.\frac{\partial (X_P)_B}{\partial \delta_T}\right|_0 \cos\alpha_0 + \left.\frac{\partial (Z_P)_B}{\partial \delta_T}\right|_0 \sin\alpha_0 \right] \tag{A.28}$$

$$Z_V \approx -\frac{\bar{q}_0 \cdot S}{m V_0^2} \cdot \left[M_0 \cdot \left.\frac{\partial C_L}{\partial M}\right|_0 + 2C_L\big|_0 \right] \tag{A.29}$$

$$Z_\gamma = \frac{g}{V_0}\sin\gamma_0 \tag{A.30}$$

$$Z_\alpha \approx -\frac{\bar{q}_0 \cdot S}{m V_0} \cdot \left[C_{L\alpha} + C_D\big|_0 \right] \tag{A.31}$$

$$Z_q = -\frac{\bar{q}_0 \cdot S}{m V_0} \cdot \frac{\bar{c}}{2V_0} \cdot C_{Lq} \tag{A.32}$$

$$Z_\eta = -\frac{\bar{q}_0 \cdot S}{m V_0} \cdot C_{L\eta} \tag{A.33}$$

$$Z_{\delta_T} = -\frac{1}{m V_0} \cdot \left[\left.\frac{\partial (X_P)_B}{\partial \delta_T}\right|_0 \sin\alpha_0 - \left.\frac{\partial (Z_P)_B}{\partial \delta_T}\right|_0 \cos\alpha_0 \right] \tag{A.34}$$

$$M_V \approx \frac{1}{I_{yy}} \cdot \frac{\bar{q}_0 \cdot S \cdot \bar{c}}{V_0} \cdot M_0 \cdot \left.\frac{\partial C_m}{\partial M}\right|_0 \tag{A.35}$$

$$M_\alpha \approx \frac{1}{I_{yy}} \cdot \bar{q}_0 \cdot S \cdot \bar{c} \cdot C_{m\alpha} \tag{A.36}$$

$$M_q = \frac{1}{I_{yy}} \cdot \bar{q}_0 \cdot S \cdot \bar{c} \cdot \frac{\bar{c}}{2V_0} \cdot C_{mq} \tag{A.37}$$

$$M_\eta = \frac{1}{I_{yy}} \cdot \bar{q}_0 \cdot S \cdot \bar{c} \cdot C_{m\eta} \tag{A.38}$$

$$M_{\delta_T} = \frac{1}{I_{yy}} \cdot \left.\frac{\partial M_P^G}{\partial \delta_T}\right|_0 \tag{A.39}$$

Omitting the decoupled differential equations for the northward position x and the altitude h, the longitudinal state-space model simplifies to:

$$\begin{bmatrix} \dot{V} \\ \dot{\gamma} \\ \dot{\alpha} \\ \dot{q} \end{bmatrix} = \begin{bmatrix} X_V & X_\gamma & X_\alpha & X_q \\ -Z_V & Z_\gamma & -Z_\alpha & -Z_q \\ Z_V & -Z_\gamma & Z_\alpha & Z_q + 1 \\ M_V & 0 & M_\alpha & M_q \end{bmatrix} \cdot \begin{bmatrix} V \\ \gamma \\ \alpha \\ q \end{bmatrix} + \begin{bmatrix} X_\eta & X_{\delta_T} \\ -Z_\eta & -Z_{\delta_T} \\ Z_\eta & Z_{\delta_T} \\ M_\eta & M_{\delta_T} \end{bmatrix} \cdot \begin{bmatrix} \eta \\ \delta_T \end{bmatrix} \tag{A.40}$$

