Chapter 8
Predictive Control for Path-Following. From Trajectory Generation to the Parametrization of the Discrete Tracking Sequences

Ionela Prodan, Sorin Olaru, Fernando A.C.C. Fontes, Fernando Lobo Pereira, João Borges de Sousa, Cristina Stoica Maniu and Silviu-Iulian Niculescu

Abstract This chapter discusses a series of developments on predictive control for path following via a priori generated trajectory for autonomous aerial vehicles. The strategy partitions itself into offline and runtime procedures with the assumed goal of moving the computationally expensive part into the offline phase and of leaving only tracking decisions to the runtime. First, it will be recalled that differential flatness represents a well-suited tool for generating feasible reference trajectory.

I. Prodan (✉)
Laboratory of Conception and Integration of Systems (LCIS EA 3747),
Université Grenoble Alpes, 26902 Valence, France
e-mail: ionela.prodan@lcis.grenoble-inp.fr

S. Olaru · C. Stoica Maniu
Laboratory of Signals and Systems, CentraleSupélec-CNRS-Université Paris-Sud, Université Paris-Saclay, 3 Rue Joliot Curie, 91192 Gif-sur-Yvette, France
e-mail: sorin.olaru@centralesupelec.fr

C. Stoica Maniu
e-mail: cristina.maniu@centralesupelec.fr

S.-I. Niculescu
Laboratory of Signals and Systems (L2S, UMR CNRS 8506),
CentraleSupélec-CNRS-Université Paris-Sud, Université Paris-Saclay, 3 Rue Joliot Curie,
91192 Gif-sur-Yvette, France
e-mail: silviu.niculescu@lss.supelec.fr

F.A.C.C. Fontes · F. Lobo Pereira
SYSTEC—Faculty of Engineering, Porto University and ISR-Porto,
4200-465 Porto, Portugal
e-mail: faf@fe.up.pt

F. Lobo Pereira
e-mail: flp@fe.up.pt

J. Borges de Sousa
LSTS—Faculty of Engineering, Porto University, 4200-465 Porto, Portugal
e-mail: jtasso@fe.up.pt

S. Olaru et al. (eds.), *Developments in Model-Based Optimization and Control*,
Lecture Notes in Control and Information Sciences 464,
DOI 10.1007/978-3-319-26687-9_8

Next, an optimization-based control problem which minimizes the tracking error for the nonholonomic system is formulated and further enhanced via path following mechanisms. Finally, possible changes of the selection of sampling times along the path and their impact on the predictive control formulation will be discussed in detail.

Keywords Model predictive control (MPC) · Differential flatness · Trajectory tracking · Path following · Autonomous aerial vehicles

8.1 Introduction

There are many situations in control and coordination of dynamical systems for which a trajectory has to be generated a priori in order to provide a reference for a tracking control problem, [1, 3, 30]. These control applications are often difficult to handle in embedded solutions (complex dynamics, difficult real-time constraints, short sampling times, limited computational resources, and the like). For all these reasons, it is essential to push as many of the reference management and control tasks offline. Therefore, enforcing the computationally demanding effort of trajectory generation for an offline stage leaves for the runtime only the relatively straightforward trajectory tracking, [14, 20, 23]. Moreover, having a priori feasible reference trajectory implies that we may offer guarantees of performance for the overall system, [22, 26, 30].

Another challenging problem frequently used in control is path following which allows dynamical (nonlinear) systems to follow a predefined path specified by points, lines, and the like. The clear specification of the difference between the following two close notions is of most importance: trajectory tracking and path following. The latter provides a desired time-independent route for the vehicle, while for the trajectory tracking the reference is represented by a function of time. Both problems have their strengths and weaknesses depending on the general control objectives. For example, we may consider that the former has the advantage of providing simultaneously both feasible input and state variables for the corresponding system. However, a disadvantage would be its time dependence, which often imposes an additional constraint on the real-time functioning. This is to be compared with the reference path which remains time-independent and, as such, provides a certain flexibility for tracking.

There is a wealth of work in the literature on trajectory tracking and path following algorithms. From an optimization-based control viewpoint, a widely used technique for solving tracking problems is model predictive control (MPC) (see, for instance, [15, 25] for an overview of MPC, and [12] for MPC of nonholonomic systems) due to its ability to handle control and state constraints, while offering good performance specifications. For example, [17] uses a predictive guidance controller for an autonomous UAV and a fault detection filter for taking into account the disturbances. Mixed-integer programming (MIP) techniques combined with receding horizon strategy were useful for coordinating the efficient interaction of multiple UAVs in scenarios with many sequential tasks and tight timing constraints (see, [16, 27]). Furthermore, some works investigate the capability of nonlinear MPC for tracking control. Among these contributions, [18] formulates a nonlinear MPC algorithm

combined with the gradient descent method for trajectory tracking, and [13] proposes a two-layer control scheme composed by a nonlinear and a linear predictive controller for a group of nonholonomic vehicles moving in formation. Reference [21] proposes as well a gradient-based optimization algorithm for trajectory generation for aircraft avoidance maneuvers where concepts like polar sets and gauge function are used to partition the feasible region in convex partitions. The combination of MPC and path following has been previously addressed in [4, 11], and [35].

The combination of MPC with flatness represents a challenging combination in the current state of the art by allowing real-time control, trajectory generation, and robustness by using set-theoretic methods, [22]. In the present work, we choose to use one of the few generic tools, those based on differential flatness for constructing a reference trajectory. Then, we propose a trade-off between trajectory and path tracking. We pre-compute a feasible trajectory but we use it as a path by considering the velocity along it as the solution of an optimization problem. By allowing this degree of freedom on how fast we move along the path, we actually increase the flexibility and robustness of the problem, while at the same time guaranteeing a feasible path.

The present chapter is motivated mainly by our previous work [22–24], where a flat trajectory was generated and further used as a reference in an output tracking MPC problem. The results were also implemented on real UAVs. This work extends the optimization-based control approach and the path following versus trajectory tracking discussions previously presented. More specifically, the original contributions are the following:

- greater flexibility of trajectory tracking by allowing a variable speed along the path in order to decrease the sensitivity to disturbances and perturbations;
- discretization and the linearization along the reference trajectory are adapted to the variable speed profile by using variable sampling intervals; and
- simulations results over a high-order unmanned aerial vehicle (UAV) model are provided.

The chapter is organized as follows. Section 8.2 introduces the prerequisites needed for trajectory generation: flat trajectory and its parametrization together with proof of concepts examples. Also, the principles of the key background underlying optimization-based control are briefly introduced. While Sect. 8.3 details the control part of the trajectory tracking problem, Sect. 8.4 presents its reconfiguration as a path. Section 8.5 presents illustrative simulation results for an UAV system, and Sect. 8.6 concludes the paper.

Notation

Let $x(k+1)$ denote the value of x at time instant $k+1$, predicted upon the information available at time $k \in \mathbb{N}$. The length of the prediction horizon is denoted by N_p, and the time step within prediction horizon is denoted by s. We write $R \succ 0$ and $R \succeq 0$ to denote that R is a positive definite and semidefinite matrix, respectively. The discretization step is denoted by T_e.

8.2 Prerequisites

This section presents some general details on flat trajectories and the optimization-based control principles.

8.2.1 Flat Trajectory

Consider the nonlinear continuous time-invariant system:

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)), \tag{8.1}$$

where $\mathbf{x}(t) \in \mathbb{R}^{n_x}$ is the state vector and $\mathbf{u}(t) \in \mathbb{R}^{n_u}$ is the input vector.

