

Advanced four-dimensional trajectory generation for tactical guidance of unmanned aerial vehicles

G Otto



orcid.org/0000-0003-2279-5871

Thesis accepted in fulfilment of the requirements for the degree
Doctor of Philosophy in Computer and Electronic Engineering
at the North-West University

Promoter: Prof G van Schoor

Co-promoter: Prof KR Uren

Graduation: May 2020

Student number: 20120974

Declaration

I, G Otto, hereby declare that the thesis entitled “**Advanced four-dimensional trajectory generation for tactical guidance of unmanned aerial vehicles**” is my own original work and has not already been submitted to any other university or institution for examination.

A handwritten signature in black ink, appearing to read 'G. Otto', with a long, sweeping underline.

Gustav Otto

Student number: 20120974

Signed on the 23rd day of February, 2020 at Potchefstroom.

Acknowledgements

A man's pride can be his downfall, and he needs to learn when to turn to others for support and guidance.

– Bear Grylls

This study has been an extremely challenging journey of discovery, and has afforded me the opportunity for extensive growth both academically and personally. I would not have been able to complete it without those that stood by me through the years it took to master the various skills and overcome the obstacles along the way.

Firstly I'm grateful to my promoters, Prof. van Schoor and Prof. Uren, for their continuous guidance and support through the years. Their feedback and motivation were pivotal in completion of this work. I would also like to thank Denel Aviation for their support to the research, both financially and with regards to technical matters.

I'm also eternally grateful to my parents for their unwavering support and ongoing encouragement to pursue my goals. Lastly, I would like to thank my wife Rocshann, for her eternal love, support and willingness to effectively put our lives on hold while I complete my studies. I will always be grateful for her efforts to keep me motivated during the late nights and weekends, and the sacrifices she has made to support me in the pursuit of my dream.

Abstract

A strategy is presented for the real-time generation of tactical, dynamically feasible 4D trajectories. The current framework for automatic flight control is analysed, along with current research into autonomous unmanned systems, to develop a technique for trajectory generation that not only works efficiently in the current architecture but also enhances the capabilities.

The algorithm is based on a segmented trajectory constructed using quartic, i.e. 4th order, Pythagorean Hodograph (PH) curves based on Bézier splines. This yields nonic spatial curves, joined so as to ensure second order geometric (G^2) continuity. The spatial trajectory is specified using way-points, tangents, and a geometric acceleration vector that controls the curvature. A velocity profile is constructed to add the time component, which can be specified as a time of arrival or velocity at each way-point. Additionally the tangent, consisting of a heading and climb angle, can be specified or calculated automatically.

The primary contribution of this study is extending the algorithm to include all the orientational information, thus including the roll angle of the craft in the Pythagorean Hodograph (PH) curve construction. The demonstration of a basic aircraft performance model (APM) serves to illustrate the use of dynamic non-linear constraints, allowing for improved utilisation of the flight envelope.

The computational advantages of using splines is shown by presenting an extensive collection of performance metrics to analyse the trajectory's efficiency and mission performance. The inclusion of the orientational dynamics allows for a more detailed analysis of the aircraft's aerodynamic performance. Using the results of the trajectory evaluation it is possible to apply kinodynamic constraints to ensure that the trajectory is within the aircraft's flight envelope.

Keywords: UAV, Navigation, Bézier Spline, Kinodynamic, Trajectory Generation, Motion Planning, Pythagorean Hodograph (PH) curve

Contents

Acronyms	xix
List of Symbols & Subscripts	xxi
Preface	xxv
1 Introduction	1
1.1 Background	1
1.2 Collaborative Control Framework	3
1.3 Study Motivation	6
1.4 Problem statement	9
1.5 Objectives	10
1.6 Methodology	10
1.7 Contribution	11
1.8 Thesis Layout	13
2 Literature Survey	15

2.1	Introduction	15
2.2	Trajectory Generation	15
2.2.1	Optimal Control Approach	17
2.2.2	Geometric-Dynamic Approach	21
2.3	Critical Review	28
3	Bézier Spline Trajectories	31
3.1	Introduction	31
3.2	Suitability of Bézier Splines	31
3.3	Problem Formulation	34
3.3.1	Spatial Position Curve	37
3.3.2	Timing Law	38
3.3.3	Boundary Conditions	40
3.4	Pythagorean Hodograph Curves	41
3.4.1	Pythagorean Quadruple	42
3.4.2	PH Curve Construction	44
3.4.3	Case studies	45
3.5	Conclusion	48
4	Trajectory Generation	49
4.1	Introduction	49

4.2	Curve Framing	49
4.2.1	Relatively Parallel Adapted Frame	51
4.2.2	Frenet-Serret Frame	52
4.2.3	Euler-Rodrigues Frame	55
4.2.4	Rotation Minimising Frame	56
4.3	Orientation Specification	58
4.3.1	Euler-Rodrigues Frame for Motion Design	58
4.4	Quaternion Spline Trajectories	60
4.4.1	Spatial Curve	62
4.4.2	Parametric Speed & Arc Length	63
4.4.3	Timing Law	65
4.4.4	Quaternion Boundary Conditions	68
4.5	Summary of Trajectory Generation Algorithm	72
4.6	Conclusion	73
5	Trajectory Evaluation	75
5.1	Introduction	75
5.2	Trajectory Evaluation & Constraints	76
5.2.1	Geometric Constraints	77
5.2.2	Translational Motion	80
5.2.3	Angular Rates	82

5.2.4	Separation Constraints	84
5.2.5	Operational Constraints	86
5.2.6	Non-linear Constraints	87
5.3	Performance Evaluation	88
5.3.1	Altitude Change	88
5.3.2	Energy	89
5.3.3	Tracking Error Risk	89
5.3.4	Mission Objectives	90
5.4	Conclusion	91
6	Case Study	93
6.1	Introduction	93
6.2	F16 Model	94
6.2.1	Generic Aircraft Model	94
6.2.2	Aerodynamic Coefficients	96
6.2.3	Wind Frame vs Body Frame	97
6.3	Inverse Model	98
6.3.1	Kinematic Inversion	98
6.3.2	Aerodynamic Angle Approximation	99
6.4	Aircraft Performance Model	99
6.5	Inverse Simulation	106

6.6	Boundary Constraints	108
6.7	Results	109
6.8	Conclusion	112
7	Conclusion	113
7.1	Overview	113
7.2	Future work	115
	Bibliography	117
A	Bézier Splines	127
A.1	Introduction	127
A.1.1	Explicit Bézier Curves	128
A.1.2	Higher Dimensional Curves	129
A.1.3	Hermite Interpolation	129
A.2	Bézier Spline Computational Properties	130
A.2.1	Convex Hull property	130
A.2.2	Derivatives (Hodograph)	131
A.2.3	Integrating Bernstein polynomials	131
A.2.4	Degree elevation	132
A.2.5	Subdivision	132
A.2.6	Addition	133

A.2.7	Multiplication	133
A.2.8	Composition	133
A.2.9	Limitations	134

List of Figures

1.1	Coordination architectures	4
3.1	PH curve variance for free parameters	46
4.1	Relatively parallel adapted frame (RPAF)	52
4.2	Frenet-Serret Frame of a curve	54
4.3	Curvature of the curve in figure 4.2	54
4.4	Nonic Euler-Rodrigues frame (ERF)	57
6.1	Wind Frame Aerodynamic Coefficients	103
6.2	Aerodynamic Constraints	104
6.3	Engine Thrust Limits	105
6.4	Specific Excess Power	105
6.5	Inverse Simulation Verification	107
6.6	Aircraft Performance Model Approximation	107
6.7	Minimum Turning Radius at max Military Thrust	109

6.8	Way-point Turning Radius Test	109
6.9	4.5 G Turn 1km Altitude	111
6.10	4.5 G Turn 6km Altitude	112
A.1	Explicit Bézier curve	128
A.2	2D Bézier curve	129
A.3	Convex hull of Bézier curve	131

List of Tables

6.1	Lift Coefficients	101
6.2	Drag Coefficients	101
6.3	Sideforce Coefficients	102
6.4	Trajectory Inputs	110
6.5	Trajectory Inputs	111

Acronyms

4DT	4 Dimensional Trajectory
APM	Aircraft Performance Model
ASV	Autonomous Surface Vehicle
ATM	Air Traffic Management
AUGV	Autonomous Unmanned Ground Vehicle
AUV	Autonomous Underwater Vehicle
CAD	Computer Aided Design
COM	centre of mass
DCM	direction cosine matrix
DMRP	Direct Method for Rapid Prototyping
DoF	degree of freedom
DP	Disjunctive Programming
EAS	Equivalent Airspeed
ECEF	earth-centred, earth-fixed
ERF	Euler-Rodrigues frame
FOV	field of view
GA	Genetic Algorithm
GJK	Gilbert-Johnson-Keerthi
ISCP	incremental SCP
MAV	Micro Air Vehicle
MILP	Mixed Integer Linear Programming

MIMO	Multiple Input, Multiple Output
MPC	Model Predictive Control
NED	north-east-down
NLP	non-linear programming
NP	Nondeterministic Polynomial time
PH	Pythagorean Hodograph
PSO	Particle Swarm Optimisation
RHC	Receding Horizon Control
RMF	rotation minimising frame
RPAF	relatively parallel adapted frame
RRMF	rational rotation minimising frame
RRT	Rapidly-exploring Random Tree
SCP	Sequential Convex Programming
SLERP	Spherical Linear Interpolation
SMC	Sliding Mode Control
SQP	Sequential Quadratic Programming
TA	terrain avoidance
TF	terrain following
UAV	Unmanned Aerial Vehicle

Nomenclature

Conventions

$\hat{t}$	Normalised form of variable t
$\bar{p}$	Bernstein coefficients of spline $p(\zeta)$
t	Scalar variable
$\psi()$	Scalar function
$\mathbf{p}$	Vector variable
$\mathcal{A}$	Quaternion variable
$\diamond^s$	Segment start value
$\diamond^f$	Segment end value
$\diamond_i$	Variable of segment i
$\diamond_{j,i}$	Variable of segment i for vehicle j

Compound function

$$(f \circ g)(x) = f(g(x))$$

Intervals

$x \in [a, b]$	$x \in \mathbb{R}, a \leq x \leq b$
$x \in [a..b]$	$x \in \mathbb{Z}, a \leq x \leq b$
$x \in (a, b)$	$x \in \mathbb{R}, a < x < b$
$x \in [a..b)$	$x \in \mathbb{Z}, a \leq x < b$

List of Symbols

Trajectory

<i>Symbol</i>	Description	[unit]
N	Number of way-points	[-]
i	Index of trajectory segments & way-points	[-]
$\mathbf{w}_i$	Way-point i (3D coordinates)	[m]
v_i	Velocity at way-point w_i	[m.s ⁻¹]
a_i	Acceleration at way-point w_i	[m.s ⁻²]
t	Universal time	[s]
t_i	Arrival time for way-point w_i	[s]
t_i^s	Segment i start time	[s]
t_i^f	Segment i end time	[s]
t^f	Final arrival time at $\mathbf{w}_N$	[s]
T_i	Segment flight time	[s]
$\hat{t}_i$	Normalised time index of segment i	$\hat{t} \in [0, 1]$
$\theta_i(t)$	Timing law for segment i	[s ⁻¹]
$\mathbf{p}_i(\zeta)$	3D Position curve for segment i	[m]
p_N	North component of position $\mathbf{p}$	[m]
p_E	East component of position $\mathbf{p}$	[m]
p_z	Altitude component of position $\mathbf{p}$	[m]
$\zeta_i(t)$	Dimensionless spatial curve index	$\zeta_i \in [0, 1]$
$s_i(\zeta)$	Distance curve of segment i	[m]
L_i	Arc length of segment i . $L_i = s_i(1)$	[m]
γ	Flight path angle	[rad]
ψ	Heading angle	[rad]
ϕ	Roll angle	[rad]
α	Angle of attack	[rad]
β	Side-slip angle	[rad]

Bézier Splines

$p(\zeta)$	Bézier curve	[-]
$\bar{p}_k$	Bézier coefficient	[-]
$\beta_{k,n}(\zeta)$	Bernstein basis polynomial	[-]
n	Order of the spline polynomial	[-]
k	Base polynomial identifier	[-]
t	Time index for time dependent curve	[s]
ζ	Dimensionless spline index	[-]

Pythagorean Hodograph Curves

$\mathcal{A}_i(\zeta)$	Quaternion polynomial form	[-]
$\sigma_i(\zeta)$	Parametric speed polynomial	[-]
$u(\zeta), v(\zeta), p(\zeta), q(\zeta)$	Real polynomial quadruple	[-]
$\mathbf{h}_i(\zeta)$	Spatial curve hodograph	[m]
$\mathbf{p}_i(\zeta)$	Spatial position curve	[m]
$\theta(t)$	Timing law	[-]
$\zeta(t)$	Spline index at time t . Timing law integral	[-]
$\hat{\mathbf{i}}, \hat{\mathbf{j}}, \hat{\mathbf{k}}$	Mutually orthogonal unit vector base frame	[-]
$\mathcal{A}^*$	Quaternion conjugate	[-]
ϕ	Free parameter parameter	[-]
τ	Free parameter parameter	[-]
$\mathbf{m}_i^s, \mathbf{m}_i^f$	Tangent vectors (derivative constraints)	[-]
$\mathbf{c}_i^s, \mathbf{c}_i^f$	Curvature control vectors (second derivative constraints)	[-]
κ_i	Curvature at way-point i	[m ⁻¹]
(λ, μ, ν)	Direction cosines of a vector	[-]
$\mathbf{r}$	Generic vector result	[-]
$\bar{\mathcal{A}}_{i,k}$	Bézier coefficients for quaternion polynomial $\mathcal{A}_i(\zeta)$	[-]
$\bar{\sigma}_{i,k}$	Bézier coefficients for parametric speed $\sigma_i(\zeta)$	[-]

$\bar{s}_i$	Bézier coefficients for arc length $s_i(\zeta)$	[-]
$\bar{\theta}_{i,k}$	Bézier coefficients for timing law $\theta_i(t)$	[-]
$\bar{\zeta}_{i,k}$	Bézier coefficients for parametric function $\zeta_i(t)$	[-]
n_θ	Order of timing law	[-]

Curve Framing

E	Generic frame	[-]
E^F	Frenet frame	[-]
E^A	Adapted frame	[-]
E^R	Euler-Rodrigues frame	[-]
$\mathbf{e}_k$	Unit vector	[-]
$\mathbf{T}$	Tangent vector of curve	[-]
$\mathbf{N}_p$	Principal normal vector of curve	[-]
$\mathbf{B}$	Binormal vector of curve	[-]
κ	Curvature of curve	$[\text{m}^{-1}]$
τ	Torsion of curve	[-]
$\mathbf{N}_k$	Generic normal vector	[-]

Preface

Obstacles are those frightful things you see when you take your eyes off your goal.

– Henry Ford

Chapter 1

Introduction

The journey of a thousand miles begins with one step.

– Lao Tzu

1.1 Background

The proliferation of unmanned systems in recent years in both military [1,2] and civil [3–6] domains, has only served to increase the need for better control over the interaction of these systems. Research into Air Traffic Management (ATM) [7], Sense, Detection and Avoidance [8], and Trajectory Generation [9] has seen a surge in interest. While some researchers are focussed on improving the efficiency of trajectories to reduce operational cost, greenhouse gas emissions, and improving airspace utilisation in congested areas [10], others are focussed on collaboration between these systems to enable them to perform more complex tasks. The latter being of particular interest in achieving true autonomy. Unmanned systems have the potential to offer great advantages over their manned counterparts, including improved efficiency, accuracy and safety, while reducing overall system and operational costs. These autonomous systems are also able to coordinate much faster, enabling more vehicles to operate in the same space.

Achieving true autonomy in unmanned systems requires a deeper understanding of the attributes the human pilot introduces into the system's operation. From the literature, the following attributes have been identified as crucial:

- *Flight Control* - Ensuring stable and accurate manoeuvring of the vehicle.
- *Supervision & Error Handling* - Evaluating system performance and adapting to deviations from expected behaviour.
- *Navigation* - Preflight mission planning and on-board real-time replanning in dynamic environments to ensure safe and efficient obstacle avoidance.
- *Coordination & Collaboration* - Sharing intent with various other actors in an area of operation and jointly optimise mission efficacy.
- *Tactical Evaluation* - Quickly evaluating various possible solutions to find the global optimal solution, both in regards to independent and collaborative operation.
- *Self-evaluation & Improvement* - The ability to compare real performance to predicted behaviour and adjust accordingly. The ability to define strategies, manoeuvres and other heuristics, and exchange them with other collaborators.

The core challenge in achieving autonomous operation, lies in the ability to coordinate with other systems. Coordination in essence is the exchange of intent, and collaboration is the alignment of that intent to a shared goal. This distinction between control and intent is important, as it fundamentally states that coordination and collaboration does not require centralised control, but rather simply ensuring that the actions of each agent are optimally aligned with the desired outcome.

Therefore a key enabling technology for autonomous systems to reach their full potential, is a universal cooperative control framework designed to emulate the characteristic capabilities of a human operator. This control architecture must allow for heterogeneous vehicle and task definition, which is essential for cooperative control. It should allow for spatial and temporal constraints used for the evaluation of vehicle dynamics, performance

and mission efficacy. These feasibility constraints are the foundation of ensuring flyable and collision free trajectories, which the system should be able to evaluate quickly and accurately. The control architecture should also be versatile enough to account for the various disciplines that affect and limit its operation, such as vehicle dynamics, communication constraints and failure of any number of subsystems. The universality of such a controller is only required at the navigation (trajectory generation) level, since it is at this level that coordination and collaboration occurs.

The primary challenge of trajectory generation, especially where multiple vehicles and objectives are concerned, is the heavy computational cost associated with the optimisation. This makes a centralised approach undesirable, as all the computational load is placed on a single node. This not only limits the scalability of the system, but also becomes a central point of failure. Therefore, it is not only desirable to have a decentralised approach, but a necessity [11]. Having such algorithms optimised for distributed computation, is also advantageous in respect to modern computer architectures optimised for parallelism.

1.2 Collaborative Control Framework

The following approach to distributed trajectory generation, serves as the control framework for this study. It serves to provide a broader context for the work discussed in later chapters, with trajectory optimisation decentralised to distribute the computational load, while retaining the ability to coordinate with a centralised authority. In the context of this framework, centralised control refers to central coordination of vehicles that collaborate on a single task (e.g. search patterns) or coordinate to a single objective (e.g. airport approach and departure coordination). It does not refer to the centralised generation and optimisation of trajectories by a single entity, and the subsequent distribution of those trajectories. The central coordinator serves as the consensus authority to ensure faster coordination in complex scenarios, such as air traffic control or a squad leader coordinating the actions of its members. This distributed approach is shown in Figure 1.1 as opposed to centralised and decentralised approaches.

The dark solid links represent primary communication channels, while the dotted lines are optional channels used based on situational requirements.

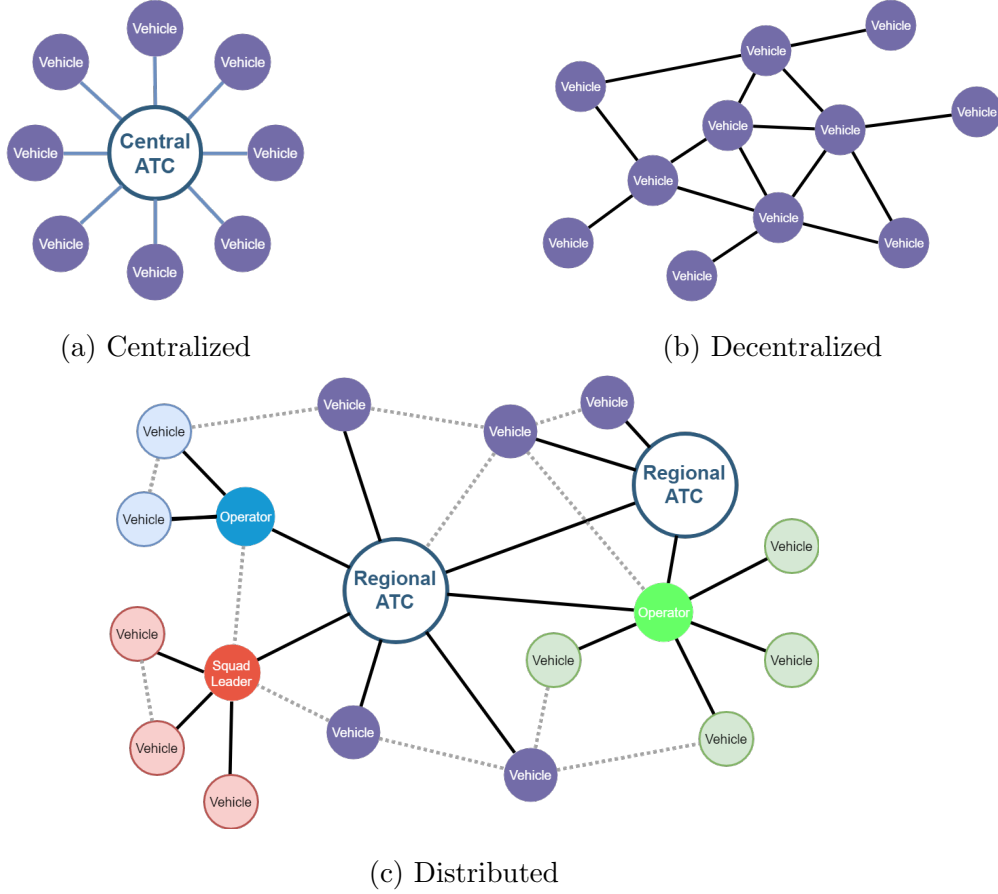


Figure 1.1: Coordination architectures

Navigation or path planning, i.e. the generation of the way-point sequence, is done either through a consensus based algorithm or centralised through a central authority, evaluating simple cubic spatial trajectories. These initial trajectories are then distributed to each vehicle for evaluation and optimisation of its own trajectory. The optimised trajectory or counter proposal is then submitted to the network for evaluation and acceptance. The network can then either accept the trajectory through consensus or offer the aircraft with a new basic (low fidelity) trajectory proposal, effectively performing a spatial, or temporal, re-routing. Standardising responses to collision scenarios can serve to ensure predictable behaviour of the system even if some vehicles experience a loss of communication.

In this approach a consensus based controller would be more generally applicable, while

a centralised controller may be desirable in congested areas such as airports. In such congested areas, the additional computational load of a centralised approach may be more desirable than the communication overhead of a consensus based approach. Additional considerations involved, such as arrival time separation and the ground logistics of the airport, make the centralised approach more practical, while the limited airspace being considered helps to constrain the computational cost. A hybrid approach at this level would centrally coordinate a group of vehicles that collaborate on a specific task, while other aircraft or groups in the airspace coordinate with the group through consensus.

The optimisation of the trajectories is done in an iterative distributed manner. Firstly the way-points of each trajectory are optimised to spatially and temporally orient the trajectory segments. This allows for efficient optimisation of task assignment and collision avoidance. These initial trajectories are then optimised on-board the aircraft by optimising the dynamics of each segment. This dynamic optimisation is where each aircraft's specific aerodynamic performance and mission efficacy is evaluated. The dynamic optimisation has a smaller impact on the trajectory's geometry than the way-point sequencing, and thus only introduces smaller changes in the spatial location and orientation. It should also be noted that the greatest gains in performance optimisation at this level, would probably be achieved through manipulation of the timing law. Having this separation between the navigational coordinator, that ensures spatial and temporal separation, and the trajectory optimiser is advantageous. This is because the trajectory optimisers used for each aircraft are dependent on the task the aircraft is performing and its aerodynamic characteristics. The separation therefore allows for various different aircraft types with unique objectives to operate simultaneously in a given airspace. Furthermore, separating the navigational control from the guidance controller, which executes the trajectory, allows heterogeneous vehicles to coordinate despite using different flight control systems, or technologies.

Trajectory generation is essentially an optimisation problem, and given its scope as a semi-infinite non-smooth optimisation problem, is a computationally challenging one. Generating a trajectory requires the optimisation of a complex set of interdependent constraints, and evaluating multiple tactical options in near real-time. Increasing the

number of vehicles and trajectory segments, makes it infeasible to solve this simultaneously as a single optimisation problem. It would be more practical to state the objective of trajectory generation as generating good (sub-optimal), rather than optimal trajectories, that are dynamically feasible and collision free. Having this baseline objective, safety and feasibility can be guaranteed, and successive iterative evaluation can be used to improve the trajectories based on the available computational resources. The fast generation of feasible trajectories are critical for response to dynamic environments and changing objectives, as the impact of short reactive manoeuvres on flight efficiency are less severe than the accumulative effect of optimising the trajectory over the entire flight.

Essentially the speed of optimisation relies on the rate at which each iteration of the trajectory can be evaluated, and the amount of information extracted from the trajectory affects the number of objectives and constraints the algorithm can consider. It is for this reason that this study focusses on this aspect of trajectory generation, i.e. efficient description and evaluation of the trajectory to eventually allow for fast distributed multi-objective optimisation. Furthermore, algorithms are presented to efficiently ensure spatial and temporal separation of segmented trajectories.

1.3 Study Motivation

Trajectory generation, or motion planning, for unmanned systems is a well established field with many novel and mature methods that exist in the literature. These methods can generally be divided into two classes based on the approach used to describe the problem, i.e. optimal control approach and geometric-dynamic approach [11]. The optimal control approach essentially solves a detailed dynamic model of the system, directly providing the control solution to perform the manoeuvre and explicitly ensuring feasibility. The geometric-dynamic approach in contrast, reduces the navigation problem to a simplified kinematic representation of the system's behaviour, placing limits on the geometric properties of the trajectory to ensure feasibility. The execution of the proposed manoeuvre is left to a guidance controller, thus the trajectory generation is only concerned with ensuring the manoeuvre is within the capabilities of the vehicle.

Many different techniques exist within each class, distinguished mainly by the mathematical tools used to find and optimise the solution. Each of these methods have their own merits and limitations, with most techniques using single segments for each aircraft that are temporally synchronized (i.e. same start and/or end times). This study is focussed on the use of the geometric-dynamic approach, specifically the use of Bézier splines for their geometric intuitiveness and computational advantages.

Consider a generic 10 degrees of freedom (DoF) aircraft model, defined by the state vector consisting of the aircraft's position $\mathbf{p} = [p_N \ p_E \ p_D]$, velocity v , angle of attack α , side-slip angle β , orientation $\Theta = [\phi \ \theta \ \psi_B]$, and thrust F_T . The trajectory can be fully defined by various combinations of four of these parameters [12]. For the purpose of navigation it is intuitive to use the three position parameters to define the trajectory, however this leaves an additional degree of freedom. Many approaches assume the specification of $\beta(t) = 0$, specifying coordinated turns. Although this approach is generally well suited to the navigation problem, it does limit the system to a smaller subset of the aircraft's capabilities. The specification of more aggressive and task specific manoeuvres would benefit from the ability to specify the roll angle ϕ as the fourth parameter. The system is further decoupled to increase the degrees of freedom (DoF) by separating the geometric specification from the temporal specification, using a timing law. The timing law allows for a limited degree of independent optimisation of the spatial position and vehicle dynamics.

In this study the specification of the roll angle is presented based on quaternion splines. This technique is not intended to replace the spatial approach with coordinated turns, but to extend the geometric dynamic approach to allow for the specification of more aggressive manoeuvres generally not possible with existing vector based approaches. The complete orientation specification in the trajectory combined with adaptive and dynamic constraints extend the usable flight envelope of the aircraft.

The purely spatial approaches to trajectory specification does not explicitly include the complete orientation, and thus the roll angle is specified separately. In this approach there is no direct mathematical coupling between the parameters, and thus discontinuities in the Euler angles must be dealt with manually. Specifying the trajectory directly using

quaternions, offers a directly coupled solution that is free of discontinuities and ambiguity. The quaternion trajectory allows for the simultaneous specification of the aircraft velocity (v), flight path angle (γ), heading (ψ_W) and roll angle (ϕ), in an elegant manner that is coupled and computationally efficient, by changing the trigonometric Euler rotations to simple scalar operations. This structure is also intuitive as the trajectory is essentially constructed by stringing together a sequence of manoeuvres (segments) in a dynamic manner, while ensuring continuity.

As shown in chapter 2, a special class of splines called Pythagorean Hodograph (PH) curves are often used for their computational advantages in path description. They are often defined in quaternion form as this offers a more concise description of the curve, however the fourth degree of freedom is left as a free parameter to optimize the curve geometry. Since the quaternion curve offers a complete orientation frame specification, we propose using this directly to include the roll angle in the spatial curve specification. Quaternions are often used in the modelling of dynamics, but to the best of our knowledge has never been directly used in trajectory generation in this context.

The quaternion Bézier spline form meets the requirements to be a Pythagorean hodograph curve, which offers exact analytical solutions to the parametric speed and arclength of the segment. The form also offers an intuitive association with the aircraft motion, as the hodograph (tangent) form directly specifies the aircraft velocity and orientation and then derives the position curve through an integration constraint in the form of way-points. The second order geometric (G^2) continuity requirements result in a quaternion polynomial of 4th order, which in turn yields a 9th order polynomial after the quaternion product ($\mathcal{A}(\zeta)\hat{\mathbf{i}}\mathcal{A}^*(\zeta)$) used to define the hodograph, and subsequent integration to get the position curve.

The order of the curve also means that the resulting frame description of the quaternion form, called the Euler-Rodrigues frame (ERF), is a rotation minimising frame (RMF). Rotation minimising frames, as the name suggests, offer an interpolation that minimises the changes in orientation, similar to the minimal curvature advantages of splines, regarding the spatial curve interpolation of the way-points.

As demonstrated in later chapters, this use of the PH curve combines the computational advantages of quaternions with those of splines, offering a concise and accurate means of defining a trajectory. The additional insight into the vehicle kinematics allows for the specification of more advanced manoeuvres without dramatically increasing the computation cost.

1.4 Problem statement

The problem addressed in this study then, is to extend the capabilities of the geometric-dynamic approach by increasing the extent to which the vehicle's dynamics are considered during trajectory generation. Current vector based (purely spatial) geometric-dynamic approaches strictly limit the motion of the vehicle to a small subset of its flight envelope, due to the simplification of the non-linear dynamics as opposed to the optimal control approaches. The optimal control approaches are computationally far more expensive than the geometric dynamic approaches, and being specific to a given vehicle's dynamic model, lacks the necessary universality to allow for global coordination and autonomy. Therefore the optimal control approach scales poorly in respect to congestion, heterogeneous vehicles, large areas and extended time periods.

Extending the capabilities of the geometric-dynamic approach to gain increased insight into the vehicle dynamics without dramatically increasing the computational cost, offers a compromise between these two approaches. The direct inclusion of the 6 DoF kinematics in the trajectory generation, removes additional computational steps reducing the overall computational cost. The segmentation of the trajectories and universality offered by the geometric-dynamic approach allows for improved scalability of the algorithms, while also being well suited to parallelism and by extension distributed optimisation. The universality of the geometric-dynamic approach allows for cooperation between heterogeneous vehicles, while retaining the versatility and other characteristics offered by a human operator.

1.5 Objectives

The primary objective of this study is to include the 6 DoF kinematics into the trajectory generation in a computationally efficient manner. The goal being to gain greater insight into the vehicle dynamics along the trajectory, and subsequently exploit this to apply non-linear kinodynamic constraints to the trajectory. The ability to apply non-linear constraints, rather than conservative static constraints, would allow the use of a greater portion of the vehicle's flight envelope than is possible with traditional geometric-dynamic approaches. Furthermore, the secondary objective of basing the trajectory generation on the geometric-dynamic approach is to overcome the discretisation and computational cost associated with the optimal control approaches. This includes the segmentation of the trajectory to improve the tractability of the trajectory generation problem by limiting the scope of each segment, ensuring the generation of feasible trajectories. The intent of this study, is to attempt to bridge the gap between the optimal control and geometric-dynamic approaches, by presenting a hybrid approach that more accurately models the vehicle dynamics without the computational cost associated with optimal control.

This study focusses on the underlying construction of the trajectory, not the subsequent optimisation nor the initial path finding. The reason being that improving the computational efficiency and conciseness of the trajectory representation, allows for a greater number of trajectory evaluators to be used during optimisation.

1.6 Methodology

In order to demonstrate this technique, the underlying construction of the trajectory using PH Bézier curves was implemented. Segmentation of the trajectory is achieved through derivative boundary conditions on each segment to ensure the required continuity. Each segment, being constructed of a spatial curve and corresponding timing law, is optimised using a basic minimisation of the bending energy and variance in the parametric speed.

Validation is done through evaluation of manoeuvres using an agile aircraft with a wide

non-linear manoeuvring envelope, specifically an F16 fighter aircraft. The non-linear aircraft model of the F16 in [13] was used in the simulations, as this offered sufficient non-linearity to demonstrate the approach. This model is similar to the model in [14], but omits the aerodynamic effects of the leading edge flap and speed brake.

The model is precomputed to generate performance metrics in the form of a performance model, which models the non-linear variation in the aircraft’s capabilities. This serves as the non-linear constraints used to evaluate the trajectory without the need to compute the model during trajectory generation. To correlate the results the final trajectory is simulated using a simplistic MPC on the non-linear model to verify that the trajectory is feasible.

1.7 Contribution

The primary contribution of the research is the application of a specific mathematical representation for the specification of aircraft trajectories, that is not only elegant in its coupling of the parameters, but the form of which offers great computational advantages. The contribution lies in taking an existing technique (Pythagorean Hodograph curves) and applying it in a new way (aircraft model with dynamic constraints) to gain new insights into the efficiency (parameter coupling and computation) of the proposed approach. The direct inclusion of the 6 DoF kinematic model in the trajectory generation, removes an additional computation step and combines the computational advantages of the geometric-dynamic approach with the dynamic evaluation provided by the direct methods (inverse dynamics). This approach is essentially a hybrid of the geometric-dynamic and direct methods, which reduces the computational cost of the combination.

Previous research has demonstrated the suitability of PH curves to distributed optimisation, single segment trajectories with dynamic constraints, and segmentation of elementary trajectories with geometric constraints. The secondary contribution of this study is extending the use of PH curves with complex dynamic constraints to trajectories consisting of multiple segments, presenting efficient algorithms to evaluate multiple

trajectories that are asynchronously segmented. Various computational heuristics are presented to improve the computational efficacy of the architecture and to demonstrate the versatility and scalability of the approach.

1.8 Thesis Layout

Chapter 2 offers a brief literature survey of the various approaches and techniques used in trajectory generation.

Chapter 3 gives an introduction to Bézier splines and discusses their suitability to the trajectory generation problem. The special case of Pythagorean Hodograph (PH) curves is introduced for their computational advantages.

Chapter 4 introduces the concepts of curve framing and how it relates to the problem of trajectory generation. These concepts are then applied to the quaternion PH curve to generation trajectories that directly include the rotational kinematics of the vehicle.

Chapter 5 presents the various evaluation functions used to quantify the performance of the trajectory for optimisation of quaternion PH trajectories.

Chapter 6 presents a case study of an F16 model, a basic proof of concept aircraft performance model, and simulation results.

Chapter 7 gives an overview of the study and the conclusions make, including possible future work.

Appendix A gives a detailed introduction to Bézier splines, their construction, manipulation and the various computational algorithms used to analyse the curve.

Chapter 2

Literature Survey

An investment in knowledge pays the best interest.

– Benjamin Franklin

2.1 Introduction

This chapter presents a brief literature survey of the various approaches and techniques used in trajectory generation. These approaches are divided into two main categories, optimal control and geometric-dynamic approaches, based on the fundamental philosophy behind them. Each category is presented with various examples of techniques that are based on these approaches, and are subsequently evaluated based on their pros and cons.

2.2 Trajectory Generation

Trajectory generation, or motion planning, is the process of generating a smooth path that connects a sequence of way-points (nodes) so that the path is free of collisions and conform to the dynamic constraints of the vehicle. Trajectory generation algorithms, in contrast to path planning, are built around an understanding of the dynamics that govern the motion of the aircraft [9]. It is for this reason that path planning algorithms, such as A*, D* and

Rapidly-exploring Random Trees (RRTs), are not directly considered in this study. These path planning algorithms also scale poorly to large 3D environments. The large scale and dynamic nature of autonomous flight makes grid based techniques computationally unsuitable, instead favouring landmark based approaches. Hybrid techniques that use these path planning algorithms in conjunction with splines to include dynamics are discussed in later sections.

There are three primary considerations (objectives) during trajectory generation beyond simply navigating between points, which are:

- *Collision Avoidance*

Finding a route between the way-points that are free of collisions.