A.4.2 Lateral State-Space Models

With the assumptions that the angle of attack α_0 and the pitch angle Θ_0 are very small (i.e. $\alpha_0 \approx 0$ and $\Theta_0 \approx 0$), the linearized lateral state-space model is:

$$\begin{bmatrix} \dot{r} \\ \dot{\beta} \\ \dot{p} \\ \dot{\Phi} \\ \dot{\psi} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} N_r & N_\beta & N_p & 0 & 0 & 0 \\ -1 & Y_\beta & 0 & g/V_0 & 0 & 0 \\ L_r & L_\beta & L_p & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & a & 0 & 0 & a & 0 \end{bmatrix} \cdot \begin{bmatrix} r \\ \beta \\ p \\ \Phi \\ \psi \\ y \end{bmatrix} + \begin{bmatrix} N_\xi & N_\zeta \\ Y_\xi & Y_\zeta \\ L_\xi & L_\zeta \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \xi \\ \zeta \end{bmatrix} \tag{A.41}$$

where

$$a = V_0 \cdot \cos\psi_0 \cdot \cos\gamma_0 \tag{A.42}$$

Furthermore, it is assumed that Y_r and Y_p equal zero. The remaining force and moment coefficients are:

$$N_r = \frac{\bar{q}_0 \cdot S \cdot b}{2\Delta} \cdot \frac{b}{2V_0} \cdot \left[I_{xz} C_{lr} + I_{xx} C_{nr}\right] \tag{A.43}$$

$$N_\beta \approx \frac{\bar{q}_0 \cdot S \cdot b}{2\Delta} \cdot \left[I_{xz} C_{l\beta} + I_{xx} C_{n\beta}\right] \tag{A.44}$$

$$N_p = \frac{\overline{q}_0 \cdot S \cdot b}{2\Delta} \cdot \frac{b}{2V_0} \cdot \left[I_{xz} C_{lp} + I_{xx} C_{np} \right] \tag{A.45}$$

$$N_\xi = \frac{\overline{q}_0 \cdot S \cdot b}{2\Delta} \cdot \left[I_{xz} C_{l\xi} + I_{xx} C_{n\xi} \right] \tag{A.46}$$

$$N_\zeta = \frac{\overline{q}_0 \cdot S \cdot b}{2\Delta} \cdot \left[I_{xz} C_{l\zeta} + I_{xx} C_{n\zeta} \right] \tag{A.47}$$

$$Y_\beta \approx \frac{\overline{q}_0 \cdot S}{m V_0} \cdot \left[C_{Q\beta} - C_D\big|_0 \right] \tag{A.48}$$

$$Y_\zeta = \frac{\overline{q}_0 \cdot S}{m V_0} \cdot C_{Q\zeta} \tag{A.49}$$

$$Y_\xi = \frac{\overline{q}_0 \cdot S}{m V_0} \cdot C_{Q\xi} \tag{A.50}$$

$$L_r = \frac{\overline{q}_0 \cdot S \cdot b}{2\Delta} \cdot \frac{b}{2V_0} \cdot \left[I_{zz} C_{lr} + I_{xz} C_{nr} \right] \tag{A.51}$$

$$L_\beta \approx \frac{\overline{q}_0 \cdot S \cdot b}{2\Delta} \cdot \left[I_{zz} C_{l\beta} + I_{xz} C_{n\beta} \right] \tag{A.52}$$

$$L_p = \frac{\overline{q}_0 \cdot S \cdot b}{2\Delta} \cdot \frac{b}{2V_0} \cdot \left[I_{zz} C_{lp} + I_{xz} C_{np} \right] \tag{A.53}$$

$$L_\xi = \frac{\overline{q}_0 \cdot S \cdot b}{2\Delta} \cdot \left[I_{zz} C_{l\xi} + I_{xz} C_{n\xi} \right] \tag{A.54}$$

$$L_\zeta = \frac{\overline{q}_0 \cdot S \cdot b}{2\Delta} \cdot \left[I_{zz} C_{l\zeta} + I_{xz} C_{n\zeta} \right] \tag{A.55}$$

