Alma Mater Studiorum
University of Bologna

School of Engineering and Architecture

department of electrical, electronic,
and information engineering
"Guglielmo Marconi" - DEI

**Master of Science in
Automation Engineering**

Thesis in
Automatic Control
and System Theory

Geometric versus Model Predictive Control based guidance algorithms for fixed-wing UAVs in the presence of very strong wind fields.

Author:
Furieri Luca

Lecturer(s):
Prof. Marconi Lorenzo (Univ. of Bologna)
Prof. Siegwart Roland (ETH - Zurich)

Supervisor(s):
Ph.D. Stastny Thomas (ETHZ - Zurich)
PostDoc. Gilitschenski Igor (ETHZ - Zurich)

Academic Year
2015/2016

Session
II

Contents

Abstract

The recent years have witnessed increased development of small, autonomous fixed-wing Unmanned Aerial Vehicles (UAVs).
In order to unlock widespread applicability of these platforms, they need to be capable of operating under a variety of environmental conditions.
Due to their small size, low weight, and low speeds, they require the capability of coping with wind speeds that are approaching or even faster than the nominal airspeed.

In this thesis, a nonlinear-geometric guidance strategy is presented, addressing this problem. More broadly, a methodology is proposed for the high-level control of non-holonomic unicycle-like vehicles in the presence of strong flowfields (e.g. winds, underwater currents) which may outreach the maximum vehicle speed.
The proposed strategy guarantees convergence to a safe and stable vehicle configuration with respect to the flowfield, while preserving some tracking performance with respect to the target path.

As an alternative approach, an algorithm based on Model Predictive Control (MPC) is developed, and a comparison between advantages and disadvantages of both approaches is drawn.
Evaluations in simulations and a challenging real-world flight experiment in very windy conditions confirm the feasibility of the proposed guidance approach.

Part of this abstract, Chapters 1 and 2 also appeared in [2] as a preprint, and were submitted for publication at ACC-2017, Seattle.

Symbols

Symbols

Indices

e	east axis
n	north axis

Acronyms and Abbreviations

ETH	Eidgenössische Technische Hochschule
IMU	Inertial Measurement Unit
UAV	Unmanned Aerial Vehicle
MPC	Model Predictive Control
NMPC	Nonlinear Model Predictive Control
NL	NonLinear
LALE	Low Altitude Long Endurance
HIL	Hardware In The Loop

Chapter 1

Introduction

In recent years, the use of small fixed-wing Unmanned Aerial Vehicles (UAVs) has steadily risen in a wide variety of applications due to increasing availability of open-source and user-friendly autopilots, e.g. Pixhawk Autopilot [3], and low-complexity operability, e.g. hand-launch.
Fixed-wing UAVs have particular merit in long-range and/or long-endurance remote sensing applications. Research in ETH Zürich's Autonomous Systems Lab (ASL) has focused on Low-Altitude, Long-Endurance (LALE) solar-powered platforms capable of multi-day, payload-equipped flight [4], already demonstrating the utility of such small platforms in real-life humanitarian applications [5].

UAVs autonomously navigating large areas for long durations will inherently be exposed to a variety of environmental conditions, namely, high winds and gusts. With respect to larger and/or faster aircrafts, wind speeds rarely reach a significant ratio of the vehicle airmass-relative speed. Conversely, wind speeds rising close to the vehicle maximum airspeed, and even surpassing it during gusts, is a frequent scenario when dealing with a small-sized, low-speed aircraft.

Usually in aeronautics, windfields are handled as an unknown low-frequency disturbance which may be dealt with either using robust control techniques, e.g. loop-shaping in low-level loops, or simply including integral action within guidance-level loops.
In the case of LALE vehicles, maximizing flight time would further require the efficient use of throttle, thus limiting airspeed bandwidth. In order to be able to use such systems safely and efficiently in a wide range of missions and different environments, it is necessary to take care of such situations directly at the guidance level of control, explicitly taking into account online wind estimates.

Goal of this thesis is to indeed develop guidance strategies that may cope with extreme wind-scenarios reliably and without the aircraft entering in emergency states.

Two different approaches will be explored. In chapter 2, a novel nonlinear-geometric guidance algorithm will be developed, building on some earlier results, in order to deal with arbitrarily strong and changing windfields. In chapter 3, we will develop a Model Predictive Controller that thanks to an underlying optimization procedure is able to automatically take care of the wind effect. In chapter 4 the performances of the two approaches will be compared, and an attempt to draw conclusions about the main advantages and disadvantages of approaches will be made. Future extensions of this work will also be discussed in chapter 3.

Chapter 2

A Nonlinear-Geometric Guidance Strategy to cope with arbitrarily strong windfields

Throughout this chapter, a nonlinear-geometric approach for the guidance of small UAVs in strong windfields will be developed. First, a brief review of existing nonlinear guidance strategies and ways to take the wind into account will be provided in Section 2.1. Then the mathematical framework and the problem definition will be presented in Section 2.2, and the novel proposed nonlinear-geometric guidance strategy will be developed through sections 2.3, 2.4. A proof of stability in the case of higher winds will be provided in Section 2.5. Continuity properties of the algorithm to changing winds will be shown in Section 2.6. The behaviour of the algorithm when the actual roll-dynamics of real aircrafts are considered is analyzed in Section 2.7. The validation of the algorithm through real flight-tests will be presented in Section 2.8.

2.1 Nonlinear guidance algorithms for fixed-wing UAVs: a brief review

Put simply, the goal of the guidance layer of control for UAVs is to make sure the vehicle will converge to and follow a given desired path. Though it should be noted there is a distinction between a *path-following problem* and a *path-tracking problem*, which is that in the path-following framework the path is defined spatially, while in the latter it is defined in the space-time domain (the vehicle is required to be in a certain place at a certain time) ([1]).

As in typical applications for small fixed UAVs it is required to cover certain areas to take images/perform patrolling task, we are most concerned about path-following: this is the problem we will consider throughout the chapter.

In [1], a point is made to remark that, ideally, the guidance layer of control for path-following is required to be simple to understand and implement, computationally light for real-time flight, and as the path is defined geometrically it should not put constraints on the actual speed of the vehicle.
Additionally, as far as non-linear guidance algorithms are concerned, it is of primary importance that the set of initial positions and velocities that allow to achieve convergence to the path is very large, if not infinite. In case this is not satisfied, it is required to switch between a manual-mode control and an automatic-mode control when the vehicle reaches a narrow neighbourhood of the path, which is far from being optimal in real applications ([1]).
Also note that many control-theoretic algorithms result in huge control inputs as the vehicle is far from the path (one could simply think of a PID controller, for example): this is another reason why the set of initial conditions might be severely restricted. This is why a generalized law allowing to perform completely autonomous flight starting from anywhere in the air is needed.
As also remarked in [1], specifically, the most used existing guidance methods for path-following can be classified into two main categories: linear and nonlinear.

The linear methods can be based on Proportional-Integral-Derivative (PID) control as in [6], and the Linear Quadratic Regulator as for example in [7], [8]. As anticipated, these methods present several disadvantages, such as the initial input-command being too large if the vehicle is not starting from an immediate neighbourhood of the path, and the impossibility to take the wind directly into account, which as we will see in the next subsection is something we are most concerned about.

The nonlinear methods can be mainly subdivided into three categories ([1]).

- The "error-regulation-based method": as for example in [9],[10], the error model for the system is derived and well-known nonlinear control techniques are applied to steer these error variables to zero (which can include cross-track error, vehicle heading error, along-track error etc . . .). The main drawback of this approach is that it's very model-dependent and therefore pretty difficult to implement. In addition to this, as the initial error is larger, the control input will be larger. This is the same problem we had with linear approaches.

- The "vector-field-based method": a vector field is designed to steer the vehicle onto the path. The main weakness is that one has to design the vector field for the particular shape of the path at hand, therefore it is not easily generalizable.

- The "virtual-target-following method": the guidance command is conceived to follow a virtual target point that moves along the path. This virtual vehicle is considered to be ahead of the vehicle, in order to obtain an "anticipation" effect on the path evolution. This method is geometric rather than control-theoretic.

A method using the virtual-target-following approach is the nonlinear path-following guidance law presented in [11], [12]. As clearly remarked in [1], the strengths of this method are its simplicity and the so-called "look-ahead effect", which enables tight tracking of curved paths by anticipating the upcoming desired path and some degree of wind effect compensation. This also provides robustness against external disturbances and smooth incidence to the desired path. However, this method also has a few drawbacks: the initial position of the vehicle should be inside of the specified look-ahead distance from the desired path, therefore a separate guidance law is needed if the initial position is outside of the look-ahead distance. Additionally, some overshoot response is shown in the initial phase, and the switching between a straight and a circular path always entails some overshoot. This method also cannot achieve exact tracking of general 3D curves of variable curvature ([1]).

In order to overcome these weaknesses, and most of all to allow the set of initial conditions to include large deviations, the authors in [1] proposed a 3-D nonlinear differential geometric path-following guidance law. This guidance law takes inspiration from pursuit guidance, similarly to the methods in the virtual-target-following approach.

Notice that these guidance laws do not take the wind disturbances into account directly. In the next subsection, some common approaches will be briefly described.

2.1.1 Wind Compensation techniques

A common strategy to eliminate the influence of wind on path following is to consider the inertial groundspeed of the vehicle, which inherently includes wind effects, see [13], [14]. Another approach is to take the wind explicitly into account, either by available wind measurements [15] [16] or by exploiting a disturbance observer, as in [9]. Another possibility is described in [17], where adaptive backstepping is used to get an estimate for the direction of the wind.
As to wind compensation techniques, a possible approach is vector fields [14] [18]. In [14], a vector field approach is used to achieve asymptotic tracking of circular and straight-line paths in the presence of non negligible persisting wind disturbances: vector fields are proposed for specific curves (e.g. straight lines, circles). This requires switching the commands when the target path is defined as the union of different parts, which makes the algorithm less uniform and its implementation trickier. Tuning of vector fields is also known to be difficult, as

highlighted in [18].
Another popular approach is based on nonlinear guidance. The strategy proposed in [12], utilizes a *look-ahead* vector for improved tracking of upcoming paths. The law was extended in [1] to any 3D path in the non-windy case. Great advantages of this law are that it is easy and intuitive to tune, the magnitude of the guidance commands is always upper-bounded, and it has flexibility in the set of feasible initial conditions.

The main contribution of this thesis is a simple, safe, and computationally efficient guidance strategy for navigation in *arbitrarily* strong windfields. To our knowledge, in particular, there is no existing guidance law directly considering the case of the windspeed being higher than the airspeed. The provided design strategy relies on the solution provided in [1] in absence of wind whose choice for the look-ahead vector will be properly modified in order to cope with arbitrarily strong windspeed.

Notation. We shall use the bold notation to denote vectors in $\mathbb{R}^3$. For a vector $\mathbf{v} \in \mathbb{R}^3$, $\hat{\mathbf{v}}$ denotes the associated versor and $\|\mathbf{v}\|$ the euclidean norm. For two vectors $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^3$, their scalar and cross products are respectively indicated by $\mathbf{v}_1 \cdot \mathbf{v}_2 \in \mathbb{R}$ and $\mathbf{v}_1 \times \mathbf{v}_2 \in \mathbb{R}^3$.

2.2 Formal Problem Definition

As a first important hypothesis, throughout this thesis, we are going to consider the most usual scenario of a mission involving path tracking of a horizontal path defined at a fixed-altitude. The wind will be considered to be horizontal as well. In addition to being the most usual scenario, one should also note that obtaining precise estimate for the vertical component of the wind is a technical challenge that is not yet completely overcome.

