Predictive Path Following Control for Fixed Wing UAVs Using the qLMPC Framework in Presence of Wind Disturbances

Ahmed Samir*, Horacio Martínez Calderón [†] and Herbert Werner [‡]
Hamburg University of Technology, Institute of Control Systems, Hamburg, Germany

Benjamin Herrmann[§], and Frank Thielecke [¶]
Hamburg University of Technology, Institute of Aircraft Systems Engineering, Hamburg, Germany

This paper presents a cascaded approach for a predictive path-following scheme using quasi-Linear-Parameter-Varying (qLPV) Model-based Predictive Control (qLMPC) applied to a 25 kg unmanned aerobatic aircraft. The qLPV representation provides a semi-linear model of the nonlinear system, while fast solvers are utilized to solve quadratic optimization problems (QOPs) in milliseconds. To incorporate integral action and overcome wind disturbances, a velocity-based linearization of the model is employed. Two scenarios with nine waypoints each are evaluated using a hybrid strategy that combines arc-length parameterization with cubic splines and a synthetic waypoint path-planner. The path-following strategy is applied to both simplified kinematic and high-fidelity models of the aircraft, with constant wind in the first scenario and time-varying wind in the second scenario. The algorithm achieves good performance in both scenarios with short execution times. Lastly, a performance analysis of the strategy with the two path-planning techniques is presented.

I. Nomenclature

$A,\ B,\ C,\ D$	=	state space matrices
g	=	gradient - gravitational acceleration, m/s^2
h	=	altitude, m
H_s	=	Hessian matrix
I_n	=	identity matrix of size n
1_N	=	one vector $[1, 1, ..., 1]^T$ of length N
0_{lxn}	=	zero matrix of l rows and n columns
J	=	cost function
n_z	=	incremental load factor, -
p	=	roll rate, rad/s
$Q,\ R,\ T$	=	weighting matrices
u	=	control input
$\bar{u}$	=	upper limit of control effort
$\underline{u}$	=	lower limit of the control effort
ΔU	=	Vector of control inputs increments
V_a	=	air speed, m/s
V_g	=	ground speed, m/s
V_s	=	synthetic waypoint speed, m/s
V_w	=	wind speed, m/s
w	=	climb rate, m/s
x	=	system states
$x_p,\ y_p,\ z_p$	=	measured position components in ECEF coordinate system, m

*Research Assistant, Institute of Control Systems, ahmed.samir@tuhh.de
[†]Research Assistant, Institute of Control Systems, horacio.martinez.calderon@tuhh.de
[‡]Professor, Institute of Control Systems, h.werner@tuhh.de
[§]Research Assistant, Institute of Aircraft Systems Engineering, benjamin.herrmann@tuhh.de
[¶]Professor, Institute of Aircraft Systems Engineering, frank.thielecke@tuhh.de

$x_r,\ y_r,\ z_r$	=	reference position components in ECEF coordinate system, m
$X_s,\ Y_s,\ Z_s$	=	moving synthetic waypoint position components, m
y	=	output vector
$\bar{y}$	=	output upper limit
$\underline{y}$	=	output lower limit
δ_a	=	aileron command, rad
δ_e	=	elevator command, rad
δ_r	=	rudder command, rad
δ_t	=	throttle command
β	=	side slip angle, rad
χ	=	course angle, rad
ϕ	=	bank angle, rad
γ	=	flight-path angle, rad
ψ	=	heading angle, rad
θ	=	pitch angle, rad
τ	=	arc length parameter, m
α	=	angle of attack, rad
$\rho(t)$	=	time-varyig vector of scheduling parameters

II. Introduction

Over the past few decades, there has been a significant surge in demand for unmanned aerial vehicles (UAVs) due to their various applications, ranging from military surveillance to civilian mapping. As a result, the need for greater automation of aircraft systems has become increasingly important. However, this has given rise to a host of complex issues related to the concepts and technologies of future aircraft. To address these issues, the Institute of Aircraft Systems Engineering at the Hamburg University of Technology launched the unmanned low-cost testing and research aircraft (ULTRA) project [1].

Achieving aircraft autonomy is essential for numerous applications, and path following is among the most critical ones. Path-following control remains one of the most crucial applications in enhancing aircraft autonomy, resulting in significant interest in this area. There are two primary approaches to addressing path-following problems. The first approach, geometric path-following, involves steering the aircraft in yaw and pitch directions based on changes in the path evolution in pitch and yaw. This approach focuses on designing the desired path's shape and ensuring accurate tracking while minimizing tracking errors. Feedback control techniques are commonly used to adjust the system's motion and trajectory to match the desired path. The main advantage of this approach is its simplicity and ease of implementation, but it may not provide optimal or robust performance in the presence of disturbances or model uncertainties. Some commonly used geometric path-following algorithms, such as pure pursuit, proportional navigation, and line-of-sight (LOS), have been targeted in research studies regarding missile guidance. Examples of such studies include those by [2], [3], [4], and [5] for autonomous marine vehicles' guidance. The second approach, control-based path-following, involves controlling the system to follow a pre-specified path or trajectory that may not necessarily be geometric in nature. This approach utilizes control theory to design a controller that generates appropriate control inputs to steer the system toward the desired path, which may require the use of sensors, feedback control, and other techniques to achieve the desired path-following performance. Control techniques, particularly nonlinear ones, can offer a level of robustness to wind disturbances. A commonly used approach is proportional-integral-derivative (PID) control [6]. In robotics applications, including UAVs, autonomous underwater vehicless (AUVs), autonomous surface vehicless (ASVs), and unmanned ground vehicless (UGVs), various control-theoretic techniques have been developed to address path-following problems. These techniques include linear quadratic regulator (LQR) [7], sliding mode control [8], model predictive control (MPC) [9], backstepping control [10], and adaptive control [11]. The theory of nested saturation is employed in [12] to propose a path-following technique for fixed-wing UAVs. For an in-depth analysis of path-following algorithms, we recommend referring to the survey paper [13]. Moreover, a survey that concentrates on quadrotors is presented in [14]. Although many of the techniques mentioned therein are for quadrotors, they can also be utilized for fixed-wing aircraft or other types of autonomous vehicles. On one hand, most of the existing control laws for path following are nonlinear. This poses challenges in ensuring stability and performance guarantees to achieve accurate path following under different paths and environmental conditions. On the other hand, linear control laws are inadequate in fully capturing the nonlinear position dynamics.

Model predictive control has emerged as one of the most effective techniques of all the control-oriented approaches for path-following. MPC, also referred to as receding horizon control (RHC), transforms the control problem into an online optimization problem. At each sampling time, a sequence of future control values is calculated by solving a finite horizon optimal control problem. Only the first element of the computed control sequence is utilized, and the process is repeated at the next sampling time. One of the MPC's key strengths is its ability to handle constraints on inputs and states while ensuring high performance. The primary disadvantage is that computational and memory resources increase rapidly with an extended time horizon. Predictive control laws, which require a significant amount of computation between sampling steps, are predominantly employed in the process industry. This is due to the slow dynamics of systems in this industry, which typically have long sampling periods. In contrast, aerospace systems have much faster dynamics, meaning that short sampling periods are required. Consequently, implementing predictive control laws in aerospace systems can be challenging. However, ongoing research aims to develop efficient MPC algorithms for aerospace systems. This accomplishment is attributed to not only the substantial rise in computing power but also to advancements in algorithms devoted to resolving nonlinear programs (NLPs) and the emergence of efficient software frameworks for nonlinear MPC. Among the software tools that have enabled this feat are *acados*[15], *GRAMPC*[16], and *OpEn*[17], which are founded on variant techniques like sequential quadratic programming (SQP) and real-time iteration [18]. Also, in [19], the authors applied nonlinear MPC (NMPC) to achieve path-following in 2D for the coordinated flight vehicle model. They employed the *PRONTO* solver [20] to solve the optimization problem. However, the computational burden is not well-defined, and it should be considered with respect to the sampling time. Typically, the low-level controller of fixed-wing aircraft operates in milliseconds because the aircraft's dynamics are relatively fast. Therefore, the high-level path-following controller can be slower, commonly, this controller is operated in tens of milliseconds at most.

Building upon the previous discussion, there is a requirement for an approach that is computationally efficient yet still captures the nonlinear dynamics of the system. Quasi-linear parameter-varying model predictive control [21] has emerged as a candidate solution, as it utilizes a quasi-linear parameter-varying (qLPV) model to capture all the model nonlinearities. The control method solves a quadratic optimization problem with linear constraints, which can be efficiently addressed using various solvers such as quadprog [22] and osqp [23], typically within milliseconds. The quasi-linear parameter-varying model predictive control (qLMPC) approach offers an improved modeling accuracy when compared to traditional linear MPC techniques. By considering the nonlinear dynamics of the system, an improved control performance can be achieved. Additionally, it addresses uncertainties and disturbances in the system by accounting for the time-varying nature of the system's parameters, leading to greater stability margins and performance robustness. This can be accomplished by incorporating offset-free MPC into the approach, which can be attained through the application of velocity-based linearization [24]. A benchmark study on State Space vs Input-Output Approaches can be found in [25] for a control moment gyroscope (CMG).

The primary contribution of this work is the ability to apply an efficient predictive path-following strategy that captures the system's nonlinearities while meeting low computational times, in the order of a few milliseconds for the full-state model of a fixed-wing aircraft. Additionally, we use a path-planning approach that combines arc length and cubic splines to provide online path evolution based on a set of waypoints, which is primarily used in computer animations. Finally, we compare this path planner with a synthetic waypoint algorithm incorporated in the qLMPC framework.

The paper is organized as follows: In Section III, we introduce the ULTRA-Extra aircraft and both its simplified kinematic and high-fidelity models, along with a low-level controller design using successive-loop-closure (SLC) for aircraft stabilization and attitude control. Section IV covers the path-following problem and outlines a method for constructing an appropriate cost function for solving within the qLMPC framework. Two different path-planning techniques are presented in Section V. Simulation results are discussed in Section VI, followed by conclusions and suggestions for future work in Section VII.