Here,

$$\Delta = I_{xx} I_{zz} - I_{xz}^2 \tag{A.56}$$

Omitting the decoupled differential equations for the heading angle ψ and the eastward position y, the simplified lateral state-space model reads:

$$\begin{bmatrix} \dot{r} \\ \dot{\beta} \\ \dot{p} \\ \dot{\Phi} \end{bmatrix} = \begin{bmatrix} N_r & N_\beta & N_p & 0 \\ -1 & Y_\beta & 0 & g/V_0 \\ L_r & L_\beta & L_p & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} r \\ \beta \\ p \\ \Phi \end{bmatrix} + \begin{bmatrix} N_\xi & N_\zeta \\ Y_\xi & Y_\zeta \\ L_\xi & L_\zeta \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \xi \\ \zeta \end{bmatrix} \tag{A.57}$$

Bibliography

Anisi, D. A. (2006), "Adaptive Node Distribution for On-Line Trajectory Planning", *Congress of the International Council of the Aeronautical Sciences (ICAS)*, Hamburg, Germany.

Beltracchi, T. J. (1992), "Decomposition Approach to Solving the All-Up Trajectory Optimization Problem", *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 3, pp. 707-716.

Betts, J. T. (1998), "Survey of Numerical Methods for Trajectory Optimization", *Journal of Guidance, Control, and Dynamics*, Vol. 21, No. 2, pp. 193-207.

Betts, J. T. (2001), *Practical Methods for Optimal Control Using Nonlinear Programming*, Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania.

Betts, J. T. and Huffman, W. P. (1998), "Mesh Refinement in Direct Transcription Methods for Optimal Control", *Optimal Control Applications and Methods*, Vol. 19, pp. 1-19.

Blakelock, J. H. (1991), *Automatic Control of Aircraft and Missiles*, 2nd ed., John Wiley & Sons, New York.

Bliss, G. A. (1946), *Lectures on the Calculus of Variations*, University of Chicago Press, Chicago.

Boiffier, J.-L. (1998), *The Dynamics of Flight - The Equations*, John Wiley & Sons, New York.

Bollino, K. P. (2006), "High-Fidelity Real-Time Trajectory Optimization for Reusable Launch Vehicles", Ph.D. Thesis, Naval Postgraduate School, Monterey, California.

Bollino, K. P., Ross, I. M. and Doman, D. D. (2006), "Optimal Nonlinear Feedback Guidance for Reentry Vehicles", *AIAA Guidance, Navigation and Control Conference*, Keystone, Colorado, AIAA 2006-6074.

Bolza, O. (1909), *Vorlesungen über Variationsrechnung*, B.G. Teubner, Leipzig.

Braun, R. D. and Kroo, I. M. (1993), "Post-Optimality Analysis in Aerospace Vehicle Design", *AIAA Aircraft Design, Systems and Operations Meeting*, Monterey, California, AIAA 1993-3932.

Braun, R. D. and Kroo, I. M. (1996), "Development and Application of the Collaborative Optimization Architecture in a Multidisciplinary Design Environment", in *Multidisciplinary Design Optimization: State of the Art*, SIAM, pp. 98-116.

Braun, R. D., Moore, A. A. and Kroo, I. M. (1996), "Use of the Collaborative Optimization Architecture for Launch Vehicle Design", *6th NASA, and ISSMO, Symposium on Multidisciplinary Analysis and Optimization*, Bellevue, Washington, AIAA-1996-4018.

Braun, R. D., Moore, A. A. and Kroo, I. M. (1997), "Collaborative Approach to Launch Vehicle Design", *Journal of Spacecraft and Rockets*, Vol. 34, No. 4, pp. 478-486.

Brockhaus, R. (2001), *Flugregelung*, 2nd ed., Springer-Verlag, Berlin.

Brown, N. (2004), "Evaluation of Multidisciplinary Design Optimization (MDO) Techniques Applied to a Reusable Launch Vehicle", Georgia Institute of Technology, AE 8900 Special Project Report.

Brown, N. F. and Olds, J. R. (2006), "Evaluation of Multidisciplinary Optimization Techniques Applied to a Reusable Launch Vehicle", *Journal of Spacecraft and Rockets*, Vol. 43, No. 6, pp. 1289-1300.

Bryson, A. E. (1998), *Dynamic Optimization*, 1st ed., Pearson Education, London.