As we wish to extend the results obtained in [1], it is useful to define the same mathematical framework. To have a better insight, we will clearly define the control problem for each different scenario, and define a state-space nonlinear formulation. This will allow us to state a robust control problem, which will be useful for analysis in future work.

2.2.1 The Frenet-Serret framework for autonomous guidance

The position of the vehicle is denoted by $\mathbf{r}_M$, which is a vector of $\mathbb{R}^3$ expressed with respect to an inertial reference frame denoted by $\mathcal{F}_i$ and described by an orthonormal right-hand basis $(\mathbf{i}, \mathbf{j}, \mathbf{k})$. We assume that $(\mathbf{i}, \mathbf{j})$ are co-planar with the flight plane, with $\mathbf{k}$ orthogonal to such a plane. The emphasis of the work is on developing a controller able to cope with strong wind. The latter is a vector $\mathbf{w} \in \mathbb{R}^3$ assumed to be constant and to lie on the flight plane, namely with zero component along $\mathbf{k}$. The vectors $\mathbf{v}_G \in \mathbb{R}^3$ and $\mathbf{a}_M \in \mathbb{R}^3$ in the plane $(\mathbf{i}, \mathbf{j})$ denote the ground speed and acceleration of the vehicle, the dynamics of the latter is described by

$$\dot{\mathbf{r}}_M = \mathbf{v}_G\,, \qquad \dot{\mathbf{v}}_G = \mathbf{a}_M\,. \tag{2.1}$$

Considering flight through a moving airmass, $\mathbf{v}_G = \mathbf{v}_M + \mathbf{w}$, in which $\mathbf{v}_M$ is the vehicle airmass-relative speed (or airspeed). Note that, since $\mathbf{w}$ is constant, $\dot{\mathbf{v}}_G = \dot{\mathbf{v}}_M$. The acceleration $\mathbf{a}_M$ represents the control input.

From a geometric viewpoint, the *vehicle path* is defined as the union of each $\mathbf{r}_M(t)$ for every time t. At each $t \geq 0$ the vehicle path can be geometrically characterized in terms of the *unit tangent vector*, the *actual orientation*, the *tangential acceleration*, the *normal acceleration*, the *tangential acceleration*, the *unit normal vector* and the *curvature* of the vehicle path respectively defined as

$$\begin{aligned} &\hat{\mathbf{T}}_G(t) := \frac{\mathbf{v}_G(t)}{\|\mathbf{v}_G(t)\|}\,, \qquad \hat{\mathbf{T}}_M(t) := \frac{\mathbf{v}_M(t)}{\|\mathbf{v}_M(t)\|}\,, \\ &\mathbf{a}_M^T(t) := (\mathbf{a}_M(t)\cdot\hat{\mathbf{T}}_M(t))\hat{\mathbf{T}}_M(t)\,, \\ &\qquad\quad \mathbf{a}_M^N(t) := (\hat{\mathbf{T}}_M(t)\times\mathbf{a}_M(t))\times\hat{\mathbf{T}}_M(t)\,, \\ &\hat{\mathbf{N}}_M(t) := \frac{\mathbf{a}_M^N(t)}{\|\mathbf{a}_M^N(t)\|}\,, \quad k_M(t) := \frac{\|\mathbf{a}_M(t)\|}{\|\mathbf{v}_G(t)\|^2}\,. \end{aligned} \tag{2.2}$$

We observe that the unit normal vector is defined only for values of the acceleration such that $\|\mathbf{a}_M^N(t)\| \neq 0$. Furthermore, all the previous vectors lie in the plane $(\mathbf{i}, \mathbf{j})$. Having in mind the application to fixed-wing UAVs, we will consider the vehicle to be unicycle-like, i.e. its speed norm $\|\mathbf{v}_M\|$ will remain unchanged in time and it will be then guided through normal acceleration commands $\mathbf{a}_M^N$. In other words, the control law for $\mathbf{a}_M$ will be chosen in such a way that $\mathbf{a}_M^T(t) \equiv 0$. According to this, and by bearing in mind (2.2), (2.1) can be rewritten as

$$\dot{\mathbf{r}}_M(t) = v_M^\star\hat{\mathbf{T}}_M(t) + \mathbf{w}(t), \quad v_M^\star\dot{\hat{\mathbf{T}}}_M(t) = \mathbf{a}_N^M(t) \tag{2.3}$$

in which $v_M^\star$ denotes the (constant) value of $\|\mathbf{v}_M\|$.

Inspired by [1], the *desired (planar) path* is a continuously differentiable space curve in the plane spanned by $(\mathbf{i}, \mathbf{j})$ represented by $\mathbf{p}(l)$, $l \in \mathbb{R}$, with associated a *Frenet-Serret* frame composed of three orthonormal vectors $(\hat{\mathbf{T}}_p(l), \hat{\mathbf{N}}_p(l), \hat{\mathbf{B}}_p(l))$, a curvature $\kappa_p(l)$ and a torsion $\tau_p(\ell)$. In the following we let $s \in \mathbb{R}$ the arc length along the curve $p(\cdot)$ defined as

$$s(l) = \int_{l_0}^{l} \|\frac{d\mathbf{p}(\ell)}{d\ell}\| d\ell\,.$$

The desired path is thus endowed with the *Frenet-Serret* dynamics given by

$$\begin{pmatrix} \hat{\mathbf{T}}_p'(s) \\ \hat{\mathbf{N}}_p'(s) \\ \hat{\mathbf{B}}_p'(s) \end{pmatrix} = \begin{pmatrix} 0 & \kappa_p(s) & 0 \\ -\kappa_p(s) & 0 & \tau_p(s) \\ 0 & -\tau_p(s) & 0 \end{pmatrix} \begin{pmatrix} \hat{\mathbf{T}}_p(s) \\ \hat{\mathbf{N}}_p(s) \\ \hat{\mathbf{B}}_p(s) \end{pmatrix} \tag{2.4}$$

in which we used the notation $(\cdot)'$ to denote the derivative with respect to s. As in [1], we define the "footprint" of $\mathbf{r}_M$ on $\mathbf{p}$ at time t as the closest point of $\mathbf{r}_M(t)$ on $\mathbf{p}(l)$ defined as

$$\mathbf{r}_P(s(t)) := \arg\min_{\mathbf{r}\in\mathbf{p}} \|\mathbf{r}_M(t) - \mathbf{r}\| \,.$$

The point P on the desired path is identified by l_P, which is the value of the curve parameter l at the closest projection. The unit tangent vector, the unit normal vector, the unit binormal vector, the curvature and the torsion of the desired path at the point P will be indicated in the following as $\hat{\mathbf{T}}_P := \hat{\mathbf{T}}_p(l_P)$, $\hat{\mathbf{N}}_P := \hat{\mathbf{N}}_p(l_P)$, $\hat{\mathbf{B}}_P := \hat{\mathbf{B}}_p(l_P)$, $\kappa_P := \kappa_p(l_P)$ and $\tau_P := \tau_p(l_P)$. They are all functions of time through $s(t)$. By bearing in mind (2.4), it turns out that the vehicle dynamics induce a *Frenet-Serret* dynamics on the desired path which is given by

$$\begin{pmatrix} \dot{\hat{\mathbf{T}}}_P(t) \\ \dot{\hat{\mathbf{N}}}_P(t) \\ \dot{\hat{\mathbf{B}}}_P(t) \end{pmatrix} = \dot{s}(t) \begin{pmatrix} 0 & \kappa_p(t) & 0 \\ -\kappa_p(t) & 0 & \tau_p(t) \\ 0 & -\tau_p(t) & 0 \end{pmatrix} \begin{pmatrix} \hat{\mathbf{T}}_P(t) \\ \hat{\mathbf{N}}_P(t) \\ \hat{\mathbf{B}}_P(t) \end{pmatrix} \tag{2.5}$$

in which $\dot{s}(t)$ can be easily computed as (see Lemma 1 and Appendix B in [1]).

$$\dot{s}(t) = \frac{(v_M^\star \hat{\mathbf{T}}_M(t) + \mathbf{w}) \cdot \hat{\mathbf{T}}_P(t)}{1 + \kappa_P(t)[(\mathbf{r}_P(t) - \mathbf{r}_M(t)) \cdot \hat{\mathbf{N}}_P(t)]} \,.$$

The (ideal) desired control objective is to asymptotically steer the position of the vehicle $\mathbf{r}_M(t)$ to the footprint $\mathbf{r}_P(s(t))$ by also aligning the unitary tangent vectors $\hat{\mathbf{T}}_G(t)$ and $\hat{\mathbf{T}}_P(t)$ and their curvature. To this end it is worth introducing an error $\mathbf{e}(t)$ defined as

$$\mathbf{e}(t) := \mathbf{r}_P(t) - \mathbf{r}_M(t)$$

and to rewrite the relevant dynamics in error coordinates. In this respect, by considering the system dynamics (2.1), the Frenet-Serret dynamics (2.5), it is simple to obtain (for compactness we omit the arguments t)

$$
\begin{aligned}
\dot{\mathbf{e}} &= -\left(\mathbf{v}_G \cdot \hat{\mathbf{T}}_P\right)\left(\frac{\kappa_P(\mathbf{e}\cdot\hat{\mathbf{N}}_P)}{1+\kappa_P(\mathbf{e}\cdot\hat{\mathbf{N}}_P)}\hat{\mathbf{T}}_P + \hat{\mathbf{N}}_P\right) \\
\dot{\hat{\mathbf{T}}}_P &= \frac{\kappa_P(\mathbf{v_G}\cdot\hat{\mathbf{T}}_P)}{1+\kappa_P(\mathbf{e}\cdot\hat{\mathbf{N}}_P)}\hat{\mathbf{N}}_P \\
\dot{\hat{\mathbf{N}}}_P &= \frac{(\mathbf{v}_G\cdot\hat{\mathbf{T}}_P)}{1+\kappa_P(\mathbf{e}\cdot\hat{\mathbf{N}}_P)}\left(\tau_P\hat{\mathbf{B}}_P - \kappa_P\hat{\mathbf{T}}_P\right) \\
\dot{\hat{\mathbf{B}}}_P &= \frac{-\tau_P(\mathbf{v}_G\cdot\hat{\mathbf{T}}_P)}{1+\kappa_P(\mathbf{e}\cdot\hat{\mathbf{N}}_P)}\hat{\mathbf{N}}_P \\
v_M^\star\dot{\hat{\mathbf{T}}}_M &= \mathbf{a}_M^N
\end{aligned}
\tag{2.6}
$$

with the ground speed $\mathbf{v}_G$ that is a function of $\hat{\mathbf{T}}_M$ and $\mathbf{w}$ according to

$$\mathbf{v}_G = v_M^\star\hat{\mathbf{T}}_M + \mathbf{w}\,.$$

This is a system with state $(\mathbf{e}, \hat{\mathbf{T}}_P, \hat{\mathbf{N}}_P, \hat{\mathbf{B}}_P, \hat{\mathbf{T}}_M)$ with control input $\mathbf{a}_M$ (to be chosen so that $\mathbf{a}_M^T \equiv 0$) subject to the wind disturbance $\mathbf{w}$. Note that for planar paths, $\tau_P = 0$.

Similarly to [1], the acceleration command will be chosen as

$$\mathbf{a}_M^N = (\mathbf{v}_M \times \mathbf{u}) \times \mathbf{v}_M \tag{2.7}$$

with $\mathbf{u} \in \mathbb{R}^3$ an auxiliary input to be chosen. Note that this choice guarantees that $\mathbf{a}_M^T(t) \equiv 0$ for all possible choices of $\mathbf{u}$. The degree-of-freedom for the problem is then the choice of the control input $\mathbf{u}$ to accomplish control goals.