III. ULTRA-Extra Aircraft and Controller Architechture

The ULTRA-Extra, shown in Fig. 1, is an unmanned replica of the Extra 330 ML unmanned aerobatic aircraft, scaled down to 1:2.5. It is powered by a 7.2kW electric motor[26], which allows for flight times up to 20 minutes. The aircraft has a total mass of 24.6 kg and a wingspan of 3.1 m. Control of the aircraft can be manual, with a remote controller and a safety pilot, or automatic, using a flight control computer. A real-time computer is present onboard the aircraft for measurement processing and GNC applications hosting. Dual antennas are used to measure heading and GPS position with an accuracy of up to 0.02 m. Additionally, the air data, which includes sideslip angle, angle of attack, airspeed, static air pressure, and air temperature, is obtained using a custom five-hole probe.

Fig. 1 ULTRA-Extra aircraft.

A. Point Mass Model

Prior to the application on the high-fidelity model, the predictive scheme is applied first to a simplified kinematic model of the ULTRA-Extra. This model describes the motion of the aircraft in space and time without considering any forces and moments that cause the motion. It assumes the aircraft is rigid and that its motion can be described by a few parameters, such as its position, velocity, and orientation. This model is as follows

$$\begin{cases} \dot{x}_{\mathrm{p}} = V_a \cos\gamma_a \cos\chi + V_w \cos\gamma_w \cos\chi_w, \\ \dot{y}_{\mathrm{p}} = V_a \cos\gamma_a \sin\chi + V_w \cos\gamma_w \sin\chi_w, \\ \dot{z}_{\mathrm{p}} = -V_a \sin\gamma_a - V_w \sin\gamma_w, \\ \dot{\chi} = \dfrac{g}{V_g} \tan\phi, \\ \dot{\gamma} = \dfrac{g}{V_g}(n_z \cos\phi - \cos\gamma). \end{cases} \tag{1}$$

B. Simulation Model

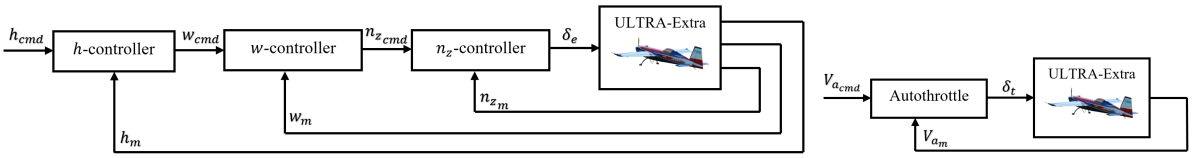
In order to perform simulation studies, a high-fidelity nonlinear model of the ULTRA-Extra aircraft has been developed. The model comprises the nonlinear aerodynamics that have been accurately identified through an extensive flight test campaign comprising 148 maneuvers [26]. A model of the electric propulsion system has been formulated to calculate the support moment and thrust for a broad range of airspeeds and throttle positions based on data obtained from a wind tunnel campaign. The dynamics of the control surface servos have been identified at component level, taking into account time delays and control surface backlash. The moment of inertia tensor has been determined via experiments. Additionally, the model incorporates computational and communication delays. Simulations of wind and atmospheric turbulence are carried out in accordance with MIL-F-8785C.

C. Stabilization and Attitude Controller Design

This section focuses on the design of the inner controller, which employs the Successive-Loop-Closure (SLC) method reported in [27] to execute the commands generated by the predictive path-following controller. The successive-loop-closure approach is a method for designing an aircraft's autopilot that involves a series of steps to progressively refine the control system design. It is based on the principle of breaking down a complex control problem into smaller, more manageable sub-problems and then designing control loops for each sub-problem. The inner and outer loops are connected and tuned to achieve the desired performance. This may involve iterative tuning and optimization to ensure that the system behaves as expected. Controlling a remotely-piloted aircraft conventionally involves using elevator (pitch), aileron (roll), rudder (yaw), and throttle (speed) inputs. Estimating attitude from the ground, such as maintaining a straight-level flight, is challenging and requires extensive training. To provide an intuitive and "care-free" ground control mode, the controller structure is chosen based on controlling the climb and sink rate of the aircraft and banking to turn, using incremental load factor and bank angle as the primary controlled quantities. Altitude-hold straight flight is the idle state of the remote control. A conventional autothrottle is used to maintain airspeed, and turn coordination is implemented through sideslip angle feedback to maintain (clean flight). SLC has been found to be a more suitable choice for the inner loop compared to other methods, such as Nonlinear Dynamic Inversion (NDI) [28], especially in

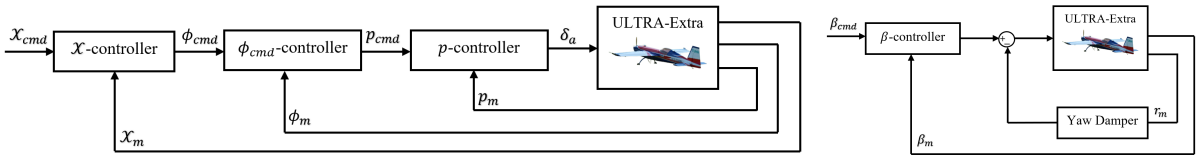
practical applications, as the inversion technique requires a high level of accuracy in the system model, which may not be possible for aircraft systems due to uncertainties. Furthermore, adjustments must be made to account for zeros in certain transfer functions, such as those found in the incremental load factor transfer function.

In order to employ the SLC method for the design of an inner controller, it is necessary to linearize the ULTRA-Extra model at various operating points that represent the diverse operational states of the aircraft. This linearization results in decoupled models for longitudinal and lateral-directional dynamics. Given that the ULTRA-Extra is designed for aerobatic maneuvers, coupling effects between roll and yaw dynamics are insignificant. Therefore, controlling all three axes can be addressed separately. To exploit the principle of integrator chains, a cascaded controller structure is selected, resulting in linear single-input-single-output control problems with a clear physical interpretation. The classical loop shaping control design technique is used to assure sufficiently high gain at low frequencies and low amplification at higher frequencies while providing robustness around the desired closed-loop bandwidth [1]. To obtain controllers that are suitable for the whole flight envelope, gain-scheduling over airspeed is applied. The linearization was performed on seven trim points with a constant altitude (h) of 200m and airspeed (Va) ranging from 20m/s to 50m/s. The structure of the low-level controller is shown in Fig. 2 and Fig.3.



(a) Altitude Channel. **(b) Airspeed Channel.**

Fig. 2 Longitudinal autopilot.



(a) Course angle channel. **(b) Side-slip channel.**

Fig. 3 Lateral autopilot.

To evaluate the controller's ability to follow commands for various attitude changes and maintain a sideslip angle $\beta = 0$, a random scenario was implemented. The results of the controller's performance are presented in Fig. 4. Careful selection of bandwidths is crucial to achieving good performance while ensuring stability and disturbance rejection. It is important to note that the inner loop should have the highest bandwidth.

IV. Problem Setup

Model predictive control is a methodology that converts the control problem into an online optimization problem, making it easier to manage constraints in control inputs and states. However, MPC has a significant drawback in that the optimization problem needs to be solved for a prediction horizon at each sampling time instant, which can be computationally expensive and requires adequate hardware resources. This can be affordable if the model is linear, but in reality, this is often not the case. To address this issue, the qLMPC approach offers a viable solution. By utilizing the qLMPC approach, the nonlinear position dynamics of the aircraft can be presented in a qLPV representation. This representation facilitates solving a quadratic cost function with linear constraints, which can be achieved using efficient solvers. As a result, the computational time required to solve the optimization problem is considerably reduced. Reference tracking is a commonly sought-after control objective, which can be achieved through predictive control laws by making slight modifications to the cost function. This section illustrates the formulation of the path-following problem using the qLMPC framework, published in [29], and how this approach can be utilized to achieve effective path-following despite the presence of wind disturbances by integration of this framework within a cascaded scheme with the ULTRA-Extra aircraft. The complete problem configuration is shown in Fig. 5. This figure demonstrates the cascaded structure in which the path planner functions as the outermost loop and produces position reference signals.

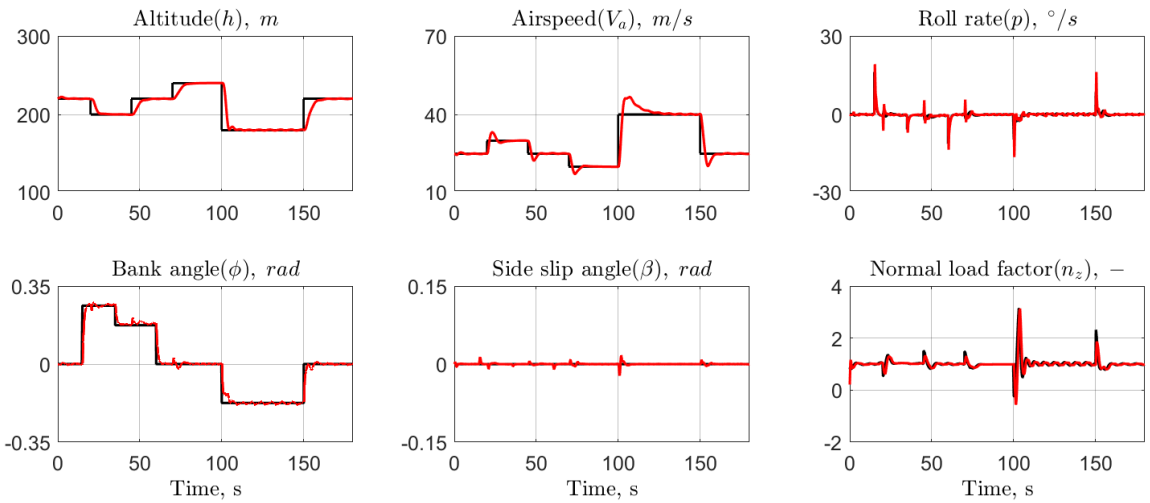


Fig. 4 Stabilization and attitude controller evaluation: command(——), measurement(——).

The path-following controller, which is responsible for producing position control commands to the low-level controller to ensure precise tracking of the path generated in the path planner module, is the least outer loop.