- *Dynamic Feasibility (Flyable)*

Ensuring the proposed route is within the capabilities of the vehicle, ensuring the aircraft can accurately track the desired trajectory.

- *Optimisation*

Ensuring the proposed route efficiently achieves the desired mission objectives.

This division of the trajectory generation problem offers insight into the sequence (or priority) for optimisation of the trajectory. Coordination and collaboration adds an additional level of complexity during optimisation and also complicates obstacle avoidance, since it requires evaluation of additional dynamic obstacles. Trajectory generation also requires the versatility of both off-line and on-line operation, as both preflight planning and dynamic replanning during flights are critically important.

As stated in section 1.3 trajectory generation can generally be divided into two classes based on the approach used to describe the problem, i.e. optimal control approach and geometric-dynamic approach [11].

2.2.1 Optimal Control Approach

The optimal control approach calculates the optimal control input that minimises a given cost function, which explicitly satisfies the dynamic feasibility constraints of the system, but requires an exact dynamic model of the vehicle. These approaches can be divided into two subcategories, the direct and indirect methods [15].

The indirect methods use the Pontryagin minimum principle to derive the analytical conditions necessary for optimality. The challenge with this approach lies in the complexity of the derivation of the Hamiltonian, the first-order optimality conditions, boundary conditions and path constraints. In the case of problems based on table lookups, such as those used in aerodynamic modelling, it is impossible to derive these optimality conditions. The limited range of values of initial costate boundary conditions for which the algorithm converges to the optimal trajectory solution, results in strict requirements for choosing these initial values. The solution of this approach is however guaranteed to be an optimal trajectory and is explicitly feasible [16].

In contrast, the direct methods solve the optimal control problem by mapping it to a non-linear parameter optimization problem. Various polynomial parameter description schemes have been proposed, often spline based, for the continuous specification of the states and/or controls [16]. These methods are further subdivided into simultaneous, which maps the problem to a finite dimensional NLP problem via discretisation of both control and state variables, and sequential, which only discretises the control variables. The advantage of the direct methods lie in that they do not require a symbolic derivation of the optimality conditions, making them more applicable to general purpose software implementation. These solvers are also able to robustly handle poor initial conditions, making the direct methods more suitable and versatile to trajectory generation, than the indirect methods.

The optimal control approach is particularly useful in space systems where exact solutions can be found, and very little to no deviation exists between the planned and physical manoeuvres due to the lack of significant external disturbances. Pseudospectral (PS)

optimal control is of particular interest in this category, as it has become especially popular in space technology to design minimal propellant manoeuvres and trajectories [17, 18].

In aviation, the computational cost of optimal control is less suitable as the accuracy of the results are greatly affected by the unpredictability of environmental disturbances and the dynamic operating environments. Many different techniques with different approximations, generalisations and assumptions are used to reduce the computational complexity of the problem. Disturbances are omitted during trajectory generation, since aerodynamic disturbances can't accurately be predicted and are compensated for by using strict dynamic constraints to avoid control saturation, leaving additional freedom to compensate for disturbances.

Direct, also called inverse dynamic, approaches [12, 19] exploit the differential flatness of linearised models to limit the computational complexity. Due the computation cost, direct methods often map the problem to a lower dimensional space, essentially classifying them as hybrid geometric-dynamic approaches.

Techniques such as manoeuvre based control, limit the dynamics to manoeuvres with known solutions, thus reducing the complexity of the dynamic models to limit the computational cost. In [20] families of trajectories are defined using a sequence of segments, each with a controller parametrised by the goal state of the manoeuvre. This permits aggressive manoeuvres such as flying through narrow vertical gaps and perching on inverted surfaces. Many manoeuvre based approaches use adaptive controllers that learn from expert human operators performing the manoeuvres, however these approaches are limited to manoeuvres defined beforehand. The advantage of approaching manoeuvre based control using first principles, is that it allows for the specification of manoeuvres that a human operator wouldn't be able to perform. Using separate controllers for each manoeuvre also reduces the dimensionality of the model, simplifying the controller design.

The essence of the manoeuvre based approach is the generation of trajectories through concatenation of well-defined motion primitives from a finite library of manoeuvres. Frazzoli et al. [21] uses this to define trajectories for time-invariant dynamic control

systems with symmetries, applying differential and algebraic constraints on the state and control to define manoeuvres. The sequencing of manoeuvres is done through a state machine to select the most suitable manoeuvres. Each manoeuvre, being a motion primitive, reduces the motion of the vehicle to simple kinematic inversion problems, implicitly ensuring feasibility. Applying these motion primitives to the dynamic model doesn't limit the controller to simplified models, rather binding the dynamics to lower dimensional behaviour. This use of motion primitives however inherently makes the trajectory sub-optimal due to lowered dimensionality of the kinematic model.

A similar approach is demonstrated in [22] on a 3 DoF rotorcraft model, using continuous parametrised classes of feasible manoeuvre primitives. Continuity between manoeuvres are achieved through boundary constraints, allowing for the use of a non-linear model with non-linear constraints. The technique offers computationally efficient interpolation with the ability to select manoeuvre styles (agile or smooth), and is intended for integration with higher level planners.

The ability to perform aggressive manoeuvres is the primary advantage of manoeuvre based approaches, as in [23] where higher order Sliding Mode Control (SMC) is used to execute combat and/or aerobatic manoeuvres. The caveat however is that the system is limited to the manoeuvres contained in its library. Additionally, many manoeuvre based approaches assume that manoeuvres are completed before transitioning to the next manoeuvre, thus reacting to unknown obstacles midway through a manoeuvre can have unforeseen consequences on the vehicle dynamics.

Other techniques only consider a limited time horizon to reduce the computational load, such as Receding Horizon Control (RHC) or Model Predictive Control (MPC). This however introduces uncertainty regarding the behaviour after that time horizon, and due to the limited scope considered during optimisation would generally result in convergence to local minima. These techniques, as opposed to the inverse methods, are forward solvers that define the control inputs and evaluate the predicted behaviour of the system using a dynamic model. This again explicitly ensures feasibility of the trajectory. Optimisation of these optimal control problems are often based on Sequential

Quadratic Programming (SQP), Sequential Convex Programming (SCP) or Mixed Integer Linear Programming (MILP), each again introducing localised approximations to reduce computational complexity during optimisation.

Non-linear MPC with set-membership estimation was used in [24], combined with a SQP solver to demonstrate robust collision avoidance in a dynamic environment. The problem is structured so as to first provide a fast initial feasible trajectory and subsequently continuous optimisation given the permitted time. Cooperative reconnaissance trajectories that maximise the available information of a target is shown in [25], evaluating both risk and vehicle constraints. The limited time horizon of MPC is advantageous here when considering unpredictable moving targets. The suitability of MPC to the modelling of complex dynamics such as the linked dynamics of an aircraft towing a drogue is demonstrated in [26].

A comparison between Mixed Integer Linear Programming (MILP) and non-linear programming optimisers for MPC of unmanned vehicles in [27] illustrates the computational considerations of increased dynamic complexity. Therefore it is suggested that an initial MILP optimisation result serve as the seed for a more detailed non-linear optimisation, resulting in faster optimisation than directly applying the non-linear solver. This is an important observation, demonstrating that a simplified model and optimiser can be used to first find the general location of the global minimum. The subsequent optimisation using the detailed non-linear model and non-linear optimiser is then able to converge faster in that limited scope.

A decentralised MILP solver for cooperative trajectory generation based on MPC is shown in [28]. Given the complexity of the problem and the strict time constraints of real-time trajectory generation, a safe fall-back solution is introduced to ensure safe operation. Also using MILP MPC, [29] demonstrates decentralised cooperative trajectory generation with external disturbances and hard coupled state constraints that are non-convex.

2.2.2 Geometric-Dynamic Approach

The second class of trajectory generation algorithms, the geometric-dynamic approach, defines the trajectory using some polynomial spatial curve that is representative of the simplified kinematic behaviour of the craft. The fundamental philosophy behind the geometric-dynamic approach is that the tactical evaluation of various scenarios requires the generation of multiple trajectories to find the global optimal solution to the mission objectives. The computation cost of optimal control, which directly computes the exact manoeuvre using the full dynamic model, is less suitable to this higher level evaluation. Therefore these approaches map the control problem to a lower dimensional space, since at this level the trajectory is only required to be a basic representation of the vehicle's motion, in order to quickly evaluate feasibility and mission efficacy. The improved computation efficiency permits the evaluation of a larger number of trajectory iterations over a large range of possible strategies, improving the probability of finding a globally optimal solution.

These trajectories are generally defined using splines for their geometric simplicity and computational efficiency. Splines are an interesting sub-field in numerical analysis used for polynomial interpolation. The use of polynomial integration allows for exact analytical solutions to derivatives and integrands, while working with a reduced numerical control set (polynomial coefficients) that defines the spline curve. Splines are piecewise polynomial curves consisting of consecutive segments, each defined by a low-order parametric curve interpolating the control points. Splines can be interpolating, i.e. passing through its control points, or approximating, i.e. passing near the control points [30]. Furthermore splines can be classified as being global or local, describing whether changes in control points affect the entire curve, or are limited to local segments.

High order interpolation techniques such as Lagrange interpolation typically require an n^{th} order polynomial to interpolate $n + 1$ points. These polynomials are infinitely differentiable, but computationally expensive, exhibits global behaviour and the resulting curve can be counter intuitive. Using global B-spline interpolation removes these odd

behaviours, resulting in smooth approximating curves. The drawback of these splines include: increased complexity as the number of way-points increase, and reduction in the impact of each way-point as the number of points increase. Any changes to the trajectory also requires re-evaluation of the entire trajectory, making global splines less suitable for large complex and dynamic environments. The lack of explicit boundary conditions for B-splines makes the construction of segmented trajectories cumbersome, instead opting for lower order B-spline interpolation to overcome these limitations.

These geometric curves can however violate the dynamic constraints of the vehicle, and hence these curves must be constrained in some way to ensure feasibility. These constraints can be linked to simple differential geometric properties such as curvature and torsion, however these become cumbersome when switching to spatial curves. This is because the torsion is undefined when the curvature is zero and experiences discontinuities at inflection points along the trajectory. Curvature is also a purely scalar property, lacking an explicit spatial orientation. The main drawback of this approach is that these geometric properties do not directly relate to the vehicle dynamics, hence requiring them to be exceedingly strict to ensure feasibility. Another approach is to apply these constraints to the geometric projections of the trajectory in respect to the earth frame, limiting properties such as flight path (climb) angle (γ) and turning rate ($\dot{\psi}$). Although these earth reference properties are more representative of the vehicle dynamics, they still place strict limits on the manoeuvrability of the vehicle, especially if kept static.

Splines have proven useful in trajectory generation for various vehicle types, from land-based Autonomous Unmanned Ground Vehicles (AUGVs), Autonomous Surface Vehicles (ASVs) and Autonomous Underwater Vehicles (AUVs), to Unmanned Aerial Vehicles (UAVs). This universality of spline trajectories, makes them ideal for achieving collaboration between these different autonomous systems.

Trajectory generation for ASVs using splines was presented in [31]. It was noted here that it is critical that a cooperative control algorithm explicitly accounts for the vehicles capabilities and environmental conditions. The continuous trajectory description provided by splines are ideal for inter vehicle collision avoidance, while permitting accurate

optimisation of the manoeuvre. The geometric properties of splines are exploited in [32] to allow an AUV to follow the terrain geometry at a fixed height above the sea floor. To improve the computational efficiency, a smoothing approximation of the terrain is generated using a cubic B-spline surface. Terrain following is achieved by keeping the trajectory tangent parallel with the terrain, subject to curvature constraints. Since the AUV always remains upright, there is no need for the inclusion of roll dynamics, reducing the complexity of the problem.

In the case of UAVs, the increased agility, complex dynamics and speed of operation complicates the trajectory generation. Due to the complexity of spatial curves, some authors have proposed planar (2D) navigational solutions that assume a constant operating altitude. One such example is found in [33], where a Voronoi diagram is generated based on threat locations and optimised using SQP to minimise threat exposure. This Voronoi diagram is then searched using a graph search algorithm such as Dijkstra's algorithm to find the initial route. This route is then smoothed using cubic splines and refined into an optimal trajectory, subject to curvature constraints. The algorithm exploits the segmentation (decomposition) of the splines to perform sequential optimisation, thus relaxing the computational time constraint and allowing for near real-time implementation. The algorithm is therefore also able to quickly find a new collision free route in the presence of dynamic obstacles.

The use of a Genetic Algorithm (GA) is demonstrated in [34], where the algorithm uses B-splines to generate smooth trajectories. The algorithm can operate both off-line, for static environments, and on-line, for static unknown environments. The off-line mode considers curvature and height constraints, while the on-line mode only considers curvature. Due to the nature of GAs, these constraints may be exceeded, requiring increased iterations. There is also inherent uncertainty as to the optimality of the solution, as the GA may converge to either a global or local minimum.

In [35] the use of cubic B-splines in an off-line path planner is demonstrated to find trajectories that minimise exposure to enemy radar. Here the terrain is represented as a contour map, thus reducing the navigation to a 2D problem of finding a route through

static obstacles, i.e. terrain peaks that exceed the altitude threshold.

Another approach in [36] uses Particle Swarm Optimisation (PSO) to find multiple path solutions for a human operator to select from. The incorporation of B-spline curves allows the algorithm to optimise the various trajectory options to minimise radar exposure and reduce fuel consumption, while enforcing surveillance target way-point intersection. The improved analysis allows the human operator to make informed decisions based on the implications of each trajectory option.

In [37], optimal trajectories are generated over a 3D terrain using quartic (4th order) B-spline curves to ensure at least C^2 continuity. The algorithm combines convex optimisation with Disjunctive Programming (DP), incorporating a receding horizon concept for high computational speed. The use of DP relaxes the non-convex constraints, improving the algorithm's performance. The optimisation is subject to constraints on the maximum and minimum velocity, maximum acceleration and curvature to ensure feasibility.

To overcome this lack of an explicit dynamic feasibility check, many authors have presented hybrid approaches combining optimal control with splines. Some of these approaches still use geometric properties for the initial trajectory generation, using techniques such as the inverse dynamics for final feasibility validation.

A particularly interesting approach here is the exploitation of the differential flatness property of the system to compute the inverse dynamics along a given trajectory. It was demonstrated in [38] that it is possible to map the trajectory of constrained mechanical systems to curves in a lower dimensional space. Here non-linear optimal control is achieved using B-splines and Sequential Quadratic Programming (SQP) to generate aggressive trajectories in real-time. The inclusion of external dynamic effects such as wind was demonstrated in [39], optimising the power required during powered dynamic soaring to extend the flight endurance.

In [40], Mellinger and Kumar exploit the differential flatness of a quad-rotor model to generate minimum snap (4th derivative of position) trajectories. Since the snap is flat to the control inputs of the model, minimising the snap minimises the actuation cost for a

given manoeuvre while explicitly ensuring feasibility.

Yakimenko [41] presented a direct method he called Direct Method for Rapid Prototyping (DMRP) of trajectories, which exploits the differential flatness of the dynamics to generate near optimal trajectories. The method incorporates a timing law to decouple the spatial and temporal components, specifying the trajectory through specification of the position curve and thrust history. The trajectory is optimised as a non-linear programming (NLP) problem, allowing for any complex cost function based on boundary or event conditions. The approach is not limited in the dimensionality of the dynamics and hence doesn't require simplification of the state equations. The dimensionality of the state vector specification (position and thrust) does result in a sub-optimal trajectory as with all similar direct methods, but is shown to have limited deviation from the optimal solution.

A similar approach is used in [42] for the generation of trajectories for very agile multirotors using SCP optimisation. The kinematic model of the trajectory is used to enforce bounded jerk and acceleration, based on the assumption that this ensures attitude continuity which ensures feasibility. Violation of the feasibility is dealt with by increasing the flight time, i.e. reducing the average velocity. Dynamically decreasing the time step when boundary constraints are violated aides to resolve issues with the interpolation accuracy. An important observation made is that the decoupling of the complex vehicle dynamics and constraints, by rather opting for a simplified kinematic representation for trajectory generation, simplifies the planning problem. This results in a fast and efficient trajectory generator for finding collision-free trajectories, and checking dynamic feasibility a posteriori using a complex vehicle model can be performed quickly.

Chen et al. [43] demonstrated a multi-agent trajectory generation algorithm for navigation in environments with static obstacles. Optimisation of the trajectory is done using SCP to approximate non-convex constraints by convex counterparts, however this may result in a set of over-constrained optimisation problems and therefore the algorithm may fail to find a feasible solution. Therefore the constraints are introduced incrementally, called incremental SCP (ISCP), creating a sequence of more relaxed and feasible intermediate optimisation problems. The decoupling of the multi-agent optimisation into a collection

of single agent optimisation problems, greatly improves the computational tractability and also increases the odds of finding a feasible solution.

A comprehensive study in [16] present a single and multi-objective 3D terrain and threat-based trajectory planning algorithm, along with a MIMO non-linear MPC controller to track the reference trajectories. The hybrid trajectory generator optimises the trajectory for optimal flight time, fuel consumption and altitude, while using a high fidelity 6 DoF quaternion aircraft model with aerodynamic and propulsion model to ensure feasibility and evaluate performance. The trajectory is optimised using a global differential-evolution-based optimisation algorithm, which allows for both terrain following (TF) and terrain avoidance (TA). The algorithm repeatedly performs the sequence of generating a spline trajectory, performing a 6 DoF dynamic simulation, evaluating the performance to compute the cost function, and then optimising the trajectory through manipulation of the spline curve. The non-linear MPC guidance controller incorporates a neurofuzzy predictor model to continuously update the aircraft model and ensure accurate tracking of the reference trajectory.

These geometric-dynamic approaches also offer a great deal of versatility in their compatibility with various techniques, including other aspects of flight automation. Hybrid sampling based methods allow for integration with path planning algorithms such as Rapidly-exploring Random Trees (RRTs) [44], which combined with spline interpolation allows for the inclusion of vehicle kinematics. Due to the complexity of path planning in dynamic environments, especially when incorporating the dynamics of agile vehicles, the advantages of random sampling approaches was considered in [45]. It was shown that this hybrid approach was computationally efficient given the planar (2D) Autonomous Unmanned Ground Vehicle (AUGV) case considered, and allowed for the decoupling of the low-level control from the motion planning. This work was extended in [46] to the case of aircraft operating in a 3D environment, applying simple geometric constraints.

In [47] a RRT is used to find a suitable route in cluttered environments resulting in the initial way-point sequence. A pruning filter is then used to remove unnecessary way-points

to limit the number of segments, and the remaining way-points are then interpolated using cubic Bézier splines. A basic feasibility check is performed through a curvature constraint.

Another motion planner based on cubic Bézier splines for fixed wing UAVs is shown in [48]. The bidirectional spline offers directional constraint at both the start and end points, while feasibility checks are performed on load factor and thrust-to-weight ratio limits using geometric properties. Geometric feasibility is also performed by comparison to terrain height. This approach offers smooth directionally (C^1) continuous trajectories that are sub-optimal, although asymptotically optimal as the number of segments increases.

A decentralised approach to 2D ground based trajectory generation is presented in [49], where a multi-agent RRT motion planner achieves coordination through a merit-based token passing strategy. Priority during replanning is granted to the agent with the greatest potential gain. A cooperation strategy is also presented where an agent can alter a teammate's plan to optimise their combined cost, which improves team performance and avoids common deadlock scenarios. A common risk in cooperative planning is deadlock, which is when there are limited path options and two or more agents compete for a given position, route or resource.

Another hybrid sampling method based on A* using a multi-stage path prediction algorithm is shown in [50]. The multi-stage trajectory generation is done in four stages: path estimation, path planning, flyable trajectory generation and finally trajectory optimisation. A localised A* algorithm is used to estimate all probable paths, serving as the input to the task assignment algorithm. After task assignment, polygonal paths are generated using a global A* algorithm which are then smoothed using cubic B-splines to generate flyable trajectories. Finally the trajectories are optimised for rendezvous to improve task coordination. The entire multi-stage algorithm is continuously repeated to allow for operation in stochastic scenarios.

Although sampling based methods are excellent at finding paths through complex congested environments, they scale poorly to large 3D environments. Adding a time component to the grid for dynamic obstacle avoidance, adds another dimension which

can also have a drastic impact on the computational tractability. Therefore many sampling methods are run repetitively to accommodate dynamic environments, rather than explicitly including the time dimension. This approach limits the time horizon for which the resulting path is valid and hence adds unnecessary computation cost. To overcome this some authors use path planning to navigate static obstacles and coordinate the dynamic obstacles using the spline stage of the algorithm.

2.3 Critical Review

The optimal control approaches are subdivided into the direct and indirect methods, with the indirect methods being complex and generally impractical for implementation in software. The direct, or inverse dynamic, methods require discretisation of the problem space, making them computationally expensive and thus scale poorly to large complex environments. The optimality of the direct methods are also subject to the resolution of the trajectory due to the discrete nature of the approach, arguably making the result inherently sub-optimal given the trade-off between accuracy and computational cost.

The geometric-dynamic approach does not discretise the trajectory, rather binding the aircraft model to a lower dimensional space that can be represented by continuous polynomials. Although the resulting trajectory is sub-optimal, this approach is highly computationally efficient and accurate, while being universally applicable to all vehicle types. There are also various hybrid approaches that tend to blur the distinction between these two categories, but they can still be categorised by considering the starting point and philosophy behind each approach.

These spline based geometric-dynamic approaches are clearly well suited to the trajectory generation problem. By mapping the dynamic model to a lower dimensional space, this approach greatly reduces the computational complexity of trajectory generation. Since they only limit the dimensionality of the state vector's behaviour, the final feasibility and evaluation can still be performed using the full dimensionality of the inverse dynamic

model. This effectively exploits the property of NP-hard¹ problems in that a proposed solution can easily be verified. Although the resulting trajectory is sub-optimal, the composite trajectory is able to asymptotically approach the optimal solution due to the segmentation. Essentially each segment is reduced in dimensionality, but due to the concatenation of the segments the overall trajectory is not limited to the dimensionality of the individual segments.

The segmented nature adds the additional computational advantage of limiting the time horizon of each iteration, hence greatly reducing the computational time constraints as the segments can be evaluated and optimised in order of priority. Avoidance of suddenly appearing obstacles is also easily dealt with due to the locality of the spline, not requiring re-evaluation of the entire trajectory. This segmentation behaves similar to the computational advantages of manoeuvre based approaches, even allowing for prior specification of manoeuvres, but not limited to a finite manoeuvre library.

As demonstrated by the hybrid and distributed techniques, the geometry-dynamic approach not only integrates well with other algorithms, but also inherently allows for multi-stage algorithms. Successively increasing the complexity of the evaluation allows the algorithm to quickly omit strategies that are obviously infeasible. This greatly improves the computational tractability, since the complex feasibility and performance evaluation using the non-linear dynamic model is only performed once a suitable path proposal is found, as opposed to the optimal control approaches. Remaining computational resources can then be applied to optimise the trajectory by evaluating the dynamic performance, prioritising the segments based on execution order. Priority can then be shifted to the next segment as the improvement in the cost function decreases per iteration.

It is important to note that the geometric-dynamic approach does not supplant the optimal control approach, simply offering a computationally efficient means of evaluating various tactics to locate the general location of the global optimal solution. More advanced dynamic models, and even optimal control, can then be used for the final evaluation and optimisation. Since the geometric-dynamic approach is combined with an advanced

¹Non-deterministic Polynomial-time (NP) hard problems are a class of decision problems.

guidance controller, the optimal control can be incorporated during execution. In this architecture, the guidance controller is responsible for the robust tracking of the trajectory, thus finding the optimal actuation while accounting for disturbances. A good example can be found in [16], where MIMO non-linear MPC with a neurofuzzy model predictor is used to track the reference trajectory. Another good option that is gaining popularity is passivity based control as presented in [51].

The segmentation and multi-stage path planning means that this approach scales well to large 3D environments with numerous dynamic and static obstacles. Splines are also well suited to landmark based obstacle maps, hence the algorithm's computation load scales according to the complexity of the environment rather than its size. The segmentation of the trajectory also scales according to the complexity of the task assigned to the system, hence the computational requirements can be correlated to the complexity of the application and operating environment. This allows low cost systems with limited capabilities to coexist with more capable and expensive systems.

Chapter 3

Bézier Spline Trajectories

My life is one long curve, full of turning points.

– Pierre Trudeau

3.1 Introduction

This chapter introduces Bézier splines and discusses their suitability to the trajectory generation problem. The details of these splines are presented here to help keep the broader literature survey of chapter 2 brief. The various case studies presented in this chapter serve to demonstrate the intuitive structure and versatility of Bézier splines to the problem of trajectory generation. The special case of Pythagorean Hodograph (PH) curves is also introduced for their computational advantages.

3.2 Suitability of Bézier Splines

The primary objective of trajectory generation is finding some smooth path connecting the way-points, while ensuring that path is free of collisions and flyable. This inherently makes trajectory generation an interpolation problem, making the use of geometric interpolation techniques a logical approach. Both B-splines and Bézier splines have seen extensive use in

trajectory generation. B-splines offer explicit continuity, but have the limitation that they lose their locality as the order of the curve increases. Interestingly B-splines are analogous to a single high order Bézier spline, making segmented Bézier splines more versatile. The trade-off in computational cost between B-splines and segmented Bézier splines is a complex issue, depending on the order and structure of the problem. Ultimately the preserved locality of Bézier splines with high order curve continuity constraints, makes them preferable in dynamic and unknown environments where frequent replanning is needed. They are also better suited to sequential optimization, as manipulation of a segment between two way-points doesn't affect neighbouring segments.

Bézier splines were first introduced by Pierre Bézier in the 1960's and have since become a powerful tool in geometric CAD [52]. Bézier splines have become particularly popular in trajectory generation, as shown in [11, 47, 48, 53–59], for their intuitive geometric construction, efficient computation algorithms and locality properties. These advantageous properties include accurate and efficient geometric-dynamic evaluation, fast collision detection, and concise trajectory description for easier coordination. A more comprehensive overview of these properties and algorithms are given in Appendix A.

Bézier splines have long been used in trajectory generation for Autonomous Unmanned Ground Vehicles (AUGVs). Planar (2D) splines are used in [60, 61] to generate time-optimal trajectories that do not exceed the acceleration limits of the vehicle. This process of constraining the differential states of the system to ensure feasibility, is called kinodynamic motion planning. A global path planner is used to find the initial collision-free straight line paths that navigate the environment, connecting the start and desired end positions. This way-point sequence (straight path route) is then interpolated using quintic Bézier splines to achieve curvature continuity, i.e. smooth acceleration, in the knots. The position curve is temporally isolated using a separate velocity profile, and constrains both the velocity and acceleration along the trajectory. Optimisation is performed until the curve converges to an optimal solution, or the planning time is exceeded.

These Bézier trajectories offers smooth, precise and predictable motion, making them ideal for accurately predicting the system behaviour. This is exploited in [58] to generate

trajectories for AUGVs that play soccer. By incorporating a directional constraint at the point of impact with the ball, the direction of the ball can be controlled. The geometric intuitiveness of Bézier splines make them ideal for this application. In this case the optimisation is focussed on maximising the agility of the robots, while the kinodynamic constraints ensure feasibility. The computational efficiency of Bézier splines also allow for fast collision avoidance in this highly dynamic environment.

Applying these splines to the trajectory generation for UAVs requires the use of spatial (3D) curves. As stated in subsection 2.2.2, these geometric trajectories don't explicitly ensure feasibility. Therefore the feasibility must be evaluated both geometrically (collisions) and dynamically (flyable). In addition to curvature, the spatial case includes torsion, which defines the rate of rotation of the plane of curvature. This is however more complex than the 2D case, as the torsion is undefined when the curvature is zero and discontinuous (inverts) at inflection points. Although curvature and torsion do not directly relate to the vehicle dynamics, they can offer insight into the kinematic behaviour along the trajectory.

Feasibility is achieved if the trajectory is flyable and free of collisions, which is enforced through various kinematic and geometric constraints. Ensuring the trajectory is flyable requires relating the geometric properties of the curve to the kinematics of the vehicle. Simple spatial constraints such as position, altitude, terrain interference, and collisions with static obstacles are easily computed without invoking the timing law. Other geometric properties such as flight path angle and heading are also directly available from the curve tangent. Kinematic properties such as turn rate, climb rate, pitch rate, velocity, longitudinal and lateral acceleration, can be computed using the timing law and differential geometry. This is however still a limited evaluation due to the omission of the aerodynamic effects of angle of attack and side-slip, and hence these constraints are generally overly strict to achieve "implied" feasibility.

These spatial Bézier curves were used in [54] to generate feasible trajectories for UAVs, using the convex hull property for spatial deconfliction at a safe separation distance. Spatial boundaries were also included to limit the operational space of the vehicle.

This results in a constrained optimisation problem that scales well, with future intent to integrate the inverse dynamics. In [59] these same properties are used for collision prediction and avoidance, demonstrating the locality and computational advantages of segmented Bézier splines for replanning in dynamic environments. Dynamic feasibility is evaluated using simple geometric constraints by exploiting the geometric properties of the curves.

As stated before, Bézier curves integrate well with other techniques to form hybrid approaches, such as the integration with RRTs in [47, 48]. In [57] a paralleled multi-UAV Genetic Algorithm (GA) is used to find a collision free route, which is then smoothed using Bézier curves. Although the Bézier splines are not directly included in the optimisation in this case, since no kinematic constraints were considered, this can easily be achieved.

Although the construction of Bézier splines makes them an intuitive choice for trajectory generation, they do have some computational limitations. The biggest limitation for this application is that a standard Bézier cannot exactly describe a circle, therefore by extension it is impossible to construct a curve with unit or constant parametric speed. This is possible through the use of rational Bézier curves, however this was not considered as part of this study. The use of rational Bézier curves does however seem counter intuitive since the idea of having constant parametric speed is to reduce the complexity of the composite functions, as the relationship becomes linear. Similarly, as will be seen later, many of the functions describing the dynamics require scaling by the parametric speed, thus resulting in rational polynomials.

3.3 Problem Formulation

Spatial trajectory generation requires the interpolation of a sequence of N way-points $\mathbf{w}_i$, with $i \in [1..N]$. Each segment connecting way-points $[\mathbf{w}_i, \mathbf{w}_{i+1}]$ is interpolated using a spatial Bézier curve $\mathbf{p}_i(\zeta_i)$, resulting in $N - 1$ segments. The trajectory is constructed through the concatenation of these segments, with the knots coinciding with the way-points. This can easily be extended to M vehicles, each with its own unique way-

point sequence $(\mathbf{w}_{j,i})$ and individual trajectory $(\mathbf{p}_{j,i})$, where vehicle index $j \in [1..M]$ and segment index $i \in [1..N_j]$. The timing of each vehicle trajectory is also subject to its own start and end times, $t \in [t_j^s, t_j^f]$, with the timing law essentially coordinating the arrival time $(t_{j,i})$ at each way-point. Sequencing of the way-points is performed by the path planner to find the best route through the environment. A separate task assignment algorithm would ideally be used to assign tasks (target way-points) to vehicles that collaborate.

Trajectory generation is essentially the inverse of the aircraft kinematics, and the required parameters can be derived from the equations of translational motion in (3.1).

$$p_N(t) = v(t) \cos \gamma(t) \cos \psi(t) \quad (3.1a)$$

$$p_E(t) = v(t) \cos \gamma(t) \sin \psi(t) \quad (3.1b)$$

$$p_D(t) = -v(t) \sin \gamma(t) \quad (3.1c)$$

The motion is defined within a right handed north-east-down (NED) Cartesian coordinate system, consisting of the north position (p_N) , east position (p_E) and altitude (p_D) . Note that the altitude is downward positive. This forms the earth-centred, earth-fixed (ECEF) inertial reference frame used for navigation, with its origin on the earth's surface. The application of these splines to a geodetic coordinate system has not been considered here.

Two additional reference frames (both right-handed) are needed to evaluate the vehicle dynamics, i.e. the body and aerodynamic frames. The body frame is aligned with the aircraft's geometric centre, with the x-axis pointing forward, y-axis pointing right, and z-axis pointing down. For the purpose of navigation, there is no distinction being made between the geometric centre and the centre of mass (COM). The position of the body frame origin is the aircraft's position $(\mathbf{p})$, and its orientation (Θ_B) in relation to the ECEF frame is defined by the roll (ϕ) , pitch (θ) and yaw (ψ_B) angles. The right-handed rotational sequence used is z-y-x, i.e. yaw-pitch-roll. The aerodynamic (or wind) frame has its origin aligned with the body frame, but offset to body frame by the angle of attack (α) and side-slip angle (β) . The orientation (Θ_W) of the aerodynamic frame to the ECEF frame is critical for navigation, consisting of the roll angle (ϕ) , flight path angle (γ) and heading (ψ) .

Considering these frame definitions in relation to the equations of translational motion in (3.1), it is clear that the trajectory is most intuitively defined by specifying the velocity ($v(t)$), flight path angle ($\gamma(t)$) and heading ($\psi(t)$). Using a continuous path specification in the form of a position spline, defines these values through the path tangent, which is computed using the derivatives of the path curve. This however does not completely define the aircraft motion, requiring a fourth parameter to bind the model to a singular solution [12]. Generally this is resolved by limiting the aircraft to coordinated turns, i.e. $\beta(t) = 0$, however it is also possible to specify the roll angle ($\phi(t)$). Other parameter options can be used, however these are not considered here.

Specifying the roll angle allows for more complex manoeuvres, but results in some computational challenges that must be addressed. The drawback of specifying the roll angle is that the independent splines are not explicitly coupled mathematically, which results in unwanted behaviour when reaching the extremes in the Euler angles. Without proper optimisation of the roll angle, it results in a sub-optimal trajectory due to uncoordinated turns ($\beta(t) \neq 0$). The advantages of specifying the roll angle may surpass these drawbacks when considering the mission requirements. Specifying the orientation can aid in aligning sensors, weapons, and payload drops with targets. Other scenarios that would benefit include trajectories that require minimising radar cross-section, formation flight, in-flight refuelling and docking. Specifying the roll angle also allows the aircraft to perform more advanced aerobatic and combat manoeuvres, improving system agility and versatility.

The segmentation of the trajectory results in the post hoc constraint of ensuring continuity of the trajectory. The continuity can either be defined as being parametric (C) or geometric (G), the latter being of interest here due the parametric independence of the segments. Since the segments interpolate the way-points, the path is inherently G^0 continuous. The addition of directional constraints in the knots (way-points), in the form of tangents, ensures the path is directionally continuous, i.e. G^1 continuous. It is however common practice to ensure that the curve is G^2 (curvature) continuous, ensuring smooth directional changes. This second order continuity constraint relates to smooth acceleration changes when transitioning between segments. The continuity constraint can

be increased to higher order derivatives, however the additional computational cost offers diminished returns as the deviation in the spatial curve reduces with each order increase. The objective is to improve the tracking performance of the vehicle when transitioning between segments, however the impact of higher order continuity in relation to external disturbances are negligible. It would be more efficient to simply have an MPC guidance controller smooth out the transition by minimising the tracking error. These continuity constraints do however add to the computational cost, but the trade-off is improved efficiency during re-evaluation in dynamic environments.

3.3.1 Spatial Position Curve

Each segment of the concatenated trajectory is a spatial curve $\mathbf{p}_{j,i}(\zeta_i)$, i.e. 3 dimensional, consisting of north position, east position and altitude splines.

$$\mathbf{p}(\zeta) = \begin{bmatrix} p_N(\zeta) \\ p_E(\zeta) \\ p_D(\zeta) \end{bmatrix}$$

Each of these segment dimensions are defined using an n^{th} order Bézier spline, as defined in (3.2).

$$B(\zeta) = \sum_{k=0}^n \beta_k b_{k,n}(\zeta) \quad (3.2)$$

The Bernstein basis polynomials are defined as

$$b_{k,n}(\zeta) = \binom{n}{k} \zeta^k (1 - \zeta)^{n-k} \quad k \in [0..n] \quad \zeta \in [0, 1] \quad (3.3)$$

where $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ is a binomial coefficient, and the dimensionless parameter ζ defines the parametric progression of the spline. The control polygon, or control points, is defined as the set of $n + 1$ Bernstein or Bézier coefficients (β_k) that define the shape of the spline.