Büskens, C. (1998), "Optimierungsmethoden und Sensitivitätsanalyse für optimale Steuerprozesse mit Steuer- und Zustands-Beschränkungen", Ph.D. Thesis, Westfälische Wilhelms-Universität, Münster.

Büskens, C. and Maurer, H. (2000), "SQP-Methods for Solving Optimal Control Problems with Control and State Constraints: Adjoint Variables, Sensitivity Analysis and Real-Time Control", *Journal of Computational and Applied Mathematics*, Vol. 120, pp. 85-108.

Bulirsch, R., Montrone, F. and Pesch, H. J. (1991), "Abort Landing in the Presence of Windshear as a Minimax Optimal Control Problem - Part 2: Multiple Shooting and Homotopy", *Journal of Optimization Theory and Applications*, Vol. 70, No. 2.

Bulirsch, R., Montrone, F. and Pesch, H. J. (1991), "Abort Landing in the Presence of Windshear as a Minimax Optimal Control Problem - Part 1: Necessary Conditions", *Journal of Optimization Theory and Applications*, Vol. 70, No. 1.

Calise, A. J., Lee, S. and Sharma, M. (2000), "Development of a Reconfigurable Flight Control Law for the X-36 Tailless Fighter Aircraft", *AIAA Guidance, Navigation and Control Conference*, Denver, Colorado, AIAA 2000-3940.

Callies, R. (2000), "Entwurfsoptimierung und optimale Steuerung. Differential-algebraische Systeme, Mehrgitter-Mehrzielansätze und numerische Realisierung", Habilitation Thesis, Fakultät für Mathematik, Technische Universität München, Garching.

Callies, R. and Wimmer, G. (2000), "Optimal Hypersonic Flight Trajectories with Full Safety in Case of Mission Abort", *AIAA Atmospheric Flight Mechanics Conference*, Denver, Colorado, AIAA 2000-3996.

Ciarcià, M. and Grillo, C. (2009), "Collision Avoidance Trajectory for an Ekranoplane", *AIAA Atmospheric Flight Mechanics Conference*, Chicago, Illinois, AIAA 2009-5931.

Colson, B., Marcotte, P. and Savard, G. (2007), "An Overview of Bilevel Optimization", *Annals of Operations Research*, Vol. 153, No. 1, pp. 235-256.

Corban, J. E., Twigg, S., Ries, T., Yang, B.-J., Johnson, E. and Calise, A. (2007), "On-Line Trajectory Optimization for Autonomous Air Vehicles", Guided Systems Techologies, Inc., GST-06-1-F.

Cox, M. G. (1972), "The Numerical Evaluation of B-Splines", *Journal of the Institute of Mathematics and its Applications*, Vol. 10, pp. 134-149.

Cremaschi, F., Weikert, S., Wiegand, A., Jung, W. and Scheuerpflug, F. (2009), "Sounding Rocket Trajectory Simulation and Optimization with ASTOS", *19th ESA Symposium on European Rocket and Balloon Programs and Related Research*, Bad Reichenhall, Germany.

Cuthrell, J. E. and Biegler, L. T. (1987), "On the Optimization of Differential-Algebraic Process Systems", *American Institute of Chemical Engineers Journal*, Vol. 33, No. 8, pp. 1257-1270.

Darby, C. L. and Rao, A. V. (2009), "A State Approximation-Based Mesh Refinement Algorithm for Solving Optimal Control Problems Using Pseudospectral Methods", *AIAA Guidance, Navigation and Control Conference*, Chicago, Illinois, AIAA 2009-5791.

de Boor, C. (1972), "On Calculating with B-Splines", *Journal of Approximation Theory*, Vol. 6, pp. 50-62.

Dempe, S. (2003), "Annotated Bibliography on Bilevel Programming and Mathematical Programs with Equilibrium Constraints", *Optimization*, Vol. 52, No. 33, pp. 333-359.

Desai, P. N. (2005), "Improved Collocation Methods with Application to Six-Degree-of-Freedom Trajectory Optimization", Ph.D. Thesis, University of Illinois, Urbana, Illinois.