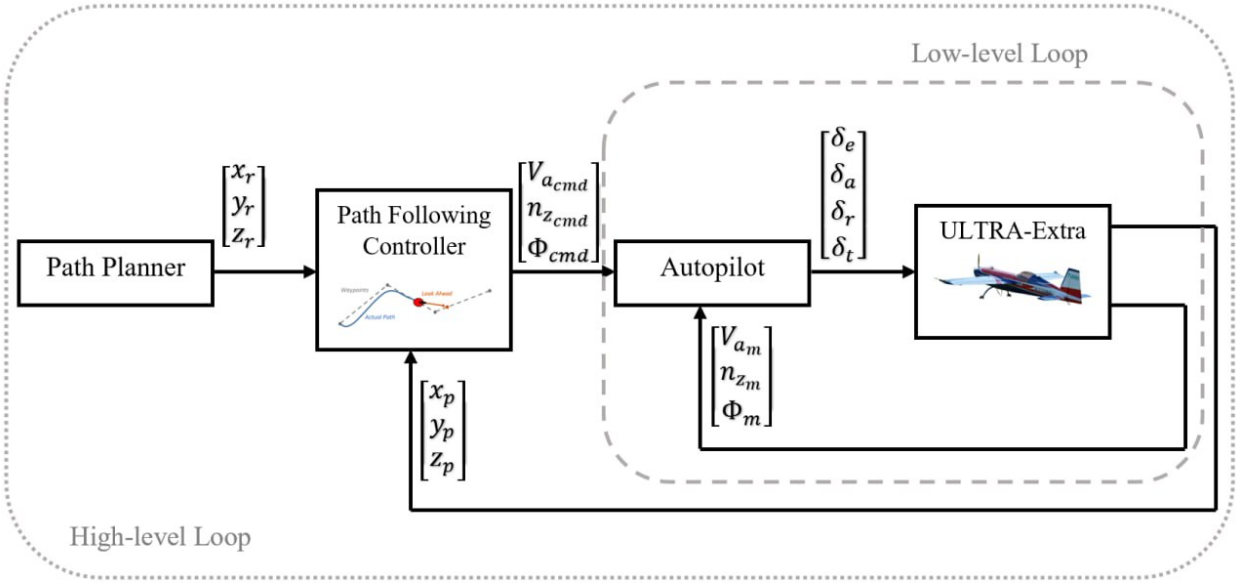


Fig. 5 Path-Following controller block diagram.

A. quasi-Linear-Parameter-Varying(qLPV) Representation

In the following, we demonstrate how to construct a qLPV model from the kinematic model (1) while maintaining the nonlinear position dynamics of the ULTRA-Extra. A general linear parameter varying (LPV) model has a state-space representation

$$\Sigma_\rho : \begin{cases} \dot{x}(t) = A(\rho(t))x(t) + B(\rho(t))u(t), \\ y(t) = C(\rho(t))x(t) + D(\rho(t))u(t), \end{cases} \tag{2}$$

where $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, and $y \in \mathbb{R}^l$. The dimensions n, m and l represent the number of states, the number of control inputs, and the number of outputs, respectively. The matrices $A(.), B(.), C(.)$, and $D(.)$ are continuous functions of

the scheduling parameter vector $\rho(t)$, and it is assumed that $\rho(t) \in \mathcal{P} \subset \mathbb{R}^{n_\rho} \quad, \forall t \geq 0$ where n_ρ is the number of scheduling parameters and $\mathcal{P}$ is a given compact set. Note that the system in (2) is linear in x and u, and it depends on the time-varying parameter ρ. In qLPV models, the parameter vector ρ is allowed to be a function of the system states and/or inputs, thus enabling an accurate representation of nonlinear system dynamics.
One possible way to obtain the nonlinear kinematic model of the ULTRA-Extra in the qLPV form (2) is through velocity-based linearization, as discussed in [30] and summarized in [24]. This technique is advantageous in the qLMPC context as it facilitates the incorporation of the integral action into the algorithm, which is beneficial in cases where the prediction model is inaccurate, or there are input disturbances such as constant or time-varying wind during flight. Consider the nonlinear model for the aircraft's position as follows

$$\dot{x} = f(x,u), \qquad\qquad y = h(x,u).$$

The use of the velocity representation of the nonlinear model makes it possible to obtain a model in the form

$$\underbrace{\begin{bmatrix}\dot{y}\\ \ddot{x}\end{bmatrix}}_{\dot{x}_v} = \underbrace{\begin{bmatrix}0 & \nabla_x h(x,u)\\ 0 & \nabla_x f(x,u)\end{bmatrix}}_{A_v}\underbrace{\begin{bmatrix}y\\ \dot{x}\end{bmatrix}}_{x_v} + \underbrace{\begin{bmatrix}\nabla_u h(x,u)\\ \nabla_u f(x,u)\end{bmatrix}}_{B_v}\dot{u}, \tag{3}$$

where $\nabla_x f(x,u)$ is the Jacobian of $f(x,u)$ with respect to x, $\nabla_u f(x,u)$ is the Jacobian of $f(x,u)$ with respect to u, $\nabla_x h(x,u)$ and $\nabla_u h(x,u)$ are the corresponding Jacobians of $h(x,u)$. Now the velocity-based model of the ULTRA-Extra kinematic model has the state vector $x = [x_\mathrm{p} \;\; y_\mathrm{p} \;\; z_\mathrm{p} \;\; \chi \;\; \gamma \;\; \dot{x}_\mathrm{p} \;\; \dot{y}_\mathrm{p} \;\; \dot{z}_\mathrm{p} \;\; \dot{\chi} \;\; \dot{\gamma}]^T$. It is important to note that velocity-based linearized models are linear in $\dot{x}$ and $\dot{u}$, rather than x and u. This means that information about x is lost and careful consideration must be given to the initial conditions. However, the model is augmented with derivative dynamics, which allows for linearization around any operating point rather than just equilibrium points, as in Jacobian linearization-based models. It should also be noted that the system matrix A_v has eigenvalues at 0, indicating the presence of integral action in the system. Now the system in (3) is in a qLPV form where the state and input matrices are ρ dependent. By applying this to the ULTRA-Extra kinematic model, we get

$$\begin{bmatrix}\dot{x}_\mathrm{p}\\ \dot{y}_\mathrm{p}\\ \dot{z}_\mathrm{p}\\ \dot{\chi}\\ \dot{\gamma}\\ \ddot{x}_\mathrm{p}\\ \ddot{y}_\mathrm{p}\\ \ddot{z}_\mathrm{p}\\ \ddot{\chi}\\ \ddot{\gamma}\end{bmatrix} = \left[\begin{array}{ccccc} 0_n & & I_n & & \\ \hline & & & -V_a\cos\gamma\sin\chi & -V_a\cos\chi\sin\gamma \\ & & & V_a\cos\chi\cos\gamma & -V_a\sin\chi\sin\gamma \\ 0_n & 0_{n\times 3} & & 0 & V_a\cos\gamma \\ & & & 0 & 0 \\ & & & 0 & \frac{g}{Va}\sin\gamma \end{array}\right]\begin{bmatrix}x_\mathrm{p}\\ y_\mathrm{p}\\ z_\mathrm{p}\\ \chi\\ \gamma\\ \dot{x}_\mathrm{p}\\ \dot{y}_\mathrm{p}\\ \dot{z}_\mathrm{p}\\ \dot{\chi}\\ \dot{\gamma}\end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\\ \cos\chi\cos\gamma & 0 & 0 & 0\\ \cos\gamma\sin\chi & 0 & 0 & 0\\ \sin\gamma & 0 & 0 & 0\\ -\frac{g}{V_a^2}\tan\phi & \frac{g\tan\phi^2+1}{V_a} & 0 & 0\\ \frac{g\cos\gamma - n_z\cos\phi}{V_a^2} & -\frac{gn_z\sin\phi}{V_a} & \frac{g\cos\phi}{V_a} & 0 \end{bmatrix}\begin{bmatrix}V_a\\ \phi\\ n_z\end{bmatrix} \tag{4}$$

It should be noted that the system and input matrices are now dependent on ρ, where $\rho = [\chi \;\; \gamma \;\; V_a \;\; \phi \;\; n_z]^T$. To utilize this model, it needs to be discretized. One possible discretization method is Runge-Kutta 4 (RK4). Thus, a discretization of model (4) will be in the form:

$$x_{v,k+1} = A_v(\rho_k)x_{v,k} + B_v(\rho_k)\dot{u}_k. \tag{5}$$

This model has the input rate as a control input. Hence, it is preferable to work with input increments to enable the use of the algorithm for MPC with integral action. Moreover, this model can be employed within the qLMPC algorithm and can be further enhanced with the inclusion of error dynamics, as we will explore in the following subsection. This augmentation can offer significant advantages in achieving effective path-following performance by transforming the tracking problem into a set point problem while still maintaining a high level of performance.

B. Cost Function Formulation

In an optimization problem, the cost function is a mathematical function that measures the quality of a particular solution to the problem subject to any constraints on the variables. In order to achieve effective path-following in three dimensions, a suitable cost function must be constructed to minimize tracking errors. Augmenting the cost function with the error dynamics is found to be a more effective approach for the underlying control problem, as it allows for the conversion of the tracking problem to a set point problem, thereby simplifying the solution process while still enabling the weighting of the system states to ensure good performance and smooth flight. One way to derive the error dynamics is to expand the model in (4) as follows

$$\begin{bmatrix} \dot{y} \\ \ddot{x} \\ \dot{e} \end{bmatrix} = \left[\begin{array}{cc|c} & & 0_{2n\times l} \\ & A_v & \\ & & \\ \hline 0_{l\times n} & \dfrac{dr}{dx}\times I_n - I_n & 0_l \end{array}\right] \begin{bmatrix} y \\ \dot{x} \\ e \end{bmatrix} + \left[\begin{array}{c} B_v \\ \hline 0_{l\times m} \end{array}\right] \begin{bmatrix} V_a \\ \phi \\ n_z \end{bmatrix}, \tag{6}$$

where $e = [x_\mathrm{r} - x_\mathrm{p} \quad y_\mathrm{r} - y_\mathrm{p} \quad z_\mathrm{r} - z_\mathrm{p} \quad \psi_\mathrm{r} - \psi_\mathrm{m} \quad \gamma_\mathrm{r} - \gamma_\mathrm{m}]^T$ is the error vector. Model (6) represents a qLPV representation of the kinematic model of the ULTRA-Extra with states and control inputs available for measurement augmented with the error dynamics. This model can be updated at each time instant to obtain an LTI representation. This allows us to create a quadratic cost function with linear constraints, which can be solved efficiently as a quadratic program using a range of available solvers. Now the cost function can be written as follows