The spatial trajectory of each vehicle is thus defined as the continuous set of segments.

$$\mathbf{p}_{j,i}: [0, 1] \rightarrow \mathbb{R}^3 \quad \zeta_i \in [0, 1], \ i = [1..N_j] \quad (3.4)$$

3.3.2 Timing Law

In order to temporally orient the spatial curve $\mathbf{p}_i(\zeta_i)$ over the flight interval $t \in [t_j^s, t_j^f]$, the dimensionless parameter (ζ_i) must be related to time.

$$\mathbf{p}_{j,i}: [t_j^s, t_j^f] \rightarrow \mathbb{R}^3 \quad i = [1..N_j], \quad (3.5)$$

To achieve this, each way-point $(\mathbf{w}_{j,i})$ must be temporally located, either by specifying the time of arrival $(t_{j,i})$ or the desired velocity $(v_{j,i})$ at the way-point. From this the segment duration $(T_{j,i})$ can be derived, and the total flight time (T_j^Σ) computed using these relationships:

$$\begin{aligned} t_j^s &= t_{j,0} \\ t_{j,i} &= t_{j,i-1} + T_{j,i-1} \quad i = [1..N-1] \end{aligned} \quad (3.6)$$

$$\begin{aligned} t_j^f &= t_{j,N} \\ T_j^\Sigma &= t_j^f - t_j^s = \sum_{i=0}^{N-1} T_{j,i} \end{aligned} \quad (3.7)$$

When specifying the desired velocities, the segment duration can initially be computed using the average velocity and segment arclength $(l_{j,i})$.

$$T_{j,i} = \frac{2l_{j,i}}{v_{j,i} + v_{j,i+1}} \quad (3.8)$$

Relating the spline progression $(\zeta_{j,i})$ to time, or velocity, forms a dynamic relation of the form:

$$\theta_{j,i}(\cdot) = \frac{d\zeta_{j,i}}{dt} > 0. \quad (3.9)$$

This dynamic relation between the rate of spline progression and rate of time progression, is called the timing law.

The intuitive approach may be to define the progression of time as a function of the spline parameter ζ , binding the position spline and the timing law to the same parameter. This approach is equivalent to defining a velocity profile $(v_j(\zeta))$ along the curve, resulting in the form

$$\theta_{j,i}(\zeta_{j,i}) = \frac{dt}{d\zeta_i}, \quad \text{for } \zeta_i \in [0, 1]. \quad (3.10)$$

This does however result in the need to calculate the inverse of the timing law when computing the position at a specific time t , i.e.

$$\zeta_{j,i}(t) = \int_{t_i}^t \theta_{j,i}^{-1}(\tau) d\tau. \quad (3.11)$$

Requiring a closed-form solution to the inverse of the timing law means that it is limited to second order polynomials, since polynomials of higher order are not guaranteed to have real roots. It is critical that the timing law be a continuous function to ensure that analytical solutions can be found for the various composite functions used to evaluate the trajectory. Limiting the order of the timing law can be problematic as it reduces the DoF for the velocity profile, limits the parametric continuity, and thus places real limitations on this approach to path planning. It is therefore favourable to define the timing law as a function of time to simplify coordination, since time is universal for all aircraft and trajectories. This form is also more efficient during execution of the trajectory, as the control reference signals can be computed directly at fixed time intervals.

The timing law is then defined over the interval $t \in [t^s, t^f]$, with t^s the start and t^f the final arrival times for the trajectory.

$$\theta_{j,i}(t) = \frac{d\zeta_{j,i}}{dt}, \quad \text{for } t \in [t_j^s, t_j^f], \quad i \in [1..N_j - 1] \quad (3.12)$$

The relevant segment can be determined by solving for i given the interval constraint $t \in [t_{j,i}, t_{j,i+1}]$, and the progression along that segment is computed by integrating the timing law.

$$\zeta_{j,i}(t) = \int_{t_{j,i}}^t \theta_{j,i}(\tau) d\tau \quad \text{for } t \in [t_{j,i}, t_{j,i+1}] \quad \text{where } \zeta_{j,i} \in [0, 1] \quad (3.13)$$

This omits the need for the inverse function, allowing for the use of higher order polynomial functions in the timing law. The trajectory $\mathbf{p}(t)$ of vehicle j is then found through concatenation of the segments defined by the composite function:

$$\mathbf{p}_{j,i}(t) = (\mathbf{p}_{j,i} \circ \zeta_{j,i})(t) = \mathbf{p}_{j,i}(\zeta_{j,i}(t)) \quad \text{for } t \in [t_j^s, t_j^f], \quad i \in [1..N_j], \quad j \in [1..M]. \quad (3.14)$$

3.3.3 Boundary Conditions

The construction of each segment is done through specification of the boundary conditions at the knots, to ensure continuity between segments. These constraints¹ in each knot include the way-point ($\mathbf{w}_i$), and the velocity vector de-constructed into its components, i.e. velocity (v_i), heading (ψ_i) and the flight path angle (γ_i), as follows:

$$\mathbf{p}_i(\zeta_i = 0) = \mathbf{w}_i \quad \gamma_i(\zeta_i = 0) = \gamma_i \quad \psi_i(\zeta_i = 0) = \psi_i \quad v_i(\zeta_i = 0) = v_i \quad (3.15a)$$

$$\mathbf{p}_i(\zeta_i = 1) = \mathbf{w}_{i+1} \quad \gamma_i(\zeta_i = 1) = \gamma_{i+1} \quad \psi_i(\zeta_i = 1) = \psi_{i+1} \quad v_i(\zeta_i = 1) = v_{i+1} \quad (3.15b)$$

Each segment is also bounded by the time interval $t \in [t_i, t_{i+1}]$, where the complete progression of ζ_i is $\zeta_i(t_i) = 0$ and $\zeta_i(t_{i+1}) = 1$. This range constraint, $\zeta_i \in [0, 1]$, must be enforced through an integration constraint on the timing law $\theta_i(t)$ to ensure correct correlation between the timing law and spatial curve. Since the start time of each segments is dependent on the end time of the previous segment, it is computationally advantageous to replace these arrival times with the interval duration $T_i = t_i^f - t_i^s$, and using only the relative normalized time in each segment, i.e. $\hat{t}_i = [0, 1]$, for the evaluation of the dynamics.

A consequence of the separation between the spatial curve and the timing law, is that the boundary conditions must also be separated. Therefore the velocity cannot directly be used to construct the spatial curve, rather using the unit tangent vector aligned with the velocity vector and a scaling parameter called the tangent magnitude (d_i). This tangent magnitude can be split between segments, i.e. as long as the unit tangents (direction) are the same the scaling can be different in a knot. This difference in the tangent magnitude must also be accounted for when enforcing G^2 continuity. The spatial curve is then bounded by the constraints $(\mathbf{w}_i, \gamma_i, \psi_i, d_i^s)$ at the segment start, and $(\mathbf{w}_{i+1}, \gamma_{i+1}, \psi_{i+1}, d_i^f)$ at the segment end. The segment is still bounded by the interval $t \in [t_i, t_{i+1}]$, which is derived from the velocities v_i and v_{i+1} and the segment arclength l_i .

¹Equations are presented for a single vehicle for the ease of reading.

3.4 Pythagorean Hodograph Curves

One particular drawback of spatial Bézier curves is the lack of a closed form solution to the arclength integral, and by association the parametric speed. The accurate and efficient computation of the arclength is vital to optimisation, while the parametric speed is critical in relating the curve geometry and timing law to the vehicle kinematics. Applying numerical techniques or polynomial approximations increases the computation cost and reduces the accuracy of the trajectory evaluation.

The parametric speed of the spatial curve is defined as the Euclidean (L^2) norm of the path derivative, as shown in (3.16).

$$\|\mathbf{p}'(\zeta)\| = \sqrt{p_N'^2(\zeta) + p_E'^2(\zeta) + p_D'^2(\zeta)} \quad (3.16)$$

The arclength progression of the spatial curve is computed using the integral of the parametric speed in (3.17), with the segment arclength being $l_i = s_i(1)$.

$$s_i(\zeta_i) = \int_0^{\zeta_i} \|\mathbf{p}'(\tau)\| d\tau \quad (3.17)$$

The velocity of the aircraft is then computed using the composition of the timing law and parametric speed.

$$v_i(t) = \|(\mathbf{p}'_i \circ \zeta_i)(t)\| \theta_i(t) \quad (3.18)$$

Remark. *The parametric speed is constrained to be positive, making the parameter mapping $\zeta_i(\hat{t}_i)$ monotonically increasing. This guarantees the convergence of numerical techniques, such as Newton-Raphson root finding, used to compute the spline index. Computation of the arclength is generally done using numerical integration techniques such as Gauss-Legendre quadrature integration.*

Many approaches have been suggested to address this issue, such as re-parametrisation of the curve or subdivision into segments of unit arclength [62–64]. Farouki et al. introduced in [65] a subset of spatial curves called Pythagorean Hodograph (PH) curves, where the

curves are limited to those curves that have a polynomial solution to the parametric speed. In this form, the spatial Bézier curve has one dimensional polynomial solution to the parametric speed which is defined as an explicit Bézier spline. The Pythagorean Hodograph (PH) curve may seem overly complex at first glance, but the computational advantages of the form is exceedingly valuable, offering exact solutions to the parametric speed and arclength, as shown in [11, 66–69].

Pythagorean Hodograph (PH) curves are a unique subset of polynomial curves where the hodograph is specified as part of the curve structure, ensuring the parametric speed has polynomial form $\sigma(\zeta)$.

$$\sigma^2(\zeta) = x'^2(\zeta) + y'^2(\zeta) + z'^2(\zeta) \quad (3.19)$$

Essentially these curves are constructed by specifying the hodograph curve $\mathbf{p}'(\zeta)$, and limiting its construction to polynomials that satisfy the Pythagorean structure of (3.19). By using the linear relationship between the spatial curve and its hodograph, a property from the use of Bézier splines, it can be ensured that the path interpolates the way-points. For these curves the parametric speed is reduced to the Bézier polynomial $\sigma(\zeta) = \|\mathbf{p}'(\zeta)\|$, and the arc length is now reduced to the integral of this polynomial, as follows:

$$s(\zeta) = \int_0^\zeta \sigma(\tau) d\tau \quad (3.20)$$

3.4.1 Pythagorean Quadruple

The caveat of spatial PH curves is that there is no linearly independent polynomial triplet that satisfies (3.19). This is analogous to the construction of quaternions, a relationship that is exploited in the construction of PH curves. The Pythagorean hodograph condition is then satisfied using the quadruple of real polynomials $u(\zeta)$, $v(\zeta)$, $p(\zeta)$ and $q(\zeta)$ in (3.21).

$$x'(\zeta) = u^2(\zeta) + v^2(\zeta) - p^2(\zeta) - q^2(\zeta) \quad (3.21a)$$

$$y'(\zeta) = 2[u(\zeta)q(\zeta) + v(\zeta)p(\zeta)] \quad (3.21b)$$

$$z'(\zeta) = 2[v(\zeta)q(\zeta) - u(\zeta)p(\zeta)] \quad (3.21c)$$

$$\sigma(\zeta) = u^2(\zeta) + v^2(\zeta) + p^2(\zeta) + q^2(\zeta) \quad (3.21d)$$

This form is however quite tedious and a more concise method for constructing PH spatial curves can be found in the quaternion representation of (3.21). In review, a quaternion $\bar{\mathcal{A}} = a + a_x\hat{\mathbf{i}} + a_y\hat{\mathbf{j}} + a_z\hat{\mathbf{k}}$, is a quadruple structure consisting of a scalar and vector part. Quaternions with a zero scalar part are called pure vector quaternions. Quaternion algebra is associative, but non-commutative, and the conjugate of a quaternion ($\bar{\mathcal{A}}^*$) can be found by taking the negative of the vector part, i.e. $\bar{\mathcal{A}}^* = a - a_x\hat{\mathbf{i}} - a_y\hat{\mathbf{j}} - a_z\hat{\mathbf{k}}$. Quaternions are commonly used to perform vector rotations and has a direct correlation to the direction cosine matrix (DCM), where the unit quaternion $\mathcal{U}$ can be defined as a rotation by an angle δ around a unit vector $\mathbf{n}$ as axis of rotation, i.e. $\mathcal{U} = \cos \frac{1}{2}\delta + \sin \frac{1}{2}\delta\mathbf{n}$. The rotation of a vector $\mathbf{v}$ using these parameters is thus achieved through the double quaternion product $\mathcal{U}\mathbf{v}\mathcal{U}^*$, resulting in a pure vector quaternion. The non-commutative nature of quaternions becomes apparent when considering rotation, as the sequence of rotations determines the resulting orientation.

The hodograph of the spatial curve can now be constructed through the double quaternion product, which is essentially a Hopf map projection of the 4D quaternion onto 3D space. This is similar to the quaternion rotation, with the key difference being that the quaternion is not of unit length². The double product has the effect of scaling the resulting polynomial in relation to the parametric speed $\sigma(\zeta)$. The hodograph then takes the form

$$\mathbf{p}'(\zeta) = \mathcal{A}(\zeta)\hat{\mathbf{i}}\mathcal{A}^*(\zeta), \quad (3.22)$$

where the spatial curve $\mathbf{p} = p_N\hat{\mathbf{i}} + p_E\hat{\mathbf{j}} + p_z\hat{\mathbf{k}}$ is found through the integration of the Bézier hodograph curve. The polynomial form of the quaternion $\bar{\mathcal{A}}$ is thus constructed using the

²The double quaternion product uses the quaternion conjugate rather than the inverse to preserve the scaling.

quadruple of Bézier splines, yielding the polynomial in (3.23).

$$\mathcal{A}(\zeta) = u(\zeta) + v(\zeta)\hat{\mathbf{i}} + p(\zeta)\hat{\mathbf{j}} + q(\zeta)\hat{\mathbf{k}} \quad (3.23)$$

It can now be shown that the quaternion product yields the result in (3.24), thus satisfying the Pythagorean hodograph condition in (3.21).

$$\begin{aligned} \mathcal{A}(\zeta)\hat{\mathbf{i}}\mathcal{A}^*(\zeta) &= [u^2(\zeta) + v^2(\zeta) - p^2(\zeta) - q^2(\zeta)]\hat{\mathbf{i}} \\ &\quad + 2[u(\zeta)q(\zeta) + v(\zeta)p(\zeta)]\hat{\mathbf{j}} \\ &\quad + 2[v(\zeta)q(\zeta) - u(\zeta)p(\zeta)]\hat{\mathbf{k}} \end{aligned} \quad (3.24)$$

It is also clear now that the arc length integral is no longer needed due to the Pythagorean structure of the polynomials.

$$\sigma(\zeta) = \mathcal{A}(\zeta)\mathcal{A}^*(\zeta) = \|\mathbf{p}'(\zeta)\| > 0 \quad (3.25)$$

This also reduces the complexity of the kinodynamic equations, such as the velocity profile:

$$v(t) = \|(\mathbf{p}' \circ \zeta)(t)\|\theta(t) \quad (3.26)$$

$$= (\sigma \circ \zeta)(t)\theta(t) \quad (3.27)$$

Beyond the computational advantages of the polynomial form of the parametric speed and arclength, the PH curves offer an efficient means of evaluating the smoothness of the curves. The PH curve form allows for the closed form computation of the bending energy of a curve, in (3.28), a metric that combines the curvature and torsion in a similar fashion to computing the stored elastic bending energy in a rod.

$$w(t) = \sqrt{\kappa^2(t) + \tau^2(t)} \quad (3.28)$$

The closed form solutions are however only available for lower order PH curves, i.e. cubic and quintic spatial curves. A similar approach is however presented in this study for higher order curves.

3.4.2 PH Curve Construction

One of the challenges in the use of quaternions for the construction of spatial PH curves is the additional degree of freedom being introduced by the fourth polynomial, that is not

bound by the boundary conditions of the previous section. Consider the simple case of the quaternion rotation

$$\mathcal{A}\hat{\mathbf{i}}\mathcal{A}^* = c, \quad (3.29)$$

which has an infinite number of rotations that satisfy this condition. This can be reduced to a one-parameter family of solutions to (3.29) of the form

$$\mathcal{A}(\phi) = \sqrt{|\mathbf{c}|} \frac{\frac{\mathbf{c}}{|\mathbf{c}|} + \hat{\mathbf{i}}}{|\frac{\mathbf{c}}{|\mathbf{c}|} + \hat{\mathbf{i}}|} \left(\cos \phi + \hat{\mathbf{i}} \sin \phi \right) \quad \text{with } \phi \in [0, 2\pi). \quad (3.30)$$

Remark. The term $\frac{\mathbf{c}}{|\mathbf{c}|} + \hat{\mathbf{i}}$ is the unit bisector of $\mathbf{c}$ and $\hat{\mathbf{i}}$, and is not uniquely defined when $\frac{\mathbf{c}}{|\mathbf{c}|} = -\hat{\mathbf{i}}$. In this case the solution converges to

$$\mathcal{A}(\phi) = \sqrt{|\mathbf{c}|} (\hat{\mathbf{j}} \cos \phi - \hat{\mathbf{k}} \sin \phi) \quad \text{with } \phi \in [0, 2\pi). \quad (3.31)$$

Similarly the quaternion product of the form $\mathcal{A}\hat{\mathbf{i}}\mathcal{B}^* = \mathbf{c}$ can be reduced to the one-parameter family of solutions

$$\mathcal{A}(\tau) = -\frac{(\tau + \mathbf{c})\mathcal{B}\hat{\mathbf{i}}}{|\mathcal{B}|^2} \quad \text{with } \tau \in \mathbb{R}. \quad (3.32)$$

The formulation of the spatial Pythagorean hodograph curve now lies in the construction of the polynomial quadruple $u(\zeta)$, $v(\zeta)$, $p(\zeta)$ and $q(\zeta)$, once again using Bézier splines for each polynomial. The order of the Pythagorean hodograph is double that of the polynomial quadruple $\mathcal{A}(\zeta)$ as a result of the double quaternion product, and thus the spatial curve $\mathbf{p}(\zeta)$ is of order $2m + 1$ given a polynomial $\mathcal{A}(\zeta)$ of order m .

Figure 3.1 shows the variance in a spatial curve constructed from a PH curve due to the additional free parameters.

3.4.3 Case studies

The computational advantages of PH curves have made them popular in trajectory generation. In [70] the planar case is presented, exploiting the low curvature achieved

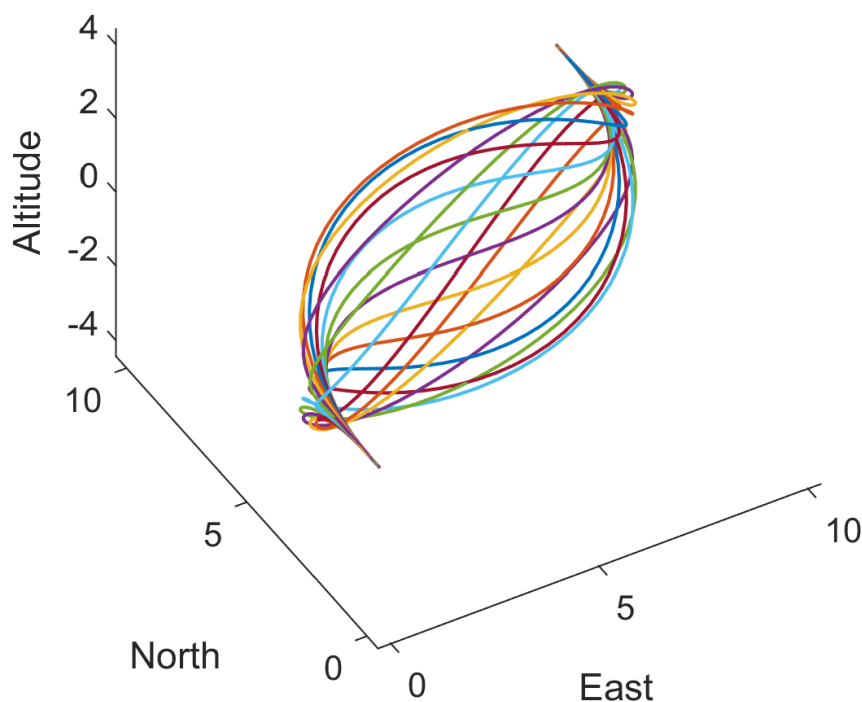


Figure 3.1: PH curve variance for free parameters

through efficient computation of curve length and bending energy. The trajectories are constructed with directional constraints in the knots, and optimised to balance the curvature and arclength. Planar PH curves are however substantially less complex than the spatial case, only resulting in four possible solutions due to the additional DoF, which can easily be evaluated to find the best solution. This demonstrates the advantage of PH curves in offering a purely analytical solution, thus introducing no numerical errors in the evaluation of the trajectory. Another planar example can be found in [66], where multiple trajectories are coordinated to achieve simultaneous time of arrival. The trajectories are subject to curvature constraints, while offering no separation between the spatial and temporal aspects of the trajectory.

The spatial PH curve, as stated before, is more complex due to the additional dimensionality. While the planar case is analogous to complex numbers, the spatial case is based on quaternions. This additional dimension introduces a far greater variance in possible solutions, which are computationally more complex to evaluate to identify the optimal solution.

A variation on the PH curve construction is used in [71], where a quadratic quaternion polynomial is constructed. Rather than directly computing a quintic (5th order) spatial curve, the control points are duplicated at the knots, ensuring the curvature in the knots are zero. This additional constraint results in a septic (7th order) spatial curve. The bending energy function is also expanded to include the climb angle, and by normalising the values in reference to the desired maxima, offers a single cost function for optimisation, as shown in (3.33).

$$w(t) = \sqrt{\left(\frac{\kappa(t)}{\kappa_{max}}\right)^2 + \left(\frac{\tau(t)}{\tau_{max}}\right)^2 + \left(\frac{\gamma(t)}{\gamma_{max}}\right)^2} \quad (3.33)$$

Cooperative trajectory generation based on quintic PH curves can be seen in [68], subject to curvature, torsion and separation distance constraints. Additional cooperative constraints are applied to ensure overlap between the camera field of view (FOV) for neighbouring UAVs, simultaneous time of arrival through path length equality and communication range separation limits. The trajectories consist of single segments, where the curve geometry is manipulated to avoid static obstacles.

Both these examples, although being only single segment trajectories, would only result in G^1 continuity if segments were concatenated. The addition of curvature (G^2) continuity constraints to construct segmented trajectories, requires the use of septic (7th order) spatial curves. This was done in [56], while enforcing curvature, torsion and climb angle constraints to ensure feasibility, although no obstacles were included. The computational advantages of using analytical curves, such as Bézier curve, was noted here for its increased flexibility in trajectory generation at a low computational cost.

Choe presented in [11] an architecture for distributed optimisation of cooperative single segment trajectories, based on quintic spatial curves combined with a quadratic timing law. Using the convex hull property in conjunction with the Gilbert-Johnson-Keerthi (GJK) distance algorithm, multi-vehicle trajectories can efficiently be evaluated for collisions. The minimum distance algorithm is also modified to serve as an efficient means of computing the global extrema of scalar Bézier polynomials. Although quintic spatial curves are popular in trajectory generation for offering curvature (G^2) continuity, this

property does not hold true for PH curves, instead requiring at least septic (7th order) spatial curves.

3.5 Conclusion

As observed by Choe in [11], these geometric-dynamic approaches allow for seamless integration with time-critical cooperative path-following algorithms, even when distributed throughout a larger cooperative framework. Dynamic constraints are also easily expressed in terms of the trajectory as a result of the differential flatness property of the system. The PH structure offers an efficient and accurate means of computing these dynamic constraints along the curve, including time derivatives. The Bézier formulation of the trajectory also allows the problem to be formulated as a finite-dimensional non-smooth optimisation problem. The advantage of using the Bézier polynomials for the trajectory regarding optimisation, lies in that the whole sub-differential of the non-smooth functions can easily be determined. The compact and accurate representation of the trajectory allows for efficient coordination during distributed optimisation, which can be done using existing non-smooth optimisation algorithms. The use of GJK on the convex hull allows for fast collision detection, making this approach scale well to scenarios with large numbers of vehicles, obstacles and trajectory segments. The ability to define rational offset curves makes the generation of formation flight trajectories much easier, only needing to define the primary trajectory and subsequently evaluating the offset curves for feasibility.

In conclusion, the geometric-dynamic approach based on PH Bézier splines offers an intuitive, efficient and concise means of accurately defining trajectories. These trajectories are versatile and universally applicable to almost any vehicle, while allowing for efficient integration with more complex control and evaluation techniques based on the required application. This makes this approach particularly useful in scenarios where various different vehicles must coordinate and collaborate, while also being ideally suited to distributed optimisation.

Chapter 4

Trajectory Generation

Do not go where the path may lead, go instead where there is no path and leave a trail.

– Ralph Waldo Emerson

4.1 Introduction

In this chapter the concepts of curve framing are introduced and related to the problem of trajectory generation. These concepts are then applied to the quaternion PH to generate trajectories that directly include the rotational kinematics of the vehicle.

4.2 Curve Framing

The 4 Dimensional Trajectories (4DTs) introduced in the previous chapter, consisting of spatial and time components, are only representative of the kinematics of a point-mass. The aircraft model contrastingly is based on a 6-DoF rigid body motion construct. It is therefore necessary to incorporate the angular motion of the body to predict the orientation of the aircraft along the trajectory. Due to the asymmetric nature of the aircraft's dynamics, the orientation of the aircraft greatly affects its capabilities, i.e. the

acceleration limits in the earth frame are effectively dependent on the orientation of the aircraft. In order to describe the orientation of the aircraft along the trajectory, a collection of techniques for curve framing are considered.

Any ordered set $E = \{\mathbf{e}_1(\zeta), \mathbf{e}_2(\zeta), \mathbf{e}_3(\zeta)\}$ of mutually orthogonal unit vector fields on a spatial curve $\mathbf{p}(\zeta)$ constitutes a frame. Through the use of differential geometry the construction of the frame can be related to the development of the curve. The frame E is called an adapted frame if $\mathbf{e}_1(\zeta)$ is the tangent of the curve, i.e. $\mathbf{e}_1(\zeta) = \frac{\mathbf{p}'(\zeta)}{|\mathbf{p}'(\zeta)|}$, and the span of $\{\mathbf{e}_2(\zeta), \mathbf{e}_3(\zeta)\}$ constitutes a normal plane to the curve.

The direction cosine matrix (DCM), which is also used to describe the orientation of an aircraft, is an example of such a frame consisting of the set $C_{e/b} = \{\mathbf{T}, \mathbf{N}_1, \mathbf{N}_2\}$. Using this association with the direction cosine matrix (DCM), which is known to be mathematically equivalent to a quaternion rotation, it becomes clear that any frame can be completely described using a quaternion, rather than the overloaded vector triple. This becomes clear when considering $\mathbf{N}_2 = \mathbf{T} \times \mathbf{N}_1$, and $\mathbf{N}_1$ only adds a single dimension of information, i.e. the rotation of the normal plane around $\mathbf{T}$.

The general development of a frame can be expressed as a state space model with the derivative in relation to arc length, as is shown in (4.1). The skew-symmetric form of the Cartan connection matrix is due to the orthogonal frame constraints $|\mathbf{T}| = |\mathbf{N}_1| = |\mathbf{N}_2| = 1$ and $\mathbf{T} \cdot \mathbf{N}_1 = \mathbf{T} \cdot \mathbf{N}_2 = \mathbf{N}_1 \cdot \mathbf{N}_2 = 0$.

$$\begin{bmatrix} \dot{\mathbf{T}} \\ \dot{\mathbf{N}}_1 \\ \dot{\mathbf{N}}_2 \end{bmatrix} = \begin{bmatrix} 0 & -\gamma & \beta \\ \gamma & 0 & -\alpha \\ -\beta & \alpha & 0 \end{bmatrix} \begin{bmatrix} \mathbf{T} \\ \mathbf{N}_1 \\ \mathbf{N}_2 \end{bmatrix} \quad (4.1)$$

This is equivalent to *Poisson's kinematic equations* which describe the development of the DCM in relation to the body frame angular rates, with $\alpha = P$, $\beta = Q$ and $\gamma = R$.

Equation (4.2) shows the quaternion form of this system.

$$\begin{bmatrix} \dot{q}_0 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 0 & -\alpha & -\beta & -\gamma \\ \alpha & 0 & \gamma & -\beta \\ \beta & -\gamma & 0 & \alpha \\ \gamma & \beta & -\alpha & 0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}. \quad (4.2)$$

Using the respective primes, $\frac{d}{ds} = \frac{1}{\sigma(\zeta)} \frac{d}{d\zeta}$, these equations can be differentiated with respect to an arbitrary parametrisation ζ , yielding the relation

$$q'(\zeta) = \frac{\sigma(\zeta)}{2} \Omega q(\zeta), \quad (4.3)$$

with Ω the skew-symmetric matrix form of the total angular velocity vector $\boldsymbol{\omega}$, where $\boldsymbol{\omega} = \alpha \mathbf{T} + \beta \mathbf{N}_1 + \gamma \mathbf{N}_2$.

4.2.1 Relatively Parallel Adapted Frame

A generalised construction of rational adapted frames was introduced in [72]. The development of the frame can be described using the state equation (4.4).

$$\begin{bmatrix} \dot{\mathbf{T}} \\ \dot{\mathbf{N}}_1 \\ \dot{\mathbf{N}}_2 \end{bmatrix} = \begin{bmatrix} 0 & k_1 & k_2 \\ -k_1 & 0 & 0 \\ -k_2 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{T} \\ \mathbf{N}_1 \\ \mathbf{N}_2 \end{bmatrix} \quad (4.4)$$

It is worth noting that the frame can be derived from the development of the curve tangent ($\mathbf{T}$), given an initial value for $\mathbf{N}_1$. Equally, the curve can be derived from the frame given the progression of $k_1(s)$ and $k_2(s)$, along with the initial orientation.

The general quaternion form for relatively parallel adapted frames (RPAFs) is shown in (4.5).

$$\begin{bmatrix} \dot{q}_0 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = \frac{\sigma}{2} \begin{bmatrix} 0 & 0 & -k_2 & -k_1 \\ 0 & 0 & k_1 & -k_2 \\ k_2 & -k_1 & 0 & 0 \\ k_1 & k_2 & 0 & 0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \quad (4.5)$$

Figure 4.1 shows a basic RPAF constructed using the parallel transport method [73], which generates a frame based purely on the development of the tangent in relation to some initial frame orientation.

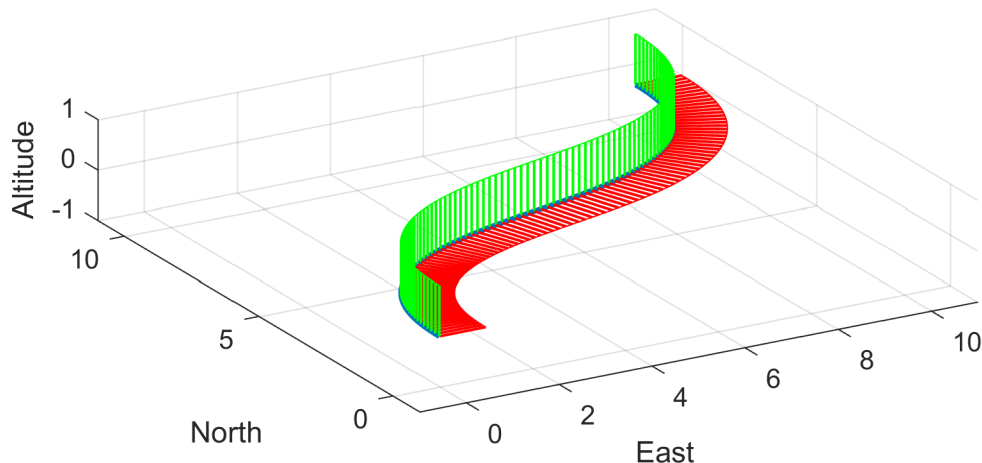


Figure 4.1: Relatively parallel adapted frame (RPAF)

4.2.2 Frenet-Serret Frame

One particularly indispensable frame from differential geometry is the Frenet Frame, that is directly related to the curve kinematics. The frame is constructed using the Frenet-Serret formulas creating the vector field $E^F = \{\mathbf{T}, \mathbf{N}_p, \mathbf{B}\}$, consisting of the tangent $\mathbf{T}$, principal normal $\mathbf{N}_p$ and the binormal $\mathbf{B}$. The tangent indicates the direction of travel, while the principal normal is related to the centripetal acceleration which is the change in direction. The following equations can be derived to determine the Frenet frame for a curve with arbitrary parametrisation.

$$\mathbf{T}(\zeta) = \frac{d\mathbf{p}}{ds} = \frac{\mathbf{p}'(\zeta)}{\|\mathbf{p}'(\zeta)\|} \quad (4.6)$$

$$\mathbf{N}_p(\zeta) = \frac{\frac{d\mathbf{T}}{ds}}{\|\frac{d\mathbf{T}}{ds}\|} = \frac{\mathbf{p}'(\zeta) \times (\mathbf{p}''(\zeta) \times \mathbf{p}'(\zeta))}{\|\mathbf{p}'(\zeta)\| \|\mathbf{p}''(\zeta) \times \mathbf{p}'(\zeta)\|} \quad (4.7)$$

$$\mathbf{B}(\zeta) = \mathbf{T}(\zeta) \times \mathbf{N}_p(\zeta) = \frac{\mathbf{p}'(\zeta) \times \mathbf{p}''(\zeta)}{\|\mathbf{p}'(\zeta) \times \mathbf{p}''(\zeta)\|} \quad (4.8)$$

Of particular interest here is that the Frenet frame is an adapted frame that is uniquely

defined on a given curve, whose state-space equation is shown in (4.9).

$$\begin{bmatrix} \dot{\mathbf{T}} \\ \dot{\mathbf{N}}_p \\ \dot{\mathbf{B}} \end{bmatrix} = \begin{bmatrix} 0 & \kappa & 0 \\ -\kappa & 0 & \tau \\ 0 & -\tau & 0 \end{bmatrix} \begin{bmatrix} \mathbf{T} \\ \mathbf{N}_p \\ \mathbf{B} \end{bmatrix} \quad (4.9)$$

Both the frame and the curve is defined by this system which is dependent on two parameters, curvature (κ) and torsion (τ). Again, reducing the frame equation to quaternion form yields the equation in (4.10).

$$\begin{bmatrix} \dot{q}_0 \\ \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{bmatrix} = \frac{\sigma}{2} \begin{bmatrix} 0 & -\tau & 0 & -\kappa \\ \tau & 0 & \kappa & 0 \\ 0 & -\kappa & 0 & \tau \\ \kappa & 0 & -\tau & 0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \quad (4.10)$$

The Frenet frame can be related to the general RPAF using these equations:

$$\begin{aligned} \kappa(s) &= \sqrt{(k_1(s))^2 + (k_2(s))^2} \\ \theta(s) &= \arctan\left(\frac{k_2(s)}{k_1(s)}\right) \\ \tau(s) &= \frac{d\theta}{ds} \end{aligned}$$

Figure 4.2 shows the Frenet-Serret frame of a curve. Note the discontinuity in the frame at the inflection point, which results from the unsigned nature of the curvature.

Curvature

Curvature (κ) is a measure of the deviation of the curve from a straight line. It is defined as the reciprocal of the radius of the osculating circle, which is a tangent circle at a point along a curve. For the Frenet frame, the curvature describes the rate of rotation of the tangent vector around the binormal vector.

$$\kappa = \frac{1}{R} = \left\| \frac{d\mathbf{T}}{ds} \right\| \quad (4.11)$$

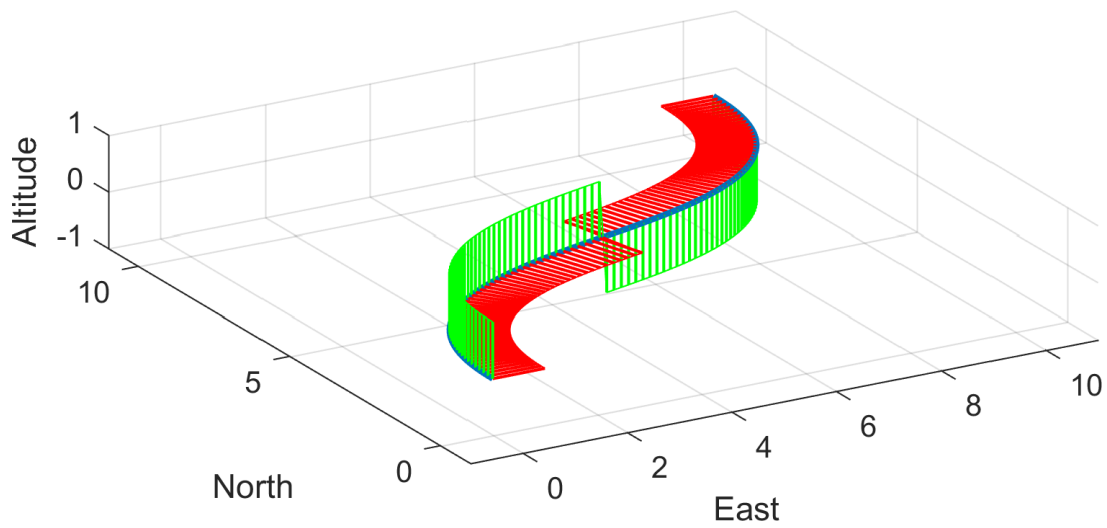


Figure 4.2: Frenet-Serret Frame of a curve

For a curve not parametrised by arc length, we can determine the curvature using (4.12).

$$\kappa(\zeta) = \frac{\|\mathbf{p}'(\zeta) \times \mathbf{p}''(\zeta)\|}{\|\mathbf{p}'(\zeta)\|^3} \quad (4.12)$$

Figure 4.3 shows the curvature of the curve in figure 4.2, noting that the curvature is unsigned and unable to distinguish the direction of the turn.