Motivated by [19], the choice of $\mathbf{u}$ presented in this work relies on the so-called *look-ahead vector*, denoted by $\hat{\mathbf{L}}$, which represents the *desired groundspeed direction* for the vehicle. The latter will be taken as a function of the system state and of the wind, according to the objective conditions in which the vehicle operates.

2.2.2 Feasibility Cone and Control Objective Formulation

Although the ideal control objective is to steer the error $\mathbf{e}(t)$ asymptotically to zero by also aligning the unitary tangent vectors $\hat{\mathbf{T}}_G(t)$ and $\hat{\mathbf{T}}_P(t)$ and their curvature, the presence of "strong" wind could make this ideal objective infeasible by forcing degraded tracking performances that take into account also safety issues. For this reason we set two objectives that will be targeted according to the wind conditions.

Ideal Tracking Objective. Ideally, the control input $\mathbf{u}$ must be chosen so that the following asymptotic objective is fulfilled

$$\begin{cases} \lim\limits_{t->\infty} \mathbf{e}(t) = 0 \\ \lim\limits_{t->\infty} (\hat{\mathbf{T}}_{\mathbf{G}}(t) - \hat{\mathbf{T}}_P(t)) = 0 \\ \lim\limits_{t->\infty} (\frac{d\hat{\mathbf{T}}_{\mathbf{G}}(t)}{dt} - \frac{d\hat{\mathbf{T}}_P(t)}{dt}) = 0 \end{cases} \tag{2.8}$$

namely position, ground speed orientation, and ground speed curvature of the vehicle converge to the path ones.

Safety Objective. When strong wind does not allow to achieve the ideal objective, the degraded safety objective consists of controlling the vehicle in such a way that the vehicle acceleration is asymptotically set to zero, the groundspeed value is asymptotically minimized (by pointing the nose the vehicle against wind) and the vehicle nose asymptotically points to $\mathbf{P}$, namely

$$\begin{cases} \lim\limits_{t\to\infty} \mathbf{a}_M^N(t) = 0 \\ \lim\limits_{t\to\infty} \hat{\mathbf{T}}_M(t) = -\hat{\mathbf{w}} \\ \lim\limits_{t\to\infty} \hat{\mathbf{e}}(t) = -\hat{\mathbf{w}}\,. \end{cases} \tag{2.9}$$

The targeted configuration, in particular, is the one in which the vehicle goes away with the wind, by minimizing the groundspeed (safety objective), and minimizing the distance to the closest point on the path. Note that this objective makes sense for *finite-length* paths: the infinite-length linear path case is briefly discussed in [2].

Ideal or degraded objectives are set according to the fulfillment of a "feasibility condition" by the look ahead vector. More precisely, with $w^\star := \|\mathbf{w}\|$ the wind strength, let β be defined as

$$\beta \;:=\; \begin{cases} \arcsin\frac{v_M^\star}{w^\star} & w^\star \geq v_M^\star \\ \pi & w^\star < v_M^\star\,. \end{cases} \tag{2.10}$$

Then, we define the "feasibility cone" $\mathcal{C}$ as the cone with apex centered at the vehicle position $\mathbf{r}_M$, main axis given by $\mathbf{w}$ and with aperture angle 2β (see Figure 2.8). All desired groundspeed vectors that lie in the cone can be indeed enforced by appropriately choosing the control input $\mathbf{u}$. This fact, and the fact that the look ahead vector represents the desired groundspeed direction for the vehicle, motivates the fact of considering the ideal tracking objective feasible at a certain time t if it's possible to shape the look ahead vector $\hat{\mathbf{L}}(t)$ so that it lies in $\mathcal{C}$. More specifically, if

$$\arccos\hat{\mathbf{w}}\cdot\hat{\mathbf{L}}(t) < \beta\,. \tag{2.11}$$

Otherwise, the ideal tracking objective is said infeasible at time t. The control objectives are set consequently. In the next section we show how to design a look-ahead vector such that if the ideal tracking objective is feasible then (2.9) is achieved, otherwise the Safety Objective is enforced.

2.2.3 The Nominal Solution in Absence of Wind in [1]

In this section we briefly present the solution chosen in [1] for the look-ahead vector in absence of wind, as it represents the basis for developing the windy solution. A graphical sketch showing the notation is provided in Figure 2.1. The authors in [1] proposed the control law

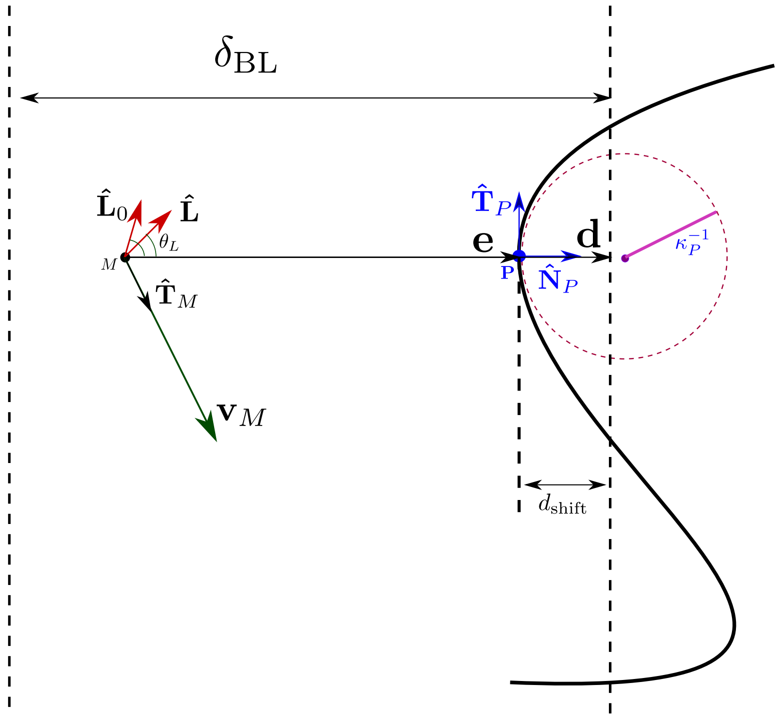

Figure 2.1: Sketch of the nominal solution

$$\mathbf{u} = k\hat{\mathbf{L}} \tag{2.12}$$

in which k is a design parameter chosen so that $k > \max\limits_{P\in p(l)} k_P$ and $\hat{\mathbf{L}}$ is the look-ahead vector chosen as

$$\hat{\mathbf{L}} = \cos\left(\theta_L(\|\mathbf{d}\|)\right)\hat{\mathbf{d}} + \sin\left(\theta_L(\|\mathbf{d}\|)\right)\hat{\mathbf{T}}_P \tag{2.13}$$

where $\mathbf{d} = \mathbf{e}+d_{\text{shift}}, \hat{\mathbf{N}}_P = (\|\mathbf{e}\|+d_{\text{shift}})\hat{\mathbf{N}}_P$ is the *radially shifted distance*, $\theta_L(\|\mathbf{d}\|)$ is a function satisfying

- $$-\infty < \frac{d\theta_L(\|\mathbf{d}\|)}{d\|\mathbf{d}\|} < 0$$
  when $|\mathbf{d}| < \delta_{BL}$, where δ_{BL} is a boundary layer to be chosen (i.e., $|\mathbf{d}| < \delta_{BL}$ means the vehicle is inside the boundary layer). This means that we require, by considering 2.13, that as the vehicle approaches the path, the look-ahead vector progressively aligns to $\hat{\mathbf{T}}_P$.

- $$\frac{d^2\theta_L(\|\mathbf{d}\|)}{d\|\mathbf{d}\|^2} < 0$$
  when $|\mathbf{d}| < \delta_{BL}$, i.e. inside the boundary layer. This means we require that the look-ahead vector smoothly converges $\hat{\mathbf{T}}_P$.

- $$\theta_L(\|\mathbf{d}\|) = 0$$
  when $|\mathbf{d}| > \delta_{BL}$, i.e. outside the boundary layer. This means the look-ahead vector points to P, as if the distance is very large our need is to approach the path as fast as possible.

- $\theta_L(\|\mathbf{d}\| = 0) = \frac{\pi}{2}$ and $\theta_L(\|\mathbf{d}\| = \delta_{BL}) = 0$ to satisfy the boundary conditions.

Technically the choice of δ_{BL}, the boundary layer parameter, is part of the control input, but we are going to consider that to be a parameter fixed in advance, as it would be in a realistic scenario. Several such functions can be found, for example

$$\theta_L(\|\mathbf{d}\|) = \frac{\pi}{2}\sqrt{1 - \mathrm{sat}(\frac{\|\mathbf{d}\|}{\delta_{BL}})} \tag{2.14}$$

As to the d_{shift} parameter, that must be chosen to guarantee the *exact tracking condition* of the command:

$$(\mathbf{a}^N_{M\text{cmd}} \cdot \hat{\mathbf{N}}_P)|_{\mathbf{r}_M=\mathbf{r}_P,\ \hat{\mathbf{T}}_M=\hat{\mathbf{T}}_P} = k_P\|\mathbf{v}_M\|^2 \tag{2.15}$$

In case we choose (2.14), then $d_{\text{shift}} = [1 - (\frac{2}{\pi}\arccos\frac{|k_P|}{k})^2]\delta_{BL}$.

Instrumental for the next results, we also introduce the look-ahead vector computed on the error $\mathbf{e}$ instead of the radially shifted distance $\mathbf{d}$, that is

$$\hat{\mathbf{L}}_0 := \hat{\mathbf{L}}|_{\mathbf{d}=\mathbf{e}}\,. \tag{2.16}$$

A way to look at the choice of $\hat{\mathbf{L}}$ as in 2.13, is that it considers a tradeoff between "aggressive" maneuvers pointing directly to the path, and "smoother" maneuvers that steer the vehicle onto the path direction. A further consideration, hinting to the convergence of the law proposed in [1] ,as $\hat{\mathbf{T}}_P$ can be considered the geometric derivative of the path at point P, is that $\hat{\mathbf{L}}$ as in 2.13 is acting as a sort of nonlinear-time-varying "PD" controller.

2.3 The Lower Wind Case

In this section, we consider the case where the wind is slower than the airspeed, i.e. $w^\star < v^\star_M$.

2.3.1 Previous solutions and their weaknesses

First thing it is worth trying is not to take the wind into account at all and apply the Nominal Solution described in 2.2.3. Convergence cannot be achieved even in the presence of very weak flowfields as can be observed in figure 2.2.

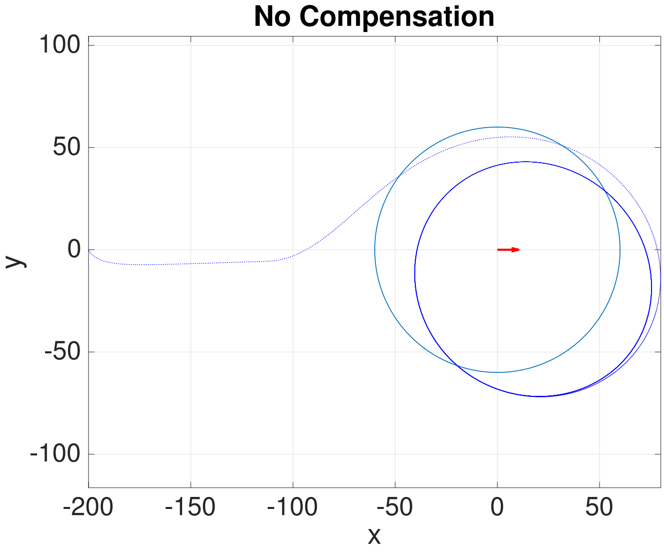

Figure 2.2: Wind is 7 m/s, airspeeed is 14 m/s. The Nominal Solution is applied, without wind compensation, is applied.