Desai, P. N. and Conway, B. A. (2008), "Six-Degree-of-Freedom Trajectory Optimization Using a Two-Timescale Collocation Architecture", *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 5, pp. 1308-1315.

Dierckx, P. (1993), *Curve and Surface Fitting with Splines*, Clarendon Press, Oxford.

Duffie, J. A. and Beckmann, W. A. (1980), *Solar Engineering of Thermal Processes*, John Wiley & Sons.

Ehtamo, H. and Raivio, T. (2001), "On Applied Nonlinear and Bilevel Programming for Pursuit-Evasion Games", *Journal of Optimization Theory and Applications*, Vol. 108, No. 1, pp. 65-96.

Etkin, B. (2005), *Dynamics of Atmospheric Flight*, Dover Publications, New York.

Etkin, B. and Reid, L. D. (1996), *Dynamics of Flight: Stability and Control*, 3rd ed., John Wiley & Sons, New York.

Falk, J. E. and Liu, J. (1995), "On bilevel programming, Part I: general nonlinear cases", *Mathematical Programming*, Vol. 70, pp. 47-72.

Fiacco, A. V. (1976), "Sensitivity Analysis for Nonlinear Programming using Penalty Methods", *Mathematical Programming*, Vol. 10, pp. 287-311.

Fiacco, A. V. (1983), *Introduction to Sensitivity and Stability Analysis in Nonlinear Programming*, Mathematics in Science and Engineering, Academic Press, New York.

Gerdts, M. (2007), "Optimal Control of Ordinary Differential Equations and Differential-Algebraic Equations", Habilitation Thesis, Department of Mathematics, University of Bayreuth, Bayreuth.

Gill, P. E., Murray, W. and Saunders, M. A. (2007), *User's Guide for SNOPT Version 7: Software for Large-Scale Nonlinear Programming*, University of California, Department of Mathematics, San Diego, CA.

Gong, Q. and Ross, I. M. (2006), "Autonomous Pseudospectral Knotting Methods for Space Mission Optimization", *AAS/AIAA 16th Space Flight Mechanics Meeting*, Tampa, Florida.

Grimm, W. and Hans, M. (1990), "Time-Optimal Turn to a Heading: An Improved Analytic Solution", *Journal of Guidance, Control, and Dynamics*, Vol. 21, No. 6, pp. 940-947.

Hancock, G. J. (1995), *Introduction to the Flight Dynamics of Rigid Aeroplanes*, Prentice Hall, Upper Saddle River, New Jersey.

Hoffman, E. (1991), "On Minimum Time Six-Degree-of-Freedom Turning Maneuvers for a High-Alpha Fighter Aircraft", Ph.D. Thesis, Georgia Institute of Technology.

Holzapfel, F. (2009), *Flugsystemdynamik 2*, Lecture Notes, Lehrstuhl für Flugsystemdynamik, Technische Universität München, München.

Holzapfel, F. (2009), *Flugregelung I*, Lecture Notes, Lehrstuhl für Flugsystemdynamik, Technische Universität München, München.

Holzapfel, F. (2009), *Flugregelung II*, Lecture Notes, Lehrstuhl für Flugsystemdynamik, Technische Universität München, München.

Holzapfel, F. and Sachs, G. (2004), "Dynamic Inversion Based Control Concept with Application to an Unmanned Aerial Vehicle", *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Providence, Rhode Island, AIAA-2004-4907.

Huesman, A. (2003), "Short note: How to implement dynamic optimization in Matlab - The sequential and simultaneous approach", Centre for Systems and Controls, Delft University of Technology,.

Jain, S. (2008), "Multiresolution Strategies for the Numerical Solution of Optimal Control Problems", Ph.D. Thesis, Georgia Institute of Technology, Georgia.

Khalil, H. K. (2001), *Nonlinear Systems*, 3rd ed., Prentice Hall, Upper Saddle River, New Jersey.

Kirk, D. E. (2004), *Optimal Control Theory: An Introduction*, Dover Publications.