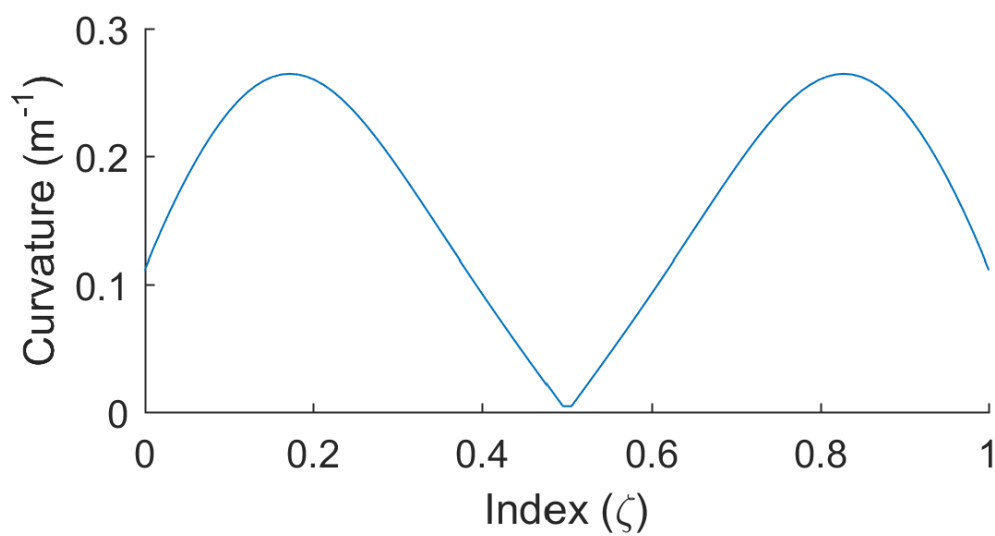


Figure 4.3: Curvature of the curve in figure 4.2

Torsion

Torsion (τ) is the rate of rotation of the osculating plane, i.e. the plane in which the curve is bending due to the curvature. In the case of the Frenet frame, the torsion describes the rate of rotation of the binormal vector around the tangent vector. The torsion can be computed using (4.13) for an arbitrarily parametrised curve.

$$\tau(\zeta) = \frac{(\mathbf{p}'(\zeta) \times \mathbf{p}''(\zeta)) \cdot \mathbf{p}'''(\zeta)}{\|\mathbf{p}'(\zeta) \times \mathbf{p}''(\zeta)\|^2} \quad (4.13)$$

Limitations of the Frenet-Serret Frame

The Frenet frame does however have a few limitations that makes its use cumbersome for orientation analysis. This is the result of the curvature not having a sign, i.e. equivalent to $\kappa = |\gamma|$. Since the two normal vectors are determined by the curvature and torsion, the Frenet frame is undefined when the curvature is zero, i.e. when the curve straightens out. These curves are referred to as degenerate curves. Furthermore the Frenet frame experiences inversion at inflection points along the curve to compensate for the lack of signed curvature, resulting in large spikes in torsion and discontinuities in the frame. In order to address these issues the Beta frame was introduced in [74]. The definition of the frame follows an inverse approach, by defining the signed curvature in reference to an initial value for the principle normal vector, i.e. $\dot{\mathbf{T}} = \kappa \mathbf{N}$. In this approach the signed curvature indicates the direction of the turning motion and the frame is defined on degenerate curves and suffers no discontinuities at inflection points. The RPAF frame in figure 4.1 is actually the Beta frame, since the initial frame orientation was based on the Frenet frame. Comparing figure 4.1 with figure 4.2, it is clear that the discontinuity at the inflection point is removed due to the ability to define signed curvature.

4.2.3 Euler-Rodrigues Frame

An interesting property of the spatial PH curve evolves from the curve being constructed from the hodograph, therefore the tangent is simply the normalised hodograph. The

quaternion form of the spatial PH curve hints to this relationship, since the hodograph is directly constructed from a series of non-unit vector rotations. Thus consider the sequence of rotations defined by the unit quaternion $\mathcal{U}(\zeta) = \frac{\mathcal{A}(\zeta)}{|\mathcal{A}(\zeta)|}$, a special rational adapted frame $E^R = \{\mathbf{T}(\zeta), \mathbf{N}_1(\zeta), \mathbf{N}_2(\zeta)\}$ is completely defined by the PH curve specification, where the vector fields can be derived using (4.14). This adapted frame is called the Euler-Rodrigues frame.

$$\mathbf{T}(\zeta) = \mathcal{U}(\zeta)\hat{\mathbf{i}}\mathcal{U}^*(\zeta) \quad \mathbf{N}_1(\zeta) = \mathcal{U}(\zeta)\hat{\mathbf{j}}\mathcal{U}^*(\zeta) \quad \mathbf{N}_2(\zeta) = \mathcal{U}(\zeta)\hat{\mathbf{k}}\mathcal{U}^*(\zeta) \quad (4.14)$$

Unlike the Frenet frame, the Euler-Rodrigues frame (ERF) is uniquely defined for each point of the spatial PH curve, including degenerate curves and inflection points, offering continuous smooth rotations along the curve. The ERF has some unique characteristics in relation to other frames [75]. In the case of cubic (3rd order) spatial PH curves, i.e. $\mathcal{A}(\zeta)$ is 1st order, the ERF has a constant offset angle to the Frenet frame. For septic (7th order) and higher order spatial PH curves the ERF is a rational rotation minimising frame (RRMF). Quintic (5th order) spatial PH curves offer C^2 continuity, but the ERF is a RRMF only if the condition $\bar{\mathcal{A}}_1\hat{\mathbf{i}}\bar{\mathcal{A}}_1^* = \text{vect}(\bar{\mathcal{A}}_0\hat{\mathbf{i}}\bar{\mathcal{A}}_2^*)$ is satisfied [76].

It is clear to see that the ERF is of the same order as the hodograph, and due to the ambiguity of the quaternion specification of the hodograph curve, it is not uniquely defined on the hodograph. Therefore the entire ERF can be rotated by a constant angle around the curve without affecting the spatial curve geometry. Figure 4.4 shows a nonic (9th order) ERF, which meets the requirement of being rotation minimising.

4.2.4 Rotation Minimising Frame

The rotation minimising frame (RMF) is a special frame specification that was developed with the goal of defining smooth swept surfaces and tubes on space curves. With this in mind the ideal was to remove the unnecessary rotation of the normal vectors around the tangent, hence minimising rotations. Since the magnitude of the tangent vector $|\mathbf{T}| = 1$ never changes, its derivative $\dot{\mathbf{T}}$ lies in the span of the normal plane, and in the case of the Frenet frame, in the direction of the principle normal $\mathbf{N}_p$. The rotation minimising frame

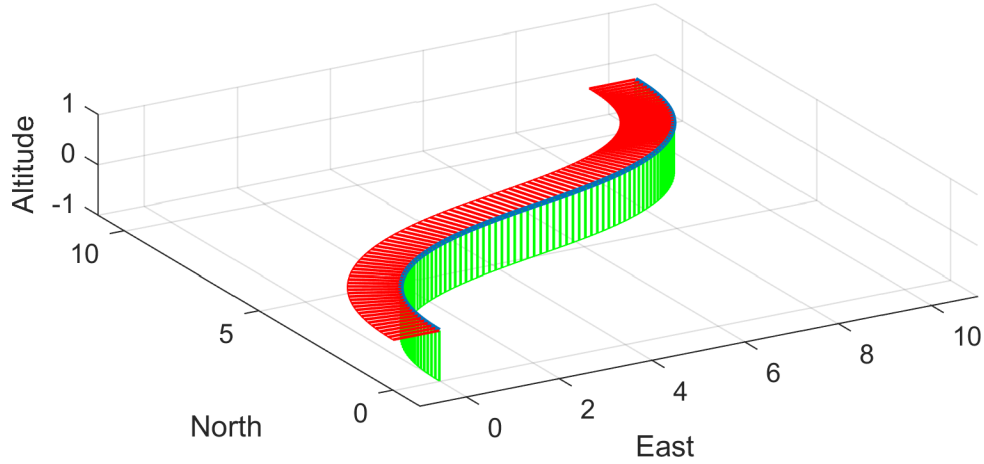


Figure 4.4: Nonic Euler-Rodrigues frame (ERF)

(RMF) is characterised by the property that the normal plane does not rotate around the tangent, i.e. $\alpha \equiv 0$.

Thus a new set of normal vectors are defined in relation to the Frenet frame through a rotation in the normal plane.

$$\begin{bmatrix} \mathbf{N}_1 \\ \mathbf{N}_2 \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} \mathbf{N}_p \\ \mathbf{B} \end{bmatrix} \quad (4.15)$$

The function $\theta(\zeta)$ is defined to remove the unnecessary rotation of the frame. It was shown in [77] that the vectors must satisfy

$$\mathbf{N}'_1(\zeta) = -\frac{\mathbf{p}''(\zeta) \cdot \mathbf{N}_1(\zeta)}{|\mathbf{p}'(\zeta)|^2} \mathbf{p}'(\zeta) \quad \mathbf{N}'_2(\zeta) = -\frac{\mathbf{p}''(\zeta) \cdot \mathbf{N}_2(\zeta)}{|\mathbf{p}'(\zeta)|^2} \mathbf{p}'(\zeta) \quad (4.16)$$

in order for the frame to be rotation minimising. Thus the relative angular rate of the frames must be

$$\frac{d\theta}{d\zeta} = -\sigma\tau = -|\mathbf{p}'| \frac{(\mathbf{p}' \times \mathbf{p}'') \cdot \mathbf{p}'''}{|\mathbf{p}' \times \mathbf{p}''|^2}, \quad (4.17)$$

and as such the relative offset angle between the rotation minimising frame (RMF) and the Frenet frame is

$$\theta(\zeta) = \theta_0 - \int_0^\zeta \tau(\xi)\sigma(\xi) d\xi. \quad (4.18)$$

As a result of the integration constant, it is clear that there are an infinite number of RMFs that satisfy (4.16), and depends on the specification of the initial orientation of the frame.

4.3 Orientation Specification

Considering the mathematical correlation between curve framing and the aircraft orientation, evident through the *Poisson kinematic equations*, these framing techniques can be evaluated in terms of their mechanical behaviour. In the case of the Frenet frame, the frame is constructed through roll and yaw motions, which is easier to identify through the Beta frame. The RPAF and RMF with $\alpha = 0$ specifies the frame through a series of pitch and yaw motions, negating the roll motion to ensure minimal rotations.

Clearly these frames do not directly relate to the aircraft orientation, with the nominal orientation achieved through pitch and roll motions, with minimal yaw motion. Even if a frame is specified as such, it would still not account for the lift and thrust vectors that also play a key role in aircraft motion. This is due to the further separation between the aircraft body frame and the aerodynamic (wind) frame, due to the aerodynamic angles. The navigation problem is primarily concerned with the construction of the aerodynamic frame, and since the roll angle is common between these two frames, omitting these aerodynamic angles during trajectory generation does not negate the results. The aerodynamic angles can then be computed along the reference trajectory using an inverse model to evaluate the aircraft performance and flight efficiency.

4.3.1 Euler-Rodrigues Frame for Motion Design

The ERF's inclusion in the PH curve construction makes it an obvious choice for the specification of an orientation inclusive trajectory. Furthermore, the rotation minimising property of septic (7th order) and higher order PH curves, is also advantageous to motion design. Using this approach to PH curve construction is actually quite intuitive for motion design, as the trajectory is effectively defined as a sequence of quaternion rotations with variable magnitude, which is the parametric speed. The challenge with this technique however, is that the spatial curve is defined by the development of the ERF, and thus manipulation of the frame translates through to the curve. It is therefore required that a better understanding of the interdependence of the variables is obtained.

There are two ways of interpreting the quaternion values that define the hodograph of the spatial trajectory, the key being to note that the projection of the quaternion polynomial defines the tangent vector of the trajectory. These relate to the case of the general solution in (3.30) for solving $\hat{\mathcal{A}}\dot{\mathcal{A}}^* = \sigma\mathbf{T}$. One is working from the perspective that each quaternion consists of a vector, its spatial projection, and a single free angular parameter that defines its rotation around the projection. The other is that each quaternion defines a frame, containing a complete three angle specification of the orientation, and a magnitude that relates to the parametric speed. It should be noted that due to the double quaternion product the parametric speed is proportional to the square of the quaternion magnitude, since $\sigma = \mathcal{A}\mathcal{A}^*$. The case shown in (3.32) relating to the solution of $\hat{\mathcal{A}}\dot{\mathcal{B}}^* = \mathbf{c}$, has a free scalar parameter that affects both the magnitude and rotation of the quaternion around the projected vector $\mathbf{c}$.

Degrees of Freedom

Previously it was determined that the required order of the spatial spline was at least fifth order, a requirement that arises from the need for G^2 continuity. The use of the PH curve construction however adds additional constraints, making it necessary to re-evaluate this conclusion.

Since the curve is constructed through specification of the hodograph, the starting position no longer affects the order of the curve and is simply an integration constant used to correctly position the spatial curve after integration. The final position, or more precisely its relation to the start position, becomes an integration constraint, to ensure that the spatial curve intersects the final position. Thus the start and final positions define, as a pair, a single constraint on the hodograph and ensures path (G^0) continuity.

The start and end velocities, consisting of the frame orientation and magnitude, binds the boundary conditions of the hodograph to ensure G^1 continuity. Similarly, the start and end curvature, along with angular rates, must be bound to ensure curvature (G^2) continuity. Combined these five constraints means the quaternion polynomial must at

least be quartic (4th order), and thus the spatial curve would be a nonic (9th order) polynomial. The included rotational kinematics for each of these quaternion constraints results in curves with continuous orientation and angular rates.

4.4 Quaternion Spline Trajectories

As discussed in section 3.3, an additional parameter is required to complete the trajectory specification to a single solution. The general approach to bind the trajectory to coordinated turns ($\beta(t) = 0$) limits the aircraft's capabilities. Although deviating from coordinated turns result in less efficient flight, it offers improved versatility and manoeuvrability. It is therefore that this study uses the approach of rather specifying the roll angle ($\phi(t)$).

Advantages of specifying the roll angle include:

- Deploying payloads.
- Minimising radar cross section.
- Aiming sensors or weapons at a target.
- Aerobatic and combat manoeuvring.
- Coordinated formation flight.
- Landing and docking manoeuvres with moving targets\vehicles.

As noted in the previous section, the quaternion PH curve offers an intuitive and efficient means of including the orientation in the trajectory specification in the form of the Euler-Rodrigues frame (ERF). Given the quartic order of the quaternion polynomial, the resulting curve meets the requirements to construct a rotation minimising frame (RMF). This ensures the orientation specification is guaranteed to be continuously defined and free of unwanted rotations, such as the inversions of the Frenet frame.

The resulting quaternion trajectory yields a nonic spatial curve that is G^2 continuous, and G^1 rotational continuity (angular rates). Achieving the angular acceleration continuity

would require further increasing the order of the quaternion polynomial, however the additional accuracy is negligible compared to the additional computational cost. Given the limited spatial and rotational continuity, this approach results in a sub-optimal trajectory as compared to optimal control approaches. However, as stated before, these trajectories are asymptotically optimal with an increasing number of segments. Furthermore, the error between the optimised spline trajectory and the optimal control solutions, is arguably negligible compared to the impact of unpredictable disturbances and environmental conditions. It is therefore advised that this approach be combined with a guidance controller, such as MPC, that optimises the transitions between segments while compensating for disturbances during execution. This hybrid approach should result in negligible differences between the physical performance of the geometric-dynamic and optimal control approaches. Thus the computational advantages of the geometric-dynamic approach are combined with the performance and accuracy of optimal control.

The quaternion PH curve then not only resolves the computational challenges regarding the arc length and parametric speed, but also directly includes the ability to define the orientation of the aerodynamic frame. Specifying the aerodynamic frame as a quaternion is not only more concise than Euler angle approaches, but computationally more efficient. The quaternion frame does not suffer from the discontinuities, nor gimbal lock, of the DCM. This makes it possible to directly define continuous roll manoeuvres, loops, and vertical climbs, without any special effort to compensate for the ambiguity of Euler angles. The quaternion curve also offers direct access to the angular rates, negating the need to numerically compute a 6-DoF model during evaluation of the trajectory, accelerating the computation of inverse dynamic models.

4.4.1 Spatial Curve

Using the quaternion polynomial form shown in (3.23), the polynomial quadruple can be defined in Bézier form as

$$\bar{\mathcal{A}}_i(\zeta_i) = \sum_{k=0}^m \bar{\mathcal{A}}_{i,k} b_{k,m}(\zeta_i), \quad (4.19)$$

with the Bernstein basis polynomials $b_{k,m}$ defined in (3.3). Given the continuity constraints determined *a priori*, the trajectory is defined using a quartic (4th order) quaternion polynomial. This results in the quaternion control polygon $[\bar{\mathcal{A}}_{i,0}, \bar{\mathcal{A}}_{i,1}, \bar{\mathcal{A}}_{i,2}, \bar{\mathcal{A}}_{i,3}, \bar{\mathcal{A}}_{i,4}]$ which defines the quaternion Bézier polynomial $\mathcal{A}_i(\zeta)$ for the i^{th} segment. Note again that these equations are easily expanded to multiple vehicles, and the additional vehicle index is omitted for brevity and ease of reading. Each of the control points $\bar{\mathcal{A}}_{i,k}$ is given by the quaternion

$$\bar{\mathcal{A}}_{i,k} = \bar{u}_{i,k} + \bar{v}_{i,k} \hat{\mathbf{i}} + \bar{p}_{i,k} \hat{\mathbf{j}} + \bar{q}_{i,k} \hat{\mathbf{k}}, \quad \text{for } k = [0..m]. \quad (4.20)$$

The quaternion polynomial can also be decomposed into the component splines:

$$\begin{aligned} u_i(\zeta_i) &= \sum_{k=0}^m \bar{u}_{i,k} b_{m,k}(\zeta_i), & v_i(\zeta_i) &= \sum_{k=0}^m \bar{v}_{i,k} b_{m,k}(\zeta_i), \\ p_i(\zeta_i) &= \sum_{k=0}^m \bar{p}_{i,k} b_{m,k}(\zeta_i), & q_i(\zeta_i) &= \sum_{k=0}^m \bar{q}_{i,k} b_{m,k}(\zeta_i). \end{aligned}$$

The control polygon of the spatial hodograph curve is then computed using the Bézier product

$$\bar{\mathbf{h}}(\zeta) = \bar{\mathcal{A}}(\zeta) \hat{\mathbf{i}} \bar{\mathcal{A}}^*(\zeta). \quad (4.21)$$

Remembering that a PH curve of order m yields a spatial curve of order $2m + 1$, the quartic quaternion polynomial results in an octic (8th order) hodograph and a nonic (9th

order) spatial curve. The hodograph curve's control polygon is then given by:

$$\begin{aligned}
\bar{\mathbf{h}}_{i,0} &= \bar{\mathcal{A}}_{i,0} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,0}^* \\
\bar{\mathbf{h}}_{i,1} &= \frac{1}{2} (\bar{\mathcal{A}}_{i,0} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,1}^* + \bar{\mathcal{A}}_{i,1} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,0}^*) \\
\bar{\mathbf{h}}_{i,2} &= \frac{1}{14} (3\bar{\mathcal{A}}_{i,0} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,2}^* + 8\bar{\mathcal{A}}_{i,1} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,1}^* + 3\bar{\mathcal{A}}_{i,2} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,0}^*) \\
\bar{\mathbf{h}}_{i,3} &= \frac{1}{14} (\bar{\mathcal{A}}_{i,0} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,3}^* + 6\bar{\mathcal{A}}_{i,1} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,2}^* + 6\bar{\mathcal{A}}_{i,2} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,1}^* + \bar{\mathcal{A}}_{i,3} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,0}^*) \\
\bar{\mathbf{h}}_{i,4} &= \frac{1}{70} (\bar{\mathcal{A}}_{i,0} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,4}^* + 16\bar{\mathcal{A}}_{i,1} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,3}^* + 36\bar{\mathcal{A}}_{i,2} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,2}^* + 16\bar{\mathcal{A}}_{i,3} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,1}^* + \bar{\mathcal{A}}_{i,4} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,0}^*) \\
\bar{\mathbf{h}}_{i,5} &= \frac{1}{14} (\bar{\mathcal{A}}_{i,1} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,4}^* + 6\bar{\mathcal{A}}_{i,2} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,3}^* + 6\bar{\mathcal{A}}_{i,3} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,2}^* + \bar{\mathcal{A}}_{i,4} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,1}^*) \\
\bar{\mathbf{h}}_{i,6} &= \frac{1}{14} (3\bar{\mathcal{A}}_{i,2} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,4}^* + 8\bar{\mathcal{A}}_{i,3} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,3}^* + 3\bar{\mathcal{A}}_{i,4} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,2}^*) \\
\bar{\mathbf{h}}_{i,7} &= \frac{1}{2} (\bar{\mathcal{A}}_{i,3} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,4}^* + \bar{\mathcal{A}}_{i,4} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,3}^*) \\
\bar{\mathbf{h}}_{i,8} &= \bar{\mathcal{A}}_{i,4} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,4}^*
\end{aligned} \tag{4.22}$$

The spatial curve control polygon can now be derived through the integration of (4.21), using the segment start position as integration constant to correctly position the curve.

$$\begin{aligned}
\bar{\mathbf{p}}_{i,0} &= \mathbf{w}_i \\
\bar{\mathbf{p}}_{i,k+1} &= \bar{\mathbf{p}}_{i,k} + \frac{1}{9} \bar{\mathbf{h}}_{i,k} \quad \text{for } k \in [1..9]
\end{aligned} \tag{4.23}$$

4.4.2 Parametric Speed & Arc Length

The key property of the Pythagorean hodograph is the polynomial solution to the parametric speed,

$$\sigma_i(\zeta_i) = \|\mathbf{h}_i(\zeta_i)\| = \mathcal{A}_i(\zeta_i) \mathcal{A}_i^*(\zeta_i). \tag{4.24}$$

Through the inherent properties of the PH Bézier curve, the parametric speed is also a Bézier curve of the form:

$$\sigma_i(\zeta_i) = \sum_{k=0}^8 \bar{\sigma}_{i,k} b_{k,8}(\zeta_i). \tag{4.25}$$

Computing this in quaternion form, the values of the control polygon are as follows:

$$\begin{aligned}
\bar{\sigma}_{i,0} &= \bar{\mathcal{A}}_{i,0} \bar{\mathcal{A}}_{i,0}^* \\
\bar{\sigma}_{i,1} &= \frac{1}{2} (\bar{\mathcal{A}}_{i,0} \bar{\mathcal{A}}_{i,1}^* + \bar{\mathcal{A}}_{i,1} \bar{\mathcal{A}}_{i,0}^*) \\
\bar{\sigma}_{i,2} &= \frac{1}{14} (3\bar{\mathcal{A}}_{i,0} \bar{\mathcal{A}}_{i,2}^* + 8\bar{\mathcal{A}}_{i,1} \bar{\mathcal{A}}_{i,1}^* + 3\bar{\mathcal{A}}_{i,2} \bar{\mathcal{A}}_{i,0}^*) \\
\bar{\sigma}_{i,3} &= \frac{1}{14} (\bar{\mathcal{A}}_{i,0} \bar{\mathcal{A}}_{i,3}^* + 6\bar{\mathcal{A}}_{i,1} \bar{\mathcal{A}}_{i,2}^* + 6\bar{\mathcal{A}}_{i,2} \bar{\mathcal{A}}_{i,1}^* + \bar{\mathcal{A}}_{i,3} \bar{\mathcal{A}}_{i,0}^*) \\
\bar{\sigma}_{i,4} &= \frac{1}{70} (\bar{\mathcal{A}}_{i,0} \bar{\mathcal{A}}_{i,4}^* + 16\bar{\mathcal{A}}_{i,1} \bar{\mathcal{A}}_{i,3}^* + 36\bar{\mathcal{A}}_{i,2} \bar{\mathcal{A}}_{i,2}^* + 16\bar{\mathcal{A}}_{i,3} \bar{\mathcal{A}}_{i,1}^* + \bar{\mathcal{A}}_{i,4} \bar{\mathcal{A}}_{i,0}^*) \\
\bar{\sigma}_{i,5} &= \frac{1}{14} (\bar{\mathcal{A}}_{i,1} \bar{\mathcal{A}}_{i,4}^* + 6\bar{\mathcal{A}}_{i,2} \bar{\mathcal{A}}_{i,3}^* + 6\bar{\mathcal{A}}_{i,3} \bar{\mathcal{A}}_{i,2}^* + \bar{\mathcal{A}}_{i,4} \bar{\mathcal{A}}_{i,1}^*) \\
\bar{\sigma}_{i,6} &= \frac{1}{14} (3\bar{\mathcal{A}}_{i,2} \bar{\mathcal{A}}_{i,4}^* + 8\bar{\mathcal{A}}_{i,3} \bar{\mathcal{A}}_{i,3}^* + 3\bar{\mathcal{A}}_{i,4} \bar{\mathcal{A}}_{i,2}^*) \\
\bar{\sigma}_{i,7} &= \frac{1}{2} (\bar{\mathcal{A}}_{i,3} \bar{\mathcal{A}}_{i,4}^* + \bar{\mathcal{A}}_{i,4} \bar{\mathcal{A}}_{i,3}^*) \\
\bar{\sigma}_{i,8} &= \bar{\mathcal{A}}_{i,4} \bar{\mathcal{A}}_{i,4}^*
\end{aligned} \tag{4.26}$$

Having the parametric speed in Bézier form, it is trivial to define the arc length polynomial,

$$\begin{aligned}
s_i(\zeta_i) &= \int_0^{\zeta_i} \sigma_i(\zeta) d\zeta = \sum_{k=0}^9 \bar{s}_{i,k} b_{k,9}(\zeta_i) \\
\bar{s}_{i,0} &= 0 \\
\bar{s}_{i,k} &= \frac{1}{9} \sum_{l=0}^{k-1} \bar{\sigma}_{i,l} \quad \text{for } k = 1, \dots, 9
\end{aligned} \tag{4.27}$$

and similarly the parametric acceleration is

$$\sigma'_i(\zeta_i) = 8 \sum_{k=0}^7 (\bar{\sigma}_{i,k+1} - \bar{\sigma}_{i,k}) b_{k,7}(\zeta_i). \tag{4.28}$$

The segment total arc length is then simply the final value of the arc length polynomial control polygon:

$$L_i = s_i(1) = \bar{s}_{i,9}. \tag{4.29}$$

4.4.3 Timing Law

The timing law, as discussed in subsection 3.3.2, is defined as

$$\theta_i(t) = \frac{d\zeta_i}{dt}, \quad \text{for } t \in [0, t^f].$$

The spline parameter (ζ_i) for each segment is computed in the segment time interval $t = [t_i, t_{i+1}]$, by integrating over the interval

$$\zeta_i(t) = \int_{t_i}^t \theta_i(\tau) d\tau, \quad (4.30)$$

As stated before, it is advantageous to normalise the time within each segment so as to facilitate the use of Bézier splines for the timing law. The timing law is primarily bounded by the time interval $T_i = t_{i+1} - t_i$, that describes the segment, rather than the actual time the segment is executed. This negates the need to continually update the arrival times of all segments when a single segment is manipulated.

$$\hat{t}_i = \frac{t - t_i}{T_i} \quad t \in [t_i, t_{i+1}] \quad (4.31)$$

The timing law is thus re-defined as a function of the normalised time,

$$\theta_i(\hat{t}_i) = \frac{d\zeta_i}{d\hat{t}_i}, \quad \hat{t}_i \in [0, 1], \quad (4.32)$$

and the time can be derived as $t = t_i + T_i \hat{t}_i$. It is then sensible to define the spatial spline parameter ζ_i as an explicit Bézier spline, parametrised by normalised time $\hat{t}_i$, as this offers a simple linear relationship between the spline index and timing law.

$$\zeta_i(\hat{t}_i) = \int_0^{\hat{t}_i} \theta_i(\tau) d\tau \quad \hat{t} \in [0, 1] \quad (4.33)$$

Although it is possible to define the Bézier spline as a function of time t using an interval change within the spline definition, performing the interval change *a priori* allows for a more concise and computationally efficient description.

Given a timing law $\theta_i(\hat{t}_i)$ defined using a Bézier spline of order n_θ , the polynomial defining $\zeta_i(\hat{t}_i)$ would also be a Bézier spline of order $n_\theta + 1$, as defined through the integration property of Bézier splines.

$$\theta_i(\hat{t}_i) = \sum_{k=0}^{n_\theta} \bar{\theta}_{i,k} b_{k,n_\theta}(\hat{t}_i) \quad (4.34)$$

$$\begin{aligned}\bar{\zeta}_{i,0} &= 0 \\ \bar{\zeta}_{i,k} &= \frac{1}{n_\theta + 1} \sum_{l=0}^{k-1} \bar{\theta}_{i,l} \quad \text{for } k = 1, \dots, (n_\theta + 1)\end{aligned} \quad (4.35)$$

This property can then be exploited to aid in the construction of the timing law, by binding the boundary conditions $\zeta_i(0) = 0$ and $\zeta_i(1) = 1$.

$$\zeta_i(1) = \frac{1}{n_\theta + 1} \sum_{k=0}^{n_\theta} \bar{\theta}_{i,k} = 1 \quad (4.36)$$

Relating the timing law back to actual time simply requires scaling the function by the time interval, and if needed adding the segment start time to determine real arrival time.

$$\theta_i(t) = \frac{d\zeta_i}{dt} = \frac{1}{T_i} \theta_i(\hat{t}) \quad (4.37)$$

Applying this to the velocity profile, the velocity at each point along the segment can be computed as

$$v_i(\hat{t}) = \frac{1}{T_i} (\sigma \circ \zeta)(\hat{t}) \theta_i(\hat{t}), \quad \text{for } \hat{t} \in [0, 1]. \quad (4.38)$$

This definition is now used to relate the boundary velocity constraints to the velocity profile.

$$\begin{aligned}v_i^s &= \frac{1}{T_i} \bar{\sigma}_{i,0} \bar{\theta}_{i,0} & v_i^f &= \frac{1}{T_i} \bar{\sigma}_{i,8} \bar{\theta}_{i,n_\theta} \\ a_i^s &= \frac{1}{T_i} (\bar{\sigma}'_0 \bar{\theta}_0^2 + \bar{\sigma}_0 \bar{\theta}'_0) & a_i^f &= \frac{1}{T_i} (\bar{\sigma}'_8 \bar{\theta}_4^2 + \bar{\sigma}_8 \bar{\theta}'_4)\end{aligned}$$

Considering the case of a quadratic timing law, the control polygon is constructed as follows:

$$\bar{\theta}_{i,0} = \frac{T_i v_i}{\bar{\sigma}_{i,0}} \quad \bar{\theta}_{i,2} = \frac{T_i v_{i+1}}{\bar{\sigma}_{i,8}} \quad \bar{\theta}_{i,1} = 3 - \bar{\theta}_{i,0} - \bar{\theta}_{i,2} \quad (4.39)$$

The control polygon for the spline index curve is then given by the cubic spline below.

$$\zeta_i(\hat{t}) = \sum_{v=0}^3 \bar{\zeta}_{i,v} b_{k,3}(\hat{t}_i) \quad (4.40)$$

$$\bar{\zeta}_{i,0} = 0 \quad \bar{\zeta}_{i,1} = \frac{1}{3} \bar{\theta}_{i,0} \quad \bar{\zeta}_{i,2} = \frac{1}{3} (\bar{\theta}_{i,0} + \bar{\theta}_{i,1}) \quad \bar{\zeta}_{i,3} = 1 \quad (4.41)$$

The quadratic case above however only offers velocity continuity and not acceleration continuity between segments, i.e. although the trajectory is G^2 (curvature) continuous, it only has C^1 parametric (with respect to time) continuity. In order to achieve C^2 continuity, i.e. continuous acceleration, the timing law must be atleast quartic (4rd order). The use of a cubic (3rd order) timing law is insufficient due to the integration constraint of $\zeta_i(1) = 1$. Therefore the use of a quartic timing law ($n_\theta = 4$) is suggested here to achieve the desired C^2 continuity. Using the above relationships the control polygon of the quartic timing law is constructed.

$$\begin{aligned}\bar{\theta}_{i,0} &= \frac{T_i v_i}{\bar{\sigma}_{i,0}} & \bar{\theta}_{i,4} &= \frac{T_i v_{i+1}}{\bar{\sigma}_{i,8}} \\ \bar{\theta}_{i,1} &= \bar{\theta}_{i,0} + \frac{T_i a_i - 8(\bar{\sigma}_{i,1} - \bar{\sigma}_{i,0})\bar{\theta}_{i,0}^2}{4\bar{\sigma}_{i,0}} & \bar{\theta}_{i,3} &= \bar{\theta}_{i,4} - \frac{T_i a_{i+1} - 8(\bar{\sigma}_{i,8} - \bar{\sigma}_{i,7})\bar{\theta}_{i,4}^2}{4\bar{\sigma}_{i,8}}\end{aligned}$$

From the constraint in (4.36) the value of the missing control point is computed.

$$\bar{\theta}_{i,2} = 5 - \bar{\theta}_{i,0} - \bar{\theta}_{i,1} - \bar{\theta}_{i,3} - \bar{\theta}_{i,4} \quad (4.42)$$

It is then clear that the control polygon for the spline index curve computes to:

$$\begin{aligned}\bar{\zeta}_{i,0} &= 0 \\ \bar{\zeta}_{i,k} &= \frac{1}{5} \sum_{v=0}^{k-1} \bar{\theta}_{i,v} \\ \bar{\zeta}_{i,5} &= 1\end{aligned} \quad (4.43)$$

The compound position spline, $\mathbf{p}(t) = (\mathbf{p} \circ \zeta)(t)$, is then of 36th order, given a nonic spatial curve and quartic timing law. This is not an issue given the use of Bézier splines, as they remain computationally efficient.

Remark. When using velocity and acceleration boundary conditions, the combination of a zero velocity and zero acceleration at a way-point must be avoided, i.e. $v_i = 0$ and $a_i = 0$. Due to the analytic nature of the splines, this results in an asymptotically infinite segment duration. This is relevant for runway taxi and hover manoeuvres.

4.4.4 Quaternion Boundary Conditions

Constructing the trajectory requires relating the spatial boundary conditions in subsection 3.3.3 to the quaternion control polygon $[\bar{\mathcal{A}}_{i,0}, \bar{\mathcal{A}}_{i,1}, \bar{\mathcal{A}}_{i,2}, \bar{\mathcal{A}}_{i,3}, \bar{\mathcal{A}}_{i,4}]$. Due to the structure of the Bézier splines, these control points relate to the higher order derivatives with increasing order. It should be remembered that the quaternion curve defines the trajectory's hodograph, and hence the way-points are no longer represented by the outermost control points, but rather by the middle control point $\bar{\mathcal{A}}_{i,2}$ as the integration constraint.