$$J = \sum_{i=1}^{N} \left(x_{k+i}^T Q x_{k+i} + u_{k+i-1}^T R u_{k+i-1} + e_{k+i}^T T e_{k+i} \right). \tag{7}$$

where $Q \in \mathbb{R}^{n\times n}$ is a symmetric positive semi-definite matrix and $R \in \mathbb{R}^{m\times m}$ is a symmetric positive definite matrix to penalise the states and the control effort, respectively. The matrix T is an additional semi-positive definite matrix to penalise the tracking errors. We can further enhance the smoothness of the flight by imposing additional constraints on the control input rate, which ensures that there are no abrupt changes in the control input. The representation commonly used for the constraints in the control input is as follows

$$\underline{u} \leq u \leq \bar{u},$$

Constraints on control input rate, states, and outputs can also be incorporated and effectively managed in the optimization problem such that:

$$U_k^* = \min_{U_k} J_k \qquad s.t. \quad AU_k \leq b,$$

where $A \in \mathbb{R}^{q\times p}$ is a matrix multiplied by the decision variables U_k and $b \in \mathbb{R}^q$ is the constraints vector. The quadratic program can be formulated as

$$\min_U \frac{1}{2} U^T H_\mathrm{s} U + g^T U \qquad s.t. \qquad AU \leq b, \tag{8}$$

where $U \in \mathbb{R}^p$ is the vector of decision variables, $H_\mathrm{s} \in \mathbb{R}^{p\times p}$ is the Hessian, $g \in \mathbb{R}^p$ is the gradient vector, . The scalar p is the number of decision variables, and q is the number of constraints. The objective is to solve the quadratic program mentioned in (8) to obtain the vector U that minimizes the given quadratic cost function. This optimization problem can be efficiently solved using e.g. the command *quadprog* in the MATLAB Optimization Toolbox. The aim is now to express the enhanced cost function in equation (7) as a quadratic program in the form of (8). The prediction equation can be written as

$$X_k = H(P_k) x_k + S(P_k) U_k, \tag{9}$$

where P_k such that $P_k = [\rho_k \quad \rho_{k+1} \cdots \rho_{k+N-1}]^T$is the vector of future scheduling variables and $X_k \in \mathbb{R}^{Nn}$ is the vector of state predictions such that $X_k = [x_{k+1}^T \quad x_{k+2}^T \quad \cdots \quad x_{k+N}^T]^T$, and $U_k \in \mathbb{R}^{Nm}$ is the future control inputs such that $U_k = [u_k^T \quad u_{k+1}^T \quad \cdots \quad u_{k+N-1}^T]^T$, $H \in \mathbb{R}^{Nn\times n}$ is the Hessian matrix multiplied by the actual state $x_k \in \mathbb{R}^n$, $S \in \mathbb{R}^{Nn\times Nm}$ is a Toeplitz matrix multiplied by the vector of future control inputs U_k. Following (9), the output prediction equation will be:

$$Y_k(P_k) = \tilde{C} H(P_k) x_k + \tilde{C} S(P_k) U_k, \tag{10}$$

where $Y_k \in \mathbb{R}^{Nl}$ is the vector of future outputs, $\tilde{C} \in \mathbb{R}^{Nl \times Nn}$ is a matrix such that $\tilde{C} = I_N \otimes C$ Now the cost function in (7) can be expanded as

$$J_k = X_k^T \tilde{Q} X_k + U_k^T \tilde{R} U_k + (R_k - \tilde{C}H(P_k)x_k - S(P_k)U_k)^T \tilde{T}(R_k - \tilde{C}H(P_k)x_k - S(P_k)U_k). \tag{11}$$

Note that R_k represents the reference position values and should not be confused with the symbol R as a weighting matrix. The same as $\tilde{C}$, $\tilde{Q} \in \mathbb{R}^{Nn \times Nn}$ such that $\tilde{Q} = I_N \otimes Q$ and $\tilde{R} \in \mathbb{R}^{Nm \times Nm}$ is the result of $I_N \otimes R$. Note that eq. (11) depends on X_k and U_k. One can get rid of the X_k dependency by substituting eq. (9) in eq. (11). So this can be written as

$$\begin{aligned} J_k = (H(P_k)x_k + S(P_k)U_k)^T \tilde{Q}(H(P_k)x_k + S(P_k)U_k) + U_k^T \tilde{R} U_k + (R_k - \tilde{C}H(P_k)x_k - \\ S(P_k)U_k)^T \tilde{T}(R_k - \tilde{C}H(P_k)x_k - S(P_k)U_k). \end{aligned} \tag{12}$$

We can then rewrite (12) as

$$J_k = \frac{1}{2} U_k^T \underbrace{2(S^T(P_k)\tilde{Q}S(P_k) + \tilde{R} + S^T(P_k)\tilde{T}S(P_k))}_{H_s} U_k + \underbrace{2(x_k^T H^T(P_k)\tilde{Q}S(P_k) - R_k^T \tilde{T}S(P_k) + x_k^T H^T(P_k)\tilde{C}^T \tilde{T}S(P_k))}_{g^T} U_k. \tag{13}$$

The cost function for the path-following problem is now in a form that can be efficiently solved by any quadratic programming solver. The next step is to incorporate the constraints into the optimization problem. In order to proceed, it is necessary to construct the matrix A and vector b that represent the required constraints as in (8), as well as identify the necessary input, output, and state constraints. By considering that the system has input and output constraints, one can build A matrix and b vector as

$$A = \begin{bmatrix} I_{Nm} \\ -I_{Nm} \\ \tilde{C}S(P_k) \\ -\tilde{C}S(P_k) \end{bmatrix}, \quad b = \begin{bmatrix} \bar{U} \\ -\underline{\mathrm{U}} \\ \bar{Y} - \tilde{C}H(P_k)x_k \\ -\underline{\mathrm{Y}} + \tilde{C}H(P_k)x_k \end{bmatrix}. \tag{14}$$

where $\bar{U} = 1_N \otimes \bar{u}$ and $\underline{U} = 1_N \otimes \underline{u}$. Similarily, $\bar{Y} = 1_N \otimes \bar{y}$ and $\underline{Y} = 1_N \otimes \underline{\mathrm{y}}$. Note that output constraints do not depend on U_k, but one can substitute eq.(10) to have a dependency on U_k. Using the matrix and the vector in (14) the QP will be

$$\begin{aligned} U_k^* = \min_{U_k} \frac{1}{2} U_k^T 2(S^T(P(k))\tilde{Q}S(P_k) + \tilde{R} + S^T(P_k)\tilde{T}S(P_k))U_k + 2(x_k^T H^T(P_k)\tilde{Q}S(P_k) - R_k^T \tilde{T}S(P_k) + \\ x_k^T H^T(P_k)\tilde{C}^T \tilde{T}S(P_k))U_K \quad s.t. \quad AU_k \leq b, \end{aligned} \tag{15}$$

which can be solved as a quadratic program.

C. qLMPC Algorithm

So far, we have discussed how to obtain a qLPV model that captures the nonlinear position dynamics of the aircraft using velocity-based linearization. We have also explained how to construct the cost function with integral action to achieve perfect path-following, which will be passed to the QP solver. The next step is to utilize this cost function in the context of the qLMPC framework. To implement the qLMPC algorithm, the following steps can be applied:

1) Define the tuning matrices Q and R, the horizon N, and the set of allowable inputs U.
2) At time k, solve problem (15) using the qLPV model; since the value of future scheduling trajectories P_k^0 is unknown where $P_k^0 = [\rho_k^0 \quad \rho_{k+1}^0 \quad \cdots \quad \rho_{k+N-1}^0]$, the model will be a linear time-invariant (LTI) system. The optimal scheduling sequence $P_k^{l=0} = 1_N \otimes f(x_k, u_{k-1})$, where l is the number of iterations used to find the optimal scheduling sequence P_k^*.
3) Solve problem (15) to obtain the future control inputs U_k^{l+1}. Using these inputs, calculate the future state predictions $X_k^{l=1}$ and therefore $P_k^{l=1}$. Then solve problem (15) again to obtain $X_k^{l=2}, U_k^{l=2}$ and $P_k^{l=2} = f(X_k^2, U_k^2)$. Since the sequence of scheduling trajectories $P_k^{l=1}$ is not constant, the qLPV model is now a linear time-varying system(LTV).

4) Iterate the last step until the scheduling sequence P_k^l approaches its optimal value $P_k^* = f(X_k^*, U_k^*)$, where X_k^* and U_k^* denote the optimal solution of problem (15).
5) Save X_k^l and U_k^l after the last iteration to calculate P_{k+1}^0 in the next time step. Then apply u_k to the system, which is the first value of U_k^l.
6) The process is then repeated again at sampling instant $k+1$, utilizing P_{k+1}^0.

This is summarised in Algorithm 1.

Algorithm 1 quasi-linear MPC [29]

Initialisation: Model, $\hat{Q}$, $\hat{R}$, N

$k \leftarrow 0$
Define $P^0 = 1_N \otimes \rho(x(0), u(0))$
repeat
  $l \leftarrow 0$
  repeat
    Solve QP(15) with P_k^l for ΔU_k^l
    Predict state $X_k^l = H(P_k^l)x_k + S(P_k^l)\Delta U_k^l$
    Define $P_k^{l+1} = \rho(X_k^l, U_k^l)$
    $l \leftarrow l + 1$
  until stop criterion
  Apply $u_k = u_{k-1} + \Delta u_k$
  Define $P_{k+1}^0 = \rho(X_k^l, U_k^l)$
  $k \leftarrow k + 1$
until end

V. Path-Planning Techniques

In this section, we will discuss two distinct techniques for path planning and demonstrate how to utilize them to generate a trackable path in ECEF coordinates. The resulting position data will then serve as a reference signal for the path-following controller. Path-following and simple waypoint flight are two different tasks. While the latter connects consecutive waypoints by straight lines, the former requires defining a continuous path segment between waypoints. Typically, Dubins path [31] has been used to create a continuous path, but both approaches often lead to discontinuities in the path curvature. This makes it challenging for a UAV to precisely follow the prescribed path due to nonholonomic constraints that restrict the aircraft's motion in a non-integrable direction. To overcome this issue, we present two path-planning techniques in this section. The first approach is an arc length parameterized path planner with combined splines, and the second approach is a synthetic waypoint-based path planner. We will use these techniques to construct two different paths defined in terms of waypoints in the ECEF coordinate system.