If wind measurements are not directly available, a simple approach to achieve path convergence with any wind, could be to apply the normal acceleration command

$$\mathbf{a}^N_M = k(v^\star \hat{\mathbf{T}}_\mathbf{M} + \mathbf{w}) \times \hat{\mathbf{L}} \times (v^\star \hat{\mathbf{T}}_\mathbf{M} + \mathbf{w}) \tag{2.17}$$

so that $\mathbf{v}_G$ will eventually be aligned with the look-ahead vector $\hat{\mathbf{L}}$. This requires the aircraft to accelerate/decelerate along its longitudinal axis: indeed, the acceleration command which is perpendicular to $\mathbf{v_G}$, which however has a longitudinal component along the vehicle, whose speed will increase/decrease falling outside our problem definition. A simulation with this approach is showed in figure 2.3. If we use this strategy, we must be sure both not to ever overcome the maximum achievable airspeed and also not to command the vehicle to go too slow and lose too much lift force. In general, this approach is hence not applicable.

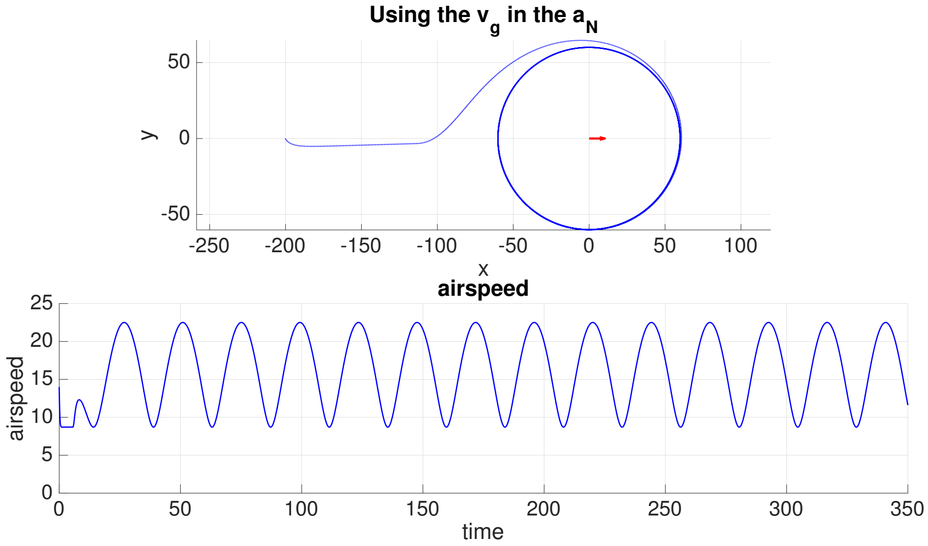

Figure 2.3: Wind is 7 m/s, airspeeed starts at 14 m/s, and varies. The ground speed is used in the nonlinear acceleration command.

A way to still preserve a constant airspeed is to consider the $\mathbf{v}_G$ in the nonlinear acceleration command, but then only apply that component of the acceleration command which lies along the lateral body axis.
Even though this approach partially compensates for the wind, it entails a severe discrepancy for slow vehicles, resulting in non easily predictable behaviours which are largely undesirable and must be taken care of one by one: as an example, at a given moment we could have $\hat{\mathbf{v}}_G \approx -\hat{\mathbf{L}}$, that incorrectly results in $\mathbf{a}_M^N \approx 0$. An example of such suboptimal behaviours is shown in figure 2.4.

2.3.2 Proposed strategy

Here the goal is to find the control input $\mathbf{u}_{\text{slow}}$ that satisfies the requirements in 2.8. We first find a *basic* control input, called $\mathbf{u}_e$, and improve on that to obtain $\mathbf{u}_{\text{slow}}$. To this end, we are going to reason in steady state, i.e.

$$\begin{cases} \mathbf{e} = 0 \\ \hat{\mathbf{T}}_{\mathbf{G}} = \hat{\mathbf{T}}_{\mathbf{P}} \\ \frac{d\hat{\mathbf{T}}_G}{dt} = \frac{d\hat{\mathbf{T}}_P}{dt} \end{cases} \tag{2.18}$$

Initial control input

Here we are going to satisfy the first two requirements in (2.8). It is useful to consider the geometry of the problem shown in Figure 2.5 and introduce the following angles

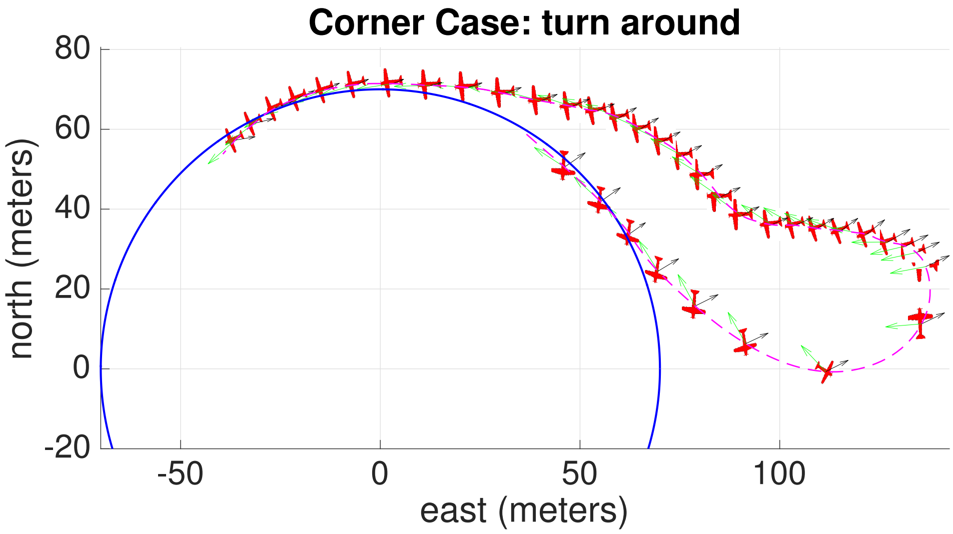

Figure 2.4: Initially, the ground speed is almost aligned with the look-ahead vector, hence the aircraft is not commanded to change its attitude and gets carried away by the wind. The aircraft is forced to perform a complete turn around to get back on track.

$$\begin{cases} \lambda_e = \arccos \hat{\mathbf{w}} \cdot \hat{\mathbf{L}}_0 \\ y = \arccos -\hat{\mathbf{w}} \cdot \hat{\mathbf{L}}_{1e} = \pi - \lambda_e - \arcsin\left(\frac{w^\star \sin(\lambda_e)}{v_M^\star}\right) \end{cases} \tag{2.19}$$

where $\hat{\mathbf{L}}_{1e}$ is an unknown target orientation for the aircraft to be computed. It should be noted that these angles are not defined in case $\mathbf{w} = 0$. We now aim to satisfy the first two requirements stated in (2.8) through the choice of a basic control input

$$\mathbf{u}_e = k\hat{\mathbf{L}}_{1e} \tag{2.20}$$

To find such a command, we assumed to already be at the Position/Orientation steady state condition. Since we assume to be on the path with the desired orientation, then $k(\mathbf{v}_M \times \hat{\mathbf{L}}_{1e}) \times \mathbf{v}_M = 0$, meaning that $\hat{\mathbf{T}}_M = \hat{\mathbf{L}}_{1e}$ ($\hat{\mathbf{T}}_M = -\hat{\mathbf{L}}_{1e}$ would be an unstable equilibrium, as showed in [1]).

The natural choice for the desired groundspeed direction is $\hat{\mathbf{L}}_0$, as it was defined in 2.16. Note that $\hat{\mathbf{L}}_0|_{\mathbf{e}=0} = \hat{\mathbf{T}}_{\mathbf{P}}$. We need to find the desired direction $\hat{\mathbf{L}}_{1e}$ for the aircraft by solving the geometry shown in picture 2.5, which means solving the following equation in $\hat{\mathbf{L}}_{1e}$:

$$\frac{\mathbf{w} + v_M^\star \hat{\mathbf{L}}_{1e}}{\|\mathbf{w} + v_M^\star \hat{\mathbf{L}}_{1e}\|} = \hat{\mathbf{L}}_0 \tag{2.21}$$

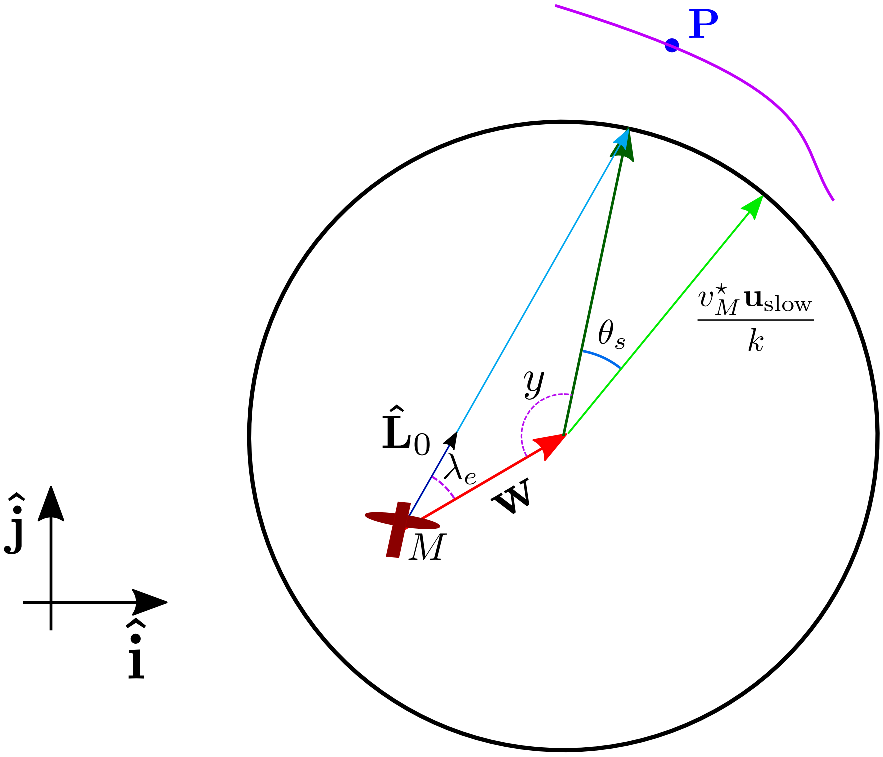

Figure 2.5: Notation.

The solution, in terms of the angles defined in (2.19), is

$$\begin{cases} \hat{\mathbf{L}}_{1e} = \mathrm{sign}([\hat{\mathbf{w}} \times \hat{\mathbf{L}}_0] \cdot \mathbf{k})\mathrm{rot}(-\hat{\mathbf{w}}, y) & w^\star > 0 \\ \hat{\mathbf{L}}_{1e} = \hat{\mathbf{L}}_0 & w^\star = 0 \end{cases} \tag{2.22}$$

where $\mathrm{rot}(a, \theta)$ is the function that rotates vector $a \in \mathbb{R}^3$ by angle θ around the vertical axis $\mathbf{k}$. The basic $\mathbf{u}_e$ will be improved in 2.3.2 to obtain curvature convergence.