Definition 8.1 The system (8.1) is called differentially flat if there exists a flat output $\mathbf{z}(\tau) \in \mathbb{R}^{n_u}$ such that the states and inputs can be algebraically expressed in terms of $\mathbf{z}(\tau)$ and a finite number of its higher order derivatives:

$$\mathbf{x}(\tau) = \Phi_0(\mathbf{z}(\tau), \dot{\mathbf{z}}(\tau), \ldots, \mathbf{z}^{(q)}(\tau)), \tag{8.2}$$
$$\mathbf{u}(\tau) = \Phi_1(\mathbf{z}(\tau), \dot{\mathbf{z}}(\tau), \ldots, \mathbf{z}^{(q)}(\tau)), \tag{8.3}$$

where $\mathbf{z}(\tau) = \gamma(\mathbf{x}(\tau), \mathbf{u}(\tau), \dot{\mathbf{u}}(\tau), \cdots, \mathbf{u}^{(q)}(\tau))$ and $q \in \mathbb{N}$ represents the maximum order of $\mathbf{z}(\tau)$ arising in the problem (see also Fig. 8.1 for a general view on differential flatness concept). □

Remark 8.1 Note that, in (8.2)–(8.3), $\tau \in \mathbb{R}$ is a scalar parameter which can be assimilated to $t \in \mathbb{R}$ in (8.1), but will be used as a decision variable interpreted as a virtual time when the reference tracking problem is casted into path following problem. □

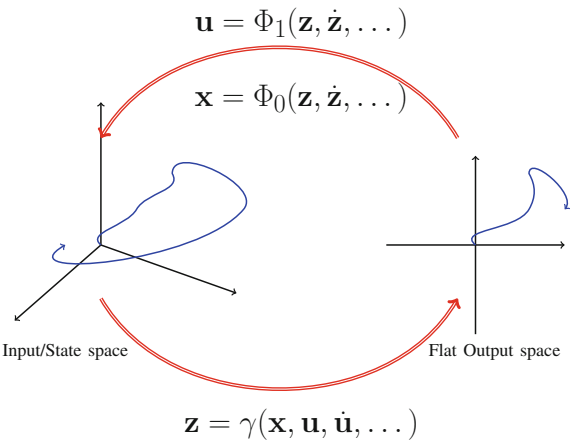

Fig. 8.1 Differentially flat systems

Remark 8.2 For any system admitting a flat description, the number of flat outputs equals the number of inputs [19]. In the case of linear systems [29], the flat differentiability (existence and constructive forms) is implied by the controllability property. □

The most important aspect of flatness is that it reduces the problem of trajectory generation to finding an adequate flat output solving an algebraic system of equalities and inequalities. This means choosing $\mathbf{z}(t)$ such that, via mappings $\Phi_0(\cdot)$, $\Phi_1(\cdot)$, various constraints on state and inputs are verified. The flat output may be itself difficult to compute. The usual solution (also followed here) is to parameterize $\mathbf{z}(t)$ by using a set of smooth basis functions $\Lambda^i(t)$:

$$\mathbf{z}(t) = \sum_{i=1}^{N} c_i \Lambda^i(t), \quad c_i \in \mathbb{R}. \tag{8.4}$$

The number of basis functions directly depends on the number of constraints imposed onto the dynamics [33].

There are multiple choices for the basis functions $\Lambda^i(t)$ in (8.4). The polynomial basis $\lambda^i(t) = t^i$ is a well-known choice but suffers from numerical deficiencies: the number of functions depends on the trajectory constraints and on the degree of the derivatives appearing in the state and input parametrizations, [6, 10, 31]. Another choice is the *Bésier basis functions* [28], they mitigate the numerical difficulties but their degree still depends on the order of derivatives that appear, [9, 34]. *B-spline* basis functions represent an alternative well suited to flatness parametrization due to their ease of enforcing continuity. Moreover, their degree depends only up to which derivative is needed to ensure continuity. This basis will be used in the simulation results presented in this chapter. For details on B-spline functions and their applications, the interested reader is referred to recent research works, [7, 31, 32].

8.2.2 Principles of Optimization-Based Control

The optimization-based control refers to the control design that optimizes a given criterion by using methods that generate optimal control laws whose parameters are such that a certain desired property, such as stability or robustness, is fulfilled. This is a broad definition which actually can cover the classical optimal control, the LMI-based techniques, MPC, or interpolation-based techniques. We provide in this chapter, a trajectory tracking MPC problem for which a control action $\mathbf{u}(k)$ for a given state $\mathbf{x}(k)$ is obtained from the control sequence $\mathfrak{u} \triangleq \{\mathbf{u}(k), \mathbf{u}(k+1), \ldots, \mathbf{u}(k+N_p-1)\}$ as the result of the optimization problem:

$$u^* = \arg\min_{\mathfrak{u}} \left\{ V_f(\mathbf{x}(k+N_p), \mathrm{r}(k+N_p)) + \sum_{s=1}^{N_p-1} V(\mathbf{x}(k+s), \mathbf{u}(k+s), \mathrm{r}(k+s)) \right\}, \tag{8.5}$$

subject to:

$$\begin{cases} \mathbf{x}(k+s+1) = f(\mathbf{x}(k+s), \mathbf{u}(k+s)), & s = 0 : N_p - 1, \\ h(\mathbf{x}(k+s), \mathbf{u}(k+s), \mathrm{r}(k+s)) \leq 0, & s = 1 : N_p - 1, \end{cases} \tag{8.6}$$

over a finite horizon N_p. Here, $V_f(\cdot,\cdot) : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \to \mathbb{R}_+$ represents the terminal cost function, $V(\cdot,\cdot,\cdot) : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_x} \to \mathbb{R}$, the cost at stage s function, $f(\cdot,\cdot)$ describes the evolution of the systems' trajectory, $h(\cdot,\cdot,\cdot)$ the constraints in a general (input state parameters) form, being $\mathrm{r}(\cdot) \in \mathbb{R}^{n_x}$ a vector of time-varying parameters which includes the reference trajectory along the prediction horizon.

Therefore, our main objective being the design of a predictive control strategy, a reference trajectory needs to be available beforehand at least for a finite prediction window. Hereinafter, we use flatness concepts previously presented in order to provide flat states and inputs of the nonlinear finite-time optimization problem (8.5) at the pre-design stage (trajectory generation), which can be updated in real-time in order to allow rescheduling and target moves. Finally, the MPC optimization problem and the a priori generated reference are linked through an optimization block which adapts the speed of tracking, being this actually the main contribution of the paper and thus it will be our main focus in the forthcoming section.

Remark 8.3 Note that, the nonlinear systems used in a wide class of practical applications are differentially flat. For the cases where the specifications of flat outputs are not possible, other strategies for trajectory generation can be employed (see, for example, [1, 5]). □

8.3 Optimization-Based Trajectory Tracking

This section starts by presenting the linearization procedure of the system model which will be further used in the optimization-based control design for trajectory tracking.