A. Hybrid Approach: Combining Arc-length with Cubic Splines

The approach presented here is based on the technique introduced in [32] for computer animation and virtual environments. However, we have made some modifications to the arc length dynamics inspired by [33]. In their work, the path evolution is determined by treating the arc length as an additional state variable that evolves online. We have adapted this technique to make it applicable to aerospace applications. To smooth the path, cubic splines are added, as was done in [26], where they were used in the context of a nonlinear guidance law (NLGL) path following application. The cubic splines provide paths that are twice differentiable, thus ensuring that there are no discontinuities in the path that could violate the nonholonomic constraints of the aircraft. The resulting augmented path takes the form of $\vec{S}_k(\tau) = [S_x^k(\tau) \;\; S_y^k(\tau) \;\; S_z^k(\tau)]^T$, $k \in \{1, ..., n-1\}$ where n represents the number of waypoints and $\vec{S}_k(\tau)$ is the spline segment and τ here is referred to as the arc length parameter. This approach allows the ULTRA-Extra to calculate the path on board once a list of waypoints is provided or updated. Unlike in [26], where the splines are time-dependent, in this design, they are a function of the arc length parameter defined in the state vector as $\dot{\tau} = \lambda\tau + v$, where v is a virtual control input that controls the path evolution and λ is a constant, which we chose here to be $-10^{-3}\ s^{-1}$ to ensure the path dynamics stability with negative eigenvalue.
The path $\mathcal{P}$ is given by a regular, sufficiently often differentiable, such that $\forall\tau \in [0, \hat{\tau}] \;:\; \tau \rightarrow p(\tau) \in \mathcal{X} \subseteq \mathbb{R}^n$ and

$p(0) = 0$ holds. Therefore, the path $\mathcal{P}$ to be followed can be defined as $\mathcal{P} = \{p(\tau) \in \mathbb{R}^n : [0, \hat{\tau}] \rightarrow p(\tau)\}$. The reference path will be now denoted by $[x_r(\tau) \;\; y_r(\tau) \;\; z_r(\tau) \;\; \psi_r(\tau) \;\; \gamma_r(\tau)]^T$ and can be defined as

$$p(\tau) = \begin{bmatrix} a\tau^3 + b\tau^2 + c\tau + d \\ e\tau^3 + f\tau^2 + g\tau + h \\ i\tau^3 + j\tau^2 + k\tau + l \\ \tan^{-1} \dfrac{dy(\tau)}{dx(\tau)} \\ \sin^{-1} \dfrac{dz(\tau)}{\sqrt{dx^2(\tau) + dy^2(\tau)}} \end{bmatrix},$$

where $a, b, c, d, e, f, g, h, I, j, k$, and l are constant coefficients that depend on the path waypoints and need to be calculated a priori depending on the path and then stored in the aircraft's memory. Then, the path will be updated online to reflect the changes in τ. To do so, the kinematic model of the aircraft will be augmented now with the arc length parameter dynamics so the model in (1) will be augmented as follows

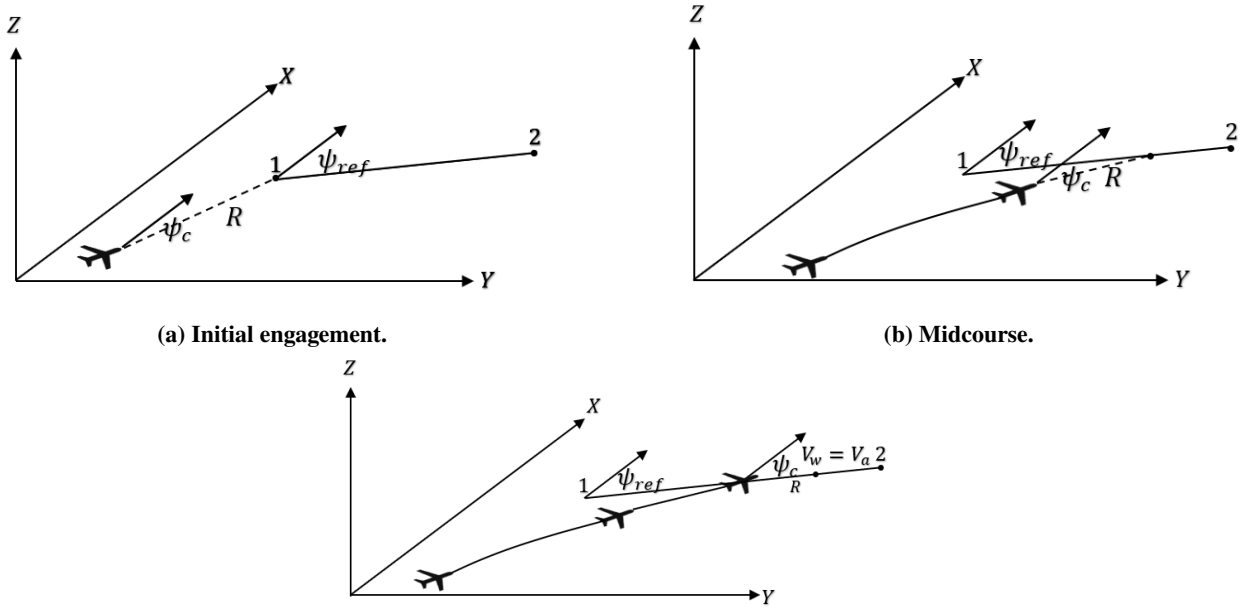
$$\begin{cases} \dot{x}_\mathrm{p} = V_a \cos\gamma \cos\chi + V_w \cos\gamma_w \cos\psi_w, \\ \dot{y}_\mathrm{p} = V_a \cos\gamma \sin\chi + V_w \cos\gamma_w \sin\psi_w, \\ \dot{z}_\mathrm{p} = -V_a \sin\gamma - V_w \sin\gamma_w, \\ \dot{\chi} = \dfrac{g}{V_a} \tan\phi, \\ \dot{\gamma} = \dfrac{g}{V_g}(n_z \cos\phi - \cos\gamma), \\ \dot{\tau} = \lambda\tau + v, \end{cases}$$

and therefore the velocity-based linearized model in (4)will also be updated as follows

$$\begin{bmatrix} \dot{x}_\mathrm{p} \\ \dot{y}_\mathrm{p} \\ \dot{z}_\mathrm{p} \\ \dot{\chi} \\ \dot{\gamma} \\ \dot{\tau} \\ \ddot{x}_\mathrm{p} \\ \ddot{y}_\mathrm{p} \\ \ddot{z}_\mathrm{p} \\ \ddot{\chi} \\ \ddot{\gamma} \\ \ddot{\tau} \end{bmatrix} = \left[\begin{array}{c|ccc} 0_n & \multicolumn{3}{c}{I_n} \\ \hline & -V_a\cos\gamma\sin\chi & -V_a\cos\chi\sin\gamma & 0 \\ & V_a\cos\chi\cos\gamma & -V_a\sin\chi\sin\gamma & 0 \\ & 0 & V_a\cos\gamma & 0 \\ 0_n \;\; 0_{n\times 3} & 0 & 0 & 0 \\ & 0 & \frac{g}{Va}\sin\gamma & 0 \\ & 0 & 0 & \lambda \end{array}\right] \begin{bmatrix} x_\mathrm{p} \\ y_\mathrm{p} \\ z_\mathrm{p} \\ \chi \\ \gamma \\ \tau \\ \dot{x}_\mathrm{p} \\ \dot{y}_\mathrm{p} \\ \dot{z}_\mathrm{p} \\ \dot{\chi} \\ \dot{\gamma} \\ \dot{\tau} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \cos\chi\cos\gamma & 0 & 0 & 0 \\ \cos\gamma\sin\chi & 0 & 0 & 0 \\ \sin\gamma & 0 & 0 & 0 \\ -\frac{g}{V_a^2}\tan\phi & \frac{g\tan\phi^2+1}{V_a} & 0 & 0 \\ \frac{g\cos\gamma - n_z\cos\phi}{V_a^2} & -\frac{gn_z\sin\phi}{V_a} & \frac{g\cos\phi}{V_a} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} V_a \\ \phi \\ n_z \\ v \end{bmatrix}.$$

B. Path Planner using Synthetic-Waypoint Guidance Algorithm

The SWG algorithm [34] utilizes a flight path that is initially defined to establish the desired trajectory for the aircraft. The flight path waypoints are specified in inertial space with reference to a fixed local frame. Additionally, a synthetic waypoint is introduced at the location of the first actual waypoint. The SWG algorithm operates by tracking the synthetic waypoint as it moves along the designated flight path. This synthetic waypoint is considered as such because its position on the flight path is projected as the position at which the aircraft intends to be within a specified time horizon. This time horizon is determined by the user and represents the time interval by which the aircraft trails the synthetic waypoint. As the aircraft approaches the synthetic waypoint on the flight path, the synthetic waypoint is repositioned, thereby compelling the vehicle to adopt a new heading to pursue it. The previous discussion highlights that the SWG can be viewed as a virtual Point Tracking (VPT) technique. In this work, we integrate it with the qLMPC algorithm to improve the path following performance. This differs from previous studies, such as [34], where the SWG was utilized in combination with a pursuit guidance algorithm.



(a) Initial engagement.

(b) Midcourse.

(c) Flight path acquisition.

Fig. 6 Synthetic waypoint engagement dynamics.