Orientation & Parametric Speed: G^1 continuity

The first pair of control points, $\bar{\mathcal{A}}_{i,0}$ and $\bar{\mathcal{A}}_{i,4}$, relate to the tangent vectors, $\mathbf{m}_i^s$ and $\mathbf{m}_i^f$, that define the direction of travel and parametric speed. The tangent vectors are decomposed into the flight path angles γ_i and γ_{i+1} , headings ψ_i and ψ_{i+1} , and magnitudes $\|\mathbf{m}_i^s\|$ and $\|\mathbf{m}_i^f\|$, as shown in (4.44). The quaternion introduces an additional angular component as shown in (3.30), which in the context of relating the Euler-Rodrigues frame (ERF) to the DCM, and thus the orientation of the aerodynamic frame, is used to define the roll angles ϕ_i and ϕ_{i+1} .

$$\mathbf{m}_i^s = |\mathbf{m}_i^s| \begin{bmatrix} \cos \gamma_i \cos \psi_i \\ \cos \gamma_i \sin \psi_i \\ \sin \gamma_i \end{bmatrix} \quad \mathbf{m}_i^f = |\mathbf{m}_i^f| \begin{bmatrix} \cos \gamma_{i+1} \cos \psi_{i+1} \\ \cos \gamma_{i+1} \sin \psi_{i+1} \\ \sin \gamma_{i+1} \end{bmatrix} \quad (4.44)$$

Note that the Euler angles are common between adjacent segments, while the magnitudes are kept independent to increase flexibility regarding curve geometry. This approach of separating the magnitudes is in no way mandatory, and can be kept mutual if desired to reduce the dimensionality of the optimisation. Equality of the magnitudes are not required for G^1 continuity, but must however be accounted for when ensuring curvature (G^2) continuity, as shown in the next section. Continuity of the parametric speed between segments is not required as parametric (C) continuity is achieved through the timing laws.

The values of the quaternions can then be derived from the general solution in (3.30), using the relationships

$$\bar{\mathbf{h}}_{i,0} = \bar{\mathcal{A}}_{i,0} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,0}^* = \mathbf{m}_i^s \quad \text{and} \quad \bar{\mathbf{h}}_{i,8} = \bar{\mathcal{A}}_{i,4} \hat{\mathbf{i}} \bar{\mathcal{A}}_{i,4}^* = \mathbf{m}_i^f. \quad (4.45)$$

Using the general solution however violates the sequence of rotation, hence it is more convenient to directly construct the quaternion using the Euler angles. The solutions for the quaternions $\bar{\mathcal{A}}_{i,0}$ and $\bar{\mathcal{A}}_{i,4}$ are then shown in (4.46) and (4.47).

$$\bar{\mathcal{A}}_{i,0} = \sqrt{|\mathbf{m}_i^s|} \begin{bmatrix} \cos(\frac{\phi_i}{2}) \cos(\frac{\gamma_i}{2}) \cos(\frac{\psi_i}{2}) + \sin(\frac{\phi_i}{2}) \sin(\frac{\gamma_i}{2}) \sin(\frac{\psi_i}{2}) \\ \sin(\frac{\phi_i}{2}) \cos(\frac{\gamma_i}{2}) \cos(\frac{\psi_i}{2}) - \cos(\frac{\phi_i}{2}) \sin(\frac{\gamma_i}{2}) \sin(\frac{\psi_i}{2}) \\ \cos(\frac{\phi_i}{2}) \sin(\frac{\gamma_i}{2}) \cos(\frac{\psi_i}{2}) + \sin(\frac{\phi_i}{2}) \cos(\frac{\gamma_i}{2}) \sin(\frac{\psi_i}{2}) \\ \cos(\frac{\phi_i}{2}) \cos(\frac{\gamma_i}{2}) \sin(\frac{\psi_i}{2}) - \sin(\frac{\phi_i}{2}) \sin(\frac{\gamma_i}{2}) \cos(\frac{\psi_i}{2}) \end{bmatrix} \quad (4.46)$$

$$\bar{\mathcal{A}}_{i,4} = \sqrt{|\mathbf{m}_i^f|} \begin{bmatrix} \cos(\frac{\phi_{i+1}}{2}) \cos(\frac{\gamma_{i+1}}{2}) \cos(\frac{\psi_{i+1}}{2}) + \sin(\frac{\phi_{i+1}}{2}) \sin(\frac{\gamma_{i+1}}{2}) \sin(\frac{\psi_{i+1}}{2}) \\ \sin(\frac{\phi_{i+1}}{2}) \cos(\frac{\gamma_{i+1}}{2}) \cos(\frac{\psi_{i+1}}{2}) - \cos(\frac{\phi_{i+1}}{2}) \sin(\frac{\gamma_{i+1}}{2}) \sin(\frac{\psi_{i+1}}{2}) \\ \cos(\frac{\phi_{i+1}}{2}) \sin(\frac{\gamma_{i+1}}{2}) \cos(\frac{\psi_{i+1}}{2}) + \sin(\frac{\phi_{i+1}}{2}) \cos(\frac{\gamma_{i+1}}{2}) \sin(\frac{\psi_{i+1}}{2}) \\ \cos(\frac{\phi_{i+1}}{2}) \cos(\frac{\gamma_{i+1}}{2}) \sin(\frac{\psi_{i+1}}{2}) - \sin(\frac{\phi_{i+1}}{2}) \sin(\frac{\gamma_{i+1}}{2}) \cos(\frac{\psi_{i+1}}{2}) \end{bmatrix} \quad (4.47)$$

Curvature & Angular rates: G^2 continuity

Curvature (G^2) continuity between segments is achieved through common specification of the second derivative in the knots, referred to here as the curvature control vectors ($\mathbf{c}_i$ and $\mathbf{c}_{i+1}$).

$$\begin{aligned} \bar{\mathbf{h}}_{i,1} &= \bar{\mathbf{h}}_{i,0} + \frac{1}{8} \mathbf{c}_i & \bar{\mathbf{h}}_{i,7} &= \bar{\mathbf{h}}_{i,8} - \frac{1}{8} \mathbf{c}_{i+1} \\ &= \mathbf{m}_i^s + \frac{1}{8} \mathbf{c}_i & &= \mathbf{m}_i^f - \frac{1}{8} \mathbf{c}_{i+1} \end{aligned} \quad (4.48)$$

The curvature in the knots can then be computed as:

$$\kappa_i = \frac{|\mathbf{m}_i^s \times \mathbf{c}_i|}{\bar{\sigma}_{i,0}^3} \quad \kappa_{i+1} = \frac{|\mathbf{m}_i^f \times \mathbf{c}_{i+1}|}{\bar{\sigma}_{i,8}^3}, \quad (4.49)$$

and if chosen correctly these values can be the maxima of the segment curvature. The direction of the curvature control vectors determine the direction of centripetal

acceleration and their magnitudes affect the magnitude of the curvature. If these vectors are kept perpendicular to their respective tangent vectors, the calculation of the curvature becomes a purely scalar computation.

The relationship between the spatial vectors and the quaternions were shown in (4.22) and repeated below.

$$\begin{aligned}\bar{\mathbf{h}}_{i,1} &= \frac{1}{2}(\bar{\mathcal{A}}_{i,0}\hat{\mathbf{i}}\bar{\mathcal{A}}_{i,1}^* + \bar{\mathcal{A}}_{i,1}\hat{\mathbf{i}}\bar{\mathcal{A}}_{i,0}^*) = \text{vect} \left(\bar{\mathcal{A}}_{i,0}\hat{\mathbf{i}}\bar{\mathcal{A}}_{i,1}^* \right) \\ \bar{\mathbf{h}}_{i,7} &= \frac{1}{2}(\bar{\mathcal{A}}_{i,3}\hat{\mathbf{i}}\bar{\mathcal{A}}_{i,4}^* + \bar{\mathcal{A}}_{i,4}\hat{\mathbf{i}}\bar{\mathcal{A}}_{i,3}^*) = \text{vect} \left(\bar{\mathcal{A}}_{i,3}\hat{\mathbf{i}}\bar{\mathcal{A}}_{i,4}^* \right)\end{aligned}$$

Using the general solution (3.32) the next pair of control points can be calculated.

$$\bar{\mathcal{A}}_{i,1}(\tau_1) = -\frac{(\tau_1 + \bar{\mathbf{h}}_{i,1})\bar{\mathcal{A}}_{i,0}\hat{\mathbf{i}}}{|\bar{\mathcal{A}}_{i,0}|^2} \quad (4.50)$$

$$\bar{\mathcal{A}}_{i,3}(\tau_3) = -\frac{(\tau_3 + \bar{\mathbf{h}}_{i,7})\bar{\mathcal{A}}_{i,4}\hat{\mathbf{i}}}{|\bar{\mathcal{A}}_{i,4}|^2} \quad (4.51)$$

The choice of the free parameters τ_1 and τ_3 affects the angular rates of the ERF in the start and end points respectively. It should be noted that these angular rates are related to the spline parameter ζ and must be combined with the timing law to determine the angular velocities with respect to time.

Way-point Interpolation & Bending Energy

The final quaternion control point $\bar{\mathcal{A}}_{i,2}$ can now be computed by applying the endpoint constraints to the path integral.

$$\int_0^1 \bar{\mathcal{A}}_i(\zeta_i)\hat{\mathbf{i}}\bar{\mathcal{A}}_i^*(\zeta_i) d\zeta_i = \frac{1}{9} \sum_{k=0}^8 \bar{\mathbf{h}}_{i,k} = \mathbf{w}_{i+1} - \mathbf{w}_i \quad (4.52)$$

Equation (4.52) can be rewritten as follows:

$$\begin{aligned}\mathbf{r}_i &= (12\bar{\mathcal{A}}_{i,2} + 10\bar{\mathcal{A}}_{i,1} + 5\bar{\mathcal{A}}_{i,0} + 5\bar{\mathcal{A}}_{i,4} + 10\bar{\mathcal{A}}_{i,3})\hat{\mathbf{i}}(12\bar{\mathcal{A}}_{i,2} + 10\bar{\mathcal{A}}_{i,1} + 5\bar{\mathcal{A}}_{i,0} + 5\bar{\mathcal{A}}_{i,4} + 10\bar{\mathcal{A}}_{i,3}) \\ &= 2520(\mathbf{w}_{i+1} - \mathbf{w}_i) - 435(\mathbf{m}_i^f + \mathbf{m}_i^s) + \frac{45}{2}(\mathbf{c}_{i+1} - \mathbf{c}_i) - (60\bar{\mathcal{A}}_{i,1}\hat{\mathbf{i}}\bar{\mathcal{A}}_{i,1} - 30\bar{\mathcal{A}}_{i,0}\hat{\mathbf{i}}\bar{\mathcal{A}}_{i,3} - 30\bar{\mathcal{A}}_{i,3}\hat{\mathbf{i}}\bar{\mathcal{A}}_{i,0} \\ &\quad - 30\bar{\mathcal{A}}_{i,1}\hat{\mathbf{i}}\bar{\mathcal{A}}_{i,4} - 30\bar{\mathcal{A}}_{i,4}\hat{\mathbf{i}}\bar{\mathcal{A}}_{i,1} + 60\bar{\mathcal{A}}_{i,3}\hat{\mathbf{i}}\bar{\mathcal{A}}_{i,3} - 21\bar{\mathcal{A}}_{i,0}\hat{\mathbf{i}}\bar{\mathcal{A}}_{i,4} - 21\bar{\mathcal{A}}_{i,4}\hat{\mathbf{i}}\bar{\mathcal{A}}_{i,0} - 36\bar{\mathcal{A}}_{i,1}\hat{\mathbf{i}}\bar{\mathcal{A}}_{i,3} - 36\bar{\mathcal{A}}_{i,3}\hat{\mathbf{i}}\bar{\mathcal{A}}_{i,1})\end{aligned}$$

where $\mathbf{r}_i$ is the pure vector solution to the above equation, and the solution to $\bar{\mathcal{A}}_{i,2}$ can then be reduced to (4.53).

$$\bar{\mathcal{A}}_{i,2}(\phi_{i,2}) = \frac{1}{12} \left(\sqrt{|\mathbf{r}|} \frac{\frac{\mathbf{r}}{|\mathbf{r}|} + \hat{\mathbf{i}}}{\frac{\mathbf{r}}{|\mathbf{r}|} + \hat{\mathbf{i}}} \mathcal{Q}(\phi_{i,2}) - 10\bar{\mathcal{A}}_{i,1} - 5\bar{\mathcal{A}}_{i,0} - 5\bar{\mathcal{A}}_{i,4} - 10\bar{\mathcal{A}}_{i,3} \right) \quad (4.53)$$

The free parameter $\phi_{i,2}$ can then be used to reduce the total bending energy of the curve, and if chosen correctly can result in planar solutions for the spatial curve.

Remark (Free Parameters). *It was shown in [78] that the spatial PH curve has an invariance with regards to the interpolants, i.e. there exist a set of quaternion polynomials related by a constant offset angle that yields the exact same spatial hodograph. Therefore the shape of the spatial curve is determined by the difference between the free angle parameters $\phi_{i,0}$, $\phi_{i,2}$ and $\phi_{i,4}$, and therefore without loss of generality the value of $\phi_{i,2}$ can be bound to a fixed value, say $\phi_{i,2} = 0$. However, this is equivalent to binding the initial value of the RMF and is therefore better in this case to bind the values of $\phi_{i,0}$ and $\phi_{i,4}$ and retaining $\phi_{i,2}$ to manipulate the bending energy of the curve. It is worth noting that the spatial curve will be planar for the following value sets for a curve with free parameter set $[\phi_0, \tau_1, \phi_2, \tau_3, \phi_4]$:*

$$[0, 0, 0, 0, 0] \quad [\pi, 0, 0, 0, 0] \quad [0, 0, 0, 0, \pi] \quad [\pi, 0, 0, 0, \pi]$$

Any value set with the same relative angles will also yield a planar curve.

4.5 Summary of Trajectory Generation Algorithm

Given the complexities of the algorithm, a summary is presented to offer an overview.

Input Constraints: Way-points $\mathbf{w}_i$, velocities v_i , roll angles ϕ_i , flight-path angles γ_i and headings ψ_i .

Algorithm

1. Compute direction cosines for $\mathbf{m}_i^s$ from (γ_i, ψ_i) and $\mathbf{m}_i^f$ from $(\gamma_{i+1}, \psi_{i+1})$.
2. Select values for $|\mathbf{m}_i^s|$ and $|\mathbf{m}_i^f|$, and calculate $\bar{\mathbf{h}}_{i,0}$ and $\bar{\mathbf{h}}_{i,8}$.
3. Calculate $\bar{\mathcal{A}}_{i,0}$ and $\bar{\mathcal{A}}_{i,4}$ respectively.
4. Select values for $\mathbf{c}_i^s$ and $\mathbf{c}_i^f$, and calculate $\bar{\mathbf{h}}_{i,1}$ and $\bar{\mathbf{h}}_{i,7}$.
5. Select values τ_1 and τ_3 , and calculate $\bar{\mathcal{A}}_{i,1}$ and $\bar{\mathcal{A}}_{i,3}$.
6. Compute vector $\mathbf{r}_i$ using (4.52).
7. Find the value for ϕ_2 that minimises the curve deviation from planarity or minimises the curve bending energy, and resolve $\bar{\mathcal{A}}_{i,2}$ using $\mathbf{r}_i$.
8. Determine the remaining control point values $\bar{\mathbf{h}}_{i,k}$ using the control set of the quaternion polynomial $\bar{\mathcal{A}}_{i,m}$.
9. Determine the control point values $\bar{\mathbf{p}}_{i,k}$ using the control set of the hodograph $\bar{\mathbf{h}}_{i,k}$.
10. Compute the control set of the parametric speed curve $\bar{\sigma}_{i,k}$ and derive the arc length and acceleration curves.
11. Compute the boundary conditions of the timing law $\bar{\theta}_{i,0}$, $\bar{\theta}_{i,1}$, $\bar{\theta}_{i,3}$ and $\bar{\theta}_{i,4}$.
12. Compute the time interval T_i using the arc length and average of the velocity constraints, or desired arrival times.
13. Compute the remaining value of the timing law $\bar{\theta}_{i,2}$ and derive the spline index curve control polygon $\bar{\zeta}_{i,k}$.

Output: Nonic spatial PH curve control set $\bar{\mathbf{p}}_{i,k}$ and quartic timing law $\bar{\theta}_{i,k}$, described by the parameter set $\Xi_i = (|\mathbf{m}_i^s|, |\mathbf{m}_i^f|, \tau_1, \phi_{i,2}, \tau_3, T_i)$, subject to the boundary constraints $(\mathbf{w}_i, v_i, \phi_i, \gamma_i, \psi_i, \mathbf{c}_i, \mathbf{w}_{i+1}, v_{i+1}, \phi_{i+1}, \gamma_{i+1}, \psi_{i+1}, \mathbf{c}_{i+1})$.

4.6 Conclusion

In this chapter, the algorithm for constructing nonic PH curves using Bézier splines was presented. The correlation between the quaternion structure of the PH curve and curve framing was illustrated, and the subsequently exploited to construct rotation minimising ERF curves. The quaternion PH curves can thus directly be used to define the 6 DoF development of the aircraft's wind reference frame, in the form of the Euler-Rodrigues frame (ERF). This then serves as the foundation for a non-linear kinodynamic and performance evaluation, as the only missing parameters are the angle of attack (α) and side-slip angle (β), which are computed as control parameters to the Aircraft Performance Model (APM).

Chapter 5

Trajectory Evaluation

True genius resides in the capacity for evaluation of uncertain, hazardous, and conflicting information.

– Winston Churchill

5.1 Introduction

The trajectory generation algorithm in the previous chapters serves as a generic means of describing the trajectory, and contains no direct consideration of the aircraft dynamics. For this reason it is necessary to optimise the trajectory for a given aircraft and ensure that it is within its capabilities. This is done through kinodynamic evaluation of the trajectory, using various functions to evaluate the kinematics and performance of the aircraft along the trajectory. Using these performance evaluators it is possible to construct a cost function optimised for a specific mission, ensuring the desired behaviour and performance are achieved. The selection of good initial values for the construction of the trajectory greatly improves the needed iterations to converge to an optimised trajectory. The optimisation is not directly part of this study, but this chapter is dedicated to the construction of the performance evaluation functions and structure needed for optimisation.

5.2 Trajectory Evaluation & Constraints

Kinodynamic trajectory generation is the process of generating a trajectory that is bounded based on various limits on its derivatives, ensuring that the craft is able to follow the trajectory. A path that is both free of obstacles and within the flight envelope of the aircraft (flyable), is called a feasible path. These limits are usually selected conservatively, leaving some performance headroom to ensure that the aircraft can still respond to disturbances and account for reductions in performance due to environmental or other factors. These constraints can be tailored to each aircraft, being defined by a combination of aerodynamic limitations, airframe structural limits, mission and safety requirements. Additional constraints can be placed on the trajectory, not based on abilities but rather the desired behaviour of the aircraft, called operational constraints. These operational constraints include following air traffic control regulations such as avoiding restricted areas, ensuring passenger comfort and avoiding damage to payloads. These trajectory evaluation parameters can either be used to define cost functions for optimisation, or as hard limits (constraints) to ensure feasibility. The selection of parameters used can be performed dynamically by an on-board state machine, increasing the versatility of the system. The flight path angle in conjunction with the angle of attack, can for example be constrained during take-off and landing to avoid tail strikes.

These parameters are classified into two main families, i.e. geometric and temporal. The geometric constraints are computed using the spatial curve and are independent of the timing law. The temporal (kinodynamic) constraints are computed using the composite trajectory, i.e. spatial curve and timing law. Due to the increased computational complexity of the temporal constraints, it is preferable to use geometric constraints where possible. The continuity constraints between segments are inherently met by the Bézier spline construction of the trajectory as discussed in the previous chapters, and hence are omitted here.

5.2.1 Geometric Constraints

Geometric constraints are the most commonly used parameters in the literature, due to their simplicity and computational efficiency. These constraints are well suited to initial optimisation of the spatial curve during path finding, before adding the temporal components.

Curvature & Torsion

Curvature and torsion was introduced in subsection 4.2.2 in reference to the Frenet-Serret frame. In summary, curvature (κ) is a measure of the deviation of the curve from a straight line, while torsion (τ) is the rate of rotation of the osculating plane, i.e. the plane in which the curve is bending. They are computed using (4.12) and (4.13) respectively. Although these parameters are useful in evaluating curve geometry, they are limited in respect to trajectory generation due to their various discontinuities and ambiguity (unsigned). The primary advantage of using them is the speed at which they can be computed, and hence serve as a cursory check to filter segments that are obviously infeasible.

Generally the maximum curvature allowed is calculated using the ratio between the max turn rate and max velocity, as shown in (5.1). Note that the curvature by its very nature doesn't have a sign, thus a single value is used to describe the limit.

$$\kappa_{max} = \frac{1}{R_{min}} = \frac{\dot{\psi}_{max}}{v_{max}} \quad (5.1)$$

Heading

The heading, although not generally used as a constraint, can be useful in evaluation various mission requirements. It is calculated using the spatial curve as shown in (5.2), or directly using the quaternion polynomial $\mathcal{A}(\zeta) = u(\zeta) + v(\zeta)\hat{\mathbf{i}} + p(\zeta)\hat{\mathbf{j}} + q(\zeta)\hat{\mathbf{k}}$ as shown

in (5.3).

$$\psi_i(\zeta_i) = \text{atan2} \left(\frac{\mathbf{p}_{E_i}'(\zeta_i)}{\sigma(\zeta_i)}, \frac{\mathbf{p}_{N_i}'(\zeta_i)}{\sigma(\zeta_i)} \right) \quad (5.2)$$

$$= \text{atan2} \left(\frac{2(v(\zeta_i)p(\zeta_i) + u(\zeta_i)q(\zeta_i))}{\|\mathcal{A}_i(\zeta_i)\|}, \frac{u^2(\zeta_i) + v^2(\zeta_i) - p^2(\zeta_i) - q^2(\zeta_i)}{\|\mathcal{A}_i(\zeta_i)\|} \right) \quad (5.3)$$

Flight Path (Climb) Angle

The climb angle, or flight-path angle, is also a purely geometric property and can be directly calculated from the spatial curve, as defined in (5.4), or the quaternion polynomial as in (5.5).

$$\gamma_i(\zeta_i) = -\arcsin \left(\frac{\mathbf{p}_{D_i}'(\zeta_i)}{\sigma_i(\zeta_i)} \right) \quad (5.4)$$

$$= -\arcsin \left(\frac{2(v(\zeta_i)q(\zeta_i) - u(\zeta_i)p(\zeta_i))}{\|\mathcal{A}_i(\zeta_i)\|} \right) \quad (5.5)$$

This property is usually defined as asymmetric due to the influence of gravity, thus the climb angle is bound to the limitations in (5.6).

$$\gamma_{dive} \leq \gamma_i(\zeta_i) \leq \gamma_{climb} \quad (5.6)$$

Roll Angle

Evaluation of the roll angle requires the normalised quaternion polynomial $\mathcal{U} = \frac{\mathcal{A}}{\|\mathcal{A}\|}$, since the roll angle is not contained in the hodograph function $\mathbf{p}'(\zeta)$.

$$\phi_i(\zeta_i) = \text{atan2} \left(\left[\mathcal{U}(\zeta_i) \hat{\mathbf{k}} \mathcal{U}(\zeta_i) \right]_y, \left[\mathcal{U}(\zeta_i) \hat{\mathbf{k}} \mathcal{U}(\zeta_i) \right]_z \right) \quad (5.7)$$

$$= \text{atan2} \left(\frac{2p(\zeta_i)q(\zeta_i) + 2u(\zeta_i)v(\zeta_i)}{\|\mathcal{A}_i(\zeta_i)\|}, \frac{u^2(\zeta_i) - v^2(\zeta_i) - p^2(\zeta_i) + q^2(\zeta_i)}{\|\mathcal{A}_i(\zeta_i)\|} \right) \quad (5.8)$$

If a symmetric limit is used to constrain the roll angle, the following approach using the unsigned bank angle would be preferable to reduce computational complexity.

$$|\phi_i(\zeta_i)| = \arcsin \left(\left[\mathcal{U}(\zeta_i) \hat{\mathbf{k}} \mathcal{U}(\zeta_i) \right]_z \right) \quad (5.9)$$

$$= \arcsin \left(\frac{u^2(\zeta_i) - v^2(\zeta_i) - p^2(\zeta_i) + q^2(\zeta_i)}{\|\mathcal{A}_i(\zeta_i)\|} \right) \quad (5.10)$$

Bending Energy

As noted in subsection 3.4.1, the bending energy of spatial curves is often used as an efficient metric for evaluating the curvature of 3D splines. Usually this is computed as the bending and twisting of the curve, represented by the curvature and torsion, as shown in (5.11).

$$E_i = \int_0^1 \sqrt{\kappa_i^2(\zeta) + \tau_i^2(\zeta)} d\zeta \quad (5.11)$$

Since the curvature results in centripetal acceleration, reducing the bending energy reduces the energy expended during the manoeuvre. Minimising the accelerations in turn reduces the stresses on the airframe and improves the comfort of passengers, making the bending energy of concern in various applications.

Although there exist closed form solutions to cubic and quintic PH curves, this is not the case for the nonic PH curve used in this algorithm. The issue lies in the calculation of the integral of the curvature, since calculating the curvature uses a rational function where the denominator is a function of the parametric speed.

Since the quaternion form of the PH curve describes a framed curve, the curving of the trajectory can be evaluated using the angular rates. Just as the relationship exists between the curvature of the frenet frame and the relatively parallel adapted frame (RPAF) parameter, i.e. $\kappa^2 = k_1^2 + k_2^2$, it is possible to use selected components of $\boldsymbol{\omega}_W$. The selection of which components depend on the desired result, for example it is possible to omit the roll rate (P_W), thus the bending energy is computed using only the pitch (Q_W) and yaw (R_W) rates. The equivalent bending energy is then computed in (5.12) using the Euclidean norm of the angular rate vector $\boldsymbol{\omega}_{W_i}(\zeta) = 2 \frac{\mathcal{A}_i^*(\zeta) \mathcal{A}_i'(\zeta)}{\sigma_i(\zeta)}$. This is computationally less complex than computing the curvature and torsion, accelerating the optimisation.

$$E_i = \int_0^1 \|\boldsymbol{\omega}_{W_i}(\zeta)\| d\zeta \quad (5.12)$$

5.2.2 Translational Motion

The first set of temporal constraints are related to the translational motion of the craft, being dependent on the composition of the spatial curve and the timing law.

Velocity

Velocity is the fundamental kinematic constraint that is needed for the evaluation of the trajectory, as it is essential to ensuring the trajectory is flyable. In this context the velocity refers to the magnitude of the velocity vector, or ground speed, which relates to the true airspeed when wind is omitted. The velocity can easily be computed using (5.13) and is bounded by the limits in (5.14). The first two equations are in relation to time, and the third equation is in terms of the normalised time within the segment $\hat{t} \in [0, 1]$.

$$v_i(t) = \|(\mathbf{p}'_i \circ \zeta_i)(t)\| \theta_i(t) \quad (5.13)$$

$$= (\sigma_i \circ \zeta_i)(t) \theta_i(t)$$

$$v_i(\hat{t}) = \frac{1}{T_i} (\sigma_i \circ \zeta_i)(\hat{t}) \theta_i(\hat{t})$$

$$0 \leq v_{min} \leq v(t) \leq v_{max} \quad (5.14)$$

Remark. *The velocity is required to be positive to ensure the relationship between the timing law and path remains stable. It is recommended that the trajectories of aircraft that are able to fly laterally or backwards be described through the use of the aerodynamic offset angles.*

Acceleration

The change in velocity, or translational acceleration, is another critical consideration that is specific to each aircraft. The translational acceleration along the spatial curve is defined as follows:

$$\mathbf{a}_i(t) = \mathbf{p}'_i(\zeta_i(t)) \dot{\theta}(t) + \mathbf{p}''_i(\zeta_i(t)) \theta_i^2(t) \quad (5.15)$$

The accelerations can be decomposed into the scalar components, i.e. the tangential (a^T) and normal acceleration (a^N):

$$\mathbf{a} = a^T \mathbf{T} + a^N \mathbf{N}_p \quad (5.16)$$

Tangential acceleration, or change in velocity, is the acceleration along the curve tangent ($\mathbf{T}$), and is computed as:

$$a_i^T(t) = \frac{\mathbf{v}_i(t) \cdot \mathbf{a}_i(t)}{\|\mathbf{v}_i(t)\|} \quad (5.17)$$

$$= \frac{(\mathbf{p}'_i \circ \zeta_i)\theta_i(t) \cdot [\mathbf{p}'_i(\zeta_i(t))\dot{\theta}(t) + \mathbf{p}''_i(\zeta_i(t))\theta_i^2(t)]}{(\sigma_i \circ \zeta_i)(t)\theta_i(t)} \quad (5.18)$$

This can however be reduced to a one dimensional function as shown below:

$$a_i^T(t) = (\sigma'_i \circ \zeta_i)(t)\theta_i^2(t) + (\sigma_i \circ \zeta_i)(t)\theta'_i(t) \quad (5.19)$$

$$a_i^T(\hat{t}) = \frac{1}{T_i^2} [(\sigma'_i \circ \zeta_i)(\hat{t})\theta_i^2(t) + (\sigma_i \circ \zeta_i)(\hat{t})\theta'_i(\hat{t})]$$

Generally the approach is to use a symmetric limit as shown in (5.20).

$$|a^T(t)| \leq a_{max} \quad (5.20)$$

It would however be more suitable to place asymmetric limits on the tangential acceleration as shown in (5.21) to better utilise the capabilities of the aircraft, allowing for distinction between aircraft with greater thrust and those that have reverse thrust or air braking capabilities.

$$a_{drag} \leq a^T(t) \leq a_{thrust} \quad (5.21)$$

The normal (or centripetal) acceleration is due to the curvature of the spatial curve, and is directed along the principle normal ($\mathbf{N}_p$).

$$a_i^N(t) = \kappa_i(t)v_i^2(t) = \frac{|\mathbf{v}_i(t) \times \mathbf{a}_i(t)|}{\|\mathbf{v}_i(t)\|} \quad (5.22)$$

$$= \frac{|(\mathbf{p}'_i \circ \zeta_i)\theta_i(t) \times [\mathbf{p}'_i(\zeta_i(t))\dot{\theta}(t) + \mathbf{p}''_i(\zeta_i(t))\theta_i^2(t)]|}{(\sigma_i \circ \zeta_i)(t)\theta_i(t)} \quad (5.23)$$

Limiting the curvature is very restrictive as the turn rates are greatly affected by the current airspeed of the aircraft and the aerodynamic conditions. It is a better option to

limit the centripetal acceleration, which directly combines the curvature with the airspeed, as shown in (5.22). Both the curvature and velocity are always positive, hence the single limitation.

$$a^N \leq a_{cent_max} \quad (5.24)$$

Climb rate

The climb rate is another consideration that is primarily determined by the excess thrust capability of the aircraft. Once again it is preferable to use an asymmetric limit, shown in (5.25), since the rate of descent is assisted by gravity, making it generally much larger than the rate of ascent.

$$\dot{z}_{descent} \leq \dot{p}_D \leq \dot{z}_{climb} \quad (5.25)$$

The climb rate is defined as the relationship between the velocity and the flight path angle, however it can be calculated directly from the velocity vector ($\mathbf{p}'_i(t)$) which is readily available from the trajectory description. Equation (5.26) shows the function used to evaluate the climb rate.

$$\begin{aligned} \dot{p}_D(t) &= (p_{D_i}' \circ \zeta)(t)\theta(t) \\ \dot{p}_D(\hat{t}) &= \frac{1}{T_i}(p_{D_i}' \circ \zeta)(\hat{t})\theta(\hat{t}) \end{aligned} \quad (5.26)$$

5.2.3 Angular Rates

The second set of temporal constraints relate to the aircraft angular motion, and is dependent on the frame of reference. These include the Euler angle rates ($\dot{\phi}$, $\dot{\gamma}$, $\dot{\psi}_w$) and wind frame rates (P_w , Q_w , R_w).

Turn Rate

The turn rate of the aircraft is commonly used as an alternative to the curvature or centripetal acceleration. The turn rate ($\dot{\psi}_w$) is computed as follows:

$$\frac{d}{dt}\psi_{wi}(t) = \frac{(\mathbf{p}'_i(\zeta_i(t)) \times \mathbf{p}''_i(\zeta_i(t))) \cdot \hat{\mathbf{k}}}{\left(\mathbf{p}'_i(\zeta_i(t)) \cdot \hat{\mathbf{i}}\right)^2 + \left(\mathbf{p}'_i(\zeta_i(t)) \cdot \hat{\mathbf{j}}\right)^2} \theta_i(t) \quad (5.27)$$

The turn rate is generally applied symmetrically using a single limit on the magnitude, however it is possible to apply an asymmetric limit on this property to account for possible deviations in performance as a result of damage or some other disturbance.

$$|\dot{\psi}_i| \leq \dot{\psi}_{max} \quad (5.28)$$

Pitch Rate

The earth frame pitch rate, or more precisely the rate of change in the flight path angle, can also be limited and is evaluated using (5.29).

$$\frac{d}{dt}\gamma_i(t) = \frac{(\sigma_i \circ \zeta_i)(t)((\mathbf{p}''_i \circ \zeta_i)(t) \cdot \hat{\mathbf{k}}) - (\sigma'_i \circ \zeta_i)(t)((\mathbf{p}'_i \circ \zeta_i)(t) \cdot \hat{\mathbf{k}})}{(\sigma_i \circ \zeta_i)(t) \sqrt{\left((\mathbf{p}'_i \circ \zeta_i)(t) \cdot \hat{\mathbf{i}}\right)^2 + \left((\mathbf{p}'_i \circ \zeta_i)(t) \cdot \hat{\mathbf{j}}\right)^2}} \theta_i(t) \quad (5.29)$$

In the above equation σ_i is defined as follows:

$$\sigma'_i(\zeta_i) = \frac{d}{d\zeta_i} \|\mathbf{p}'_i(\zeta_i)\| = \frac{(\mathbf{p}'_i(\zeta_i) \cdot \mathbf{p}''_i(\zeta_i))}{\|\mathbf{p}'_i(\zeta_i)\|} \quad (5.30)$$

The limiting of the pitch rate can either be done using symmetric or asymmetric limits, but generally a symmetric approach is suggested.

$$|\dot{\gamma}(t)| \leq \dot{\gamma}_{max} \quad (5.31)$$

Roll Rate

The earth frame roll rate, is computed directly from the quaternion polynomial $\mathcal{A}(\zeta) = u(\zeta) + v(\zeta)\hat{\mathbf{i}} + p(\zeta)\hat{\mathbf{j}} + q(\zeta)\hat{\mathbf{k}}$ as follows:

$$\frac{d}{dt}\phi_i(t) = \frac{-v_i(\zeta_i)u'_i(\zeta_i) + u_i(\zeta_i)v'_i(\zeta_i) + q_i(\zeta_i)p'_i(\zeta_i) - p_i(\zeta_i)q'_i(\zeta_i)}{(\sigma_i \circ \zeta_i)(t)} \theta_i(t) \quad (5.32)$$

Wind Frame Angular Rates

The combined modelling of the orientation along trajectory has the advantage that the kinematic rates of the wind (aerodynamic) frame are directly available from the frame specification of the hodograph curve, $\mathcal{A}(\zeta)$. These angular rates can be calculated through the simple quaternion product of the unit quaternion frame and its derivative, however since the quaternion function does not maintain unit length, the equation yields a rational polynomial function.

$$\begin{aligned}\boldsymbol{\omega}_{w_i}(\zeta) &= 2\mathcal{U}_i^*(\zeta)\mathcal{U}_i'(\zeta) \\ &= \frac{2\mathcal{A}_i^*(\zeta)\mathcal{A}_i'(\zeta) - \sigma_i'(\zeta)}{\sigma_i(\zeta)}\end{aligned}$$

An interesting observation is that the final term $\frac{\sigma_i'(\zeta)}{\sigma_i(\zeta)}$ is a pure scalar value and therefore does not affect the values of $\boldsymbol{\omega}$ that are of interest, since $\boldsymbol{\omega}_w$ is a pure vector quaternion of the form $\boldsymbol{\omega}_w = 0 + P_w\hat{\mathbf{i}} + Q_w\hat{\mathbf{j}} + R_w\hat{\mathbf{k}}$. Therefore the computation of the final term can be omitted in implementation and the scalar part of $\boldsymbol{\omega}_w$ can simply be ignored.

Since the above function is defined as a function of the spline index ζ , it is necessary to relate the angular rates to the timing law just as the velocity was.

$$\begin{aligned}\boldsymbol{\omega}_{w_i}(t) &= 2(\mathcal{U}_i^* \circ \zeta_i)(t)(\mathcal{U}_i' \circ \zeta_i)(t)\hat{\theta}_i(t) \\ &= \frac{2(\mathcal{A}_i^* \circ \zeta_i)(t)(\mathcal{A}_i' \circ \zeta_i)(t) - (\sigma_i' \circ \zeta_i)(t)}{(\sigma_i \circ \zeta_i)(t)}\hat{\theta}_i(t)\end{aligned}$$

Remark. The Euler angle rates can also be computed using $\boldsymbol{\omega}_w(t)$ by performing angular transformation using γ and ϕ .

$$\begin{bmatrix} \dot{\phi} \\ \dot{\gamma} \\ \dot{\psi}_w \end{bmatrix} = \begin{bmatrix} 1 & \tan \gamma \sin \phi & \tan \gamma \cos \phi \\ 0 & \cos \phi & -\sin \phi \\ 0 & \sin \phi / \cos \gamma & \cos \phi / \cos \gamma \end{bmatrix} \begin{bmatrix} P_w \\ Q_w \\ R_w \end{bmatrix}$$

5.2.4 Separation Constraints

The physical separation between two objects can be analysed with greater accuracy than standard obstacle avoidance and path planning techniques. By describing the path of

another vehicle or the shape of a obstacle using Bézier splines, the distance can be calculated for both spatial and temporal separation. Note that due to the computational nature of Bernstein polynomials, it is preferable to compare the square of the distance to the square of the separation limit, as this negates the need to calculate the square root of the polynomial. The techniques presented here are exact separation calculations of the separation distance, and more generalised computations are used for collision detection in later sections.