Improvement of the control input to satisfy the curvature convergence requirement

Again, we will reason in steady state. In order to satisfy the curvature convergence requirement, we need that

$$(\mathbf{a}_M^N \cdot \hat{\mathbf{N}}_P)|_{\mathbf{r}_M=\mathbf{r}_P,\ \hat{\mathbf{T}}_G=\hat{\mathbf{T}}_P} = k_P \|\mathbf{v}_G\|^2 \tag{2.23}$$

So we define a (scalar) amount of *additional normal acceleration* to be applied for the $\mathbf{v}_G$ to keep the curvature as

$$\|\mathbf{a}_{N\mathrm{res}}^G\| = k\|\mathbf{v}_G\|^2 \|(\hat{\mathbf{L}}_0 \times \hat{\mathbf{L}}) \times \hat{\mathbf{L}}_0\| \tag{2.24}$$

$\hat{\mathbf{L}}$ and $\hat{\mathbf{L}}_0$ as defined in 2.2.3. It's easy to verify that

$$\begin{aligned} \|\mathbf{a}_{N\mathrm{res}}^G\| \ |_{\mathbf{e}=0} &= k\|\mathbf{v}_G\|^2 \|(\hat{\mathbf{T}}_P \times \hat{\mathbf{L}}_{|\mathbf{d}|=\mathbf{d}_{\mathrm{shift}}}) \times \hat{\mathbf{T}}_P\| \\ &= \|k_P\| \|\mathbf{v}_G\|^2 \end{aligned} \tag{2.25}$$

The idea is to rotate the $\mathbf{u}_e = k\hat{\mathbf{L}}_{1e}$ by a proper angle θ_s, which is shaped soon after. If we do so, then in steady state the normal acceleration applied to the vehicle will not be null but

$$\|\mathbf{a}_{N\,\mathrm{res}}^M\| = k{v_M^\star}^2 \sin(\theta_s) \tag{2.26}$$

When $\mathbf{w} = 0$ we have $v_M^\star = \|\mathbf{v}_G\|$, then we can trivially observe that $\|\mathbf{a}_{N\mathrm{res}}^G\| = \|\mathbf{a}_{N\,\mathrm{res}}^M\|$. As $\|\mathbf{a}_{N\mathrm{res}}^G\|$ is obtained through (2.24), then considering (2.26) we can already obtain the needed shifting angle:

$$\theta_s = \arcsin\left[\mathrm{sat}(\|(\hat{\mathbf{L}}_0 \times \hat{\mathbf{L}}) \times \hat{\mathbf{L}}_0\|)\right] \tag{2.27}$$

where $\hat{\mathbf{L}}$ defined as in 2.14, the $\mathrm{sat}(\cdot)$ function bounds the argument to be in the interval [-1,1]. As to the general case $w^\star > 0$, where the angles λ_e and y are defined, remember that in general for any normal acceleration $\mathbf{a}_N = \boldsymbol{\Omega} \times \mathbf{V}$, where $\boldsymbol{\Omega}$ is the angular speed vector $\mathbf{V}$ is the linear speed vector. Then, using the angles introduced before, it holds

$$\begin{cases} \dot{\lambda}_e = \dfrac{\|\mathbf{a}^G_{N\,\text{res}}\|}{\|\mathbf{v}_G\|} \\ \dot{y} = -\dot{\lambda}_e - \dfrac{w^\star \cos(\lambda_e)\dot{\lambda}_e}{v^\star_M\sqrt{1-(\frac{w^\star \sin(\lambda_e)}{v^\star_M})^2}} \end{cases} \tag{2.28}$$

Since it is also true that $\dot{y} = \dfrac{\|\mathbf{a}^M_{N\,\text{res}}\|}{v^\star_M}$, by comparison with the (2.26) we obtain the formulation for θ_s

$$\theta_s = \arcsin\left[\text{sat}(\frac{\dot{y}}{kv^\star_M})\right] \tag{2.29}$$

The conclusion is that by choosing the control input

$$\mathbf{u}_{\text{slow}} = \text{rot}(\mathbf{u}_e, \theta_s) \tag{2.30}$$

$\mathbf{u}_e$ defined as in 2.20, the goals in (2.8) are satisfied. We report in Figure 2.6 a phase portrait showing global convergence in numerical simulations for a large variety of initial conditions and different windspeeds. That said, attractiveness to the equilibrium will have to be formally proved in future work. In Figure 2.7, we can observe the performance of the algorithm for strong constant wind, still slower than the airspeed.

Choice of k

Still we have to determine the parameter k. In order for the algorithm to keep null error in steady state, we have as a lower bound on the choice of k:

$$k > \max_{|k_P|}\frac{4w^\star}{v^\star_M}|k_P| \tag{2.31}$$

The lower bound on k can be derived from the worst case scenario ($\hat{\mathbf{L}}_0 = \hat{\mathbf{w}}$, groundspeed in favour of the wind) by posing the argument of the arcsin in equation (2.29) to be in the interval [-1,1] when $\mathbf{e} = 0$. If we pick an even greater k, it is also possible to guarantee that the argument of the arcsin will not ever saturate during the transient when $\mathbf{e} \neq 0$, resulting in better transient performance.

2.4 The Higher Wind Case

Let us define the desired direction for the groundspeed $\hat{\mathbf{L}}_0$ as in equation (2.16), and the corresponding basic control input $\mathbf{u}_e$ as in equation (2.20). It is convenient to reason considering the angles introduced in (2.19): refer to Figure 2.8 for a better visualization.

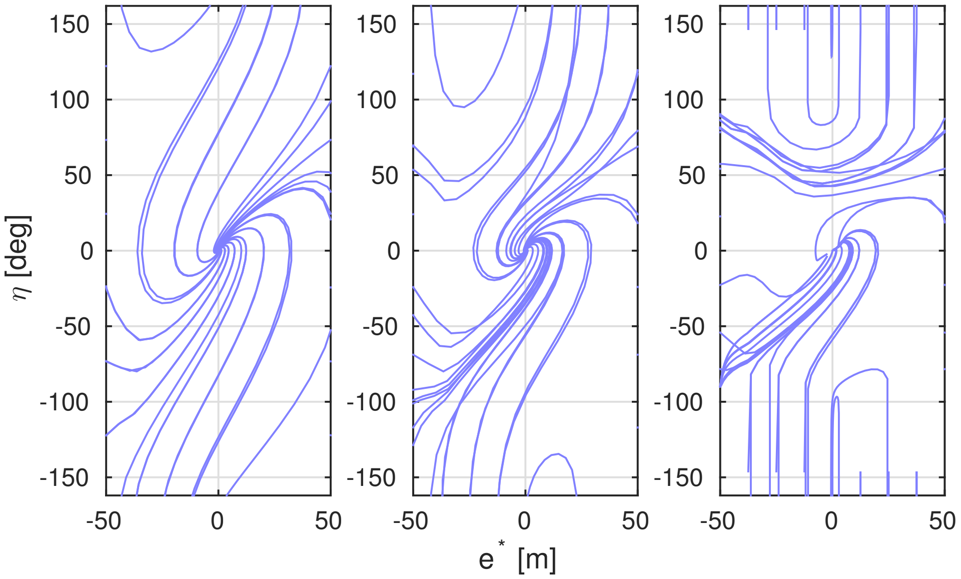

Figure 2.6: Phase portraits of the proposed lower wind solution for $w^{\star}$=0 m/s (**left**), 7 m/s (**middle**), and 13.5 m/s (**right**), respectively. The tracking angular error $\eta = \text{atan2}\left(\hat{\mathbf{T}}_{P_y}, \hat{\mathbf{T}}_{P_x}\right) - \text{atan2}\left(\hat{\mathbf{T}}_{G_y}, \hat{\mathbf{T}}_{G_x}\right) \in [-\pi, \pi]$ is compared with the signed, one-dimensional cross-track error $e^{*} = \mathbf{e} \cdot \frac{\mathbf{r}_M}{\|\mathbf{r}_M\|}$ to demonstrate algorithm convergence within the bounds of $\delta_{BL} = 50$ m, for $k = 0.05$, $R = 100$ m, and $v_M^{\star}$=14 m/s

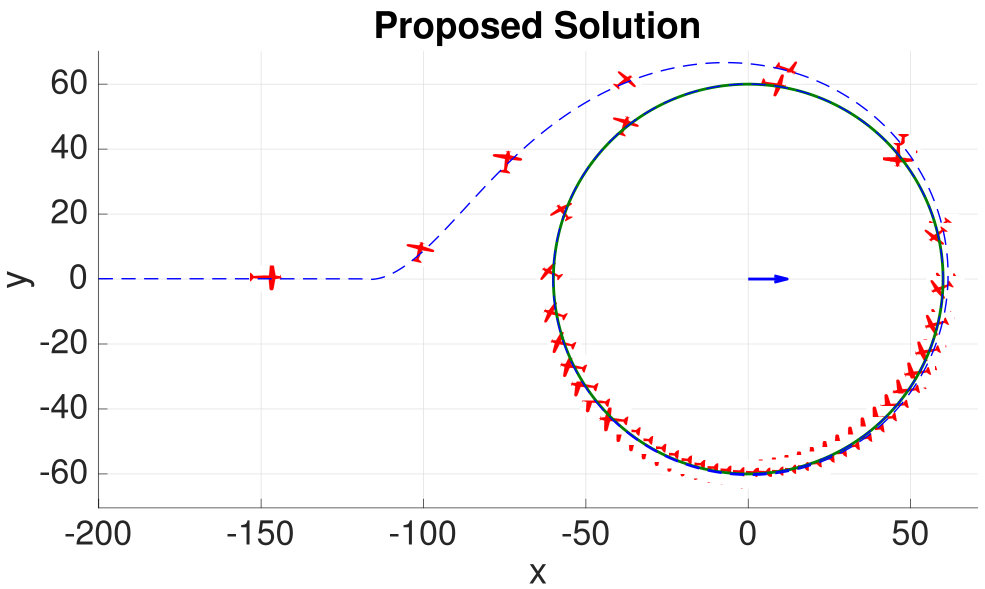

Figure 2.7: Airspeed 14 m/s. Windspeed 12 m/s. The proposed control input lets the vehicle achieve the objectives stated in 2.8.

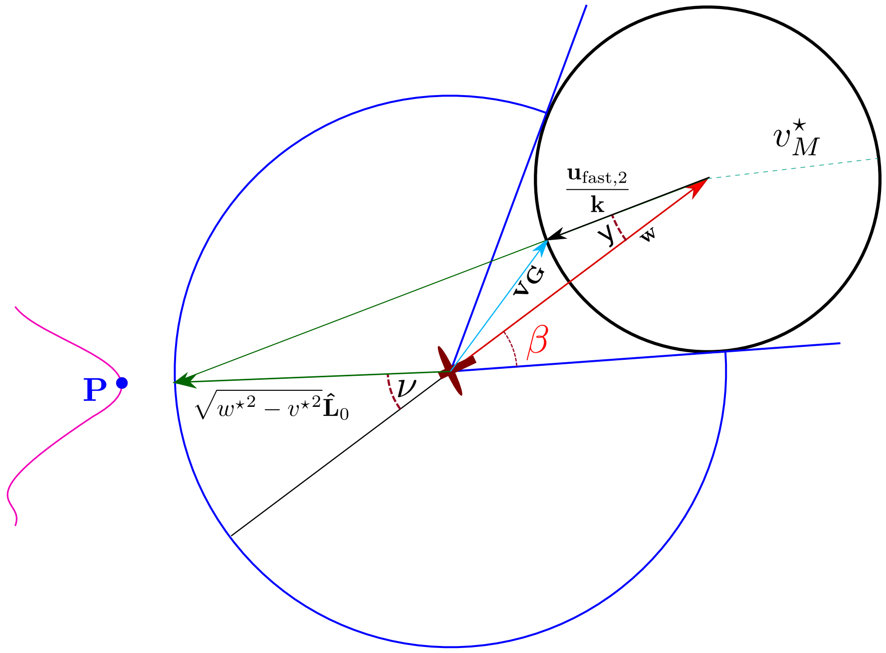
$v_M^\star$
$\frac{\mathbf{u}_{\text{fast},2}}{\mathbf{k}}$
y
w
vG
β
P
ν
$\sqrt{w^{\star 2} - v^{\star 2}}\hat{\mathbf{L}}_0$