8.3.1 Linearization

Hereinafter, we choose to discretize and linearize the dynamics (8.1) along the flat trajectory (for the construction details regarding the discretization method, linearization points along the flat trajectory and the like, the reader can consult our previous work, [22]). For the time discretization via Euler explicit method, we compute a first-order approximation of the state of the system at a time later. The one at the current time is as follows:

$$\mathbf{x}(k+1) = \mathbf{x}(k) + T_e \cdot f(\mathbf{x}(t), \mathbf{u}(t))|_{t=k\cdot T_e}, \tag{8.7}$$

where T_e is the discretization step. Furthermore, for the discretized model of the nonlinear system (8.1) defined as:

$$\mathbf{x}(k+1) = f^d(\mathbf{x}(k), \mathbf{u}(k)), \tag{8.8}$$

we consider a collection of points along the reference trajectory in which we precompute linear approximations of (8.8):

$$\mathbb{L} \triangleq \{l^j = (\mathbf{x}^j, \mathbf{u}^j), \quad j = 0 : N_l\}, \tag{8.9}$$

with N_l the number of chosen linearization points. For a given point $l^j \in \mathbb{L}$ we consider the following Taylor decomposition:

$$f^d(\mathbf{x}(k), \mathbf{u}(k)) = f^d(\mathbf{x}^j, \mathbf{u}^j) + \mathbf{A}_j(\mathbf{x}(k) - \mathbf{x}^j) + \mathbf{B}_j(\mathbf{u}(k) - \mathbf{u}^j) + \beta_j(\mathbf{x}(k), \mathbf{u}(k)), \tag{8.10}$$

where the matrices $\mathbf{A}_j \in \mathbb{R}^{n_x \times n_x}$ and $\mathbf{B}_j \in \mathbb{R}^{n_x \times n_u}$ are defined by

$$\mathbf{A}_j = \frac{\partial f^d}{\partial \mathbf{x}}|_{(\mathbf{x}^j, \mathbf{u}^j)}, \qquad \mathbf{B}_j = \frac{\partial f^d}{\partial \mathbf{u}}|_{(\mathbf{x}^j, \mathbf{u}^j)} \tag{8.11}$$

and $\beta_j(\mathbf{x}(k), \mathbf{u}(k)) \in \mathbb{R}^{n_x}$ represents the terms of the Taylor decomposition of rank greater than 1 (i.e., the nonlinear residue of the linearization):

$$\beta_j(\mathbf{x}(k), \mathbf{u}(k)) = f^d(\mathbf{x}(k), \mathbf{u}(k)) - f^d(\mathbf{x}^j, \mathbf{u}^j) - \mathbf{A}_j(\mathbf{x}(k) - \mathbf{x}^j) - \mathbf{B}_j(\mathbf{u}(k) - \mathbf{u}^j), \tag{8.12}$$

for all $j = 0, \ldots, N_l$. Therefore, the system (8.8) is linearized in $l^j \in \mathbb{L}$ as follows:

$$\mathbf{x}(k+1) = f_j^d(\mathbf{x}(k), \mathbf{u}(k)) \triangleq \mathbf{A}_j\mathbf{x}(k) + \mathbf{B}_j\mathbf{u}(k) + d_j, \tag{8.13}$$

where $f_j^d(\mathbf{x}(k), \mathbf{u}(k)) = f^d(\mathbf{x}(k), \mathbf{u}(k)) - \beta_j(\mathbf{x}(k), \mathbf{u}(k))$, and the affine constant terms $d_j \in \mathbb{R}^{n_x}$ are given by

$$d_j = f^d(\mathbf{x}^j, \mathbf{u}^j) - \mathbf{A}_j\mathbf{x}^j - \mathbf{B}_j\mathbf{u}^j, \tag{8.14}$$

for all $j = 0, \ldots, N_l$.

Remark 8.4 Note that, in general, the linearization and the discretization operations can be interchanged and even mixed in order to obtain a discrete-time linear system which best approximates the sampling of the continuum time dynamics. □

Furthermore, for selecting between the predefined linearization points (8.9) for the current input/state values, we partition the state space into a collection of *Voronoi cells*:

$$\mathcal{V}_j = \left\{(\mathbf{x}, \mathbf{u}) : ||(\mathbf{x}, \mathbf{u}) - (\mathbf{x}^j, \mathbf{u}^j)|| \leq ||(\mathbf{x}, \mathbf{u}) - (\mathbf{x}^i, \mathbf{u}^i)||, \ \forall i \neq j\right\}, \tag{8.15}$$

where each cell consists of all points whose linearization error around point $(\mathbf{x}^j, \mathbf{u}^j)$ is lower than the one with respect to any other point $(\mathbf{x}^i, \mathbf{u}^i)$ from $\mathbb{L}$, with $i, j = 0, \ldots, N_l$. This allows a practical criterion for the selection of the linearization point during runtime:

$$\text{if} \quad (\mathbf{x}(k), \mathbf{u}(k)) \in \mathcal{V}_j \quad \text{then} \quad \mathbf{x}(k+1) = f_j^d(\mathbf{x}(k), \mathbf{u}(k)), \quad \forall\, j = 0, \ldots, N_l. \tag{8.16}$$

It is worth mentioning that, for a given error norm, the Voronoi decomposition is unique (by its geometrical properties) and, as such, it offers a generic design tool for any localization of the linearization points. The drawback is that this criterion is purely geometric and does not take into account the dynamical properties of the model. This disadvantage can be mitigated by two practical procedures: increasing the number of linearization points, and computing the maximal linearization error (see [8] for a discussion on the accuracy of the linearization and its impact in the design of stabilizing control laws). Since $\beta_j(\mathbf{x}, \mathbf{u}) = f^d(\mathbf{x}, \mathbf{u}) - f_j^d(\mathbf{x}, \mathbf{u})$, it follows that the linearization error is related to the topology of its corresponding cell, $\mathcal{V}_j$: For all $k = 0, 1, \ldots$, we have

$$||\beta_j(\mathbf{x}(k), \mathbf{u}(k))|| \leq \max_{(\mathbf{x}, \mathbf{u}) \in \mathcal{V}_j} ||f^d(\mathbf{x}, \mathbf{u}) - f_j^d(\mathbf{x}, \mathbf{u})||. \tag{8.17}$$

Basically, a Voronoi decomposition with decreasing volume of the cells leads to an increasing quality of the PWA approximation for the function (8.8).

The following remarks are in order.

Remark 8.5 An a priori computation of the linearization (8.11), (8.12), and (8.14) in all feasible combinations of inputs and states is difficult to handle. As such, we prefer to select the linearization points (8.9) along the flat trajectory under the assumption (to be verified along the system evolution) that the real trajectory will stay in the corresponding validity domain (Voronoi cell), and thus the chosen linearization points will remain relevant to the problem at hand. □

Remark 8.6 Here, we have chosen the linearization points equidistantly along the reference trajectory. This choice is acceptable as long as the trajectory tracking error is contained by similar uncertainty bounds over the associated Voronoi cells. Adaptive curve sampling can be employed via a different parametrization scheme in order to select these points. For example, the selection of linearization points can be seen as an optimization problem where the goal is to position the points in such a way as to minimize the linearization errors $\beta_j(\mathbf{x}(k), \mathbf{u}(k))$ in (8.10). Such an approach becomes relevant when the control problem is specified in a high-dimensional space and an automatic implementation of the scheme is required. □

Remark 8.7 Note that, a detailed exposition on the procedure for selecting between the linearization points can be found in [22]. □

Next, an optimization problem is formulated in a predictive control framework. It includes the minimization of the system tracking error since the nominal trajectory is generated by taking into account the state and input constraints, but the real vehicle state may not follow exactly the reference flat trajectory. Hence, the system will be controlled in real time to remain adequately close to the reference trajectory over a finite-time horizon in the presence of constraints by using the available information.