Figure 6 illustrates the SWG algorithm, which starts with the aircraft located at a distance R from the first waypoint, which is also the first synthetic waypoint. The commanded heading angle ψ_c is used to direct the aircraft towards the path. As the aircraft approaches the synthetic waypoint, the distance R decreases, but to ensure algorithm stability, it should not be greater than a predetermined distance R^* called virtual distance and set during the design phase ($R \geq R^*$). The calculation of the maximum allowed virtual distance R^* can be performed using the relation $R^* = V_a T_H$, where T_H represents the desired time horizon for the aircraft to respond to changes in the flight path. It is worth noting that this time horizon is similar to the prediction horizon used in predictive control algorithms. The synthetic waypoint is only allowed to travel along the flight path between the required waypoints to ensure it evolves in the correct direction. This condition can be expressed mathematically as follows

$$\begin{bmatrix} X_s \\ Y_s \\ Z_s \end{bmatrix}_i = \begin{bmatrix} X_s \\ Y_s \\ Z_s \end{bmatrix}_{i-1} + V_s \begin{bmatrix} \cos\gamma_{ref} \cos\psi_{ref} \\ \cos\gamma_{ref} \sin\psi_{ref} \\ -\sin\gamma_{ref} \end{bmatrix} \Delta T.$$

The three-dimensional position of the moving synthetic waypoint in inertial space is defined by X_s, Y_s, and Z_s. V_s represents the speed of the synthetic waypoint. The reference heading and flight-path angles are denoted by ψ_{ref} and γ_{ref}, respectively. These angles determine the required orientation between each pair of consecutive waypoints and are updated after the aircraft successfully visits each waypoint. ΔT is the sampling time of the position update of the synthetic waypoint. To examine the stability of this algorithm, please refer to [35].

C. Comparison Between the Hybrid and Synthetic Waypoint Path Planners

In this subsection, we compare the two path-planner approaches and discuss the advantages of each. The Hybrid approach offers the advantage of ensuring the smoothness of the path, which is crucial for applications requiring precise and continuous motion. This approach, which is arc length dependent, also ensures that the path is continuous without sudden changes in direction or velocity. Therefore, tracking errors are smaller compared to the synthetic waypoint approach, as discussed in the following section. Moreover, the use of splines reduces the computational cost of path planning by eliminating the need to compute the path curvature explicitly. However, the disadvantage of this method is that the path needs to be arc length parameterized beforehand, making it a preprocessing step. This can be done offline, so there is no online computational burden.

On the other hand, the synthetic waypoint path-planner also provides a smooth and continuous reference path by creating a straight line path between waypoints and smoothing the transition between waypoints to provide a smooth transition that the aircraft can follow while obeying the nonholonomic constraints of the aircraft. Another strength of the synthetic waypoint path planner is its better ability to handle obstacles since it is more flexible in mission planning compared to the hybrid approach, where the path dynamics evolve online as an additional state.

VI. Simulation Results

This section presents a comprehensive analysis of the simulation results of the qLMPC path-following. In order to verify the effectiveness of the path planning and path following algorithms, two different three-dimensional paths were generated prior to commencement. The scenarios in this study consist of nine waypoints, with the first and last waypoints being the same, and the altitude varies between waypoints. This was done to rigorously test the algorithm's performance on a 3D path. It should be noted that the aircraft's airspeed is considered as a degree of freedom and is commanded by the path-following controller, unlike in previous studies such as [26] and [36] where the airspeed was controlled separately from the path-following controller. To prevent the ULTRA-Extra from flying at excessively high airspeeds, which could be determined by the controller, and since the path-following mode is assumed to be activated during flight, the aircraft's airspeed cannot be reduced to zero. Therefore, the airspeed was constrained between 20 m/s and 50 m/s. This is also advantageous for high-fidelity simulations that rely on an inner loop controller designed based on a gain-scheduled controller for seven different linearized models at different airspeeds ranging between 20-50 m/s. The simulations were performed on a computer equipped with an 8-Core processor running at 3.6 GHz. The quadprog solver in the MATLAB Optimization Toolbox was utilized for the simulations. In the first scenario, the waypoints are set up to resemble an aerodrome circuit with altitudes ranging from 200 m to 280 m. A constant wind with a mean value of $V_w = 4$ m/s and an orientation of $\chi_w = 89^\circ$ is simulated. The second scenario is more challenging, with nine waypoints forming a path similar to a roller-coaster ride, with altitude variations ranging from 180 m to 280 m. During the simulation test, a time-varying wind with V_w ranging from 1 m/s to 5 m/s and an orientation of $\chi_w = 335^\circ$ is simulated, as shown in Fig. 7. Table 1 demonstrates the two scenario's waypoints in ECEF coordinates. Performance evaluation is conducted by means of a cost function $J = \frac{1}{t_f - t_0} \int_{t_0}^{t_f} \|e\| \, dt$, which represents the average track error along the path. This is similar to a previous study on the ULTRA-Extra [36], where the Nonlinear Guidance Law (NLGL) and Nonlinear Differential Geometric Path-Following Guidance Law (NDGPFG) were utilized, but for a different scenario. To validate the efficacy of the qLMPC algorithm with a fixed-wing aircraft in a preliminary simulation, we opt for the point mass acceleration model, which implies that the movement of the aircraft is independent of its orientation and that moments are disregarded. This choice is based on its straightforwardness, reliable estimation of the aircraft's motion, and computational efficiency. Moreover, it is capable of withstanding measurement errors or uncertainties while providing reliable estimates of the vehicle's trajectory, even when the measurements are subject to noise or uncertainty. We then proceed to apply the algorithm to the high-fidelity model, replicating the same scenarios while assuming that the safety pilot would manually guide the aircraft close to the desired path before activating the path-following mode at the first waypoint. It is important to note that the controller would require further tuning to achieve good results with the full model and that the performance would not be as high as that on the kinematic model due to several factors, including the finite bandwidth of the full model and the time delays resulting from the actuators and computational times. Despite this, no structural changes to the controller were necessary. In this paper, we present the tracking figures solely for the high-fidelity model to avoid redundancy while providing a comparison of errors between simulations using both the kinematic and full models.

A. Hybrid Path-Planner

In the case of the hybrid path-planner, the first flight test scenario demonstrated that the aircraft was able to accurately follow the path with an average error of 0.66 m in the kinematic model simulation and 1.09 m in the full model simulation. However, the maximum error in the kinematic model simulation was around 1.6 m, which increased to 2.06 m in the full model simulation, as expected. The performance of the path-following algorithm for the first scenario is shown in Fig. 8. The figure shows that the error stays below 1.1 m for approximately 50% of the path duration and below 1.7 m for 80% of the path duration, indicating the efficacy of the path-following algorithm employed.

The results for a more difficult scenario, with a maximum curvature about twice that of the first scenario, are shown in Fig.9. The same simulation procedures were followed as in the first scenario, and a maximum error of 1.93 m was obtained in the kinematic model, which increased to 4.41 m in the full model simulation. The average error along the

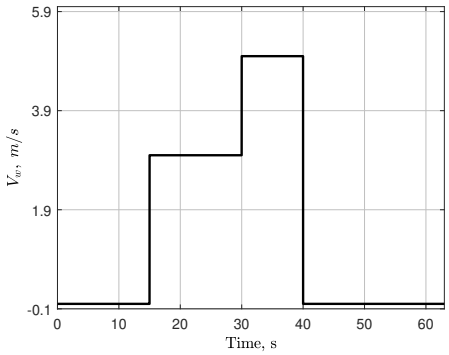


Fig. 7 Wind speed for the second scenario.

Table 1 Waypoints, $[x, y, z]$ used for path definition.

No.	Position, m	No.	Position, m	No.	Position, m
Aerodrome Circuit Path					
1	[110,-40,220]	4	[551,122,270]	7	[-157,272,200]
2	[333,-177.7,220]	5	[203.5,480,250]	8	[-57,72,220]
3	[511,-100,250]	6	[-97,422.3,200]	9	[110,-40,220]
'Roller coaster' like Path					
1	[155,315,220]	4	[400,420,250]	7	[-190,490,280]
2	[260,50,250]	5	[155,315,180]	8	[40,600,250]
3	[500,150,280]	6	[-80,220,250]	9	[155,315,220]

path was 0.33 m in the point mass model simulation, which increased to 1.94 m in the high-fidelity model simulation. Table 2 presents a comparison of the errors between the kinematic model and the full model simulations for both scenarios.

One can compare this method with [26], where identical scenarios were tested using the Nonlinear Guidance Law(NLGL) path following controller with time-dependent cubic splines, resulting in maximum errors of approximately 2.6 m and 5 m for the first and second scenarios, respectively. Additionally, the average path time (APT) in [26] was approximately 85 s and 62 s for the first and second scenarios, respectively, while in this work, it was approximately 56 s and 60 s for the first and second scenarios, respectively. The difference in APTs could be due to the fact that in this work, the airspeed was left as a degree of freedom for the path following controller to decide between 20-40 m/s, whereas in [26], it was always commanded to be between 25-30 m/s, separately from the path following controller. Furthermore, it can be observed from Fig. 8 and Fig. 9 that the computation time required to solve the optimization problem for implementing the qLMPC algorithm did not exceed the sampling time, which was 0.01 s and was always in the range of a few milliseconds.

Table 2 Average error comparison - Hybrid approach path-planner.

	'Aerodrome Circuit' Path		'Roller coaster' like Path	
	Full Model	Kinematic Model	Full Model	Kinematic Model
Min Error, m	0.112	0.02	0.45	0.04
Max Error, m	2.0643	1.61	4.41	1.93
Average Error, m	1.09	0.66	1.94	0.33
Average Path Time(APT), s	56		60	

B. Synthetic Waypoint

In the following, we evaluate the effectiveness of the path-following controller using the synthetic waypoint path planner described in V.B with the qLMPC algorithm. As in the previous section, we tested the controller on the same scenarios using both the kinematic and full models of the ULTRA-Extra. Although the tracking was satisfactory, the errors were slightly higher than in the previous subsection. The average tracking error for the first scenario on the kinematic model was approximately 0.81 m, while it was around 3.15 m for the full model, with a maximum error of approximately 4.89 m for the full model and 1.724 m for the kinematic model. Similarly, for the second scenario, the average tracking error was approximately 1.65 m for the kinematic model and 4.98 m for the full model, with a maximum error of 6.79 m for the kinematic model and 8.75 m for the full model. These errors were most likely due to the tuning parameter of the error distance between the transition points of the waypoints. In the first scenario, the error distance between transition points of the synthetic waypoint was set to 7 m, while in the second scenario, it was set to 12[m] to ensure successful waypoint visitation. As a result, the controller aims to minimize the tracking error after each waypoint transition, which is not the case in the arc length parameterization path planner, where the transitions are smoothed using cubic splines. The application of the synthetic waypoint path planner algorithm in the context of the qLMPC framework is demonstrated in Fig. 10 and Fig. 11. It can be observed that the error was less than 4 m for around 50% of the path duration in the first scenario and less than 6 m for around 60% of the path duration for the second scenario and also the minimum error is around 1.36 m. These errors are acceptable for an aircraft with a wingspan of around 3.1 m and a weight of 24.6 kg and for paths that span an area of around 3.6 km^2. It is worth mentioning that the execution times remained within a few milliseconds, ensuring the successful application of the qLMPC algorithm without any issues. The average path time for the first scenario was approximately 57 s, and for the second scenario, it was approximately 61 s, which is comparable to the times obtained using the hybrid approach. For a comparison of the errors observed when the algorithm was applied to the kinematic model and the full model, please refer to Table 3.