Figure 2.8: Geometry: $w^\star > v_M^\star$

2.4.1 Solution for $\hat{\mathbf{L}}_0$ feasible

As the desired groundspeed direction $\hat{\mathbf{L}}_0$ is feasible, the idea is then to reason as in 2.3.2: choose the basic control input $\mathbf{u}_e$ as in (2.20) and shift it by the correct shifting angle θ_s in order to achieve curvature convergence: this would mean $\mathbf{u}_{\text{fast},1} = \mathbf{u}_{\text{slow}}$, and doing so we would achieve curvature convergence as long as the $\hat{\mathbf{L}}_0$ continues to be feasible. However, with usual shapes for the target curved path, at some point the desired direction *will* become infeasible: when this happens, we need the control input not to change abruptly, i.e. to be a continuous function of the desired $\hat{\mathbf{L}}_0$. Since we cannot in general achieve the goals in (2.8), we actually make a slightly different choice for $\mathbf{u}_{\text{fast},1}$ that guarantees continuity in the sense mentioned before (as better explained in [2]), while keeping almost perfect curvature convergence as long as the $\hat{\mathbf{L}}_0$ is feasible:

$$\theta_{s2} = \frac{\sqrt{[1 - (\frac{w^\star \sin(\lambda_e)}{v_M^\star})^2]}}{\cos(\lambda_e)} \theta_s \tag{2.32}$$

So, in the end,

$$\mathbf{u}_{\text{fast},1} = \text{rot}(\mathbf{u}_e, \theta_{s2}) \tag{2.33}$$

2.4.2 Infeasible desired direction: the state of the art.

As anticipated, the case of windspeeds being so strong as to force the aircraft to fly backwards is usually neglected in the field of aeronautics, as it is a pretty rare event. This is indeed quite frequent with LALE UAVs.
When this happens, the main goal we can easily think of, as the aircraft is bound to flow away from the target path together with the wind, is to “minimize the damage”: this results in trying to minimize the norm of the groundspeed. This is achieved if the aircraft turns against the windfield.

The easiest way to do so is to instantaneously ask the aircraft to turn against the wind as soon as it is not possible for it to follow the desired path, and hope for the wind to stop: this is what was done before starting this thesis.
Although this could be a suitable emergency way of acting in case the windspeed rarely overcomes the aircraft airspeed, this is indeed suboptimal and could result in very poor tracking performance in the case of persisting strong winds.

In Figure 2.9, an example of such suboptimal behaviour is shown:
As soon as the target direction becomes infeasible, the aircraft abruptly turns against the wind, and cannot change its mind until the direction becomes feasible again “by chance”. Another reason why this behaviour is undesirable can be seen by thinking of a wind profile that often crosses the line between $w^\star < v_M^\star$ and $w^\star > v_M^\star$: as the commands would be largely discontinuous, this would result in dangerous aircraft behaviour.

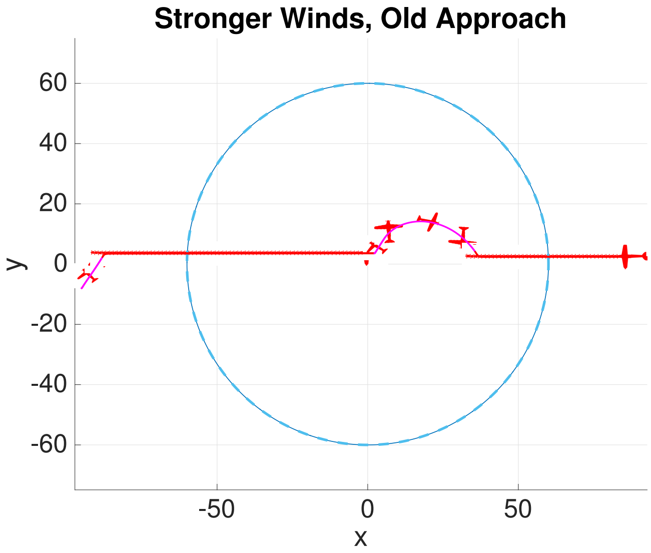

Figure 2.9: Wind is $15m/s$, directed from left to right, airspeed is $14m/s$.

2.4.3 Solution for $\hat{\mathbf{L}}_0$ infeasible

We define an *infeasibility parameter* α_{out} and a *safety function* $\sigma_{\text{safe}}(\alpha_{\text{out}})$ as follows:

$$\alpha_{\text{out}} = \frac{\lambda - \beta}{\pi - \beta} \qquad \sigma_{\text{safe}} = \frac{\frac{\pi}{2} - \beta - y(\alpha_{\text{out}})}{\frac{\pi}{2} - \beta} \tag{2.34}$$

both indices have maximum value equal to 1. When $\sigma_{\text{safe}} = 1$, it means that we act conservatively and choose $\frac{\mathbf{u}_{\text{fast},2}}{k} = -\hat{\mathbf{w}}$: this has to happen only in the absolutely worst scenario of $\hat{\mathbf{L}}_0 = -\hat{\mathbf{w}}$, which corresponds to the maximum $\alpha_{\text{out}} = 1$. In all the intermediate cases, we want to guarantee a tradeoff between conservatism and tracking performance, i.e we want $\sigma_{\text{safe}}(\alpha_{\text{out}})$ to be increasing with α_{out}. This can be achieved by finding a proper mapping f from angle $\nu = \pi - \lambda_e$ to angle y in the following form

$$f : \nu \in [0, \pi - \beta] \to y \in [0, \frac{\pi}{2} - \beta] \tag{2.35}$$

This mapping should satisfy the following 3 properties:

$$\begin{aligned} &f(0) = 0 \\ &f(\pi - \beta) = \frac{\pi}{2} - \beta \\ &f(a) < f(b)\ \forall a > b,\ \ a, b \in [0, \frac{\pi}{2} - \beta] \end{aligned} \tag{2.36}$$

The first requirement is to guarantee that $\sigma_{\text{safe}} = 1$ when $\hat{\mathbf{L}}_0 = -\mathbf{w}$. The second one is a boundary condition to guarantee that the input is continuous to the $\hat{\mathbf{L}}_0$ switching from being feasible to infeasible (or vice versa). The third requirement

is for finding a tradeoff between safety and performance: put in words, the more the $\hat{\mathbf{L}}_0$ is infeasible for the groundspeed, the more we want to turn against the wind and wait for it to stop.
By looking at picture 2.8, a natural choice that follows geometric intuition and is coherent with the requirements that we have just stated, is

$$\mathbf{u}_{\text{fast},2} = k \frac{\sqrt{w^{\star 2} - v_M^{\star}{}^2}\hat{\mathbf{L}}_0 - \mathbf{w}}{\|\sqrt{w^{\star 2} - v_M^{\star}{}^2}\hat{\mathbf{L}}_0 - \mathbf{w}\|} \tag{2.37}$$

In terms of the mapping that has been defined before, this choice corresponds to

$$f(\nu = \pi - \lambda) = y = \arcsin \frac{\sin\nu\cos\beta}{\sqrt{1 + \cos^2\beta + 2\cos\beta\cos\nu}} \tag{2.38}$$

this mapping satisfies the requirements (2.36), as can easily be verified by substitution and derivation with respect to ν. For clarity, the function is plotted in Figure 2.10 for different values of the wind-cone opening angle β

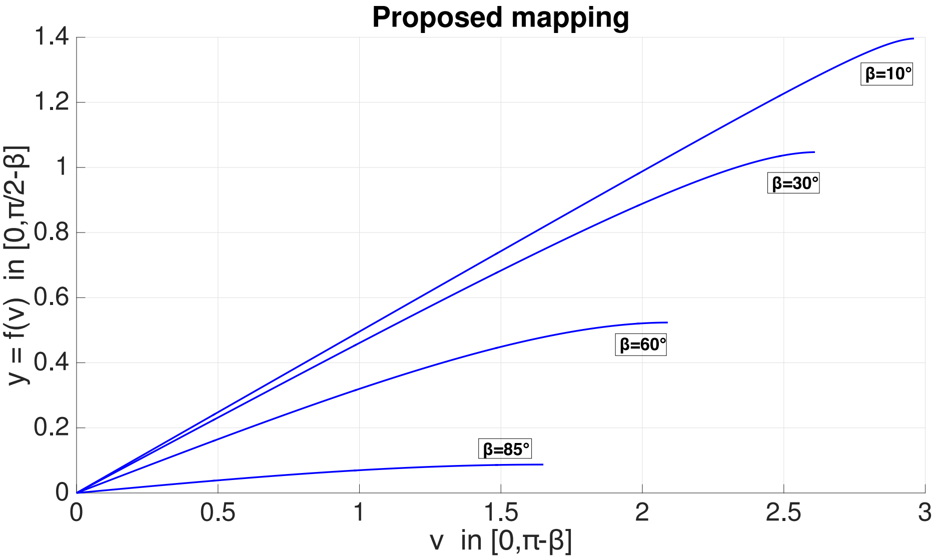

Figure 2.10: proposed mapping $y = f(\nu)$ for different values of β

In Figure 2.11 the performance of the algorithm is shown.
It is also worth highlighting the tradeoff introduced between performance and security (**incremental safety**) by computing the safety function $\sigma_{\text{safe}}(\alpha_{\text{out}})$:

$$\sigma_{\text{safe}} = \frac{\beta - \frac{\pi}{2} + \arcsin\left(\frac{\sin(\pi-\beta-(\pi-\beta)\alpha_{\text{out}})\cos\beta}{\sqrt{1+\cos\beta^2+2\cos\beta\cos(\pi-\beta-(\pi-\beta)\alpha_{\text{out}})}}\right)}{\beta - \frac{\pi}{2}} \tag{2.39}$$

as is also shown in Figure 2.12.

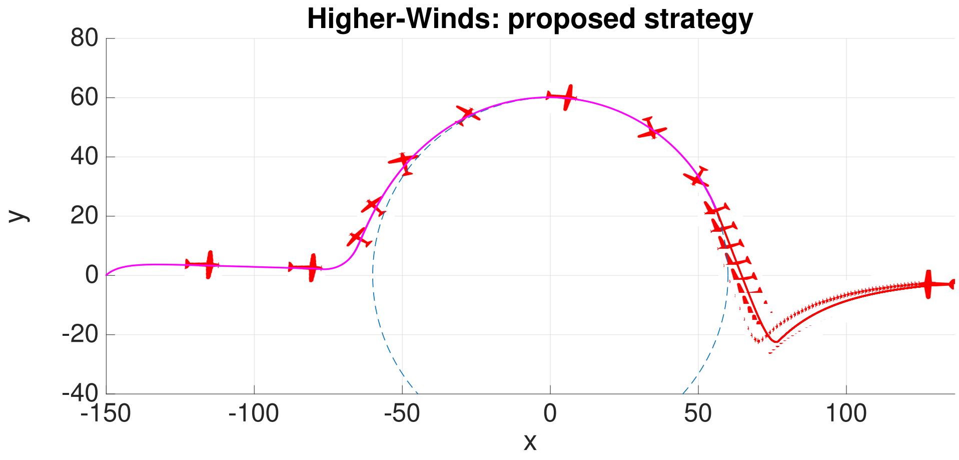

Figure 2.11: wind is 16 m/s, airspeed is 14 m/s. The proposed law is used. Magenta line: feasible desired direction. Red line: infeasible desired direction