Spatial Separation

Spatial separation is the geometric distance between two splines and is independent of time. This analysis compares the distance between two splines in reference to both spline indexes, effectively comparing the distance between each point along one line to each point on the other, creating a 2D distance map. Spatial separation is ensured when the minimum distance is greater than the required separation distance. The separation between two vehicles, a and b in a group of v vehicles, is computed using (5.33).

$$\min_{\substack{a,b=1\dots v \\ a \neq b}} \|\mathbf{p}_{a,i}(\zeta_i) - \mathbf{p}_{b,j}(\zeta_j)\|^2 \geq E^2, \quad \zeta_i, \zeta_j \in [0, 1] \quad \forall i \in [0, N_a], j \in [0, N_b] \quad (5.33)$$

Temporal Separation

Temporal separation ensures that two aircraft never collide, allowing their trajectories to intersect, or even fly the same geometric trajectory, but never occupying the same space at a given point in time. Since this analysis is only in reference to time, the resulting function is one dimensional and the analysis is actually computationally less complex than the spatial separation.

$$\min_{\substack{a,b=1\dots v \\ a \neq b}} \|\mathbf{p}_{a,i}(\zeta_i(t)) - \mathbf{p}_{b,j}(\zeta_j(t))\|^2 \geq E^2, \quad t \in [0, t_{ab}^f] \quad \forall i \in [0, N_a], j \in [0, N_b] \quad (5.34)$$

Here $t_{ab}^f = \min\{t_a^f, t_b^f\}$, the final flight time of the aircraft that finishes its trajectory first.

5.2.5 Operational Constraints

The operational constraints are based not on the dynamics of the craft, instead they are used to ensure desired behaviour and safe operation of the aircraft.

Altitude

The limit in altitude is determined by the aircraft flight ceiling (aerodynamic) and the operational limits from air traffic control. Along with these considerations, the terrain and obstacles in the area of operation also determine the selection of the minimum altitude.

$$h_{min} \leq -p_D \leq h_{max} \quad (5.35)$$

Positional Boundaries

Positional constraints can also be placed on the aircraft ensuring the aircraft stays within a specific area. This can be done for regulatory or safety considerations, or based on the usable range of the aircraft as a quick means of ensuring feasibility. Positional constraints can also be used to ensure the aircraft remains within communications range or within line of sight. These positional constraints can be in the shape of a box,

$$N_{min} \leq p_N \leq N_{max} \quad (5.36)$$

$$E_{min} \leq p_E \leq E_{max} \quad (5.37)$$

or defined as an operational radius around a center point $\mathbf{p}_c$,

$$\max_{i=1,\dots,N} \|\mathbf{p}_i(\zeta_i) - \mathbf{p}_c\|^2 \leq R^2. \quad (5.38)$$

It is also possible to define a moving boundary following a moving point, who's trajectory is also defined using a Bézier spline,

$$\max_{\substack{a,b=1\dots v \\ a \neq b}} \|\mathbf{p}_{a,i}(\zeta_i(t)) - \mathbf{p}_{b,j}(\zeta_j(t))\|^2 \leq R^2, \quad (5.39)$$

which would be practical for systems operating from moving platforms such as naval vessels. Note that the comparison is done to the square of the radius, negating the need to calculate the square root of the spline products, which greatly improves the computational efficiency.

5.2.6 Non-linear Constraints

As demonstrated with some of the constraints, the use of asymmetric constraints allows for some interesting applications. Since the aircraft dynamics are asymmetric in many of the cases, this allows the system to utilise a larger portion of the flight envelope rather than limiting it to the smaller value. The capabilities can be further increased if the algorithm compensates for the non-linear nature of the dynamics, as the abilities of the aircraft varies greatly based on various parameters. For example, considering how the mass limits the manoeuvrability, the aircraft abilities would normally be limited to the worst case of maximum weight if a static constraint is chosen. The amount of lift that can be generated is determined by the airspeed, thus the system would be limited in manoeuvrability if a conservative constraint is used. Similarly, the maximum climb rates are dependent on the excess power, which varies with altitude. The principle of dynamically adjusting the kinodynamic constraints based on the current state of the aircraft and operational environment, allows the trajectory generation algorithm to operate in a wide array of applications. From an operational constraint perspective, consider the advantages of applying orientation constraints on an aircraft during take-off and landing operations. The system can also be used for automated taxiing manoeuvres, regulating airspeed in controlled airspace, ensuring passenger comfort, or relaxing the constraints in emergencies to increase manoeuvrability. Any damage to the aircraft or component failures that result in reduced or even asymmetric performance could easily be included in the algorithm to allow for safe and predictable operation.

5.3 Performance Evaluation

The kinodynamic constraints serve as limits to ensure that the trajectory is flyable for a given aircraft, but the optimisation of the trajectory requires the evaluation of the trajectory based on some performance measure. These performance evaluators can be adjusted based on the required application, optimising for example fuel cost, time of flight, comfort, etc. Each segment of a given trajectory can also be optimised based on different objectives, such as minimising for take-off and landing distance, while maintaining kinematic continuity between segments. Other more advanced evaluators can be derived to address specific missions such as minimising radar exposure for stealth, maximising the search area, or improving collaboration by evaluating the combined performance of a swarm.

There are a few fundamental performance evaluators that are inherently included in the trajectory that are not covered in detail here. These include the arc-length of the trajectory (i.e. distance) and time of flight, as they are trivial to include. The evaluation of object separation can also be used as an objective for target tracking or threat avoidance.

The performance evaluation is highly aircraft specific, requiring aerodynamic and thrust models. There are however a few performance metrics that can directly be derived from the trajectory to serve as a cursory metric for initial optimisation. These include change in altitude, specific energy and power, risk and miscellaneous mission objectives.

5.3.1 Altitude Change

In order to reduce unnecessary changes in altitude, the optimisation algorithm can evaluate the accumulated altitude change along the trajectory.

$$\Delta Z = \int_0^1 |p'_z(\zeta)| d\zeta \quad (5.40)$$

5.3.2 Energy

For systems requiring more advanced manoeuvrability which exceeds the sustainable performance levels, it is necessary to keep track of the aircraft's energy. Evaluating the energy of the aircraft allows for faster evaluation of manoeuvre selection in highly dynamic environments through the principles of energy-maneuvrability theory. If the specific energy state of the aircraft is known, higher level tactical controllers can make quick decisions regarding which manoeuvre options are possible. Rather than having to evaluate various scenarios for feasibility in real-time, the computational focus can be placed on the tactical consequences of each manoeuvre.

The specific energy (h_e) can be computed directly from the trajectory, as this is independent of mass, as shown below:

$$\begin{aligned} h_e(t) &= h + \frac{v^2}{2g} \\ &= -p_{D_i}(t) + \frac{v_i^2(t)}{2g} \end{aligned} \quad (5.41)$$

Similarly, the specific power can also be computed directly as follows:

$$\begin{aligned} P_s(t) &= \dot{h} + \frac{v\dot{v}}{g} \\ &= -\dot{p}_D(t) + \frac{v_i(t)a_i^T(t)}{g} \end{aligned} \quad (5.42)$$

Evaluating the specific excess power as a constraint, does however require modelling of the aircraft's thrust and mass.

5.3.3 Tracking Error Risk

When using a larger portion of the flight envelope, the controllers are getting closer to the limits of the craft's capabilities and thus there exists a greater risk that disturbances or variations in performance due to turbulence can result in deviations from the reference trajectory. If the aircraft is for example following a trajectory that requires it to perform a sharp turn that is on the limit of its abilities, a sudden large disturbance such as a surge in

wind, may increase the turning radius beyond the point where the aircraft can compensate to rejoin the reference trajectory. If it is needed to utilise such a large portion of the flight envelope, it is possible to add a risk function that represent the risk of deviation from the reference trajectory, which increases the closer the manoeuvre comes to the limits of the aircraft. Alternatively multiple possible trajectories can be generated with reduced turn rates, that ensure should the aircraft be unable to perform the tight turn it would not result in a collision, and allowing for fast response and correction should the aircraft fail to follow the preferred trajectory.

5.3.4 Mission Objectives

Through the inclusion of the orientation in the trajectory description, the impact of the orientation on mission efficiency can also be evaluated. When constructing mission specific evaluators, it is now possible to ensure that the target is not only within a specific range, but that the target is aligned with the sensor payload, such as a camera system. Even considering gimbal mounted camera and communication systems, excessive bank angles can result in losing line of sight or loosing the communications uplink. Including these sort of heuristics into the trajectory generation can dramatically improve mission efficiency and reliability.

Another consideration is the optimisation of aerial photography for the purpose of mapping terrain or structures. Through the simultaneous evaluation of the mission and flight efficiency, the algorithm can ensure not only the most efficient route to map the area, but also ensure sufficient tolerances to ensure that disturbances don't affect the mission outcome. Consider for example a quad rotor MAV taking images of a tall building for 3D modelling by following a helical trajectory to spirals upwards around the building. If the camera was mounted on the side or underneath the aircraft, it would have to maintain a fairly small pitch and roll angles, limiting the velocity, however if the camera was mounted on top of the aircraft it would be allowed to travel much faster since more aggressive pitch and roll angles would be favoured. This is the purpose of including the orientation in the trajectory generation and subsequent evaluation of the mission efficiency, to allow

autonomous systems to make these distinctions.

5.4 Conclusion

In this chapter various evaluation metrics were presented that can be used during optimisation depending on the required objective, operating condition or aircraft type. This wide range of metrics allow for increase versatility in multi-objective optimisation through dynamic cost function specification, and is critical to the universality of this approach. In summary, these metrics include:

- *Geometric parameters:* Curvature, torsion, heading, flight path angle, roll angle, and bending energy.
- *Translational motion parameters:* Velocity, acceleration (tangential and centripetal), and climb rate.
- *Angular rate parameters:* Turn rate, pitch rate, roll rate, and wind frame angular rates.
- *Separation parameters:* Geometric and temporal separation.
- *Operational parameters:* Altitude and positional boundaries.
- *Performance parameters:* Altitude change, specific energy and specific power.

The geometric parameters are well suited to initial optimisation of the spatial curve during path planning, producing smooth continuous curves for the next stages. Subsequently temporal parameters are used to ensure the trajectory is flyable, using purely kinematic constraints. Final optimisation can then be done by incorporating the performance metrics, or a performance model, to optimise the dynamic and mission performance of the aircraft. This sequential optimisation allows the algorithm to first locate the general location of the global minimum using geometric evaluation, i.e. finding the optimal route while accounting for curve continuity and curvature. This reduces the computation cost of optimisation during the more complex stages of trajectory generation, ultimately ensuring faster trajectory generation.

Chapter 6

Case Study

If at first you don't succeed, try, try, try again.

– William Edward Hickson

6.1 Introduction

In this chapter a case study is presented to demonstrate the capabilities of this approach to a non-linear aircraft model. The aircraft used in this case study is an F16 single engine fighter aircraft, chosen for the agility and availability of high fidelity non-linear models. The model used in this case study is based on the model found in [13], with the simulation implementation derived from [79]. This model is a lower fidelity model than that of [14], and omits the leading-edge flap and speed (air) brake. The model covers the angle of attack range of $-10 \leq \alpha \leq 45$, and side-slip angle range of $-30 \leq \beta \leq 30$.

An overview of the model is presented to give insight into its structure and to serve as a reference for the introduction of the inverse model. At the hand of the inverse model, a few performance metrics are computed to demonstrate the fundamental principle behind the use of non-linear kinodynamic constraints. These performance metrics combine to form the aircraft performance model used to approximately verify that the trajectory is

flyable, and evaluate the efficiency of the proposed trajectory. This is then combined and applied to a trajectory to demonstrate the integration of these aspects, and to validate the results.

6.2 F16 Model

In this section the details of the model used in this study is presented.

6.2.1 Generic Aircraft Model

The non-linear state space model in [13] is defined by the following state vector:

$$\mathbf{X} = [p_N \ p_E \ p_D \ \phi \ \theta \ \psi \ v \ \alpha \ \beta \ P \ Q \ R]^T \quad (6.1)$$

The components of this state vector are as follows:

- Position in earth frame: $\mathbf{p} = [p_N \ p_E \ p_D]^T$
- Orientation in Euler angles: $\boldsymbol{\Theta} = [\phi \ \theta \ \psi]^T$
- Velocity: $v = \|\mathbf{v}_b\|$
- Aerodynamic angles: Angle of attack (α) and angle of side-slip (β)
- Body frame angular rates: $\boldsymbol{\omega}_b = [P \ Q \ R]^T$

The model is composed of various stages, the first being the kinematic model consisting of the navigational equations in (6.3) and Euler kinematic equations in (6.4).

$$C_{b/e} = \text{DCM}(\boldsymbol{\Theta}) \quad (6.2)$$

$$\dot{\mathbf{p}} = C_{e/b} \mathbf{v}_b \quad (6.3)$$

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \tan \theta \sin \phi & \tan \theta \cos \phi \\ 0 & \cos \phi & -\sin \phi \\ 0 & \sin \phi / \cos \theta & \cos \phi / \cos \theta \end{bmatrix} \begin{bmatrix} P \\ Q \\ R \end{bmatrix} \quad (6.4)$$

The kinematic equations in (6.3) and (6.4) can be replaced by the quaternion form in (6.5).

$$\begin{aligned}\dot{\mathcal{Q}}_{b/e} &= \frac{1}{2} \mathcal{Q}_{b/e} \omega_{\mathbf{b}} \\ \dot{\mathbf{p}} &= \mathcal{Q}_{b/e} \mathbf{v}_b \mathcal{Q}_{b/e}^*\end{aligned}\tag{6.5}$$

The dynamic component of the model consists of the force (6.6) and moment (6.7) equations. The force equations consist of the Coriolis acceleration ($\omega_b \times \mathbf{v}_b$), gravity ($\mathbf{g}$), thrust (T) and the aerodynamic forces ($[F_x \ F_y \ F_z]^T$).

$$\begin{aligned}\dot{U} &= RV - QW - g \sin \theta + \frac{F_x}{m} + \frac{T}{m} \\ \dot{V} &= PW - RU + g \sin \phi \cos \theta + \frac{F_y}{m} \\ \dot{W} &= QU - PV + g \cos \phi \cos \theta + \frac{F_z}{m}\end{aligned}\tag{6.6}$$

The moment equations consist of the gyroscopic coupling ($\omega_b \times J \omega_b$), and the aerodynamic moments ($[L \ M \ N]^T$).

$$\begin{aligned}\dot{P} &= (c_1 R + c_2 P)Q + c_3 L + c_4 N \\ \dot{Q} &= c_5 PR - c_6(P^2 - R^2) + c_7 M \\ \dot{R} &= (c_8 P - c_2 R)Q + c_4 L + c_9 N\end{aligned}\tag{6.7}$$

Here the coefficients c_x represent the rotational inertia and inertial coupling present in the system.

$$\begin{aligned}\Gamma c_1 &= (J_y - J_z)J_z - J_{xz}^2 & \Gamma c_2 &= (J_x - J_y + J_z)J_{xz} \\ \Gamma c_3 &= J_z & \Gamma c_4 &= J_{xz} \\ c_5 &= \frac{J_z - J_x}{J_y} & c_6 &= \frac{J_{xz}}{J_y} \\ c_7 &= \frac{1}{J_y} & \Gamma c_8 &= J_x(J_x - J_y) + J_{xz}^2 \\ \Gamma c_9 &= J_x & \Gamma &= J_x J_z - J_{xz}^2\end{aligned}$$

The following auxiliary equations are also needed to compute the progression of the

aerodynamic angles:

$$\mathbf{v}_b = v \begin{bmatrix} \cos \alpha \cos \beta \\ \sin \beta \\ \sin \alpha \cos \beta \end{bmatrix} \quad \dot{v} = \frac{U\dot{U} + V\dot{V} + W\dot{W}}{v}$$

$$\dot{\beta} = \frac{\dot{V}v - V\dot{v}}{v^2 \cos \beta} \quad \dot{\alpha} = \frac{U\dot{W} - W\dot{U}}{U^2 + W^2}$$

6.2.2 Aerodynamic Coefficients

The generic model presented above requires the computation of the aerodynamic forces and angles, using lookup tables of the aerodynamic coefficients. These coefficients are then combined with the standard atmosphere model, to determine the dynamic pressure ($\bar{q}$) used to compute the resulting forces and moments. The aerodynamic forces and moments are computed as follows:

$$\text{drag, } F_x = \bar{q}SC_X$$

$$\text{lift, } F_z = \bar{q}SC_Z$$

$$\text{sideforce, } F_y = \bar{q}SC_Y$$

$$\text{rolling moment, } L = \bar{q}SbC_L$$

$$\text{pitching moment, } M = \bar{q}S\bar{c}C_M$$

$$\text{yawing moment, } N = \bar{q}SbC_N$$

where $\bar{q}$ is the free-stream dynamic pressure, S is the wing reference area, b is the wing span, and $\bar{c}$ is the mean geometric chord of the wing. The aerodynamic coefficients are computed as functions of the angle of attack (α), side-slip angle (β), body angular rates ($\boldsymbol{\omega}_b$), and the control surface deflections. The control surface deflections being the elevator (δ_e), ailerons (δ_a) and rudder (δ_r) respectively. These coefficient consist of the primary geometric coefficients ($C_{\diamond}$), damping coefficients ($D_{\diamond}$), and control surface efficiency coefficients ($C_{\delta_{\diamond}}$). The remaining factor that affects the coefficients is the torque

produced by the offset of the center of gravity ($x_{cgr} - x_{cg}$) from the aerodynamic center.

$$\begin{aligned}
C_X &= C_x(\alpha, \delta_e) + \frac{\bar{c}D_{xq}(\alpha)}{2v}Q \\
C_Y &= C_y(\beta, \delta_a, \delta_r) + \frac{\bar{c}D_{yr}(\alpha)}{2v}R + \frac{\bar{c}D_{yp}(\alpha)}{2v}P \\
C_Z &= C_z(\alpha, \beta, \delta_e) + \frac{\bar{c}D_{zq}(\alpha)}{2v}Q \\
C_L &= C_l(\alpha, \beta) + \frac{bD_{lr}(\alpha)}{2v}R + \frac{bD_{lp}(\alpha)}{2v}P + C_{\delta la}(\alpha, \beta)\delta_a + C_{\delta lr}(\alpha, \beta)\delta_r \\
C_M &= C_m(\alpha, \delta_e) + C_Z(x_{cgr} - x_{cg}) + \frac{\bar{c}D_{mq}(\alpha)}{2v}Q \\
C_N &= C_n(\alpha, \beta) - C_Y(x_{cgr} - x_{cg})\frac{\bar{c}}{b} + \frac{bD_{nr}(\alpha)}{2v}R + \frac{bD_{np}(\alpha)}{2v}P + C_{\delta na}(\alpha, \beta)\delta_a + C_{\delta nr}(\alpha, \beta)\delta_r
\end{aligned} \tag{6.8}$$

6.2.3 Wind Frame vs Body Frame

The trajectory of the aircraft is determined by the wind frame, which is offset from the body frame by the aerodynamic angles. The aerodynamic angles are also the controlling inputs of the various aerodynamic coefficients, alongside the control surface deflections. The DCM relating converting between the body and wind frame is given by:

$$C_{w/b} = \begin{bmatrix} \cos \alpha \cos \beta & \sin \beta & \sin \alpha \cos \beta \\ -\cos \alpha \sin \beta & \cos \beta & -\sin \alpha \sin \beta \\ -\sin \alpha & 0 & \cos \alpha \end{bmatrix} \tag{6.9}$$

Relating wind frame angular rates (ω_w) to the body angular rates (ω_b) requires the following transformation, which includes compensating for the rate of change in the aerodynamic angles.

$$\begin{bmatrix} P_w \\ Q_w \\ R_w \end{bmatrix} = \begin{bmatrix} \cos \alpha \cos \beta & \sin \beta & \sin \alpha \cos \beta \\ -\cos \alpha \sin \beta & \cos \beta & -\sin \alpha \sin \beta \\ -\sin \alpha & 0 & \cos \alpha \end{bmatrix} \begin{bmatrix} P_b - \dot{\beta} \sin \alpha \\ Q_b - \dot{\alpha} \\ R_b + \dot{\beta} \cos \alpha \end{bmatrix} \tag{6.10}$$

6.3 Inverse Model

The state space model above computes the motion on the aircraft as a function of the control surface deflections and thrust. During trajectory generation, specifically the geometric dynamic approaches, the position of the craft is specified using a continuous polynomial. Essentially the state vector is partially defined, and by deriving the remaining values, the model can be solved in reverse. As shown in [12], specifying only the position ($\mathbf{p}(t)$) is insufficient to bind the system to a single solution. It is necessary to add an addition state, either binding the angle of side-slip ($\beta(t)$), or as in this case the roll angle ($\phi(t)$). Given that the position is specified as a continuous polynomial, the velocity vector ($\dot{\mathbf{p}}(t)$) is also readily available, which includes the heading ($\psi_w(t)$), flight path angle ($\gamma(t)$) and velocity ($v(t)$).

6.3.1 Kinematic Inversion

The presented approach using a quaternion PH curve concisely includes all these parameters in the quaternion polynomial $(\mathcal{A} \circ \zeta)(t)$. In order to complete the state space model, the accelerations must be derived. The challenge here is that the body frame is not known and only the development of the wind frame is presented. Therefore the kinematic model must be derived from the perspective of the wind reference frame.

Translating the state space model to the wind frame, yields the following model:

$$C_{w/e} = \text{DCM}(\phi, \gamma, \psi_w) \quad (6.11)$$

$$\mathbf{v}_e = C_{e/w} \mathbf{v}_w = \hat{\mathcal{A}} \mathbf{i} \mathcal{A}^* \quad (6.12)$$

$$= v [\cos \gamma \cos \psi_w \quad \cos \gamma \sin \psi_w \quad -\sin \gamma]^T$$

$$\boldsymbol{\omega}_w = \frac{2\mathcal{A}^* \mathcal{A}'_i - \sigma'}{\sigma} \quad (6.13)$$

$$\mathbf{a}_b = \mathbf{a}_e - \boldsymbol{\omega}_w \times \mathbf{v}_b \quad (6.14)$$

$$\mathbf{a}_e = \frac{1}{m} C_{e/w} (\mathbf{F}_w + \mathbf{T}_w) + \mathbf{g} \quad (6.15)$$

The parameters that need to be computed from the quaternion polynomial then include the

inertial acceleration ($\mathbf{a}_e$), angular velocity ($\boldsymbol{\omega}_w$), velocity (v), roll angle (ϕ), flight path angle (γ) and heading (ψ_w).

6.3.2 Aerodynamic Angle Approximation

The challenge in relating the wind frame kinematics with the aerodynamic forces and thrust is the offset by the aerodynamic angles. These forces must be balanced with the inertial acceleration, accounting for gravitation acceleration, as follows:

$$\mathbf{T}_w + \mathbf{F}_w = mC_{w/e}(\mathbf{a}_e - \mathbf{g}) \quad (6.16)$$

Solving the model thus requires solving for the roots of α , β and T . The thrust and aerodynamic forces must however still be converted to the wind reference frame.

$$\mathbf{T}_w = T \begin{bmatrix} \cos \beta \cos \alpha \\ \sin \beta \cos \alpha \\ -\sin \alpha \end{bmatrix} \quad (6.17)$$

$$\mathbf{F}_w = \begin{bmatrix} \cos \alpha \cos \beta & \sin \beta & \sin \alpha \cos \beta \\ -\cos \alpha \sin \beta & \cos \beta & -\sin \alpha \sin \beta \\ -\sin \alpha & 0 & \cos \alpha \end{bmatrix} \begin{bmatrix} F_x \\ F_y \\ F_z \end{bmatrix} = -\bar{q}S \begin{bmatrix} D_{rag}(\alpha, \beta) \\ S_{lip}(\alpha, \beta) \\ L_{ift}(\alpha, \beta) \end{bmatrix} \quad (6.18)$$

In order to solve this, a model is needed for the non-linear aerodynamic coefficients.

6.4 Aircraft Performance Model

In order to accelerate the computation of the inverse model, a necessity for trajectory optimisation, the aerodynamic coefficients can be precomputed as a quick reference to get approximate solutions to the aerodynamic angles. The accuracy of these results naturally depends on the complexity of the performance model, as the coefficients depend not only on the aerodynamic angles, but also on the control surface deflections ($\delta_e, \delta_a, \delta_r$), dynamic pressure ($\bar{q}$), velocity (v) and body angular rates ($\boldsymbol{\omega}_b$). The decomposition of the aerodynamic force coefficients in (6.8) show that the primary coefficients depend on the

aerodynamic angles and the related control surface deflections. The additional damping parameters depend on the angle of attack, velocity and body angular velocity.

A basic performance model, that only serves as an estimate of the kinodynamic limits of the aircraft, can be computed based on steady state conditions. Firstly by omitting the damping coefficients, the dependence on the angular velocity and velocity is removed. The force coefficients now still depend on the dynamic pressure and control surface deflections. The forces are then computed into lookup tables with the control surface deflections set to zero, and in reference to the equivalent airspeed. The control surface deflections are deliberately set to zero, as this provides a slightly more conservative constraints than the actual dynamic limit, allowing for improved robustness against disturbances and deviations in performance. These wind frame aerodynamic coefficients, for lift, drag and sideforce, are then computed as follows:

$$\begin{bmatrix} D_{rag}(\alpha, \beta) \\ S_{lip}(\alpha, \beta) \\ L_{ift}(\alpha, \beta) \end{bmatrix} = \begin{bmatrix} \cos \alpha \cos \beta & \sin \beta & \sin \alpha \cos \beta \\ -\cos \alpha \sin \beta & \cos \beta & -\sin \alpha \sin \beta \\ -\sin \alpha & 0 & \cos \alpha \end{bmatrix} \begin{bmatrix} C_x(\alpha) \\ C_y(\beta) \\ C_z(\alpha, \beta) \end{bmatrix} \quad (6.19)$$

The resulting coefficients are listed in Table 6.1, 6.2 and 6.3, and is visualised in Figure 6.1.

Using these tables, estimates of the aerodynamic angles can be derived for each point along the trajectory, by combining these coefficients with the dynamic pressure ($\bar{q}$) and wing area (S). This effectively relates the force to the equivalent airspeed, which reduces the dimensionality by combining the altitude and true airspeed into a single metric. This is also useful during implementation, as the dynamic pressure is readily available during flight, and allows the system to evaluate the trajectory using dynamic environmental conditions. The performance model can also be scaled for changes in aircraft mass, which also affects the manoeuvrability of the aircraft. The relationship between the equivalent airspeed and true airspeed is as follows:

$$V_{EAS} = V_{TAS} \sqrt{\frac{\rho}{\rho_0}} = \sqrt{\frac{2\bar{q}}{\rho_0}} \quad (6.20)$$

Table 6.1: Lift Coefficients

AoA	Side-slip angle (β)						
	-30°	-20°	-10°	0°	10°	20°	30°
-10°	0.54662	0.6621	0.73139	0.75448	0.73139	0.6621	0.54662
-5°	0.17253	0.20909	0.23103	0.23834	0.23103	0.20909	0.17253
0°	-0.072588	-0.087817	-0.096954	-0.1	-0.096954	-0.087817	-0.072588
5°	-0.30047	-0.36358	-0.40145	-0.41407	-0.40145	-0.36358	-0.30047
10°	-0.52812	-0.63775	-0.70353	-0.72545	-0.70353	-0.63775	-0.52812
15°	-0.76264	-0.91753	-1.0105	-1.0414	-1.0105	-0.91753	-0.76264
20°	-0.97554	-1.171	-1.2883	-1.3274	-1.2883	-1.171	-0.97554
25°	-1.1378	-1.365	-1.5013	-1.5467	-1.5013	-1.365	-1.1378
30°	-1.2821	-1.5349	-1.6866	-1.7372	-1.6866	-1.5349	-1.2821
35°	-1.3529	-1.6174	-1.7761	-1.8289	-1.7761	-1.6174	-1.3529
40°	-1.3497	-1.6119	-1.7693	-1.8217	-1.7693	-1.6119	-1.3497
45°	-1.2417	-1.4817	-1.6257	-1.6737	-1.6257	-1.4817	-1.2417

Table 6.2: Drag Coefficients

AoA	Side-slip angle (β)						
	-30°	-20°	-10°	0°	10°	20°	30°
-10°	-0.40282	-0.26751	-0.18373	-0.15537	-0.18373	-0.26751	-0.40282
-5°	-0.33046	-0.17286	-0.074406	-0.040928	-0.074406	-0.17286	-0.33046
0°	-0.31819	-0.15654	-0.055411	-0.021	-0.055411	-0.15654	-0.31819
5°	-0.32624	-0.17047	-0.073272	-0.040242	-0.073272	-0.17047	-0.32624
10°	-0.35251	-0.21194	-0.1249	-0.095423	-0.1249	-0.21194	-0.35251
15°	-0.39269	-0.27639	-0.20553	-0.18174	-0.20553	-0.27639	-0.39269
20°	-0.48953	-0.40932	-0.36236	-0.34692	-0.36236	-0.40932	-0.48953
25°	-0.63526	-0.60013	-0.5829	-0.57781	-0.5829	-0.60013	-0.63526
30°	-0.78705	-0.80245	-0.81858	-0.82513	-0.81858	-0.80245	-0.78705
35°	-0.95019	-1.0163	-1.0659	-1.0841	-1.0659	-1.0163	-0.95019
40°	-1.1055	-1.2177	-1.2975	-1.3262	-1.2975	-1.2177	-1.1055
45°	-1.2063	-1.3458	-1.4436	-1.4786	-1.4436	-1.3458	-1.2063

Table 6.3: Sideforce Coefficients

AoA	Side-slip angle (β)						
	-30°	-20°	-10°	0°	10°	20°	30°
-10°	0.46025	0.32831	0.17069	0	-0.17069	-0.32831	-0.46025
-5°	0.50203	0.36275	0.18997	0	-0.18997	-0.36275	-0.50203
0°	0.50912	0.36869	0.19331	0	-0.19331	-0.36869	-0.50912
5°	0.50446	0.36362	0.19017	0	-0.19017	-0.36362	-0.50446
10°	0.4893	0.34853	0.18106	0	-0.18106	-0.34853	-0.4893
15°	0.4661	0.32507	0.16684	0	-0.16684	-0.32507	-0.4661
20°	0.41019	0.27669	0.13919	0	-0.13919	-0.27669	-0.41019
25°	0.32605	0.20724	0.10031	0	-0.10031	-0.20724	-0.32605
30°	0.23842	0.1336	0.058748	0	-0.058748	-0.1336	-0.23842
35°	0.14423	0.055761	0.015141	0	-0.015141	-0.055761	-0.14423
40°	0.054537	-0.017517	-0.025697	0	0.025697	0.017517	-0.054537
45°	-0.0036427	-0.064146	-0.051452	0	0.051452	0.064146	0.0036427

During inverse simulation, the exact solution is needed along each point, which is useful in evaluating the performance by evaluating drag. Ensuring however the trajectory is flyable, only requires that the trajectory remains within the flight envelope of the aircraft. For this, only the extremes need to be evaluated, rather than solving for the roots of (6.16). Calculating these extremes for a given equivalent airspeed, and maximum allowed angle of attack, yield the flight envelope restrictions shown in Figure 6.2. These graphs, being a subset of the data, show the maximum acceleration achievable at a given equivalent airspeed. Ensuring the trajectory is flyable, hence requires that the acceleration along the trajectory lies below the given limit defined by the extrema of the aerodynamic angles. These limits on the aerodynamic angles can either be operational limits, or derived based on the maximum drag the engine can overcome for a given limit on engine thrust. The maximum sustained angle of attack that achieves the desired change in airspeed, can then be calculated as follows:

$$\begin{aligned}
& \underset{\alpha}{\text{maximize}} \quad D_{rag}(\alpha, \beta) \\
& \text{subject to} \quad T_{max} \cos \alpha \cos \beta - D_{rag}(\alpha, \beta) = m(\dot{v} - g \sin \gamma)
\end{aligned}$$

Note that the max thrust used in this equation can be scaled down to leave additional performance margins to increase robustness. The above equation can be computed conservatively based on the maximum allowed side-slip angle, rather than idealised

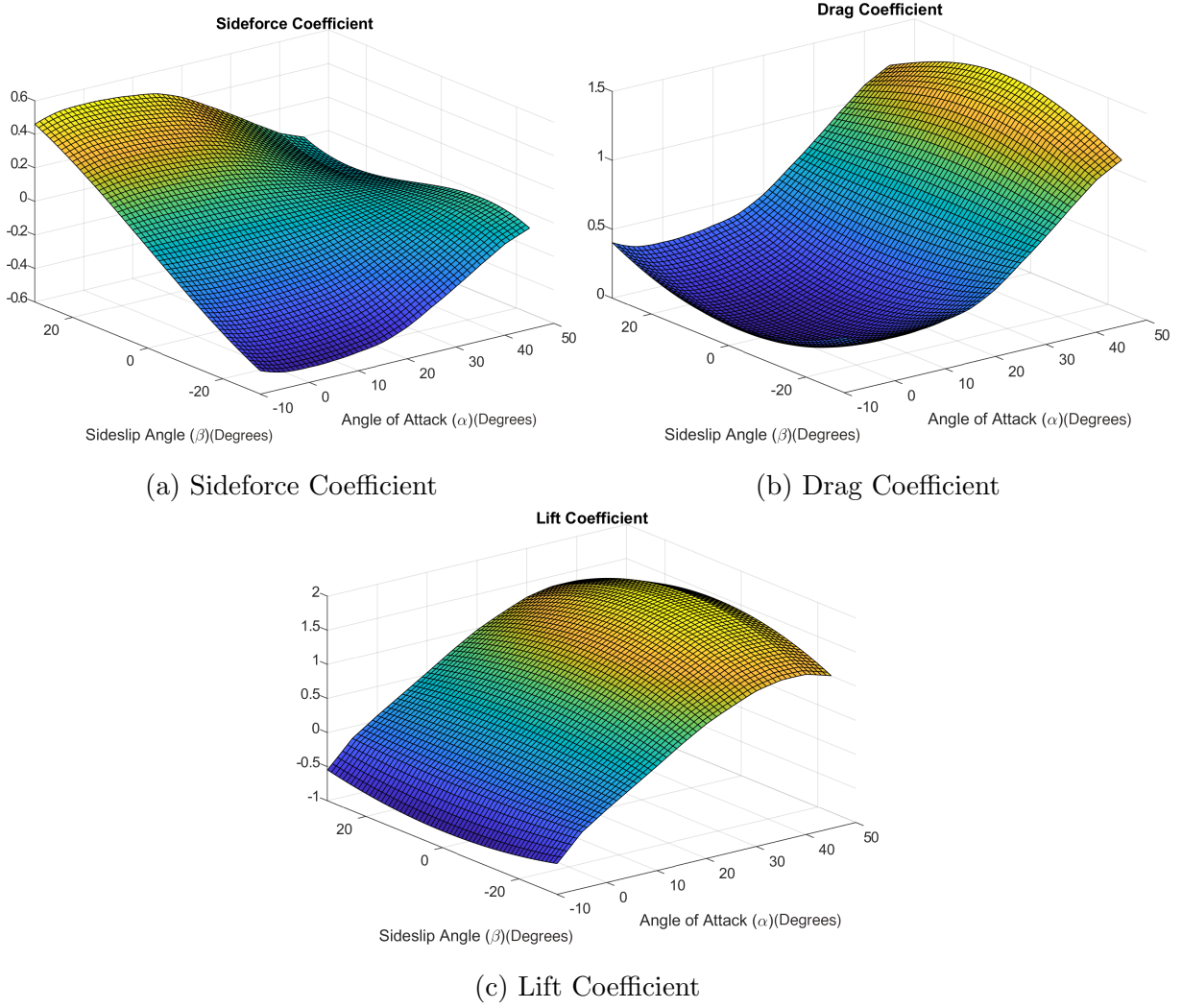


Figure 6.1: Wind Frame Aerodynamic Coefficients

coordinated turns.