8.3.2 Real-Time Control

We consider the recursive construction of an optimal open-loop control sequence $\mathrm{u} = \{\mathbf{u}(k), \mathbf{u}(k+1), \ldots, \mathbf{u}(k+N_p-1)\}$ over a *finite* constrained receding horizon, which leads to a feedback control policy by the effective application of the first control action as system input (see also Fig. 8.2 which illustrates the general trajectory tracking mechanism):

$$u^* = \arg\min_{\mathrm{u}} \sum_{s=0}^{N_p-1} \left(||\mathbf{x}(k+s) - \mathbf{x}^{ref}(k+s)||_{\mathbf{Q}} + ||\mathbf{u}(k+s) - \mathbf{u}^{ref}(k+s)||_{\mathbf{R}} \right), \tag{8.18}$$

subject to the set of constraints:

$$\begin{cases} \mathbf{x}(k+s+1) = \mathbf{A}_j \mathbf{x}(k+s) + \mathbf{B}_j \mathbf{u}(k+s) + r_j, \ s = 0 : N_p - 1, \\ \mathbf{x}(k+s) \in \mathcal{X}, \ s = 1 : N_p - 1, \\ \mathbf{u}(k+s) \in \mathcal{U}, \ s = 1 : N_p - 1, \\ \mathbf{x}(k) = \mathbf{x}_p(k), \end{cases} \tag{8.19}$$

for some $j \in \{1 : N_l\}$. Here, $\mathbf{x}_p(k)$ denotes the state of the plant measured at instant k, the matrices $\mathbf{Q} = \mathbf{Q}^T \succeq 0$, $\mathbf{R} \succ 0$ are weighting matrices and N_p denotes the length of the prediction horizon.

The solution of the optimization problem (8.18)–(8.19) needs to satisfy the dynamical constraints, expressed by the equality constraints in (8.19). At the same time, other security or performance specifications can be added to the system trajectory. These physical limitations (velocity and bank control inputs) are stated in terms of pointwise hard constraints on both the state variables, and input control action as

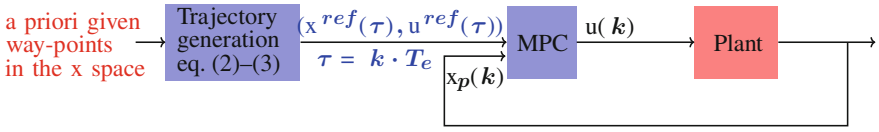

Fig. 8.2 Trajectory tracking mechanism using MPC

detailed by the set inclusion constraints in (8.19). Practically, the closed and compact sets $\mathcal{X}, \mathcal{U}$ denote in a compact formulation the magnitude constraints on states and inputs, respectively. In the following, all these sets are supposed to be polytopic (and, thus, bounded) and to contain the reference control process. This means that $\mathbf{x}^{ref}(k) \in \mathcal{X}$ and $\mathbf{u}^{ref}(k) \in \mathcal{U}$.

Stability constraints can be considered by adding to problem (8.18)–(8.19) a terminal set-terminal cost arguments, the particularity being the time-varying nature of the prediction model.

8.4 Optimization-Based Path Following

This section recalls first the difference between path and trajectory tracking. The former only provides a desired route which may or may not be feasible when taking into account the dynamics of the vehicle or the input and state constraints. At each point in time, the latter provides a pair of reference state and input and thus guarantees the feasibility of the problem. While superior to deal with real-time requirements, the trajectory tracking problem is more challenging. Besides the increased difficulty in generating a trajectory rather than a path there is also the issue of time dependence, which may imply the infeasibility of the real-time optimization problem (8.18)–(8.19).

The synchronization of the absolute time t in (8.1) with the virtual time τ in (8.2)–(8.3), which parametrizes the flat trajectory, forces a constant "velocity" in the sense that the state has to track the current values of the reference. If for some reason (disturbances, unforeseen obstacles), the vehicle "lags", the tracking controller has forced the vehicle to remain in a reachable domain around the current values of the reference trajectory, otherwise the closed-loop control design is compromised. Conversely, if the vehicle is slightly ahead the reference trajectory, it may perform "complex" maneuvers (with an inefficient energy use) when trying to be "in sync" with the reference. This highlights the need for a mechanism which the vehicle can "decelerate" or "accelerate" as desired along the reference trajectory, by adjusting the virtual time flow with respect to the real controller time. Specifically, if for some reason the vehicle remains constantly behind it is not reasonable to follow the trajectory with a constant or an increasing tracking error which can lead to infeasibility in predictive control terms. The alternative will be to reconfigure the trajectory in terms of a path to be followed.

In this section, we propose a formal relaxation of the trajectory tracking scheme. This is done by choosing the optimal point (in the sense specified above) at each sampling time on the path to be followed. Subsequently, further enhancement of the trajectory tracking scheme is provided by varying the speed profile along the path, thus increasing the flexibility and robustness of the problem while guaranteeing the path feasibility.

8.4.1 Selection of the Initial Point of the Reference Trajectory

The basic idea is to find the point and the associated time value τ_c on the predefined trajectory, which is the closest one to the current position of the vehicle $\mathbf{x}(k)$:

$$\tau_c = \arg \min_{\tau \in [\tau_{k-1}, t_f]} \quad ||\mathbf{x}^{ref}(\tau) - \mathbf{x}(k)||_2^2. \tag{8.20}$$

Note that it is assumed implicitly in (8.20) that $\tau_k \geq \tau_{k-1}$ which means that the only possibility for the vehicle is to advance forward along the path (rather than go backward).

In order to better explain the proposed strategy, Fig. 8.3 illustrates a reference trajectory (depicted in blue), which passes through some a priori given waypoints (denoted by the green dots), and the optimal time obtained by solving (8.20), for which the current position is the closest one from the reference trajectory. If the real-time position and the one in virtual time are synchronized, then the vehicle is forced to track a point which is behind, whereas the optimal choice is to reorient the vehicle to track the reference using the optimal time τ_c obtained as in (8.20). Note that the same figure illustrates where to look for the optimal time by using solid blue and dashed blue to denote the past and the future, respectively, along the trajectory.

Once the optimal time along the reference is found, an optimization-based control problem similar to (8.18) is implemented (see also a similar mechanism illustrated in Fig. 8.2):

$$u^* = \arg \min_{\mathrm{u}} \sum_{s=0}^{N_p-1} \left(||\mathbf{x}(k+s) - \mathbf{x}^{ref}(\tau_c(k) + s \cdot T_e)||_{\mathbf{Q}} + ||\mathbf{u}(k+s) - \mathbf{u}^{ref}(\tau_c(k) + s \cdot T_e)||_{\mathbf{R}} \right), \tag{8.21}$$

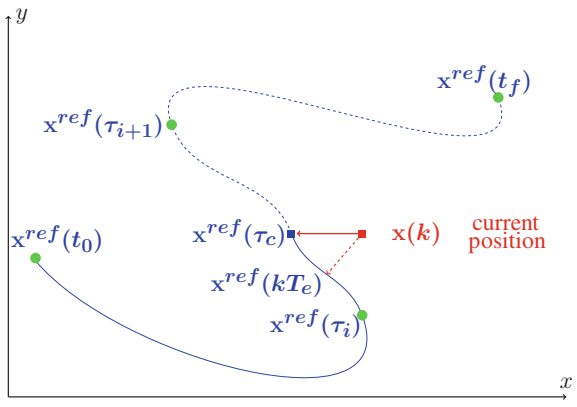

Fig. 8.3 Optimal time for which the current position is the closest from the reference trajectory

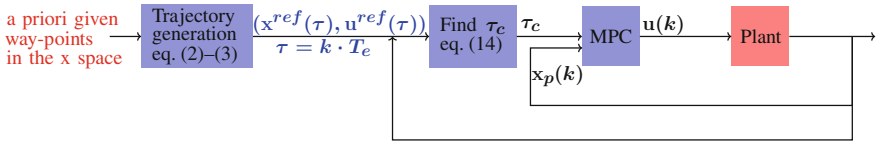

Fig. 8.4 Conversion to path following mechanism using MPC

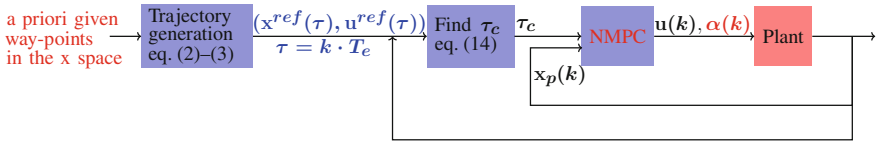

Fig. 8.5 Conversion to path following mechanism using NMPC

subject to the set of constraints (8.19), with T_e the sampling time as in (8.7) and τ_c the optimal time obtained as the result of the optimization (8.20) (Fig. 8.4).