Table 3 Average error comparison - Synthetic waypoint path-planner.

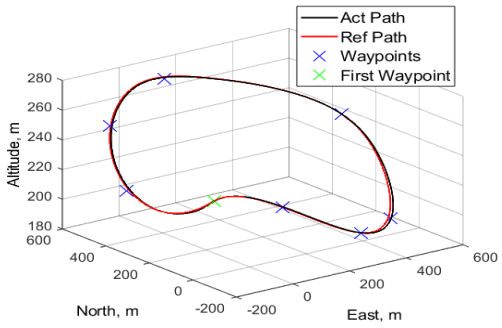
	'Aerodrome Circuit' Path		'Roller coaster' like Path	
	Full Model	Kinematic Model	Full Model	Kinematic Model
Min Error, m	0.15	0.10	1.36	1.24
Max Error, m	4.89	1.724	8.75	6.79
Average Error, m	3.15	0.81	4.98	1.65
Average Path Time(APT), s	57		61	

VII. Conclusion

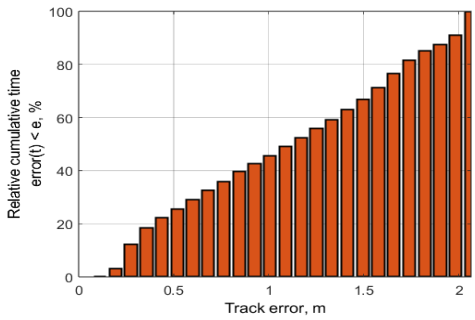
This paper presented an efficient quasi-linear model predictive path-following controller for the ULTRA-Extra aircraft based on velocity-based linearization of its simplified kinematic model. The controller was applied to both the kinematic and high-fidelity models and tested on two different scenarios, each consisting of nine waypoints generated using two different approaches. The first approach utilizes a combination of arc length and cubic splines while the second approach uses the synthetic waypoint guidance algorithm. Both scenarios yield good results for both models. This work serves as a precursor to future experiments on the ULTRA-Extra aircraft preceded by conducting Hardware-in-the-loop simulations. Additionally, the stability analysis of the qLMPC algorithm can be conducted by leveraging the extensive literature on stability analysis for LTI and LPV systems.

Acknowledgments

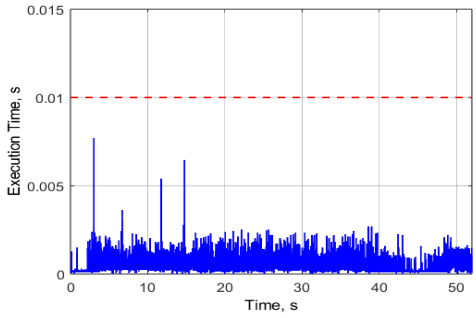
The authors express their gratitude to the ***Institut für Flugzeug-Systemtechnik (FST)*** for providing the high-fidelity ULTRA-Extra model utilized in this research, as well as the waypoints of the two scenarios previously used in an NLGL path following study, which facilitated the comparison between the two path-following algorithms.



(a) Path-Following for the first scenario.

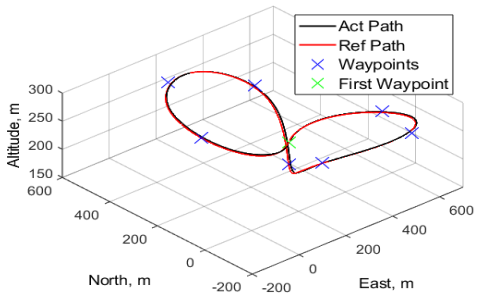


(b) Absolute track error.

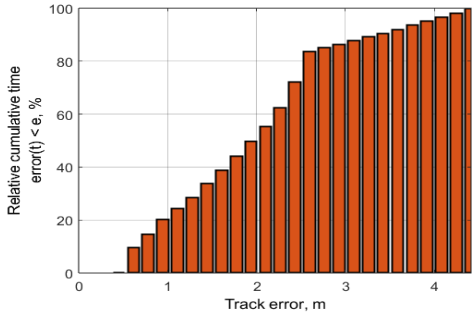


(c) Execution time.

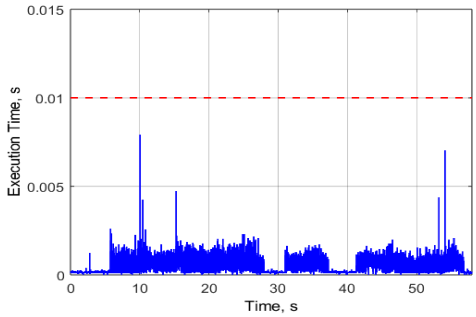
Fig. 8 'Aerodrome Circuit' path-following - Hybrid Approach path planner.



(a) Path-Following for the second scenario.

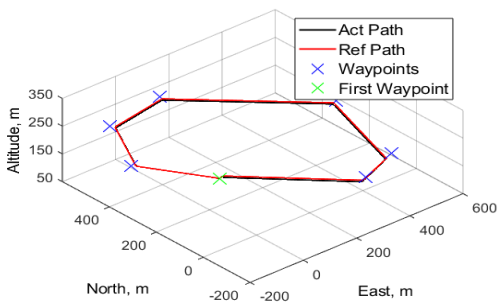


(b) Absolute track error.

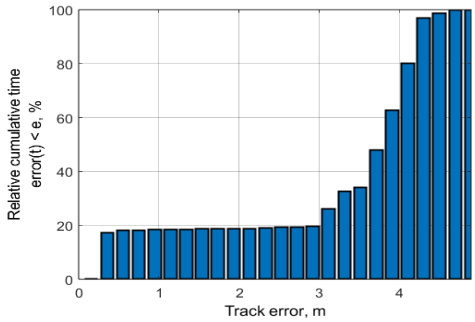


(c) Execution time.

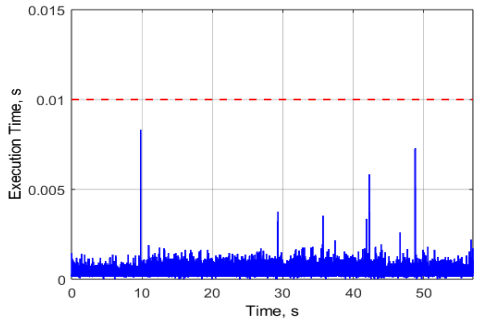
Fig. 9 'Roller-coaster' like path-following - Hybrid approach path planner.



(a) Path-Following for the first scenario.

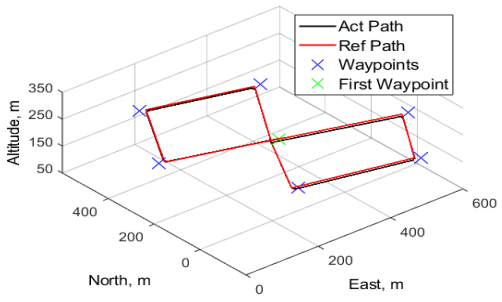


(b) Absolute track error.

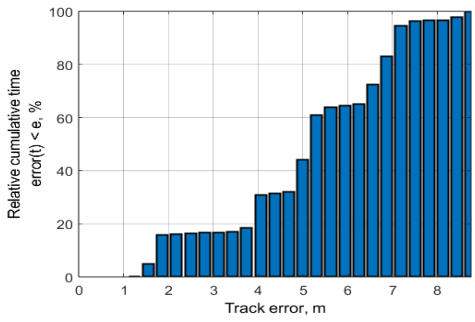


(c) Execution time.

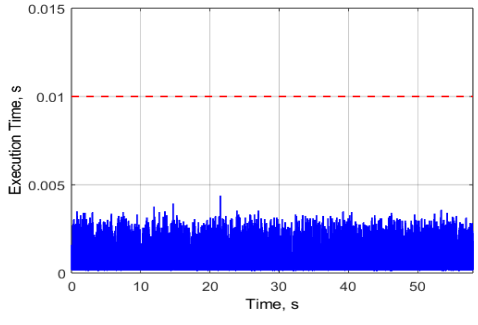
Fig. 10 'Aerodrome Circuit' path-following - Synthetic waypoint path planner.



(a) Path-Following for the second scenario.



(b) Absolute track error.



(c) Execution time.

Fig. 11 'Roller-coaster' like path-following - Synthetic waypoint path planner.

References

[1] Krings, M., Annighöfer, B., and Thielecke, F., "ULTRA - Unmanned Low-cost Testing Research Aircraft," *American Control Conference*, 2013, pp. 1472–1477.

[2] Rysdyk, R., "Unmanned Aerial Vehicle Path Following for Target Observation in Wind," *Journal of Guidance, Control, and Dynamics*, Vol. 29, No. 5, 2006, pp. 1092–1100. https://doi.org/10.2514/1.19101, URL https://doi.org/10.2514/1.19101.

[3] Conte, G., Duranti, S., and Merz, T., "Dynamic 3D path following for an autonomous helicopter," *IFAC Proceedings Volumes*, Vol. 37, No. 8, 2004, pp. 472–477. https://doi.org/https://doi.org/10.1016/S1474-6670(17)32021-9, URL https://www.sciencedirect.com/science/article/pii/S1474667017320219, iFAC/EURON Symposium on Intelligent Autonomous Vehicles, Lisbon, Portugal, 5-7 July 2004.