Kiusalaas, J. (2005), *Numerical Methods in Engineering with MATLAB*, Cambridge University Press, New York.

Knauer, M. (2009), "Bilevel-Optimalsteuerung mittels hybrider Lösungsmethoden am Beispiel eines deckengeführten Regalbediengerätes in einem Hochregallager", Ph.D. Thesis, Fachbereich 3 (Mathematik & Informatik), Universität Bremen, Bremen.

Lane, S. H. (1988), "Theory and Development of Adaptive Flight Control Systems using Nonlinear Inverse Dynamics", Ph.D. Thesis, Department of Mechanical and Aerospace Engineering, Princeton University, Princeton.

Ledsinger, L. A. (2000), "Solutions to Decomposed Trajectories with Powered Flyback Using Multidisciplinary Design Optimization", Ph.D. Thesis, Georgia Institute of Technology, Atlanta, Georgia.

Ledsinger, L. A. and Olds, J. R. (1998), "Multidisciplinary Design Optimization Techniques for Branching Trajectories", *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, St. Louis, Missouri, AIAA 1998-4713.

Ledsinger, L. A. and Olds, J. R. (2000), "Optimized Solutions for the Kistler K-1 Branching Trajectory Using MDO Techniques", *8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Long Beach, California, AIAA 2000-4884.

Magnus, J. R. and Neudecker, H. (1985), "Matrix Differential Calculus with Applications to Simple, Hadamard and Kronecker Products", *Journal of Mathematical Psychology*, Vol. 29, No. 4, pp. 474-492.

McCormick, B. W. (1994), *Aerodynamics, Aeronautics and Flight Mechanics*, 2nd ed., John Wiley & Sons, New York.

McRuer, D., Ashkenas, I. and Graham, D. (1990), *Aircraft Dynamics and Automatic Control*, Princeton University Press, Princeton, New Jersey.

Miele, A., Wang, T. and Melvin, W. W. (1986), "Optimal Take-Off Trajectories in the Presence of Windshear", *Journal of Optimization Theory and Applications*, Vol. 49, No. 1, pp. 1-45.

Mor, M. and Livne, E. (2006), "Multidisciplinary Design Optimization of Reentry Vehicles: Trajectory Optimization and Sensitivities", *47th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Newport, Rhode Island, 2006-1718.

Mor, M. and Livne, E. (2007), "Coupled Aeroelastic / Trajectory Optimization of Reentry Vehicles", *48th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Honolulu, Hawaii, AIAA 2007-1859.

N.N. (1975), *The Standard Atmosphere*, DIN ISO 2533, International Organization for Standardization, Genf, Schweiz.

N.N. (2000), "World Geodetic System 1984 - Its Definition and Relationships with Local Geodetic Systems", National Imagery and Mapping Agency, Department of Defense,, TR8350.2.

Nelson, R. (1997), *Flight Stability and Automatic Control*, 3rd ed., McGraw-Hill, New York.

Pamadi, B. N. (1998), *Performance, Stability, Dynamics and Control of Airplanes*, AIAA Educational Series, New York.

Perez, R. E., Liu, H. H. T. and Behdinan, K. (2004), "Evaluation of Multidisciplinary Optimization Approaches for Aircraft Conceptual Design", *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Albany, New York, AIAA 2004-4537.

Perez, R. E., Liu, H. H. T. and Behdinan, K. (2008), "Decomposition-Based Simultaneous Stabilization with Optimal Control", *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 3, pp. 647-655.

Petersen, F. M., Cornick, D. E., Bauer, G. L. and Rehder, J. R. (1977), "A Two-Level Trajectory Decomposition Algorithm Featuring Optimal Intermediate Target Selection", *Journal of Spacecraft and Rockets*, Vol. 14, No. 11, pp. 676-682.

Phillips, W. F. (2004), *Mechanics of Flight*, John Wiley & Sons, New York.

Pontryagin, L. S., Boltyanskii, V. G., Gamkrelidze, R. V. and Mishchenko, E. (1962), *The Mathematical Theory of Optimal Processes*, John Wiley & Sons, New York.