Note that the optimization problem (8.21) retains its linear structure, while (8.20) is the simplest nonlinear optimization problem that can be formulated since it involves only one decision variable τ subject to the constraints in (8.2)–(8.3).

8.4.2 *Selection of the Speed Profile Along the Path*

In the following, we go further in the design of the trajectory tracking scheme by introducing a scalar term α to adjust the speed along the trajectory. This is equivalent to a decorrelation of the flow along the virtual time by linear acceleration ($\alpha > 1$) or deceleration ($\alpha < 1$). Therefore, the optimization problem (8.21) is reformulated as follows

$$(u^*, \alpha^*) = \arg\min_{\mathbf{u},\alpha} \sum_{s=0}^{N_p-1} \left(||\mathbf{x}(k+s) - \mathbf{x}^{ref}(\tau_{T_e}(s;k,\alpha(k)))||_{\mathbf{Q}} + ||\mathbf{u}(k+s) - \mathbf{u}^{ref}(\tau_{T_e}(s;k,\alpha(k)))||_{\mathbf{R}} \right), \tag{8.22}$$

subject to the set of constraints (8.19), where $\tau_{T_e}(s;k,\alpha(k)) = \tau_c(k) + s\alpha(k)T_e$, with T_e being the sampling time as in (8.7), τ_c the optimal time obtained as in (8.20), and α the time-varying speed profiling factor. Now, the structure of the optimization problem (8.21) becomes nonlinear, and, thus, more complex, but still simple in the sense that, it involves only one additional dimension, the decision variable α, which permits to adjust the way the path is followed. Figure 8.5 illustrates the path following mechanism using a nonlinear model predictive control (NMPC) strategy.

8.4.3 Linearization with Varying Speed Profile

In Sect. 8.3.1, we introduced a general linearization procedure for the system. Following the previous sections, the goal is to compare the system trajectory with the reference trajectory in a discrete set of time instances when the reference trajectory is followed at varying speed. To do that, we consider here non-equidistant consecutive time instants $\{t_k\}_{k\geq 0}$ such that:

$$t_{k+1} = t_k + \alpha(k)T_e, \quad k \in \mathbb{N}, \quad \alpha(k) \in (0, 1], \tag{8.23}$$

where T_e is the nominal sampling period.

As with (8.7), we consider Euler explicit method for the discretization:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \alpha(k)T_e f(\mathbf{x}(t_k), \mathbf{u}(t_k)), \quad k \in \mathbb{N}. \tag{8.24}$$

Next, for the linearization procedure, the collection of linearization points along the reference trajectory is redefined by:

$$\mathbb{L} \triangleq \{l^j = (\mathbf{x}^j, \mathbf{u}^j, \alpha^j), \quad j = 1, 2, \ldots, N_l\}, \tag{8.25}$$

with the additional component α^j, whose nominal value is chosen to be $\alpha^j = 1$, corresponding to the reference trajectory at the nominal speed. Furthermore, the linearization around a point $l^j \in \mathbb{L}$ yields the following dynamics:

$$f(\mathbf{x}(t_k), \mathbf{u}(t_k)) \simeq f(\mathbf{x}^j, \mathbf{u}^j) + f_x(\mathbf{x}^j, \mathbf{u}^j)(\mathbf{x}(t_k) - \mathbf{x}^j) + f_u(\mathbf{x}^j, \mathbf{u}^j)(\mathbf{u}(t_k) - \mathbf{u}^j) \tag{8.26}$$

and

$$\begin{aligned}\mathbf{x}(t_{k+1}) \simeq\ & \mathbf{x}(t_k) + \alpha^j T_e f_x(\mathbf{x}^j, \mathbf{u}^j)\mathbf{x}(t_k) + \alpha^j T_e f_u(\mathbf{x}^j, \mathbf{u}^j)\mathbf{u}(t_k) \\ & + \alpha(k)T_e(f(\mathbf{x}^j, \mathbf{u}^j) - f_x(\mathbf{x}^j, \mathbf{u}^j)\mathbf{x}^j - f_u(\mathbf{x}^j, \mathbf{u}^j)\mathbf{u}^j),\end{aligned} \tag{8.27}$$

which can be rewritten as:

$$\begin{aligned}\mathbf{x}(t_{k+1}) \simeq\ & (I + \alpha^j T_e f_x(\mathbf{x}^j, \mathbf{u}^j))\mathbf{x}(t_k) + \alpha^j T_e f_u(\mathbf{x}^j, \mathbf{u}^j)\mathbf{u}(t_k) \\ & + \alpha(k)T_e(f(\mathbf{x}^j, \mathbf{u}^j) - f_x(\mathbf{x}^j, \mathbf{u}^j)\mathbf{x}^j - f_u(\mathbf{x}^j, \mathbf{u}^j)\mathbf{u}^j).\end{aligned} \tag{8.28}$$

Note that (8.27) is a bilinear model and a further approximation ($\alpha(k) = \alpha^j$ in the second and third terms) leads to the linear model (8.28).

As with Sect. 8.3.1, we denote $\mathbf{x}(k+1) = f^d(\mathbf{x}(k), \mathbf{u}(k), \alpha(k))$, thus, $\mathbf{x}(t_k)$ and $\mathbf{u}(t_k)$ become $\mathbf{x}(k)$ and $\mathbf{u}(k)$, respectively. The discrete-time linearized model can be written as:

$$\mathbf{x}(k+1) = \mathbf{A}_j\mathbf{x}(k) + \mathbf{B}_j^1\mathbf{u}(k) + \mathbf{B}_j^2\alpha(k), \tag{8.29}$$

where

$$\mathbf{A}_j = I + \alpha^j T_e f_x(\mathbf{x}^j, \mathbf{u}^j), \tag{8.30}$$

$$\mathbf{B}_j^1 = \alpha^j T_e f_u(\mathbf{x}^j, \mathbf{u}^j), \tag{8.31}$$

$$\mathbf{B}_j^2 = T_e(f(\mathbf{x}^j, \mathbf{u}^j) - f_x(\mathbf{x}^j, \mathbf{u}^j)\mathbf{x}^j - f_u(\mathbf{x}^j, \mathbf{u}^j)\mathbf{u}^j). \tag{8.32}$$

8.4.4 Real-Time Control

The optimization problem (8.22) is now reformulated as follows:

$$(u^*, \alpha^*) = \arg\min_{\mathbf{u},\alpha} \sum_{s=0}^{N_p-1} \Big(||\mathbf{x}(k+s) - \mathbf{x}^{ref}(\tau(k+s))||_{\mathbf{Q}} + ||\mathbf{u}(k+s) - \mathbf{u}^{ref}(\tau(k+s))||_{\mathbf{R}} + ||\alpha(k+s) - 1|| \Big), \tag{8.33}$$

subject to the set of constraints:

$$\begin{cases} \mathbf{x}(k+s+1) = \mathbf{A}_j \mathbf{x}(k+s) + \mathbf{B}_j^1 \mathbf{u}(k+s) + \mathbf{B}_j^2 \alpha(k+s), \ s = 0 : N_p - 1, \\ \tau(k+s+1) = \tau(k+s) + \alpha(k) T_e, \ s = 0 : N_p - 1, \\ \mathbf{x}(k+s) \in \mathcal{X}, \ s = 1 : N_p - 1, \\ \mathbf{u}(k+s) \in \mathcal{U}, \ s = 1 : N_p - 1, \\ \mathbf{x}(k) = \mathbf{x}_p(k), \\ \tau(k) = \tau_c(k), \end{cases} \tag{8.34}$$

for the appropriate index $j \in \{1, \ldots, N_l\}$ corresponding to the active Voronoi cell and τ_c the optimal time obtained as in (8.20) and α the speed profile factor.