[4] Ambrosino, G., Ariola, M., Ciniglio, U., Corraro, F., Pironti, A., and Virgilio, M. A. D., "Algorithms for 3D UAV Path Generation and Tracking," *Proceedings of the 45th IEEE Conference on Decision and Control*, 2006, pp. 5275–5280.

[5] Gu, N., Wang, D., Peng, Z., Wang, J., and Han, Q.-L., "Advances in Line-of-Sight Guidance for Path Following of Autonomous Marine Vehicles: An Overview," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, Vol. 53, No. 1, 2023, pp. 12–28. https://doi.org/10.1109/TSMC.2022.3162862.

[6] Sun, M., Zhu, R., and Yang, X., "UAV Path Generation, Path Following and Gimbal Control," *2008 IEEE International Conference on Networking, Sensing and Control*, 2008, pp. 870–873. https://doi.org/10.1109/ICNSC.2008.4525338.

[7] Ratnoo, A., Sujit, P., and Kothari, M., "Adaptive Optimal Path Following for High Wind Flights," *IFAC Proceedings Volumes*, Vol. 44, No. 1, 2011, pp. 12985–12990. https://doi.org/https://doi.org/10.3182/20110828-6-IT-1002.03720, URL https://www.sciencedirect.com/science/article/pii/S147466701645706X, 18th IFAC World Congress.

[8] Nelson, D. R., Barber, D. B., McLain, T. W., and Beard, R. W., "Vector Field Path Following for Miniature Air Vehicles," *IEEE Transactions on Robotics*, Vol. 23, No. 3, 2007, pp. 519–529. https://doi.org/10.1109/TRO.2007.898976.

[9] Li, Z., Sun, J., and Oh, S., "Handling roll constraints for path following of marine surface vessels using coordinated rudder and propulsion control," *Proceedings of the 2010 American Control Conference*, 2010, pp. 6010–6015. https://doi.org/10.1109/ACC.2010.5531275.

[10] Encarnacao, P., and Pascoal, A., "Combined trajectory tracking and path following: an application to the coordinated control of autonomous marine craft," *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No.01CH37228)*, Vol. 1, 2001, pp. 964–969 vol.1. https://doi.org/10.1109/CDC.2001.980234.

[11] Aguiar, A. P., Kaminer, I., Ghabcheloo, R., Pascoal, A., Xargay, E., Hovakimyan, N., Cao, C., and Dobrokhodov, V., *Time-Coordinated Path Following of Multiple UAVs over Time-Varying Networks using L1 Adaptation*, American Institute of Aeronautics and Astronautics Inc. (AIAA), 2008. https://doi.org/10.2514/6.2008-7131, URL https://arc.aiaa.org/doi/abs/10.2514/6.2008-7131.

[12] Beard, R. W., Ferrin, J., and Humpherys, J., "Fixed Wing UAV Path Following in Wind With Input Constraints," *IEEE Transactions on Control Systems Technology*, Vol. 22, No. 6, 2014, pp. 2103–2117. https://doi.org/10.1109/TCST.2014.2303787.

[13] Sujit, P., Saripalli, S., and Sousa, J. B., "Unmanned Aerial Vehicle Path Following: A Survey and Analysis of Algorithms for Fixed-Wing Unmanned Aerial Vehicless," *IEEE Control Systems Magazine*, Vol. 34, No. 1, 2014, pp. 42–59. https://doi.org/10.1109/MCS.2013.2287568.

[14] Rubí, B., Pérez, R., and Morcego, B., "A Survey of Path Following Control Strategies for UAVs Focused on Quadrotors," *Journal of Intelligent & Robotic Systems*, Vol. 98, No. 2, 2020, pp. 241–265.

[15] Verschueren, R., Frison, G., Kouzoupis, D., Frey, J., van Duijkeren, N., Zanelli, A., Novoselnik, B., Albin, T., Quirynen, R., and Diehl, M., "acados: a modular open-source framework for fast embedded optimal control," , 2020.

[16] Englert, T., Völz, A., Mesmer, F., Rhein, S., and Graichen, K., "A software framework for embedded nonlinear model predictive control using a gradient-based augmented Lagrangian approach (GRAMPC)," *Optimization and Engineering*, 2018, pp. 1–41.

[17] Sopasakis, P., Fresk, E., and Patrinos, P., "OpEn: Code Generation for Embedded Nonconvex Optimization," , 2020.

[18] Diehl, M., Bock, H. G., and Schlöder, J. P., "A Real-Time Iteration Scheme for Nonlinear Optimization in Optimal Feedback Control," *SIAM Journal on Control and Optimization*, Vol. 43, No. 5, 2005, pp. 1714–1736. https://doi.org/10.1137/S0363012902400713, URL https://doi.org/10.1137/S0363012902400713.

[19] Rucco, A., Aguiar, A. P., Pereira, F. L., and de Sousa, J. B., "A Predictive Path-Following Approach for Fixed-Wing Unmanned Aerial Vehicles in Presence of Wind Disturbances," *Robot 2015: Second Iberian Robotics Conference*, edited by L. P. Reis, A. P. Moreira, P. U. Lima, L. Montano, and V. Muñoz-Martinez, Springer International Publishing, Cham, 2016, pp. 623–634.

[20] Hauser, J., "A PROJECTION OPERATOR APPROACH TO THE OPTIMIZATION OF TRAJECTORY FUNCTIONALS," *IFAC Proceedings Volumes*, Vol. 35, No. 1, 2002, pp. 377–382. https://doi.org/https://doi.org/10.3182/20020721-6-ES-1901.00312, URL https://www.sciencedirect.com/science/article/pii/S1474667015387334, 15th IFAC World Congress.

[21] Cisneros, P. S. G., Voss, S., and Werner, H., "Efficient Nonlinear Model Predictive Control via quasi-LPV representation," *2016 IEEE 55th Conference on Decision and Control (CDC)*, IEEE, 2016. https://doi.org/10.1109/cdc.2016.7798752.

[22] The MathWorks, Inc., *MATLAB Optimization Toolbox*, 2023. URL https://www.mathworks.com/products/optimization.html, version R021b.

[23] Stellato, B., Banjac, G., Goulart, P., Bemporad, A., and Boyd, S., "OSQP: An Operator Splitting Solver for Quadratic Programs," *Mathematical Programming Computation*, Vol. 12, No. 4, 2020, pp. 637–672. https://doi.org/10.1007/s12532-020-00179-2.

[24] Cisneros, P. S. G., and Werner, H., "A Velocity Algorithm for Nonlinear Model Predictive Control," *IEEE Transactions on Control Systems Technology*, Vol. 29, 2021, pp. 1310–1315.

[25] Calderón, H. M., Cisneros, P. S., and Werner, H., "qLPV Predictive Control - A Benchmark Study on State Space vs Input-Output Approach⁎⁎HMC acknowledges support from CONACYT-Mexico." *IFAC-PapersOnLine*, Vol. 52, No. 28, 2019, pp. 146–151. https://doi.org/https://doi.org/10.1016/j.ifacol.2019.12.362, URL https://www.sciencedirect.com/science/article/pii/S2405896319322621, 3rd IFAC Workshop on Linear Parameter Varying Systems LPVS 2019.

[26] Sedlmair, N., Theis, J., and Thielecke, F., "Design and experimental validation of UAV control laws - 3D spline-path-following and easy-handling remote control," *Proceedings of the 5th CEAS Conf. Guid., Navigation, Control*, 2019.

[27] Beard, R. W., *Small unmanned aircraft*, Princeton University Press, 2012.

[28] Hastedt, P., "Load Factor Control of a Scaled Flight Test Vehicle using Nonlinear Dynamic Inversion," , jan 2022. https://doi.org/10.2514/6.2022-0283.vid.

[29] González Cisneros, P. S., and Werner, H., "Nonlinear model predictive control for models in quasi-linear parameter varying form," *International Journal of Robust and Nonlinear Control*, Vol. 30, No. 10, 2020, pp. 3945–3959. https://doi.org/https://doi.org/10.1002/rnc.4973, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rnc.4973.

[30] Leith, D. J., and Leithead, W. E., "Gain-scheduled and nonlinear systems : dynamic analysis by velocity-based linearization families," *International Journal of Control*, Vol. 70, 1998, pp. 289–317.

[31] Dubins, L. E., "On Curves of Minimal Length with a Constraint on Average Curvature, and with Prescribed Initial and Terminal Positions and Tangents," *American Journal of Mathematics*, Vol. 79, No. 3, 1957, pp. 497–516. URL http://www.jstor.org/stable/2372560.

[32] Wang, H., Kearney, J., and Atkinson, K., "Arc-length parameterized spline curves for real-time simulation," *Proc. 5th International Conference on Curves and Surfaces*, Vol. 387396, 2002.

[33] Faulwasser, T., Kern, B., and Findeisen, R., "Model predictive path-following for constrained nonlinear systems," *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, IEEE, 2009. https://doi.org/10.1109/cdc.2009.5399744.

[34] Medagoda, E. D. B., and Gibbens, P. W., "Synthetic-Waypoint Guidance Algorithm for Following a Desired Flight Trajectory," *Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 2, 2010, pp. 601–606. https://doi.org/10.2514/1.46204, URL https://doi.org/10.2514/1.46204.

[35] Medagoda, E. D., "Closed Loop Stability of Synthetic Waypoint Guidance Algorithm," *IFAC Proceedings Volumes*, Vol. 43, No. 15, 2010, pp. 75–80. https://doi.org/https://doi.org/10.3182/20100906-5-JP-2022.00014, URL https://www.sciencedirect.com/science/article/pii/S147466701531819X, 18th IFAC Symposium on Automatic Control in Aerospace.

[36] Sedlmair, N., Theis, J., and Thielecke, F., "Experimental Comparison of Nonlinear Guidance Laws for Unmanned Aircraft," *IFAC-PapersOnLine*, Vol. 53, No. 2, 2020, pp. 14805–14810. https://doi.org/10.1016/j.ifacol.2020.12.1922.