Pourtakdoust, S. H. and Kiani, M. (2009), "Analysis of Optimal Trajectory Planning for Flight through Microburst Wind Shears", *AIAA Atmospheric Flight Mechanics Conference*, Chicago, Illinois, AIAA 2009-5933.

Quarteroni, A., Sacco, R. and Saleri, F. (2000), *Numerical Mathematics*, Springer-Verlag, New York.

Rahn, M. (1998), "Eine numerische Methodik zur simultanen Flug- und Systemoptimierung von Raumtransportsystemen", Ph.D. Thesis, Institut für Raumfahrtsysteme, Universität Stuttgart, Stuttgart.

Rahn, M. and Schoettle, U. (1996), "Decomposition Algorithm for Performance Optimization of a Launch Vehicle", *Journal of Spacecraft and Rockets*, Vol. 33, No. 2, pp. 214-221.

Rahn, M., Schoettle, U. and Messerschmid, E. (1996), "Multidisciplinary Design Tool for System and Mission Optimization of Launch Vehicles", *6th NASA, and ISSMO, Symposium on Multidisciplinary Analysis and Optimization*, Bellevue, Washington, AIAA 1996-4130.

Raivio, T. and Ehtamo, H. (2000), "Visual Aircraft Identification as a Pursuit-Evasion Game", *Journal of Guidance, Control, and Dynamics*, Vol. 23, No. 4, pp. 701-708.

Rao, P. P., Sutter, B. M. and Hong, P. E. (1996), "Six Degrees-Of-Freedom Trajectory Targeting and Optimization for Titan Launch Vehicles", *AIAA Guidance, Navigation and Control Conference*, Denver, Colorado, AIAA 1996-3776.

Riehl, J. P., Paris, S. W. and Sjauw, W. K. (2006), "Comparison of Implicit Integration Methods for Solving Aerospace Trajectory Optimization Problems", *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, Keystone, Colorado, AIAA 2006-6033.

Ries, T. (2005), "Full Dynamics 6DoF-Trajectory-Optimizations of an Unmanned Aerial Vehicle (UAV)", Diploma Thesis, Georgia Institute of Technology, Flight Mechanics and Control Group, Georgia.

Ringertz, U. (2000), "Optimal Trajectory for a Minimum Fuel Turn", *Journal of Aircraft*, Vol. 37, No. 5, pp. 932-934.

Rolfe, J. M. and Staples, K. J. (1986), *Flight Simulation*, Cambridge Aerospace Series, Cambridge.

Roskam, J. (2001), *Airplane Flight Dynamics and Automatic Flight Controls: Part I*, Darcorporation, Laurence, Kansas.

Ross, I. M. and Fahroo, F. (2002), "A Perspective on Methods for Trajectory Optimization", *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, Monterey, California, AIAA 2002-4727.

Roth, B. D. and Kroo, I. M. (2008), "Enhanced Collaborative Optimization: A Decomposition-Based Method for Multidisciplinary Design", *Proceedings of the ASME 2008 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*, Brooklyn, New York, DETC 2008-50038.

Roth, B. D. and Kroo, I. M. (2008), "Enhanced Collaborative Optimization: Application to an Analytic Test Problem and Aircraft Design", *12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, Victoria, British Columbia, AIAA 2008-5841.

Russell, J. B. (2003), *Performance and Stability of Aircraft*, Butterworth-Heinemann, Oxford.

Schmidt, L. V. (1998), *Introduction to Aircraft Flight Dynamics*, AIAA Education Series, American Institute of Aeronautics and Astronautics, Reston, Virginia.

Schultz, R. L. (1987), "Three-Dimensional Trajectory Optimization for Aircraft", *Journal of Guidance*, Vol. 13, No. 6, pp. 936-943.

Slotine, J.-J. E. and Li, W. (1991), *Applied Nonlinear Control*, Pentice Hall, Englewood Cliffs, New Jersey.