Remark 8.8 Note that adapting the discretization step via a time-varying coefficient α and a constant prediction horizon N_p leads to a time-varying prediction horizon on the absolute continuous timescale. Indeed, in continuous time the prediction time will be $N_p T_e \sum_{s=0}^{N_p-1} \alpha(s)$. In order to mitigate this phenomenon, the MPC prediction horizon, N_p can be adapted, by choosing

$$N_p(k) = \min\{n \in \mathbb{N} : n T_e \sum_{s=0}^{N_p-1} \alpha(s) \geq N_p\}. \qquad \square$$

Finally, let us wrap-up in Algorithm 1 the mechanism implemented based on the theoretical elements previously presented.

Algorithm 8.1 Path following optimization-based control problem

Input: Specify a collection of waypoints
-construct the flat trajectory as in (8.2)–(8.3), passing through the waypoints;
-choose a collection of linearization points $\mathbb{L}$ (8.19);
-construct the PWA function as in (8.23);
for $k = 1 : k_{max}$ **do**
  -measure the current state of the plant $\mathbf{x}_p(k)$;
  -find the optimal time τ_c by solving (8.20);
  -select the linearization point $l^j \in \mathbb{L}$ and consequently the pair $(\mathbf{A}_j, \mathbf{B}_j^1, \mathbf{B}_j^2)$ as in Remark 8.7;
  -find the optimal control action u^* and the speed profile factor α by solving (8.27);
  -apply to the plant the first value of the control sequence $u^*(k)$ during the time $\alpha^*(k)T_e$;
end

8.5 Simulation Example for an UAV System

In this section, we start with the case of a 2D 3-DOF model (8.35) of an Unmanned Aerial Vehicle (UAV) in which the autopilot forces coordinated turns (zero side-slip) at a fixed altitude:

$$\begin{aligned} \dot{x}(t) &= v_{\mathrm{a}}(t)\cos\Psi(t) + d_x, \\ \dot{y}(t) &= v_{\mathrm{a}}(t)\sin\Psi(t) + d_y, \\ \dot{\Psi}(t) &= \tfrac{g\,\tan\Phi(t)}{v_{\mathrm{a}}(t)}. \end{aligned} \tag{8.35}$$

The state variables are represented by the position $(x(t), y(t))$ and the heading (yaw) angle $\Psi(t) \in [0, 2\pi]$ rad, which we denote as $\mathbf{x}(t) = [x^T(t)y^T(t)\Psi^T(t)]^T$. The input signals are the airspeed $v_{\mathrm{a}}(t)$ and the bank (roll) angle $\Phi(t)$, respectively, denoted as $\mathbf{u}(t) = [v_a^T(t)\Phi^T(t)]^T$. Also, the airspeed and the bank angle are regarded as the autopilot pseudocontrols. Furthermore, we assume a small angle of attack and that the autopilot provides a higher bandwidth regulator for the bank angle, making its dynamics negligible when compared to the heading dynamics. Also, in (8.35), d_x and d_y represent the wind velocity components on the x and y axes. The dynamical model of the vehicle corresponds to a nonholonomic system, which is completely controllable (under the natural assumption that the velocity is different from zero), but it cannot make instantaneous turns in certain directions.

We take as flat output the position components of the state, $\mathbf{z}(\tau) = [z_1(\tau)\, z_2(\tau)]^T = [x(t)y(t)]^T$, which permits to compute the remaining variables:

$$\mathbf{x}^{ref}(\tau) = \left[z_1(\tau)\, z_2(\tau) \arctan\left(\frac{\dot{z}_2(\tau)}{\dot{z}_1(\tau)}\right)\right]^T, \tag{8.36}$$

$$\mathbf{u}^{ref}(\tau) = \left[\sqrt{\dot{z}_1^2(\tau) + \dot{z}_2^2(\tau)}\ \arctan\left(\frac{1}{g}\frac{\ddot{z}_2(\tau)\dot{z}_1(\tau) - \dot{z}_2(\tau)\ddot{z}_1(\tau)}{\sqrt{\dot{z}_1^2(\tau) + \dot{z}_2^2(\tau)}}\right)\right]^T, \tag{8.37}$$

where $\tau \in [t_0, t_f]$ is a scalar parameter which can be assimilated to t in (8.35), but will be used as a decision variable interpreted as a virtual time when the reference tracking problem is recast in a path following problem.

Note that, in the heading component of the state variable appears first-order derivatives of the flat outputs, while in the roll angle input appears the second-order derivatives of the flat outputs. Hence, for obtaining a smooth state and input (i.e., their derivatives are continuous) it follows that the B-spline parametrization has to have at least degree 4. Further, the linearized model was used for the control part of the trajectory tracking problem.

Next, for testing the proposed trajectory tracking method and its reformulation into a path following problem we use an extended model of (8.35) (for low-level control of an UAV [2]) with 12 states in a 6-DOF simulation. More precisely, the 12-state model includes the positions (x [m], y [m], z [m]), the velocities (v_x [m/s], v_y [m/s], v_z [m/s]), the roll, pitch and yaw angles (ϕ [rad], θ [rad], ψ [rad]), and the angular rates (p [rad/s], q [rad/s], and r [rad/s]), all measured along body frame coordinates, X, Y, and Z. The simulations also incorporate perturbations like the wind with an intensity bounded by some reasonable values (e.g., a maximum speed of 8 m/s).

The objective here is to force the UAV to track 6 given waypoints (denoted as red dots in the forthcoming simulations). The following data and tuning parameters were used for the simulations:

- the list of waypoints: {(500, 200, 150), (450, −250, 150), (0, −350, 150), (−350, 0, 150), (−350, 300, 150), (−200, 500, 150), (50, 450, 150), (400, 0, 150)}.
- the sampling time is $T_e = 0.01$ s;
- constraints on the input: the velocity $v_a \in [18,\ 25]$ m/s, the bank angle $\Phi \in [-0.43, 0.43]$ rad and the wind components d_x, d_y with $|| \begin{bmatrix} d_x & d_y \end{bmatrix} |_2 \leq 8$ m/s;
- small variations on the velocity and bank command are admitted: the rate of change of v_a is limited to the maximum acceleration the aircraft can produce, i.e., $0.1 \sim 0.2\,\text{m/s}^2$; the variation of Φ is limited to 0.04 rad/s;
- the weights matrices are: $\mathbf{Q} = [10e1\,0\,0;\ 0\ 10e1\,0;\ 0\,0\,0.1], \mathbf{R} = 10e4 \cdot [10\,0;\ 0\ 1]$;
- the prediction horizon is $N_p = 7$.