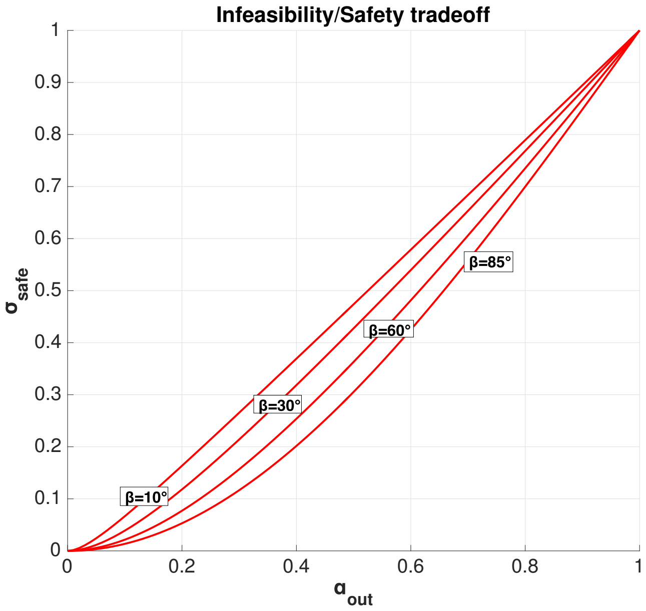

Figure 2.12: Infeasibility/safety relationship, for different βs

2.5 Proof Of Stability

We are in the scenario of $w^\star > v_M^\star$. For finite-length paths, we want to show that we achieve the requirements in 2.9.
Here we will also consider briefly the case of infinite paths: the only realistic case in UAV application is that of infinite linear paths. In this case we want to show that:

$$\begin{cases} \lim\limits_{t\to\infty} \mathbf{a}_N^M(t) = 0 \\ \underset{t\to\infty}{\hat{\mathbf{T}}_M} = -\mathrm{rot}(-\hat{\mathbf{w}}, f(\frac{\pi}{2} - \mu)) \end{cases} \tag{2.40}$$

where $\mu = \arccos \hat{\mathbf{w}} \cdot \hat{\Lambda}$, $\hat{\Lambda}$ is the direction of the target linear path, mapping $f : \nu \to y$ has to be chosen. The second requirement in 2.40 asks for a tradeoff between the linear-path direction and the anti-wind direction for the $\hat{\mathbf{T}}_M$, that results in an efficient direction for the actual $\hat{\mathbf{T}}_G$.
In both cases, the proof for the proposed algorithm will be structured as follows:

- First the so called *geometric case* will be tackled: the vehicle is considered to always be at the desired heading angle, i.e. $\hat{\mathbf{T}}_M(t) = \frac{\mathbf{u}_{\text{fast},2}}{k}(t)$, $\forall t$.
- Then, the so called *dynamical case* (the vehicle is not always at the desired heading angle) will be considered and shown to fall into the geometrical case as time goes to infinity.

2.5.1 Geometric case: finite paths

Subcase 0. Single point path

Here we consider the path to be very far away and hence similar to a single point P for the aircraft to be reached. The radially shifted distance is indistinguishable from the error, so $\theta_s(t) \approx 0\ \forall t$. Also, notice that with a point-path, $\hat{\mathbf{e}} = \hat{\mathbf{L}}_0$. By defining

$$a = \sqrt{w^{\star 2} - {v_M^\star}^2}, \qquad l = \|a\hat{\mathbf{L}}_0 - \mathbf{w}\| \tag{2.41}$$

we obtain

$$\begin{aligned} \mathbf{v}_G \times \hat{\mathbf{L}}_0 &= (w^\star + v_M^\star \hat{\mathbf{L}}_{1e}) \times \hat{\mathbf{L}}_0 = \\ &= (\mathbf{w} + \frac{v_M^\star(\hat{\mathbf{L}}_0 a - \mathbf{w})}{l}) \times \hat{\mathbf{L}}_0 = \\ &= (\underbrace{(1 - \frac{v_M^\star}{l})}_{>0} \mathbf{w} + \frac{a v_M^\star}{l}\hat{\mathbf{L}}_0) \times \hat{\mathbf{L}}_0 \\ &= \underbrace{(1 - \frac{v_M^\star}{l})}_{>0} \mathbf{w} \times \hat{\mathbf{L}}_0 + 0 \end{aligned} \tag{2.42}$$

Now let the line directed as $\hat{\mathbf{L}}_0$ divide the plane into two half-planes: the previous considerations imply that $\mathbf{v}_G$ and $\mathbf{w}$ both lie in the same half-plane, so the $\hat{\mathbf{L}}_0$ will rotate more towards the $-\mathbf{w}$ direction in time until eventually $\lim_{t\to\infty} \hat{\mathbf{L}}_0 = \lim_{t\to\infty} \hat{\mathbf{e}} = -\hat{\mathbf{w}}$. Another way to see this: the path-point P acts as a rotational joint for the error vector $\mathbf{e}$, which is fixed at one end in P: the $\mathbf{v}_G$ is rotating the error vector in the same direction as the wind would rotate it, meaning that it will point instantaneously more in the anti-wind direction, i.e. even more outwardly with respect to the cone, until it reaches the anti-wind direction (the "torque" around point P is null at that point).
As in this case $\hat{\mathbf{L}}_0 = \hat{\mathbf{e}}$, the $\hat{\mathbf{L}}_0$ rotation must stop here. Since $f(0) = 0$, then also $\lim_{t\to\infty} \frac{\mathbf{u}_{\text{fast},2}}{k} = -\hat{\mathbf{w}}$ by construction. By hypothesis of geometrical case, this means $\lim_{t\to\infty} \hat{\mathbf{T}}_M(t) = -\hat{\mathbf{w}}$. Then, by definition of the normal acceleration command, also $\lim_{t\to\infty} \mathbf{a}_N^M = 0$ so we reach **asymptotic safety** as defined in 2.9.
As an additional feature, note that

$$\begin{aligned} &\text{sign}[(\mathbf{v}_G \times \mathbf{w}) \cdot \mathbf{k}](t_0) = \\ &\quad \text{sign}[(\mathbf{v}_G \times \mathbf{w}) \cdot \mathbf{k}](t), \ \forall t > t_0 \end{aligned} \tag{2.43}$$

so we reach the equilibrium without oscillations around that line such that $\hat{\mathbf{e}} = -\hat{\mathbf{w}}$.

Subcase 1. Finite length paths

In this case, the $\hat{\mathbf{L}}_0$ versor is a function of the particular path we are considering, so we can no longer assume it to coincide with $\hat{\mathbf{e}}$ as in the single point path case. However, consider the following two facts:

- The path is finite
- $\forall t \in \mathbb{R}, \ \ \mathbf{r}_M(t) \cdot \hat{\mathbf{w}} \ \geq \ \mathbf{r}_M(0) \cdot \hat{\mathbf{w}} + \underbrace{(w^\star - v_M^\star)}_{>0} t$

That is, as the wind is constantly stronger than the airspeed, the minimum growth of the projection of the error onto the wind direction has rate $(w^\star - v_M^\star)t$. This implies that

$$\lim_{t\to+\infty} \|\mathbf{r}_M(t)\| = +\infty \tag{2.44}$$

and since the path is finite

$$\lim_{t\to+\infty} \|\mathbf{d}(t)\| = \lim_{t\to+\infty} \|\mathbf{e}(t)\| = +\infty \tag{2.45}$$

As the distance grows to infinity, the path will look like a single point $\mathbf{P}_\infty$, that is the center of the smallest circle that contains the whole path. Then, we fall into the single point-path subcase.

2.5.2 Geometric case: infinite linear paths

Here the path is not finite. However, a common case in UAV applications is when the path is an infinite line. If this line is outside the wind-cone or the intersection with the cone is finite, it is not possible for the $\mathbf{v}_G$ to align to it. In this case, the proposed algorithm achieves **efficient wind stability**, i.e. the objectives in 2.40. To show this, simply notice that if $\mathbf{d} > \delta_{BL}$, then whatever the vehicle position, we have $\hat{\mathbf{L}}_0 \perp \hat{\Lambda}$, as the error direction will always be perpendicular to the line. A simulation for this situation is shown in Figure 2.13.

The interpretation for this result is that the proposed algorithm finds some efficient compromise for the $\mathbf{v}_G$ direction between the anti-wind direction and the path direction, which is a tradeoff between safety and tracking performance.

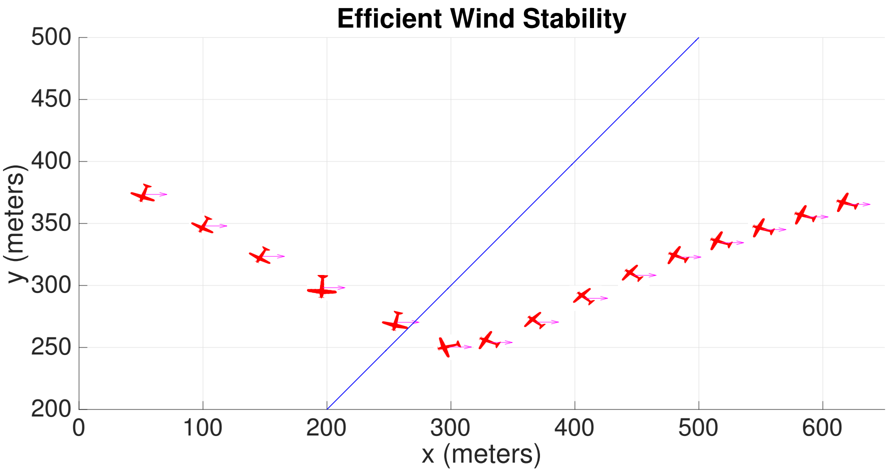

Figure 2.13: Airspeed is 14 m/s. Windspeed is 30 m/s, indicated by the magenta arrow on the aircraft. $\hat{\Lambda} = (\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2})$. Line direction is hence infeasible.

2.5.3 Dynamical case

Here we will extend the proof for the geometric case, so as to consider the dynamics imposed by the nonlinear acceleration command. As the subcase of finite-length paths was shown to fall into the subcase of single-point paths, studying the dynamic extension for the single-point paths is all we need. The extension for the infinite linear path is trivial and will be omitted, as the $\hat{\mathbf{L}}_0$ stops changing as soon as $\mathbf{d} > \delta_{BL}$.

In the following, it is clearer to directly refer to Figure (2.14) for the symbols definition.

Depending on the desired groundspeed direction $\hat{\mathbf{L}}_0$, we have two subcases.