Beyond these aerodynamic constraints, there are additional derived performance metrics that reduce the dimensionality of the constraints, in turn reducing the computational cost of feasibility checks. These performance metrics include the stall curves for given angle of attack limits as shown in Figure 6.2d, and is computed using (6.21). In this equation ρ is the air density at the current altitude.

$$v_{stall} = \sqrt{\frac{mg}{0.5\rho SC_Z(\alpha_{max})}} \quad (6.21)$$

The non-linear nature of the engine thrust in relation to altitude and velocity changes, also has a large impact on the flight envelope. The engine model for the F16 in [13]

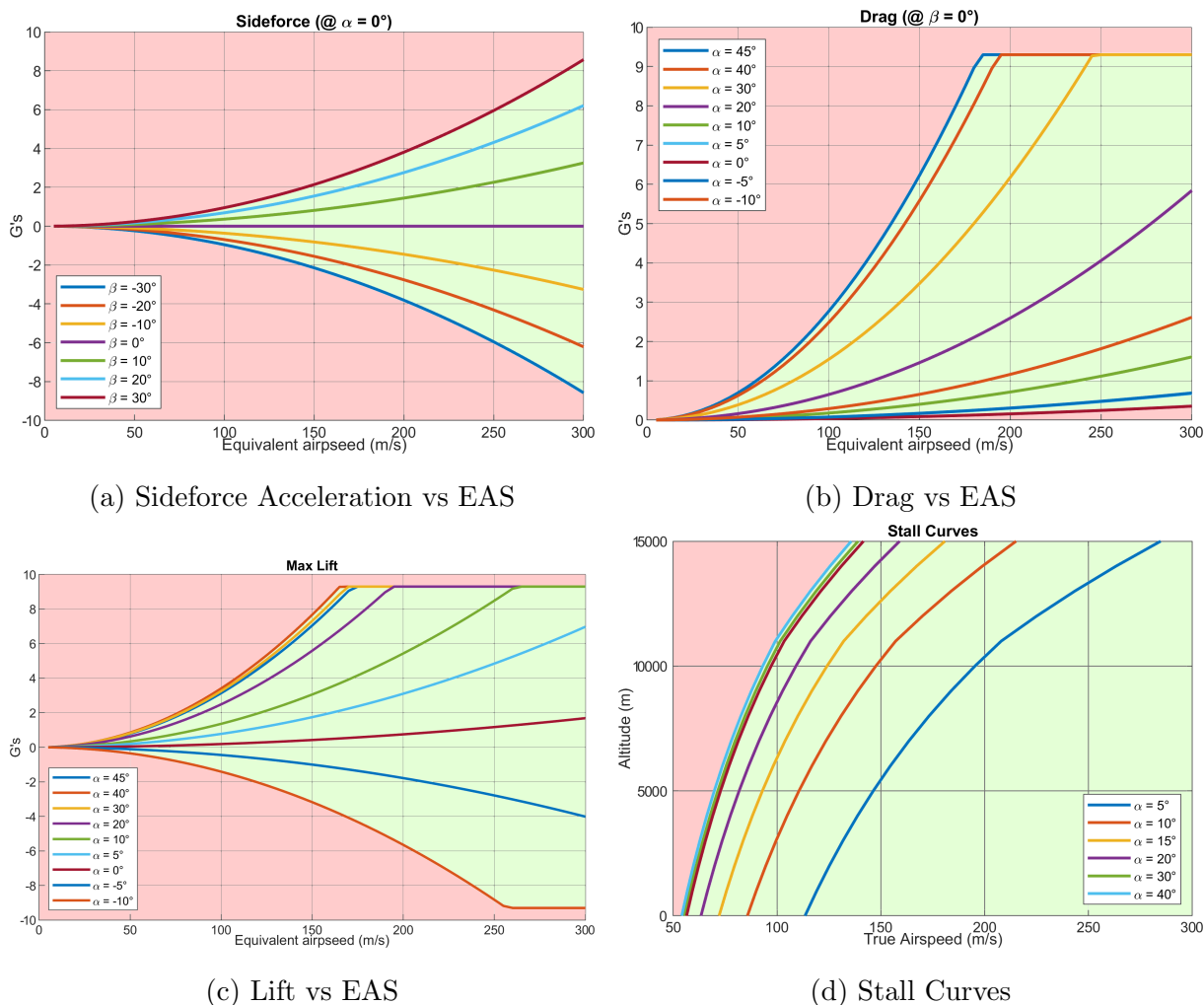


Figure 6.2: Aerodynamic Constraints

has 3 distinct limits, the lower bound being the minimum thrust at idle, the upper bounds being the military thrust (without afterburner) and max thrust (with afterburner). This distinction is useful to demonstrate that the performance model can use different constraints based on operating requirements. These thrust limits are shown in Figure 6.3. The engine thrust model can also be converted to specific excess power for each operating mode, as shown in Figure 6.4, which can directly be used for comparison with the trajectory to ensure feasibility.

The presented performance model is however still limited, as angular rate limits, damping, mass variance and center of gravity offsets are not been included. The purpose of this model is to demonstrate the concept of using precomputed performance metrics as quick

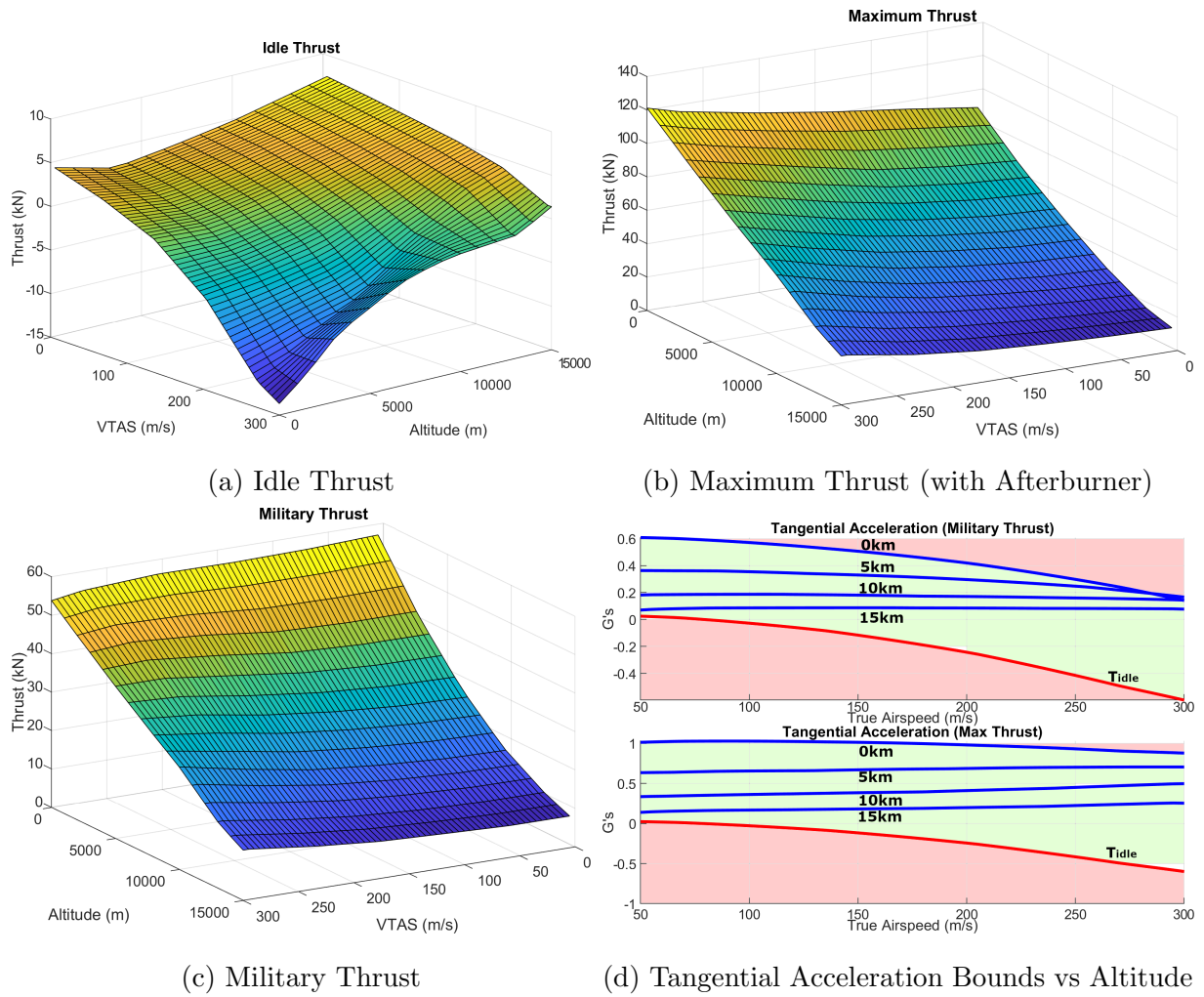


Figure 6.3: Engine Thrust Limits

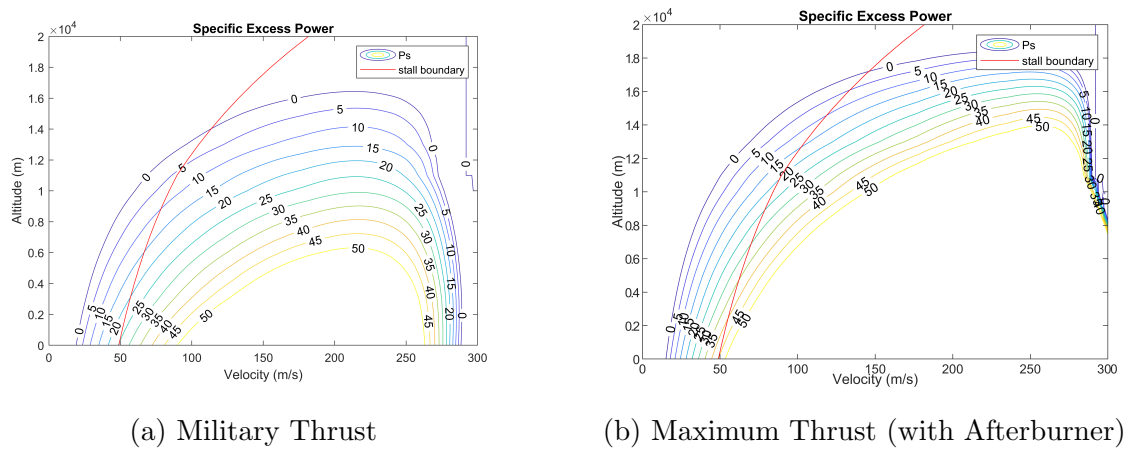


Figure 6.4: Specific Excess Power

references to derive dynamic non-linear constraints for ensuring the trajectory is flyable. Expansions to the model is possible as needed, with increases in fidelity allowing for increased flight envelope utilisation.

6.5 Inverse Simulation

The use of an inverse simulation to model the aircraft performance along a trajectory requires modelling of the aerodynamic characteristics of the aircraft. Directly using the a high fidelity non-linear model such as presented in section 6.2, results in a non-linear optimisation problem to solve for all the various interacting coefficients. Due to the extensive interaction of the variables, and the under-actuated nature of the system, the inverse simulation could have multiple solutions, often converging to local minima. It was also observed that when solving in single points, the system would converge to minima that don't accurately account for the rate limits (continuity) in the control surface deflections between iterations. Therefore the solution of the inverse computation, requires accounting for smooth control input changes between iteration steps. This requires expanding the time horizon, and greatly depends on the accuracy of the initial approximation of the aerodynamic angles and other states.

Essentially the inverse model, due to it's extensive non-linearity, expanded time horizon and subsequent computational complexity, effectively results in a basic model predictive controller (MPC). Therefore it is essentially better to use a MPC to perform the final high fidelity simulation for trajectory validation, using the lower resolution APM for approximate evaluation during trajectory generation. The inverse simulation using the non-linear model was implemented as a constrained non-linear optimisation problem, that minimizes the error in position, roll, heading and flight path angle, subject to constraints to match the reference trajectory's wind frame angular rates and change in velocity (tangential acceleration). Additional constraints were placed to limit that rate of change in the control deflections.

The following sample shown in Figure 6.5 was computed using the 4th order Runge-

Kutta integration of the non-linear model at a time-step of 0.2s, with the inverse model accurately tracking the reference trajectory. The same simulation was then analysed using the Aircraft Performance Model (APM) to get approximate values for the aerodynamic angles and thrust for performance evaluation of the trajectory, with the results shown in Figure 6.6. As expected the APM is not as accurate as the full model due to the approximations, however it still provides sufficiently accurate results to serve as a reference during trajectory generation. The oscillation seen in the side-slip angle in Figure 6.5 is the result of the non-optimal initial values in this open-loop simulation.

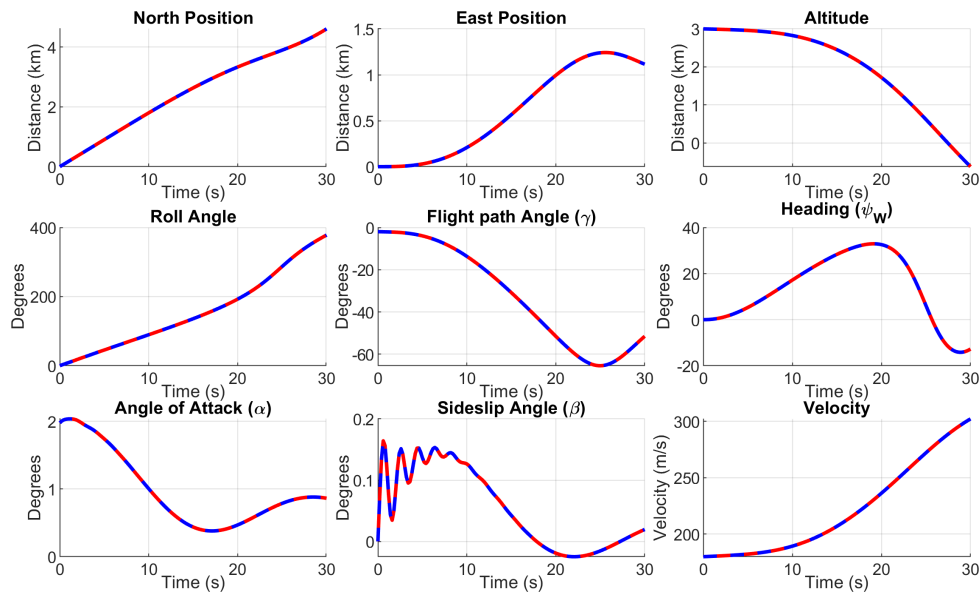


Figure 6.5: Inverse Simulation Verification

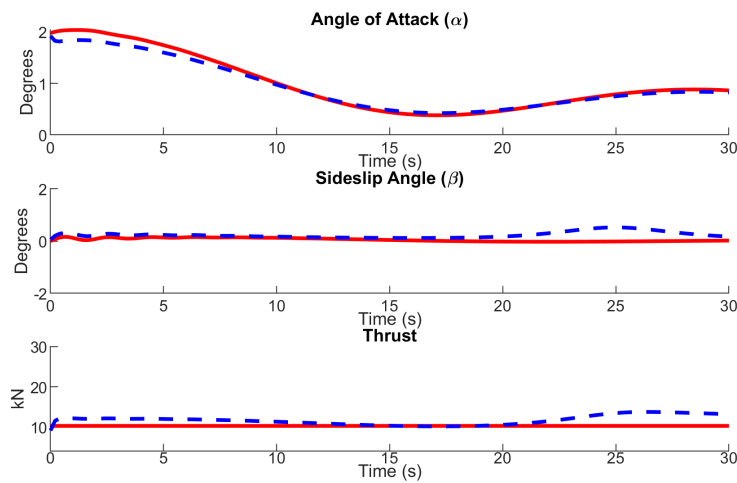


Figure 6.6: Aircraft Performance Model Approximation

6.6 Boundary Constraints

Although the performance model can be used for continuous evaluation of the trajectory, it does not require iterative computation up to the desired point, rather allowing for evaluation of the extrema in the trajectory. By computing the boundary limits for given segment, the segment can quickly be validated for feasibility without having to compute each point within the segment. The segmentation also means that conservative constraints don't limit the entire trajectory. If necessary, a given segment can be analysed in greater detail to exploit as much of the flight envelope as possible, thus focussing processing power only on the extrema of the trajectory.

Generating these outer boundary constraints, requires evaluating the worst case scenario for the given segment, i.e. taking the minimum equivalent airspeed (or dynamic pressure) and lowest value for the max thrust, and comparing these value to the maximum tangential and normal acceleration. The tangential acceleration determines the maximum angle of attack the given thrust limit can sustain, which in turn determines the maximum normal acceleration and thus turn radius. This can be applied during trajectory generation to confirm whether the next way-point is achievable for the given tangent $\hat{\mathbf{m}}_i$ in the current way-point $\mathbf{w}_i$. Computing the turning radius at the given altitude and velocity, as shown in Figure 6.7 at full military thrust, results in a turning radius of approximately $10.5km$. Calculating the cross between the tangent and the connecting vector $\mathbf{w}_{i+1} - \mathbf{w}_i$, gives the normal vector $\mathbf{N}_c$ that defines the turning plane. If the distance from the centers of the turning circles $\mathbf{P}_c$, both left hand right turns, is greater than the turning radius, then the way-point is reachable.

$$\mathbf{N}_c = \hat{\mathbf{m}}_i \times \frac{\mathbf{w}_{i+1} - \mathbf{w}_i}{\|\mathbf{w}_{i+1} - \mathbf{w}_i\|} \quad (6.22)$$

$$\mathbf{P}_c = \mathbf{w}_i \pm r(v, h) \hat{\mathbf{m}}_i \times \hat{\mathbf{N}}_c \quad (6.23)$$

$$\text{subject to } \|\mathbf{w}_{i+1} - \mathbf{P}_c\| \geq r(v, h) \quad (6.24)$$

This is an example, as shown in Figure 6.10, of a highly non-linear dynamically computed constraint used in a computationally efficient manner to ensure trajectory feasibility.

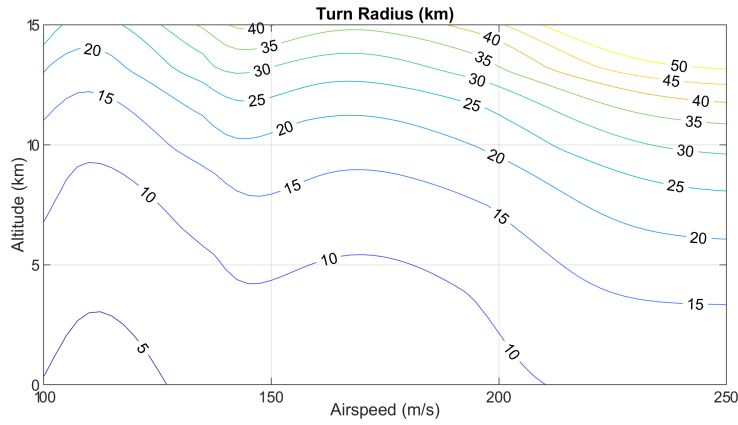


Figure 6.7: Minimum Turning Radius at max Military Thrust

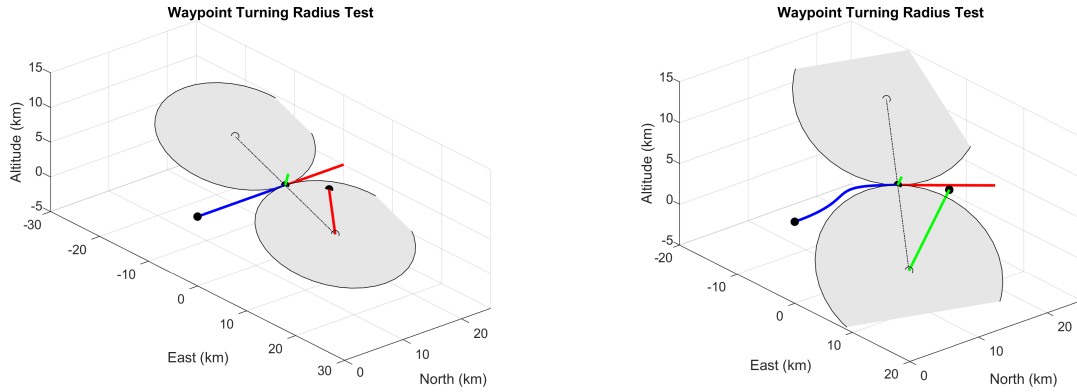


Figure 6.8: Way-point Turning Radius Test

6.7 Results

These boundary constraints can now be combined with the trajectory generation to quickly evaluate feasibility. The following example trajectories are presented of an F16 performing a constant bank turn with radius of 5km at an airspeed of 150m.s^{-1} at 1km and 6km altitudes respectively. These manoeuvres are constrained to not exceed the maximum military thrust at the given altitude.

The trajectory is first constructed using the input values in the table below to the PH curve algorithm.

This results in the quaternion polynomial control sets for the two segments. These

Table 6.4: Trajectory Inputs

i	P_N	P_E	Alt	ϕ	γ	ψ_W	d_{mag}	C_x	C_y	C_z	v	a
1	0	0	1000	77.4°	0°	0°	7854	0	12566	0	150	0
2	5000	5000	1000	77.4°	0°	90°	7854	-12566	0	0	150	0
3	0	10000	1000	77.4°	0°	180°	7854	0	-12566	0	150	0

segments have been optimised to minimise the bending energy of the curves.

$$\bar{\mathcal{A}}_1 = \begin{bmatrix} 69.1294 & 69.1294 & 60.0794 & 58.6582 & 48.8819 \\ 55.4536 & 55.4537 & 58.7736 & 47.0540 & 39.2116 \\ 0 & 11.0907 & 24.3448 & 31.3693 & 39.2116 \\ 0 & 13.8259 & 24.8857 & 39.1055 & 48.8819 \end{bmatrix}$$

$$\bar{\mathcal{A}}_2 = \begin{bmatrix} 48.8819 & 39.1065 & 24.8858 & 13.8259 & 0.0000 \\ 39.2116 & 31.3681 & 24.3448 & 11.0907 & 0.0000 \\ 39.2116 & 47.0527 & 58.7738 & 55.4516 & 55.4536 \\ 48.8819 & 58.6592 & 60.0794 & 69.1310 & 69.1294 \end{bmatrix}$$

Computing the quaternion polynomial product, yield the control sets for the spatial curves of each segment.

$$\bar{\mathbf{p}}_1 = \begin{bmatrix} 0 & 873 & 1745 & 2577 & 3330 & 3976 & 4480 & 4826 & 5000 & 5000 \\ 0 & 0 & 175 & 520 & 1024 & 1670 & 2423 & 3255 & 4127 & 5000 \\ 1000 & 1000 & 1000 & 986 & 971 & 971 & 986 & 1000 & 1000 & 1000 \end{bmatrix}$$

$$\bar{\mathbf{p}}_2 = \begin{bmatrix} 5000 & 5000 & 4825 & 4480 & 3976 & 3330 & 2577 & 1745 & 873 & 0 \\ 5000 & 5873 & 6745 & 7577 & 8330 & 8976 & 9480 & 9825 & 10000 & 10000 \\ 1000 & 1000 & 1000 & 986 & 971 & 971 & 986 & 1000 & 1000 & 1000 \end{bmatrix}$$

The result are shown in Figure 6.9, noting that the acceleration does not exceed the predicted performance limit. Applying the APM to approximate the aerodynamic angles, it is clearly seen that the aircraft does not exceed the thrust limit imposed.

The same manoeuvre is now performed at an altitude of $6km$, with the control sets and inputs listed below. This time the acceleration exceeds the predicted performance limit, and once the APM is applied, it is seen that the manoeuvre would require exceeding the thrust constraint placed on the aircraft.

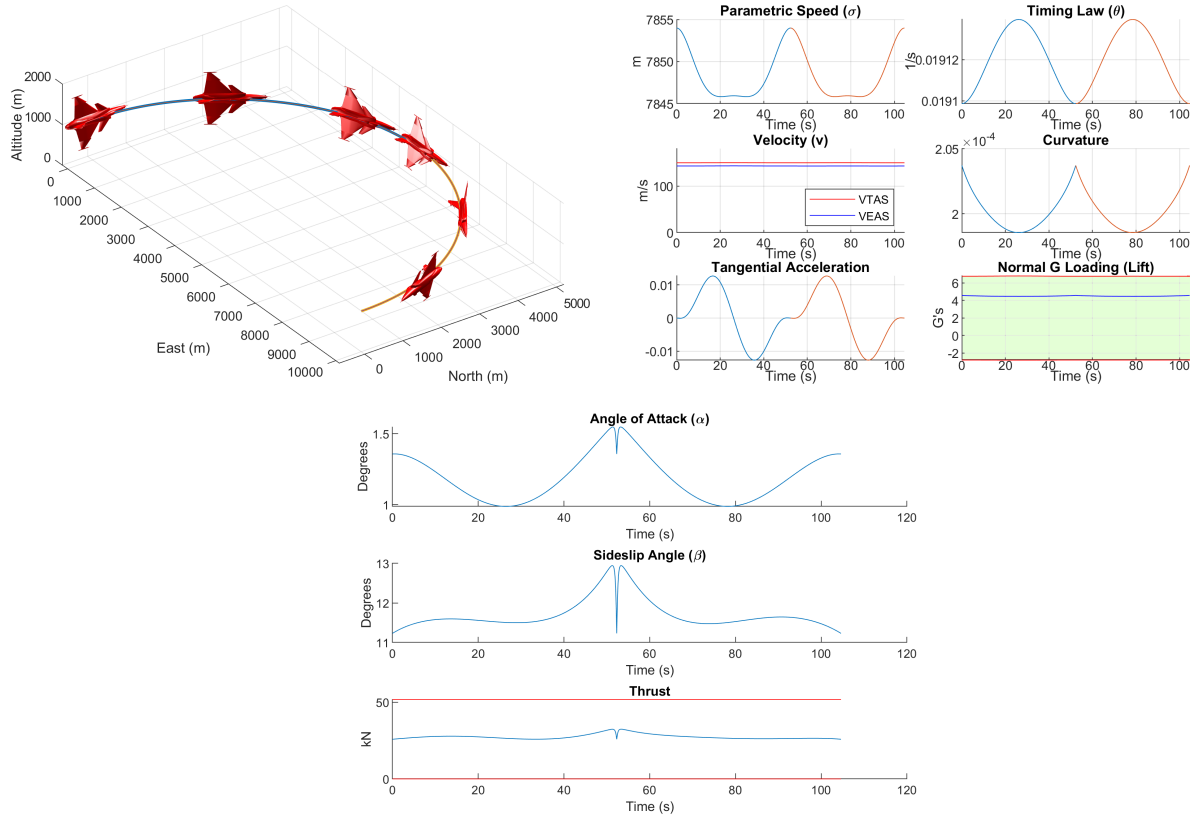


Figure 6.9: 4.5 G Turn 1km Altitude

Table 6.5: Trajectory Inputs

i	P_N	P_E	Alt	ϕ	γ	ψ_W	d_{mag}	C_x	C_y	C_z	v	a
1	0	0	6000	77.4°	0°	0°	7854	0	12566	0	150	0
2	5000	5000	6000	77.4°	0°	90°	7854	-12566	0	0	150	0
3	0	10000	6000	77.4°	0°	180°	7854	0	-12566	0	150	0

$$\bar{\mathcal{A}}_1 = \begin{bmatrix} 69.1294 & 69.1294 & 60.0794 & 58.6582 & 48.8819 \\ 55.4536 & 55.4537 & 58.7736 & 47.0540 & 39.2116 \\ 0 & 11.0907 & 24.3448 & 31.3693 & 39.2116 \\ 0 & 13.8259 & 24.8857 & 39.1055 & 48.8819 \end{bmatrix}$$

$$\bar{\mathcal{A}}_2 = \begin{bmatrix} 48.8819 & 39.1065 & 24.8858 & 13.8259 & 0.0000 \\ 39.2116 & 31.3681 & 24.3448 & 11.0907 & 0.0000 \\ 39.2116 & 47.0527 & 58.7738 & 55.4516 & 55.4536 \\ 48.8819 & 58.6592 & 60.0794 & 69.1310 & 69.1294 \end{bmatrix}$$

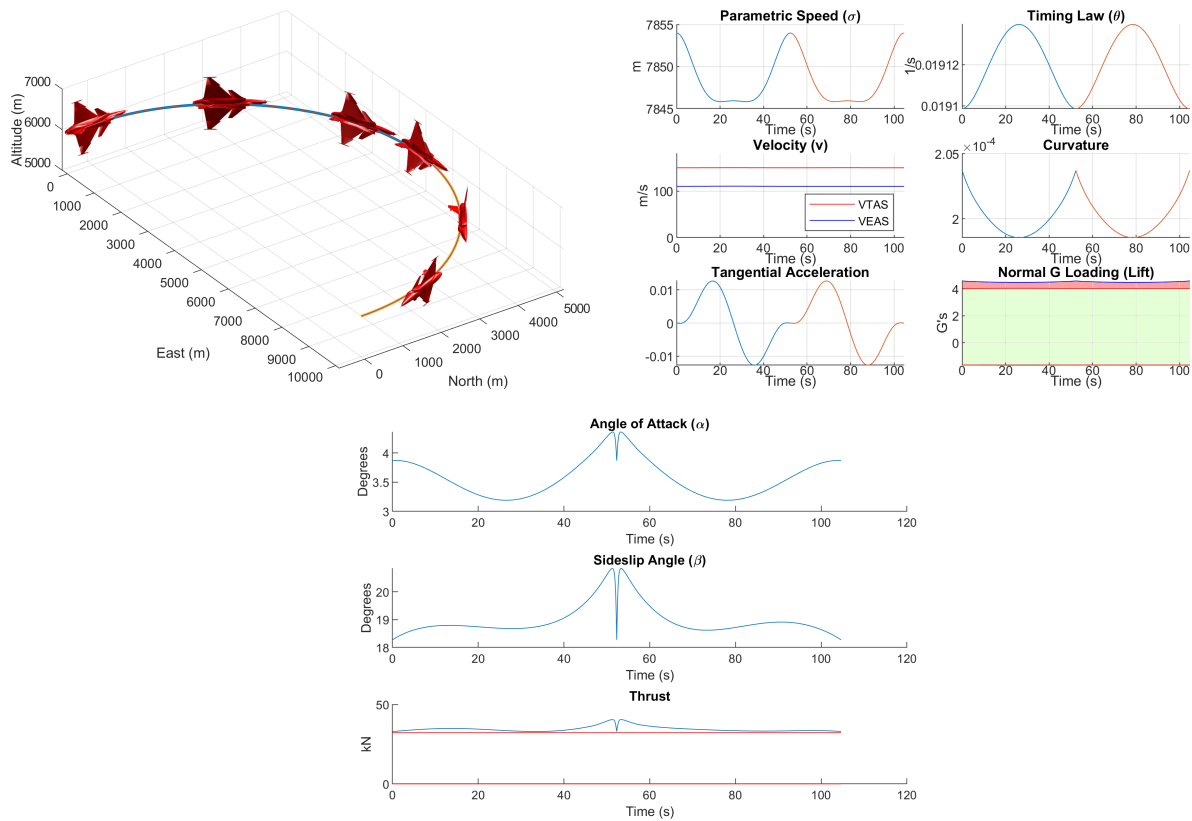


Figure 6.10: 4.5 G Turn 6km Altitude

6.8 Conclusion

In this chapter, the concept of using a lower fidelity Aircraft Performance Model (APM) to predict the performance limits of the aircraft at it's current state was introduced. It was shown that the boundary box placed on the expected acceleration limit, resulted in accurate prediction of a constraint violation. The result being that dynamic constraints can be applied to segments without the computational load of analysing each point along the trajectory. Although these constraints are still conservative, the fact that their range are limited to local segments, doesn't limit the entire trajectory to overly conservative constraints. Therefore it is possible to apply complex non-linear constraints without excessive computation load. The analytical nature of the Bézier PH curves also guarantees that the extrema can be identified and tested against these limits.

Chapter 7

Conclusion

“Sometimes it’s the journey that teaches you a lot about your destination.”

- Drake

7.1 Overview

A technique for the accurate and concise description of a 4D trajectory has been presented using Bézier splines. Hermite interpolation was used to ease the generation of the initial values due to the intuitive relation between the boundary continuity constraints and the curve control set. The quaternion Pythagorean Hodograph (PH) curve structure, frequently used for its efficient and exact method of computing the parametric speed and arc length, was exploited as a means of defining a continuous frame specification. The use of a timing law to separate the temporal properties of the trajectory from the geometry, allows for additional manipulation of the trajectory for temporal separation and dynamic optimisation. Having the analytical relationship between the timing law and the spatial curve means that there are no errors introduced through the use of numerical

approximations to calculate the various performance evaluation functions.

The main contribution of this thesis is extending the PH curve specification to include the desired orientation, which isn't present in other trajectory generation techniques. Using the quaternion PH curve naturally define the trajectory as the rotational development of the aerodynamic reference frame, which serves as a natural reference point to analyse the aerodynamic performance of the aircraft. It was shown that the inverse implementation can efficiently and accurately track the reference trajectory due to convenient availability of exact solutions to the tangential acceleration ($\dot{v}$) and aerodynamic frame angular rates (ω_w). Increasing the resolution of the inverse simulation, even the oscillatory behaviour due to cross-coupling is accurately modelled. The computation of the bending energy as the integral of the body angular rates, although still not a closed form solution, is computationally far more efficient than computing the curvature and torsion.

Inclusion of the rigid body (6 DoF) kinematics of in the trajectory specification, accelerates the trajectory evaluation by removing an additional step of having to model the rotation afterwards. This means that the entire spatial trajectory and orientational kinematics is concisely defined using the small set of quaternion values that constitutes the control polygon of the quaternion polynomial. The addition of the orientation in the trajectory description enables the algorithm to perform a more detailed analysis on the aircraft's dynamics, as shown with the use of an Aircraft Performance Model (APM).

The segmented nature of the trajectory, allows the algorithm to scale well to complex environments, as the trajectory can be segmented to reduce the scope both spatially and temporally. Bézier splines have also previously been proven to be well suited to distributed optimisation, due to their concise and accurate representation of the trajectory. The control polygon structure, specifically through the convex hull property and other efficient algorithm for determining extrema, are well suited to obstacle avoidance using established and efficient algorithms such as the Gilbert-Johnson-Keerthi (GJK) algorithm. This means that these splines co-exist well with other techniques useful in environmental mapping, including spline surfaces, point clouds and polygons. These algorithms are also

well suited to finding the extrema in the dynamic trajectory evaluation function due to their shared Bézier structure.

All these characteristics result in an approach that is suitable to various vehicle types, scales well to complex environments and dynamics, and can be implemented in a distributed system. The ability to distribute and segment the trajectory is critical to dynamic environments, and is well suited to modern computed architectures focussed on parallelism. The computational advantages of using splines, being that there exists simple linear relationships between the control sets of the input functions and the results, makes the analysis of the trajectory very efficient. This computational behaviour extends to addition and subtraction, scalar and polynomial multiplication, integration and differentiation, root finding, and computation of extrema.

7.2 Future work

The trajectory generation algorithm and various evaluation functions have been presented and a logical next step in the research would be the development of a suitable multi-objective optimisation algorithm. It is recommended that the optimisation algorithm should be constructed in such a way that the optimisation is done in increasing detail, rather than evaluating all the various performance factors from the start. The algorithm should first focus on the geometry of the trajectory, ensuring obstacle avoidance (geometric separation) and limiting the curvature and arclength. Subsequent optimisation of the velocity profile or timing law allows for improved coordination, general performance and temporal separation. Finally the optimisation of the orientation allows for maximising the performance of the aircraft in regards to energy cost and mission efficiency.

In order for the system to truly take advantage of the full flight envelope of the aircraft, a detailed performance model of the aircraft is needed. This performance model presented only serves as a proof of concept, lacking damping and rotational performance metrics. Although it was shown that the additional computational complexity would result in little additional insight into the trajectory performance. Even with such a basic model, various

performance metrics can be derived to help guide the optimisation and tactical evaluation through simple heuristics.

The goal of the performance mode, along with the iterative optimisation structure, is to first find the general location of the global optimal. Furthermore, this is well suited to tactical evaluation, as these low fidelity models can offer enough insight to select between various strategies without excessive computational cost.

The challenge with the use of Bézier splines regarding non-constant parametric speed, raises the question whether another splining technique should be considered. The quaternion form of the Pythagorean Hodograph (PH) curve has shown to be a suitable approach for the description of the trajectory, allowing for complete specification of the position and orientation using a single polynomial. Evaluating the use of other splines as the polynomial foundation for the quaternion PH curves, especially extending the work to geodetic coordinates systems.