First, we add the current position of the UAV to the list of waypoints. Next, by using the theoretical results presented in Sect. 8.2.1 and (8.36)–(8.37), we generate a flat trajectory starting from the current position of the UAV and passing through the given waypoints. Figure 8.6 illustrates the a priori generated flat trajectory and the heading angle, whereas Fig. 8.7 depicts the control input signals and their derivatives. The linearized model is used for the control part of the trajectory tracking problem with the above-mentioned tuning parameters.

Second, by considering the trajectory tracking mechanism detailed in Fig. 8.2 and solving (8.18)–(8.19) with no wind conditions we obtain good tracking performance as illustrated in 3D in Fig. 8.8a. In green dashed line, we represent the UAV actual motion, and, in magenta dashed line, the path projection on the ground.

However, sometimes it may be the case that, under different wind conditions, the UAV may track the trajectory with an increasing tracking error as proved by the

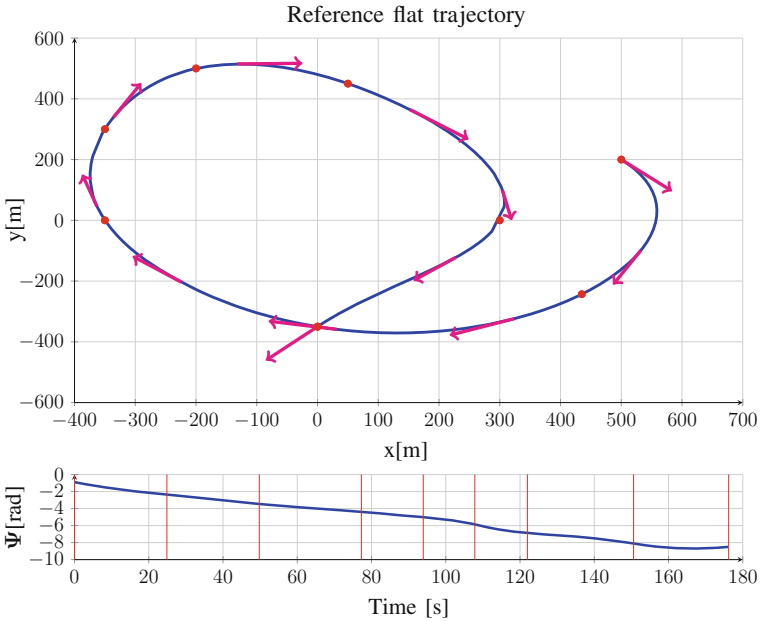

Fig. 8.6 Reference trajectory passing through the waypoints and the corresponding heading angle

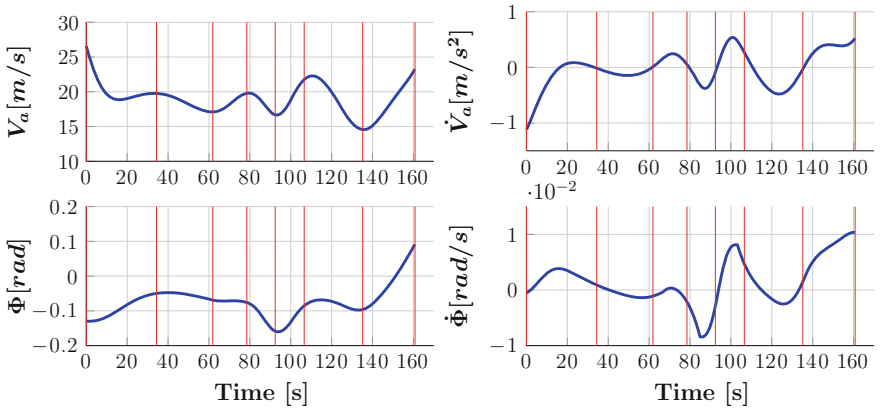

Fig. 8.7 Velocity and roll angle control signals and their derivatives

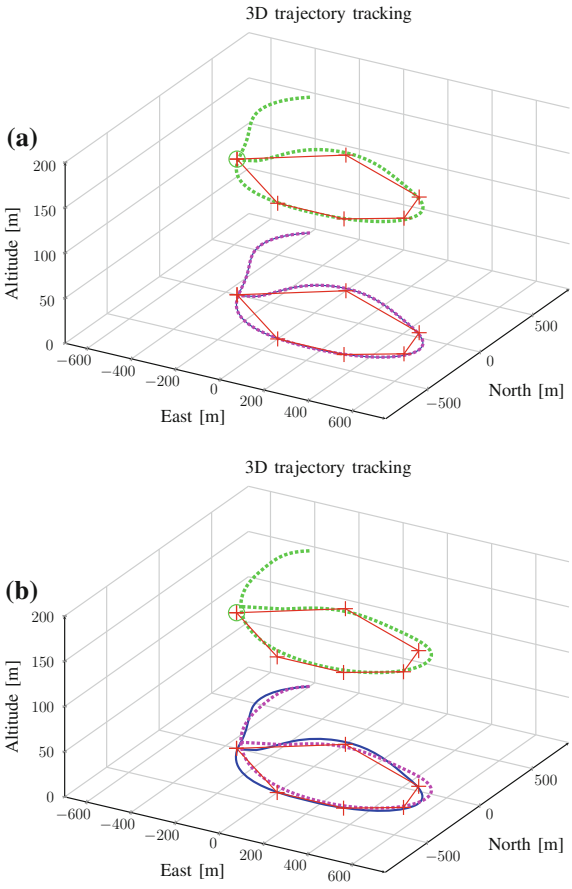

Fig. 8.8 Trajectory tracking: **a** UAV actual motion with no wind conditions. **b** UAV actual motion with wind conditions

simulation scenario with a wind velocity of 5[m/s] from East depicted in Fig. 8.8b in blue continuous line.

To deal with these type of situations, the trajectory tracking problem is reformulated via a path following problem. Equation (8.20) is useful in various scenarios. For instance, we can consider the initial time instant when the real trajectory is far away from the reference. Also, we can consider it during the runtime whenever we need to reinitialize the reference time τ: whenever the real trajectory steers too far away from the reference, we have to recalculate the best time τ_c as addressed in (8.20). The scalar α permits to shrink or expand the sampling period and, in this way, to adjust the speed of the virtual reference vehicle. Note that the use of a speed profile as in (8.22) shows no discernible difference with respect to the simpler method (8.21). Starting from a different initial position $(-30, -800, 150)$, Fig. 8.9 illustrates the performance of the proposed path following mechanism.

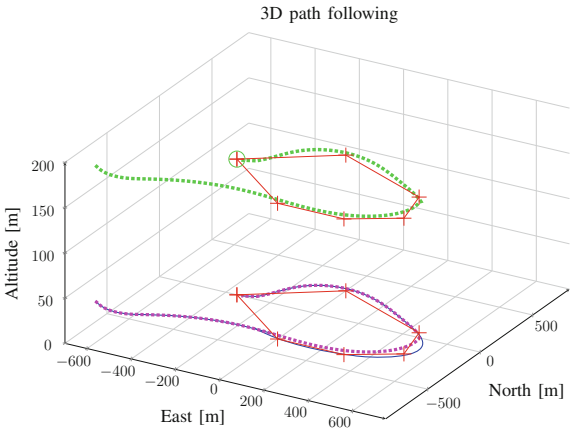

Fig. 8.9 Path following: UAV actual motion with wind conditions

8.6 Conclusions

This chapter presents a predictive control strategy for path following of autonomous aerial vehicles. The strategy is decomposed in a two-stage procedure, where a reference trajectory was pre-specified using differential flatness formalism and than an optimization-based control problem is formulated for minimizing the tracking error for the vehicle. The discussions provided highlight the advantages of reconfiguring the generated feasible trajectory in terms of a path along with the structural properties of the resulting optimization-based control problem. By allowing a variable speed along the path the control problem sensitivity to disturbances and perturbations is decreased. Moreover, the discretization and the linearization along the reference trajectory are adapted to the variable speed profile by using time-varying sampling intervals. Some simulation examples for the control of autonomous aerial vehicles are presented in order to illustrate the proposed approach.