Snell, S. A. (1991), "Nonlinear Dynamic-Inversion Flight Control of Supermaneuverable Aircraft", Ph.D. Thesis, University of Minnesota, Minneapolis.

Snell, S. A., Enns, D. F. and Garrard, W. L. (1992), "Nonlinear Inversion Flight Control for a Supermaneuverable Aircraft", *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 4, pp. 976-984.

Sobieszczanski-Sobieski, J., Agte, J. S. and Sandusky, R. R. (1998), "Bi-Level Integrated System Synthesis (BLISS)", NASA Langley Research Center, NASA/TM-1998-208715.

Sobieszczanski-Sobieski, J., Agte, J. S. and Sandusky, R. R. (2000), "Bilevel Integrated System Synthesis", *AIAA Journal*, Vol. 38, No. 1, pp. 164-172.

Sobieszczanski-Sobieski, J., Barthelemy, J.-F. and Riley, K. M. (1982), "Sensitivity of Optimum Solutions of Problem Parameters", *AIAA Journal*, Vol. 20, No. 9, pp. 1291-1299.

Stevens, B. L. and Lewis, F. L. (1992), *Aircraft Control and Simulation*, John Wiley & Sons, New York.

Stevens, B. L. and Lewis, F. L. (2003), *Aircraft Control and Simulation*, John Wiley & Sons, Hoboken, New Jersey.

Sugar, R. D. and Stubberud, A. R. (1974), "Decomposition Technique for Minimum Time Trajectories", *Journal of Optimization Theory and Applications*, Vol. 14, No. 2, pp. 233-250.

Tanartkit, P. and Biegler, L. T. (1997), "A Nested, Simultaneous Approach for Dynamic Optimization Problems - II - The Outer Problem", *Computers and Chemical Engineering*, Vol. 21, No. 12, pp. 1365-1388.

Teo, K. L. (2005), "Control Parameterization Enhancing Transform to Optimal Control Problems", *Numerical Analysis*, Vol. 63, pp. 2223-2236.

Vasantharajan, S. and Biegler, L. T. (1990), "Simultaneous Strategies for Optimization of Differential-Algebraic Systems with Enforcement of Error Criteria", *Computers and Chemical Engineering*, Vol. 14, No. 10, pp. 1083-1100.

Vicente, L. N. and Calamai, P. H. (1994), "Bilevel and Multilevel Programming: A Bibliography Review", *Journal of Global Optimization*, Vol. 5, No. 3, pp. 291-306.

von Stryk, O. (1994), "Numerische Lösung optimaler Steuerungsprobleme: Diskretisierung, Parameteroptimierung und Berechnung der adjungierten Variablen", Ph.D. Thesis, Mathematics Centre, Technical University of Munich, Munich.

von Stryk, O. and Bulirsch, R. (1992), "Direct and Indirect Methods for Trajectory Optimization", *Annals of Operations Research*, Vol. 37, pp. 357-373.

Wächter, A., Laird, C., Margot, F. and Kawajir, Y. (2009), *Introduction to IPOPT: A tutorial for downloading, installing and using IPOPT*, Department of Mathematical Sciences, IBM T.J. Watson Research Center, Yorktown Heights, New York.

Well, K. H., Markl, A. and Mehlem, K. (1997), "ALTOS - A Software Package for Simulation and Optimization of Trajectories of Launch- and Reentry Vehicles", *48th International Astronautical Congress*, Turin, Italy.

Wiegand, A., Markl, A., Mehlem, K., Ortega, G., Steinkopf, M. and Well, K. (1999), "ALTOS - ESA's Trajectory Optimization Tool Applied to X-38 Reentry Vehicle Trajectory Design", *49th International Astronautical Congress*, Amsterdam, Netherlands.

Winter, C.-J., Sizman, R. L. and Vant-Hull, L. L. (1991), *Solar Power Plants*, Springer-Verlag, New York.

Zhao, Y. and Tsiotras, P. (2009), "Mesh Refinement Using Density Functions for Solving Optimal Control Problems", *AIAA Infotech@Aerospace Conference*, Seattle, Washington, AIAA 2009-2019.