The use of hermite interpolation would still be a requirement to ensure trajectory continuity between segments, but some new technique is needed that is able to interpolate points on the surface of the 4 dimensional 3-sphere (S^3), and that has higher degrees of freedom than the standard Spherical Linear Interpolation (SLERP) approach. Although rational Bézier splines are able to exactly describe a circle, the rational function structure of this approach doesn't provide the desired reduction in complexity, rather it increases the complexity of the algorithm. The purpose of reducing the parametric speed to a constant is to avoid the use of rational polynomials that can not be reduced further. It is however important that the locality of these splines are preserved, as this is key to the scalability of Bézier splines.

Bibliography

- [1] K. Israel and R. Nesbit, “Defense science board study on unmanned aerial vehicles and uninhabited combat aerial vehicles,” DEFENSE SCIENCE BOARD WASHINGTON DC, Tech. Rep., 2004. (Cited on page 1.)
- [2] “United States Air Force Unmanned Aircraft Systems Flight Plan 2009-2047,” Headquarters, United States Air Force, Tech. Rep., 18 May, 2009. (Cited on page 1.)
- [3] A. Sipe and J. Moore, “Air traffic functions in the NextGen and SESAR airspace,” in *Digital Avionics Systems Conference, 2009. DASC’09. IEEE/AIAA 28th.* IEEE, 2009, pp. 2–A. (Cited on page 1.)
- [4] FAA, “NextGen implementation plan,” *Washington, DC, Available: <http://www.faa.gov/nextgen/media/ng2011implementationplan.pdf> (accessed March 18, 2011),* 2010. (Cited on page 1.)
- [5] SESAR Joint Undertaking, “European ATM master plan,” 2012. (Cited on page 1.)
- [6] SESAR, “Master plan,” 2009. (Cited on page 1.)
- [7] G. Enea and M. Porretta, “A comparison of 4D-trajectory operations envisioned for Nextgen and SESAR, some preliminary findings,” in *28th Congress of the International Council of the Aeronautical Sciences*, 2012, pp. 23–28. (Cited on page 1.)
- [8] B. Korn and C. Edinger, “UAS in civil airspace: Demonstrating “sense and avoid” capabilities in flight trials,” in *Digital Avionics Systems Conference, 2008. DASC 2008. IEEE/AIAA 27th.* IEEE, 2008, pp. 4–D. (Cited on page 1.)

-
- [9] C. Goerzen, Z. Kong, and B. Mettler, “A survey of motion planning algorithms from the perspective of autonomous UAV guidance,” *Journal of Intelligent and Robotic Systems*, vol. 57, no. 1-4, p. 65, 2010. (Cited on pages 1 and 15.)
- [10] A. Gardi, R. Sabatini, and S. Ramasamy, “Multi-objective optimisation of aircraft flight trajectories in the atm and avionics context,” *Progress in Aerospace Sciences*, vol. 83, pp. 1–36, 2016. (Cited on page 1.)
- [11] R. Choe, “Distributed cooperative trajectory generation for multiple autonomous vehicles using pythagorean hodograph bézier curves,” Ph.D. dissertation, University of Illinois at Urbana-Champaign, 2017. (Cited on pages 3, 6, 16, 32, 42, 47, and 48.)
- [12] D. B. Quanbeck, “Methods for generating aircraft trajectories,” CENTER FOR NAVAL ANALYSES ALEXANDRIA VA OPERATIONS EVALUATION GROUP, Tech. Rep., 1982. (Cited on pages 7, 18, 36, and 98.)
- [13] B. Stevens and F. Lewis, *Aircraft Control and Simulation*. Wiley, 2003. [Online]. Available: <https://books.google.co.za/books?id=T0Ux6av4btIC> (Cited on pages 11, 93, 94, and 103.)
- [14] L. T. Nguyen, M. E. Ogburn, W. P. Gilbert, K. S. Kibler, P. W. Brown, and P. L. Deal, “Simulator study of stall/post-stall characteristics of a fighter airplane with relaxed longitudinal static stability.[f-16],” 1979. (Cited on pages 11 and 93.)
- [15] J. T. Betts, “Survey of numerical methods for trajectory optimization,” *Journal of guidance, control, and dynamics*, vol. 21, no. 2, pp. 193–207, 1998. (Cited on page 17.)
- [16] R. Kamyar and E. Taheri, “Aircraft optimal terrain/threat-based trajectory planning and control,” *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 2, pp. 466–483, 2014. (Cited on pages 17, 26, and 30.)
- [17] N. Bedrossian, S. Bhatt, M. Lammers, L. Nguyen, and Y. Zhang, “First ever flight demonstration of zero propellant maneuver (tm) attitude control concept,” in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2007, p. 6734. (Cited on page 18.)
-

-
- [18] I. M. Ross and M. Karpenko, “A review of pseudospectral optimal control: From theory to flight,” *Annual Reviews in Control*, vol. 36, no. 2, pp. 182–197, 2012. (Cited on page 18.)
- [19] W. Blajer, J. Graffstein, and M. Krawczyk, “Prediction of the dynamic characteristics and control of aircraft in prescribed trajectory flight,” *Journal of Theoretical and Applied Mechanics*, vol. 39, pp. 79–103, 2001. (Cited on page 18.)
- [20] D. Mellinger, N. Michael, and V. Kumar, “Trajectory generation and control for precise aggressive maneuvers with quadrotors,” *The International Journal of Robotics Research*, vol. 31, no. 5, pp. 664–674, 2012. (Cited on page 18.)
- [21] E. Frazzoli, M. A. Dahleh, and E. Feron, “Maneuver-based motion planning for nonlinear systems with symmetries,” *IEEE transactions on robotics*, vol. 21, no. 6, pp. 1077–1091, 2005. (Cited on page 18.)
- [22] C. Dever, B. Mettler, E. Feron, J. Popovic, and M. McConley, “Nonlinear trajectory generation for autonomous vehicles via parametrized maneuver classes,” *Journal of guidance, control, and dynamics*, vol. 29, no. 2, pp. 289–302, 2006. (Cited on page 19.)
- [23] N. K. Ure and G. Inalhan, “Autonomous control of unmanned combat air vehicles: Design of a multimodal control and flight planning framework for agile maneuvering,” *IEEE Control Systems*, vol. 32, no. 5, pp. 74–95, 2012. (Cited on page 19.)
- [24] E. Scholte and M. E. Campbell, “Robust nonlinear model predictive control with partial state information,” *IEEE Transactions on Control Systems Technology*, vol. 16, no. 4, pp. 636–651, 2008. (Cited on page 20.)
- [25] J. Ousingsawat and M. E. Campbell, “Optimal cooperative reconnaissance using multiple vehicles,” *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 1, pp. 122–132, 2007. (Cited on page 20.)
- [26] L. Sun, J. D. Hedengren, and R. W. Beard, “Optimal trajectory generation using model predictive control for aerially towed cable systems,” *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 2, pp. 525–539, 2014. (Cited on page 20.)
-

-
- [27] F. Borrelli, D. Subramanian, A. U. Raghunathan, and L. T. Biegler, “A comparison between milp and nlp techniques for centralized trajectory planning of multiple unmanned air vehicles,” in *American Control Conference*, 2006. (Cited on page 20.)
- [28] T. Schouwenaars, J. How, and E. Feron, “Decentralized cooperative trajectory planning of multiple aircraft with hard safety guarantees,” in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2004, p. 5141. (Cited on page 20.)
- [29] Y. Kuwata and J. P. How, “Cooperative distributed robust trajectory optimization using receding horizon milp,” *IEEE Transactions on Control Systems Technology*, vol. 19, no. 2, pp. 423–431, 2011. (Cited on page 20.)
- [30] T. D. DeRose and B. A. Barsky, “Geometric continuity, shape parameters, and geometric constructions for Catmull-Rom splines,” *ACM Transactions on Graphics*, vol. 7, pp. 1–41, 1988. (Cited on page 21.)
- [31] A. J. Häusler, R. Ghabcheloo, A. M. Pascoal, A. P. Aguiar, I. I. Kaminer, and V. N. Dobrokhodov, “Temporally and spatially deconflicted path planning for multiple autonomous marine vehicles,” *IFAC Proceedings Volumes*, vol. 42, no. 18, pp. 376–381, 2009. (Cited on page 22.)
- [32] K. Murthy and S. Rock, “Spline-based trajectory planning techniques for benthic auv operations,” in *Autonomous Underwater Vehicles (AUV), 2010 IEEE/OES*. IEEE, 2010, pp. 1–9. (Cited on page 23.)
- [33] K. B. Judd and T. W. McLain, “Spline based path planning for unmanned air vehicles,” in *AIAA Guidance, Navigation, and Control Conference and Exhibit*, vol. 9. Montreal, Canada, 2001. (Cited on page 23.)
- [34] I. K. Nikolos, K. P. Valavanis, N. C. Tsourveloudis, and A. N. Kostaras, “Evolutionary algorithm based offline/online path planner for uav navigation,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 33, no. 6, pp. 898–912, 2003. (Cited on page 23.)
- [35] K. E. May, D. Van Khanh, T. C. Seng, Y. S. Ping, and H. J. Sien, “Contour based path planning for unmanned aerial vehicles (uavs) over hostile terrain,” in *Soft*
-

-
- Computing and Pattern Recognition, 2009. SOCPAR'09. International Conference of.* IEEE, 2009, pp. 732–735. (Cited on page 23.)
- [36] L. Swartzentruber, J. L. Foo, and E. Winer, “Three-dimensional multi-objective uav path planner using terrain information,” in *50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference 17th AIAA/ASME/AHS Adaptive Structures Conference 11th AIAA No*, 2009, p. 2222. (Cited on page 24.)
- [37] A. Babaei and A. Karimi, “Optimal trajectory-planning of uavs via b-splines and disjunctive programming,” *arXiv preprint arXiv:1807.02931*, 2018. (Cited on page 24.)
- [38] M. B. Milam, K. Mushambi, and R. M. Murray, “A new computational approach to real-time trajectory generation for constrained mechanical systems,” in *Decision and Control, 2000. Proceedings of the 39th IEEE Conference on*, vol. 1. IEEE, 2000, pp. 845–851. (Cited on page 24.)
- [39] N. Akhtar, J. F. Whidborne, and A. K. Cooke, “Real-time trajectory generation technique for dynamic soaring uavs,” in *Proceedings of the UKACC International Conference on Control*, 2008. (Cited on page 24.)
- [40] D. Mellinger and V. Kumar, “Minimum snap trajectory generation and control for quadrotors,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2520–2525. (Cited on page 24.)
- [41] O. A. Yakimenko, “Direct method for rapid prototyping of near-optimal aircraft trajectories,” *Journal of Guidance, Control, and Dynamics*, vol. 23, no. 5, pp. 865–875, 2000. (Cited on page 25.)
- [42] F. Augugliaro, A. P. Schoellig, and R. D’Andrea, “Generation of collision-free trajectories for a quadrocopter fleet: A sequential convex programming approach,” in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 1917–1922. (Cited on page 25.)
- [43] Y. Chen, M. Cutler, and J. P. How, “Decoupled multiagent path planning via incremental sequential convex programming,” in *Robotics and Automation (ICRA)*,
-

-
- 2015 IEEE International Conference on.* IEEE, 2015, pp. 5954–5961. (Cited on page 25.)
- [44] S. M. LaValle, “Rapidly-exploring random trees: A new tool for path planning,” 1998. (Cited on page 26.)
- [45] E. Frazzoli, M. A. Dahleh, and E. Feron, “Real-time motion planning for agile autonomous vehicles,” *Journal of guidance, control, and dynamics*, vol. 25, no. 1, pp. 116–129, 2002. (Cited on page 26.)
- [46] W. Zu, G. Fan, Y. Gao, Y. Ma, H. Zhang, and H. Zeng, “Multi-uavs cooperative path planning method based on improved rrt algorithm,” in *2018 IEEE International Conference on Mechatronics and Automation (ICMA)*. IEEE, 2018, pp. 1563–1567. (Cited on page 26.)
- [47] K. Yang and S. Sukkarieh, “3D smooth path planning for a UAV in cluttered natural environments,” in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on.* IEEE, 2008, pp. 794–800. (Cited on pages 26, 32, and 34.)
- [48] D. Lee and D. H. Shim, “Path planner based on bidirectional spline-RRT* for fixed-wing uavs,” in *Unmanned Aircraft Systems (ICUAS), 2016 International Conference on.* IEEE, 2016, pp. 77–86. (Cited on pages 27, 32, and 34.)
- [49] V. R. Desaraju and J. P. How, “Decentralized path planning for multi-agent teams with complex constraints,” *Autonomous Robots*, vol. 32, no. 4, pp. 385–403, 2012. (Cited on page 27.)
- [50] W. Yao, N. Qi, and Y. Liu, “Online trajectory generation with rendezvous for UAVs using multistage path prediction,” *Journal of Aerospace Engineering*, vol. 30, no. 3, 2016. (Cited on page 27.)
- [51] C. Ha, Z. Zuo, F. B. Choi, and D. Lee, “Passivity-based adaptive backstepping control of quadrotor-type uavs,” *Robotics and Autonomous Systems*, vol. 62, no. 9, pp. 1305–1315, 2014. (Cited on page 30.)
-

-
- [52] R. E. Barnhill and R. F. Riesenfeld, *Computer aided geometric design*. Academic Press, 2014. (Cited on page 32.)
- [53] V. Dobrokhodov, “Cooperative path planning of unmanned aerial vehicles,” *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 5, pp. 1601–1602, 2011. (Cited on page 32.)
- [54] M. Lizarraga and G. Elkaim, “Spatially deconflicted path generation for multiple uavs in a bounded airspace,” in *Position, Location and Navigation Symposium, 2008 IEEE/ION*. IEEE, 2008, pp. 1213–1218. (Cited on pages 32 and 33.)
- [55] V. Cichella, R. Choe, B. S. Mehdi, E. Xargay, N. Hovakimyan, A. C. Trujillo, and I. Kaminer, “Trajectory generation and collision avoidance for safe operation of cooperating uavs,” in *AIAA Guidance, Navigation, and Control Conference*, 2014, p. 0972. (Cited on page 32.)
- [56] A. A. Neto, D. G. Macharet, and M. F. Campos, “Feasible path planning for fixed-wing UAVs using seventh order Bézier curves,” *Journal of the Brazilian Computer Society*, vol. 19, no. 2, pp. 193–203, 2013. (Cited on pages 32 and 47.)
- [57] O. K. Sahingoz, “Generation of bezier curve-based flyable trajectories for multi-UAV systems with parallel genetic algorithm,” *Journal of Intelligent & Robotic Systems*, vol. 74, no. 1-2, p. 499, 2014. (Cited on pages 32 and 34.)
- [58] K. Jolly, R. S. Kumar, and R. Vijayakumar, “A Bézier curve based path planning in a multi-agent robot soccer system without violating the acceleration limits,” *Robotics and Autonomous Systems*, vol. 57, no. 1, pp. 23–33, 2009. (Cited on page 32.)
- [59] S. B. Mehdi, R. Choe, and N. Hovakimyan, “Avoiding multiple collisions through trajectory replanning using piecewise bézier curves,” in *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*. IEEE, 2015, pp. 2755–2760. (Cited on pages 32 and 34.)
- [60] C. Sprunk, “Planning motion trajectories for mobile robots using splines,” *University of Freiburg*, 2008. (Cited on page 32.)
-

-
- [61] B. Lau, C. Sprunk, and W. Burgard, “Kinodynamic motion planning for mobile robots using splines,” in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*. IEEE, 2009, pp. 2427–2433. (Cited on page 32.)
- [62] M. Walter and A. Fournier, “Approximate arc length parameterization,” in *Proceedings of the 9th Brazilian symposium on computer graphics and image processing*. Citeseer, 1996, pp. 143–150. (Cited on page 41.)
- [63] J. Gravesen, “Adaptive subdivision and the length and energy of bézier curves,” *Computational Geometry*, vol. 8, no. 1, pp. 13–31, 1997. (Cited on page 41.)
- [64] H. Wang, J. Kearney, and K. Atkinson, “Arc-length parameterized spline curves for real-time simulation,” in *5th international conference on Curves and Surfaces*, 2002. (Cited on page 41.)
- [65] R. T. Farouki and T. Sakkalis, “Pythagorean hodographs,” *IBM Journal of Research and Development*, vol. 34, no. 5, pp. 736–752, Sept 1990. (Cited on page 41.)
- [66] M. Shanmugavel, A. Tsourdos, R. Zbikowski, B. A. White, C. Rabbath, and N. Lechevin, “A solution to simultaneous arrival of multiple UAVs using Pythagorean hodograph curves,” in *American Control Conference, 2006*. IEEE, 2006, pp. 6–pp. (Cited on pages 42 and 46.)
- [67] S. Subchan, B. White, A. Tsourdos, M. Shanmugavel, and R. Zbikowski, “Pythagorean hodograph (PH) path planning for tracking airborne contaminant using sensor swarm,” in *Instrumentation and Measurement Technology Conference Proceedings, 2008. IMTC 2008. IEEE*. IEEE, 2008, pp. 501–506. (Cited on page 42.)
- [68] M. Shah and N. Aouf, “3d cooperative pythagorean hodograph path planning and obstacle avoidance for multiple uavs,” in *Cybernetic Intelligent Systems (CIS), 2010 IEEE 9th International Conference on*. IEEE, 2010, pp. 1–6. (Cited on pages 42 and 47.)
-

-
- [69] R. Choe, J. Puig-Navarro, V. Cichella, E. Xargay, and N. Hovakimyan, “Trajectory generation using spatial Pythagorean hodograph Bézier curves,” in *AIAA Guidance, Navigation and Control Conference, Kissimmee*, 2015. (Cited on page 42.)
- [70] H. Bruyninckx and D. Reynaerts, “Path planning for mobile and hyper-redundant robots using Pythagorean hodograph curves,” in *Advanced Robotics, 1997. ICAR’97. Proceedings., 8th International Conference on.* IEEE, 1997, pp. 595–600. (Cited on page 45.)
- [71] A. A. Neto and M. F. Campos, “A path planning algorithm for UAVs with limited climb angle,” in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on.* IEEE, 2009, pp. 3894–3899. (Cited on page 47.)
- [72] R. L. Bishop, “There is more than one way to frame a curve,” *The American Mathematical Monthly*, vol. 82, no. 3, pp. 246–251, 1975. [Online]. Available: <http://www.jstor.org/stable/2319846> (Cited on page 51.)
- [73] A. J. Hanson and H. Ma, “Parallel transport approach to curve framing,” *Indiana University, Techreports-TR425*, vol. 11, pp. 3–7, 1995. (Cited on page 52.)
- [74] D. Carroll, E. Kose, and I. Sterling, “Improving Frenet’s frame using Bishop’s frame,” *Journal of Mathematics Research*, vol. 5, no. 4, p. 97, 2013. [Online]. Available: <http://www.ccsenet.org/journal/index.php/jmr/article/view/30630> (Cited on page 55.)
- [75] H. I. Choi and C. Y. Han, “Euler–Rodrigues frames on spatial Pythagorean-hodograph curves,” *Computer Aided Geometric Design*, vol. 19, no. 8, pp. 603 – 620, 2002. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0167839602001656> (Cited on page 56.)
- [76] R. Farouki, C. Giannelli, C. Manni, and A. Sestini, “Design of rational rotation–minimizing rigid body motions by Hermite interpolation,” *Mathematics of Computation*, vol. 81, no. 278, pp. 879–903, 2012. (Cited on page 56.)
- [77] R. Komanduri, K. Subramanian, and B. Von Turkovich, “High speed machining,” *New Orleans*, p. 1984, 1984. (Cited on page 57.)

-
- [78] Z. Šír and B. Jüttler, “ C^2 Hermite interpolation by Pythagorean hodograph space curves,” *Mathematics of Computation*, vol. 76, no. 259, pp. 1373–1391, 2007. (Cited on page 71.)
- [79] R. S. Russell, “Non-linear f-16 simulation using simulink and matlab,” *University of Minnesota, Tech. paper*, 2003. (Cited on page 93.)

Appendix A

Bézier Splines

A.1 Introduction

This appendix is included for the convenience of the reader as a concise introduction to Bézier splines as introduced in chapter 3. The theoretical foundation of Bézier splines are presented alongside various useful algorithms that allow for efficient computation of polynomial arithmetic using these splines.

A Bézier spline is a polynomial defined in Bézier form, which allows for the manipulation of the polynomial using intuitive geometric properties. Larger curves can be constructed by joining a series of Bézier curves, called segments, end to end, subject to continuity constraints at these knots. Bézier splines of degree n are constructed using the $n + 1$ Bernstein basis polynomials, as shown in (A.1),

$$\beta_{v,n}(\zeta) = \binom{n}{v} \zeta^v (1 - \zeta)^{n-v} \quad v \in [0..n] \quad \zeta \in [0, 1] \quad (\text{A.1})$$

where $\binom{n}{v} = \frac{n!}{v!(n-v)!}$ is a binomial coefficient and ζ is the parametric progression of the spline. The spline (A.2) is a linear combination of Bernstein basis polynomials in (A.1),

$$p(\zeta) = \sum_{v=0}^n \bar{p}_v \beta_{v,n}(\zeta) \quad (\text{A.2})$$

where $\bar{p}_v$ are called the Bernstein or Bézier coefficients. This set of coefficients $\{\bar{p}_0, \bar{p}_1, \dots, \bar{p}_n\}$ that define the spline segment, is called the control polygon.

Considering the case of the cubic Bézier, the equation would be

$$p(\zeta) = (1 - \zeta)^3 \bar{p}_0 + 3(1 - \zeta)^2 \zeta \bar{p}_1 + 3(1 - \zeta) \zeta^2 \bar{p}_2 + \zeta^3 \bar{p}_3 \quad \zeta \in [0, 1] \quad (\text{A.3})$$

The spline is defined over the interval of $\zeta \in [0, 1]$ and is hence referred to as a uniform spline. The spline can also be defined over an arbitrary interval, i.e. $\zeta \in [\zeta_0, \zeta_1]$, shown in (A.4).

$$\begin{aligned} p_{[\zeta_0, \zeta_1]}(\zeta) &= \frac{\sum_{v=0}^n \binom{n}{v} (\zeta - \zeta_0)^v (\zeta_1 - \zeta)^{n-v} \bar{p}_v}{(\zeta_1 - \zeta_0)^n} \\ &= \sum_{v=0}^n \binom{n}{v} \left(\frac{\zeta - \zeta_0}{\zeta_1 - \zeta_0} \right)^v \left(\frac{\zeta_1 - \zeta}{\zeta_1 - \zeta_0} \right)^{n-v} \bar{p}_v \end{aligned} \quad (\text{A.4})$$

In this form the spline is called non-uniform as each segment has a unique interval.

A.1.1 Explicit Bézier Curves

An explicit Bézier curve, is a curve with control points defined over regular intervals, i.e. $x \equiv \zeta$ and simply $y = p(x)$. It is simply a polynomial function in Bernstein polynomial form. The control points are defined as $\bar{\mathbf{p}}_i = (\frac{\zeta_0(n-i) + \zeta_1 i}{n}, y_i)$ with $i = [0..n]$, as shown in figure A.1.

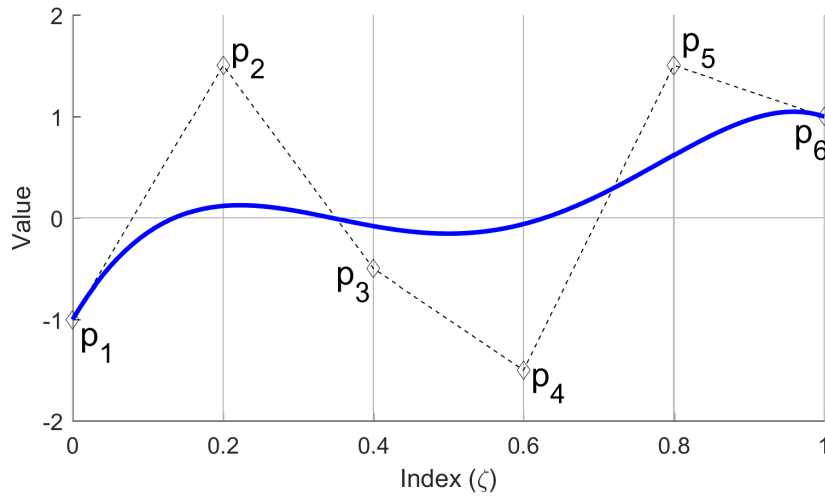


Figure A.1: Explicit Bézier curve

A.1.2 Higher Dimensional Curves

Higher dimensional curves are achieved by defining individual splines for each dimension. The spline index ζ no longer represents an axis but rather only serves to define the progression of the curve. Consider for example a 2D spline curve,

$$\mathbf{p}(\zeta) = \sum_{v=0}^n \bar{\mathbf{p}}_v \beta_{v,n}(\zeta) \quad (\text{A.5})$$

with $\bar{\mathbf{p}}_i$ defined as a sequence of coordinate pairs $[\bar{p}_{x,i} \ \bar{p}_{y,i}]$. Thus this spline can be decomposed into the two explicit splines that define each axis.

$$p_x(\zeta) = \sum_{v=0}^n \bar{p}_{x,v} \beta_{v,n}(\zeta) \quad p_y(\zeta) = \sum_{v=0}^n \bar{p}_{y,v} \beta_{v,n}(\zeta)$$

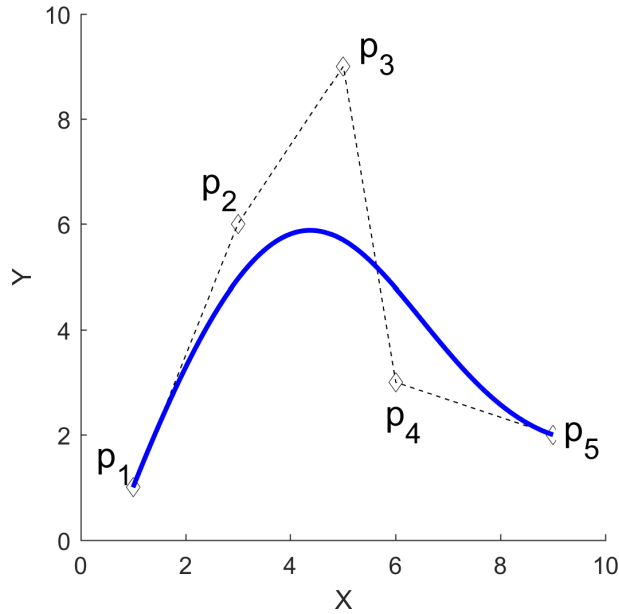


Figure A.2: 2D Bézier curve

A.1.3 Hermite Interpolation

Hermite splines, although being of a different form, can easily be related to a Bézier spline of the same order. This relationship is useful in relating the geometric properties of the

control polygon to the behaviour of the spline polynomial. Consider the cubic Hermite spline in (A.6), with starting point w_0 , end point w_1 , and tangents in these points m_0 and m_1 .

$$p(\zeta) = (2\zeta^3 - 3\zeta^2 + 1)\mathbf{w}_0 + (\zeta^3 - 2\zeta^2 + \zeta)\mathbf{m}_0 + (-2\zeta^3 + 3\zeta^2)\mathbf{w}_1 + (\zeta^3 - \zeta^2)\mathbf{m}_1 \quad (\text{A.6})$$

Rearranging this polynomial into Bézier form, demonstrates the relationship between the control polygon and the spline polynomial as shown in (A.7).

$$\begin{array}{ccc}
\text{Hermite} & & \text{Bézier} \\
\text{-----} & & \text{-----} \\
\mathbf{w}_0 = \bar{\mathbf{p}}_0 & & \\
\mathbf{m}_0 = 3(\bar{\mathbf{p}}_1 - \bar{\mathbf{p}}_0) & & (\text{A.7}) \\
\mathbf{m}_1 = 3(\bar{\mathbf{p}}_3 - \bar{\mathbf{p}}_2) & & \\
\mathbf{w}_1 = \bar{\mathbf{p}}_3 & &
\end{array}$$

This relationship remains for higher degree polynomials, and shows the intuitive correlation between the control polygon and the derivatives of the spline polynomial at the ends. Exploiting this to join segments to as to ensure parametric continuity, i.e. continuity of the derivatives, is called Hermite interpolation.

A.2 Bézier Spline Computational Properties

A.2.1 Convex Hull property

The convex hull property of Bézier curves is a very important one, as it allows for quick analysis of the bounds of the curve. The convex hull property states that the curve is always contained within the area bounded by the control points of the curve, as shown in figure A.3. This is particularly useful for obstacle avoidance as it allows us to quickly determine if there is a possibility of collision.

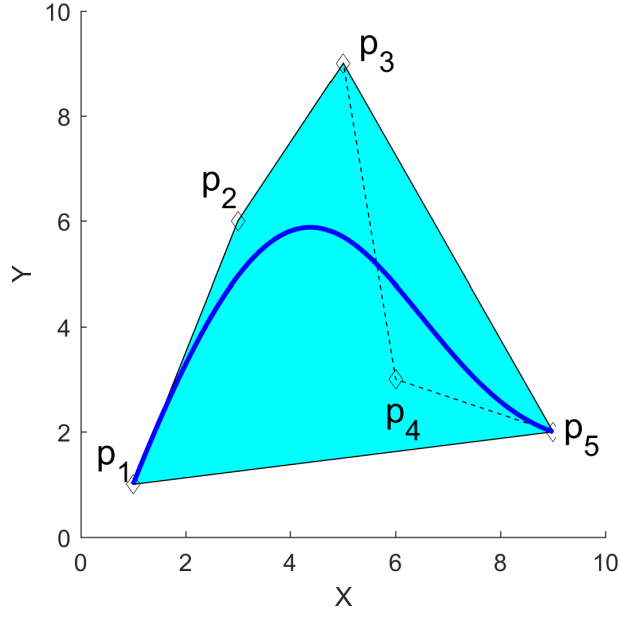


Figure A.3: Convex hull of Bézier curve

A.2.2 Derivatives (Hodograph)

The derivative of a Bézier curve can be determined directly from the control points and defined as a Bézier curve of one order lower. Consider a Bézier curve $p(\zeta)$ of degree n over the interval $\zeta \in [\zeta_0, \zeta_1]$, the derivative $h(\zeta)$ would be a Bézier curve of degree $n - 1$ with control points as shown in (A.8).

$$\bar{h}_i = \frac{n}{\zeta_1 - \zeta_0} (\bar{p}_{i+1} - \bar{p}_i) \quad i = [0..n - 1] \quad (\text{A.8})$$

A.2.3 Integrating Bernstein polynomials

Similar to the hodograph, the integral of a Bézier curve can be derived from the control points and described as a Bézier curve of one order higher. Consider a Bézier curve $p(\zeta)$ of order n over the interval $\zeta \in [\zeta_0, \zeta_1]$, then its integral would be a Bézier curve $q(\zeta)$ of order $n + 1$ with control points defined in (A.9). The initial value of the integral is defined as

$\bar{q}_0 = c$, which is the integration constant.

$$\bar{q}_i = \frac{1}{n+1} \sum_{j=0}^{i-1} \bar{p}_j \quad i = [1, n+1] \quad (\text{A.9})$$

A.2.4 Degree elevation

Degree elevation is the process of increasing the order of a Bézier curve control set, without affecting the curve itself. Consider a curve $p(\zeta)$ of order n , the elevated curve $q(\zeta)$ of order $n+1$ is computed as follows:

$$\begin{aligned} \bar{q}_0 &= \bar{p}_0 \\ \bar{q}_i &= \alpha_i \bar{p}_{i-1} + (1 - \alpha_i) \bar{p}_i \quad \alpha = \frac{i}{n+1} \quad \text{for } i \in [1..n+1] \end{aligned} \quad (\text{A.10})$$

This process can be repeated until the desired order is reached.

A.2.5 Subdivision

Subdivision of a curve $p(\zeta)$ of order n at index $\zeta = \alpha$ can easily and efficiently be done using the *de Casteljau* algorithm. This algorithm simply derives each set of coordinates by linearly dividing the distance between the previous set of coordinates at the desired index. The final set of values yields the control polygons of two new bezier curves $p^a(\zeta)$ and $p^b(\zeta)$ of order n , so that

$$\begin{aligned} p^a(\zeta) &= p(\alpha\zeta) \\ p^b(\zeta) &= p(\alpha + (1 - \alpha)\zeta). \end{aligned}$$

The algorithm uses the original control polygon of the original curve as the initial values, i.e. $\bar{a}_{i,0} = \bar{p}_i$ for $i = [0 \dots n]$. The *de Casteljau* algorithm is then computed using

$$\bar{a}_{i,j} = \alpha \bar{a}_{i+1,j-1} + (1 - \alpha) \bar{a}_{i,j-1} \quad \text{for } i = [0 \dots n], j = [0 \dots n-i] \quad (\text{A.11})$$

and the control polygons of the new curves are found using the values

$$\bar{p}_i^a = \bar{a}_{0,i} \quad \text{for } i = [0 \dots n] \quad (\text{A.12})$$

$$\bar{p}_i^b = \bar{a}_{i,n-i} \quad \text{for } i = [0 \dots n]. \quad (\text{A.13})$$

A.2.6 Addition

Addition of two Bézier splines is easily achieved by simply adding the control sets of the two splines together. It should be noted that the order of the two splines must be the same, thus degree elevation can be used to increase the order of the lower degree spline.

$$f(\zeta) + g(\zeta) = \sum_{v=0}^n (\bar{f}_v + \bar{g}_v) b_{v,n}(\zeta) \quad (\text{A.14})$$

A.2.7 Multiplication

The product of two Bernstein polynomials, $f(\zeta)$ of order m and $g(\zeta)$ of order n , yields a Bernstein polynomial of order $m+n$. The coefficients of the product curve are computed as follows:

$$\bar{p}_i = \sum_{j=\max(0,i-n)}^{\min(m,i)} \frac{\binom{m}{j} \binom{n}{i-j}}{\binom{m+n}{i}} \bar{f}_j \bar{g}_{i-j} \quad (\text{A.15})$$

$$p(\zeta) = f(\zeta)g(\zeta) = \sum_{i=0}^{m+n} \left(\sum_{j=\max(0,i-n)}^{\min(m,i)} \frac{\binom{m}{j} \binom{n}{i-j}}{\binom{m+n}{i}} \bar{f}_j \bar{g}_{i-j} \right) \beta_{i,m+n}(\zeta) \quad (\text{A.16})$$

A.2.8 Composition

The composition of two Bernstein polynomials, $r(\zeta)$ of order n and $s(\zeta)$ of order m , yields a Bernstein polynomial $c(\zeta) = (r \circ s)(\zeta)$ of order mn . The compound curve is then found using the initial values $\bar{a}_{i,0}^0 = \bar{r}_i$ for $i = [0 \dots n]$, and the subsequent expansion

$$\bar{a}_{i,j}^k = \frac{1}{\binom{km}{j}} \sum_{l=\max(0,j-m)}^{\min(j,km-m)} \binom{km-m}{l} \binom{m}{j-l} [(1 - \bar{s}_{j-l}) \bar{a}_{i,l}^{k-1} + \bar{s}_{j-l} \bar{a}_{i+1,l}^{k-1}] \quad (\text{A.17})$$

where $k = [0 \dots n]$, $i = [0 \dots n - k]$ and $j = [0 \dots km]$. The control polygon for the resulting compound curve is then given by

$$\bar{c}_j = \bar{a}_{0,j}^n \quad \text{for } j = [0 \dots mn]. \quad (\text{A.18})$$

A.2.9 Limitations

Bernstein polynomials however do not have similar reductions for rational polynomials, i.e. polynomial division, and by extension inverse function. Computation of the square root of a Bernstein polynomial suffers the same issue, and thus magnitude and absolute value functions cannot be performed directly on the control set of the spline. Therefore it is necessary to compensate for this limitation when using Bernstein polynomials.

Consider for example the distance between two spatial curves, $d(\zeta) = \|\mathbf{f}(\zeta) - \mathbf{g}(\zeta)\|$. The euclidean distance requires the computation of a square root of the sum of the squares of the individual splines of each dimension.

$$d(\zeta) = \sqrt{q_x^2(\zeta) + q_y^2(\zeta)}$$

This limitation can be avoided by rather evaluating the square of the distance ($d^2(\zeta)$), as this yields a Bézier spline that is easily calculated through subtraction ($\bar{q}_x = \bar{f}_x - \bar{g}_x$), multiplication (q_x^2) and then addition ($q_x^2 + q_y^2$).