Acknowledgments All the authors acknowledge the support of FCT/ANR PESSOA Project "Advanced control of a fleet of heterogeneous autonomous vehicles." F.A.C.C. Fontes, F. Lobo Pereira, and J. Borges de Sousa acknowledge the support of LSTS - Laboratory of Underwater Systems and Technologies of FEUP. F.A.C.C. Fontes and F. Lobo Pereira acknowledge the support of FCT underlying the funding of the R& D Unit SYSTEC-Research Center for Systems and Technologies.

References

1. A.P. Aguiar, J.P. Hespanha, Trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modeling uncertainty. IEEE Trans. Autom. Control **52**(8), 1362–1379 (2007)
2. R. Bencatel, M. Faied, J. Sousa, A.R. Girard, Formation Control with Collision Avoidance (2011), pp. 591–596

3. M. Burger, K.Y. Pettersen, Smooth transitions between trajectory tracking and path following for single vehicles and formations, in *Estimation and Control of Networked Systems*, (2010), pp. 115–120
4. A.C.D. Caldeira, F.A.C.C. Fontes. Model predictive control for path-following of nonholonomic systems in *Proceedings of the 10th Portuguese Conference in Automatic Control* (Coimbra, Portugal, 2010), pp. 374–379
5. C.L. Castillo, W. Moreno, K.P. Valavanis, Unmanned helicopter waypoint trajectory tracking using model predictive control, in *Proceedings of the IEEE Mediterranean Conference on Control and Automation* (2007), pp. 1–8
6. M. Daniel, J.C. Daubisse, The numerical problem of using bé zier curves and surfaces in the power basis. Comput. Aided Geom. Des. **6**(2), 121–128 (1989)
7. J.A. De Doná, F. Suryawan, M.M. Seron, J. Lévine, A flatness-based iterative method for reference trajectory generation in constrained NMPC, in *Nonlinear Model Predictive Control*, vol. 384, LNCIS, ed. by L. Magni, D.M. Raimondo, F. Allgöwer (Springer, Heidelberg, 2009), pp. 325–333
8. L. Fagiano, M. Canale, M. Milanese, Set membership approximation of discontinuous NMPC laws, pp. 8636–8641
9. G.E. Farin, Curves and Surfaces for Computer-Aided Geometric Design: A Practical Guide. Morgan Kaufmann, 5th edn (2001)
10. R.T. Farouki, V.T. Rajan, On the numerical condition of polynomials in bernstein form. Comput. Aided Geomet. Des. **4**(3), 191–216 (1987)
11. T. Faulwasser, B. Kern, R. Findeisen, Model Predictive Path-following for Constrained Nonlinear Systems, pp. 8642–8647
12. F.A.C.C. Fontes, A general framework to design stabilizing nonlinear model predictive controllers. Syst. Control Lett. **42**(2), 127–143 (2001)
13. F.A.C.C. Fontes, D. Fontes, A. Caldeira, *Model predictive control of vehicle formations* (Optimization and Cooperative Control, Strategies, 2009), pp. 371–384
14. A. Grancharova, E.I. Grøtli, D. Ho, T.A. Johansen, Uavs trajectory planning by distributed MPC under radio communication path loss constraints. J. Intell. Robotic Syst. 1–20 (2014)
15. L. Grüne, J. Pannek, Nonlinear Model Predictive Control (Springer, 2011)
16. J. How, E. King, Y. Kuwata, Flight demonstrations of cooperative control for UAV teams, in *AIAA 3rd Unmanned Unlimited Technical Conference, Workshop and Exhibit* (2004), pp. 20–23
17. T. Keviczky, G.J. Balas, Software-enabled receding horizon control for autonomous unmanned aerial vehicle guidance. J. Guid. Control Dyn. **29**(3), 680–694 (2006)
18. H.J. Kim, D.H. Shim, S. Sastry, Nonlinear model predictive tracking control for rotorcraft-based unmanned aerial vehicles, in *Proceedings of the 2002 American Control Conference*, vol. 5 (IEEE, 2002) pp. 3576–3581
19. J. Lévine, Analysis and control of nonlinear systems: A flatness-based approach (Springer, Hidelberg, 2009)
20. D. Mellinger, N. Michael, V. Kumar, Trajectory generation and control for precise aggressive maneuvers with quadrotors. Int. J. Robotics Res. 027–36 (2012)
21. R.B. Patel, P.J. Goulart, Trajectory generation for aircraft avoidance maneuvers using online optimization. J. Guid. Control Dyn. **34**(1), 218–230 (2011)
22. I. Prodan, R. Bencatel, S. Olaru, J. Sousa, C. Stoica, S.I. Niculescu, Predictive control for autonomous aerial vehicles trajectory tracking, in *Proceedings of the IFAC Nonlinear Model Predictive Control Conference* (Noordwijkerhout, The Netherlands, 23–27 August 2012), pp. 508–513
23. I. Prodan, S. Olaru, R. Bencatel, J. Sousa, C. Stoica, S.I. Niculescu, Receding horizon flight control for trajectory tracking of autonomous aerial vehicles. Control Eng. Pract. **21**(10), 1334–1349 (2013)
24. I. Prodan, S. Olaru, F.A.C.C. Fontes, C. Stoica, S.-I. Niculescu, A predictive control-based algorithm for path following of autonomous aerial vehicles, in *2013 IEEE International Conference on Control Applications (CCA)* (IEEE, 2013), pp. 1042–1047

25. J.B. Rawlings, D.Q. Mayne, Model Predictive Control: Theory and Design (2009)
26. A.P. Schoellig, F.L. Mueller, R. D'Andrea, Optimization-based iterative learning for precise quadrocopter trajectory tracking. Auton. Robots **33**(1–2), 103–127 (2012)
27. T. Schouwenaars, M. Valenti, E. Feron, J. How, Implementation and flight test results of milp-based uav guidance, in *Aerospace Conference* (IEEE, 2005), pp. 1–13
28. L.L. Schumaker, Spline Functions: Basic Theory (Cambridge University Press, 2007)
29. H. Sira-Ramírez, S. Agrawal, *Differential Flatness* (Marcel Dekker, New York, 2004)
30. F. Stoican, I. Prodan, D. Popescu, Flat trajectory generation for way-points relaxations and obstacle avoidance, in *The 23st IEEE Mediterranean Conference on Control and Automation* (2015), pp. 729–734
31. F. Suryawan, Constrained trajectory generation and fault tolerant control based on differential flatness and B-splines. Ph.D thesis, School of Electrical Engineering and Computer Science (The University of Newcastle, Australia, 2012)
32. F. Suryawan, J. De Dona, M. Seron, Methods for trajectory generation in a magnetic-levitation system under constraints, in *18th Mediterranean Conference on Control and Automation* (2010), pp. 945–950
33. J.H. Wilkinson, *The Algebraic Eigenvalue Problem*, vol. 155 (Oxford University Press, 1965)
34. F. Yamaguchi, F. Yamaguchi, *Curves and Surfaces in Computer Aided Geometric Design* (Springer, Berlin, 1988)
35. S. Yu, X. Li, H. Chen, F. Allgower, Nonlinear model predictive control for path following problems. In In Proceedings of the IFAC Nonlinear Model Predictive Control Conference (Noordwijkerhout, The Netherlands, 23–27 August 2012), pp. 508–513