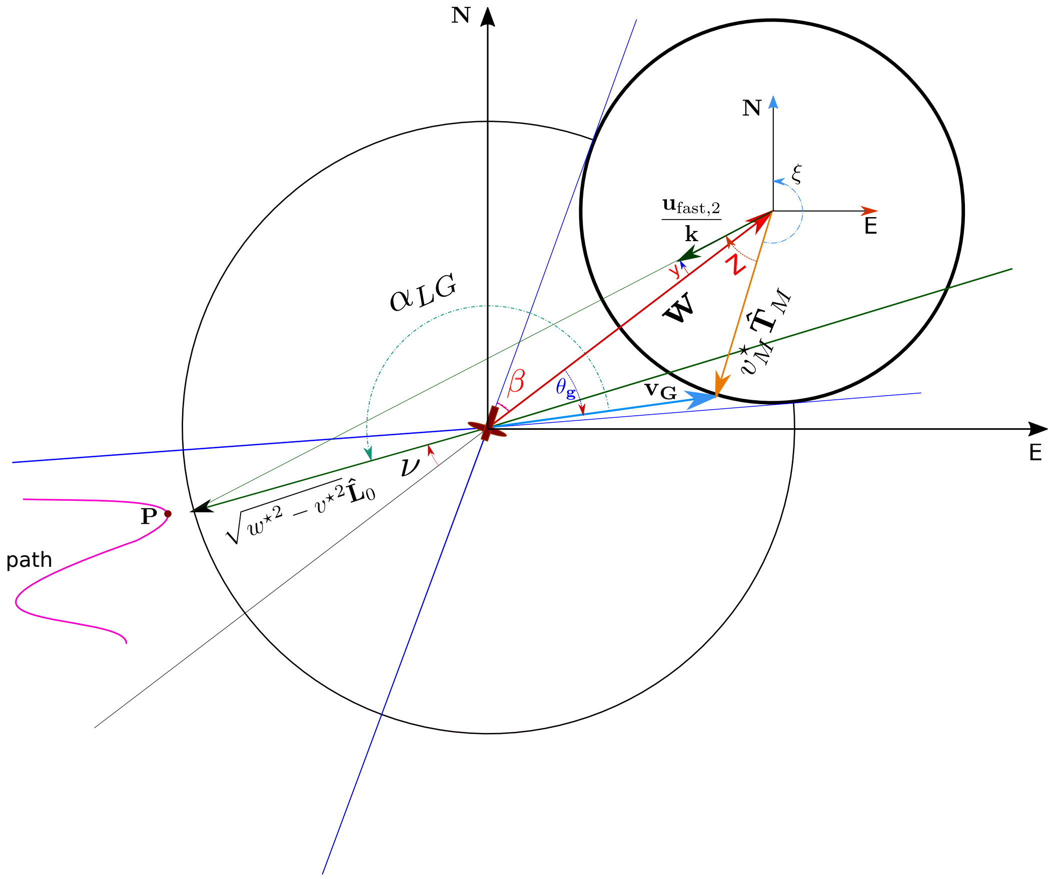

Figure 2.14: Symbols used in the proof

Subcase 1

If

$$\beta < \nu < \pi - \beta \tag{2.46}$$

corresponding to $\hat{\mathbf{L}}_0$ pointing outside of the "specular" cone, then it is easy to see that

$$\alpha_{Lg} < \pi, \ \forall \theta_g \tag{2.47}$$

meaning that

$$\dot{\nu} < 0 \tag{2.48}$$

independently from the actual aircraft orientation. This holds until we fall into subcase 2.

Subcase 2

If

$$0 < \nu < \beta \tag{2.49}$$

corresponding to $\hat{\mathbf{L}}_0$ pointing inside of the "specular" cone, we need further considerations. It is not true anymore that $\alpha_{Lg} < \pi,\ \forall \theta_g$. Instead we have

$$\begin{cases} \alpha_{Lg} < \pi, \text{ if } \theta_g > -\nu \\ \alpha_{Lg} \geq \pi \text{ otherwise} \end{cases} \tag{2.50}$$

Then it's possible that, depending on how the aircraft is oriented, ν will increase while the angle between $\hat{\mathbf{T}}_M$ and the commanded direction $\frac{\mathbf{u}_{\text{fast},2}}{k}$ is smaller than π, which is undesirable as it would mean the $\hat{\mathbf{L}}_0$ is "running away" from $\hat{\mathbf{T}}_M$.
To show that eventually the aircraft can be considered to be aligned with its commanded control input versor $\frac{\mathbf{u}_{\text{fast},2}}{k}$, consider the following:

- We can increase parameter k in order to make the vehicle turn with faster dynamics.
- As time goes to infinity, eventually the "chasing" angle z will decrease to 0.

To show this last fact, first notice that for any given $\hat{\mathbf{T}}_M$, if $\dot{\nu} > 0$ then $\dot{\nu}$ is a decreasing function of $|\mathbf{e}\cdot\mathbf{w}|$ that goes to 0 as $\frac{1}{|\mathbf{e}\cdot\mathbf{w}|}$ or faster. Indeed, consider the case when $\dot{\nu} > 0$ and has the maximum value, i.e. $\nu = 0$ and $\hat{T}_M \perp \mathbf{w}$. We have

$$\dot{\nu}_{\text{MAX}} = \frac{v_M^\star}{|\mathbf{e}\cdot\mathbf{w}|} \tag{2.51}$$

which acts as an upperbound for all the other situations. Irrespectively from $\hat{\mathbf{T}}_M$, since $w^\star > v_M^\star$, $|\mathbf{e}\cdot\mathbf{w}|$ indeed increases, hence $\dot{\nu}$ must decrease and tend to 0. Since y is a function of ν such that $\forall\ \nu,\ y(\nu) < \nu$, than also $\dot{y}$ decreases and tends to 0 as time goes to infinity. Now consider the time derivative of the "chasing" angle z

$$\dot{z} = \dot{y} + \dot{\xi} \tag{2.52}$$

Since we showed $\lim\limits_{t\to+\infty} \dot{\nu}(t) = 0 = \lim\limits_{t\to+\infty} \dot{y}(\nu(t))$, irrespectively of what the orientation of the vehicle could be at any time, then, as ξ indicates the heading angle of the aircraft,

$$\lim_{t\to\infty} \dot{z}(t) = \dot{\xi}(t) \tag{2.53}$$

As the acceleration command is designed to steer the vehicle orientation onto the chosen look-ahead vector, which now is $\frac{\mathbf{u}_{\text{fast},2}}{k}$, as the look-ahead is bound to asymptotically stop changing as the vehicle gets further away from the path, then we actually have that $\lim\limits_{t\to\infty} \dot{\xi}(t) = 0$, with the vehicle aligned to the look-ahead vector. This, together with the 2.53, translates in

$$\lim_{t\to\infty} z(t) = 0 \tag{2.54}$$

Then we can say that we asymptotically fall into the geometrical case, and the proof holds.

2.6 Continuity

In realistic scenarios, the wind is not going to be constant, but will likely switch between $w^\star < v_M^\star$ and $w^\star > v_M^\star$ several times. Not only that, the path is going to be curved, so the desired direction for the groundspeed $\hat{\mathbf{L}}_0$ is going to switch between being feasible and infeasible. All these switchings mean that it is very important for the command input $\mathbf{u}$ to be continuous to changing winds and changing $\hat{\mathbf{L}}_0$.

The control input was derived separately for the three subcases (slower winds, higher winds with feasible desired direction, higher winds with infeasible desired direction) in 2.3, 2.4.1, 2.4.3. We want to show here that the complete control input

$$\mathbf{u} = \begin{cases} \mathbf{u}_{\text{slow}}, & w^\star \le v_M^\star \\ \mathbf{u}_{\text{fast},1}, & w^\star > v_M^\star,\ \lambda \le \beta \\ \mathbf{u}_{\text{fast},2}, & w^\star > v_M^\star,\ \lambda > \beta \end{cases} \tag{2.55}$$

indeed guarantees continuity in this sense.

- **Switching between $\mathbf{u}_{\text{slow}}$ and $\mathbf{u}_{\text{fast,1}}$**: this happens as the wind passes from $w^\star < v_M^\star$ to $w^\star > v_M^\star$. Let t^* be the boundary time instant in which $w^\star(t^*) = v_M^\star(t*)$. Also, in this case, $\lambda_e(t^*) \le \frac{\pi}{2}$. Looking at the formulation for θ_s and θ_{s2} in (2.32), we have:

  $$\theta_{s2}|_{w^\star = v_M^\star} = \theta_s|_{w^\star = v_M^\star} \tag{2.56}$$

  and so $\mathbf{u}_{\text{slow}}(t^*) = \mathbf{u}_{\text{fast},1}(t^*)$

  so the command $\mathbf{u}(t)$ is continuous at this boundary condition

- **Switching between $\mathbf{u}_{\text{slow}}$ and $\mathbf{u}_{\text{fast,2}}$**: this happens as the wind passes from $w^\star < v_M^\star$ to $w^\star > v_M^\star$. Let t^* be the boundary time instant in which $w^\star(t^*) = v_M^\star(t*)$. Also, in this case, $\lambda_e(t^*) > \frac{\pi}{2}$. Solving the geometry in 2.5, we have that $\mathbf{u}_e = -\mathbf{w}$. Since $\beta(t^*) = \frac{\pi}{2}$, this implies that $y(t^*) = 0$ as computed in (2.38): so $\mathbf{u}_{\text{fast},2}(t^*) = -\mathbf{w}$ as well. Assuming $\hat{\mathbf{T}}_M \approx \hat{\mathbf{L}}_{1e}$, which is the case after some transient, we have that $\|\mathbf{v}_G\| \approx 0$. So by (2.28), we have $\theta_s \approx 0$, implying

  $$\mathbf{u}_{\text{slow}}(t^*) \approx -\mathbf{w} \approx \mathbf{u}_{\text{fast},2}(t^*) \tag{2.57}$$

- **Switching between $\mathbf{u}_{\text{fast,1}}$ and $\mathbf{u}_{\text{fast,2}}$**: this happens when $w^\star > \mathbf{v}_M$ and $\hat{\mathbf{L}}_0$ passes from being feasible to being infeasible. Let t^* be the boundary time instant. Then

  $$w^\star(t^*) \sin \lambda_e(t^*) = v_M^\star(t^*) \tag{2.58}$$

so $\theta_{s2}(t^*) = 0$. This guarantees continuity, as no shifting angle is applied in the infeasible case.

In Figure 2.15 we report a plot that highlights the continuity of the input as the wind increases and for a fixed $\hat{\mathbf{L}}_0$. We indicate the associated angles y for the $\mathbf{u}$ direction and ν for the $\hat{\mathbf{L}}_0$.

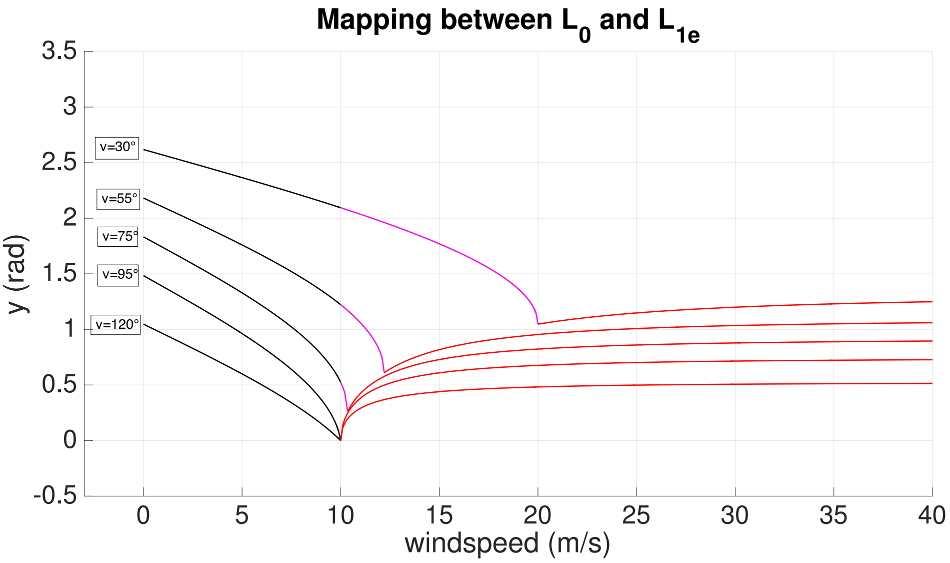

Figure 2.15: Switching $w^\star$ lower/higher than $v_M^\star$. Magenta: inside the cone.

2.6.1 Sinusoidal Winds

As an example of a more realistic varying wind profile, in order to show that the commands do not switch abruptly and are continuous, we consider the case of the wind having this sinusoidal profile

$$\mathbf{w}(t) = W \sin(\varOmega t) \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T \tag{2.59}$$

for some wind pulsation $\varOmega$ and amplitude $W > v_M^\star$. The result is shown in Figure 2.16, and the same is shown (more clearly) in an accompanying video[1].
Switching between any couple of the three parts of the control input can happen in this case. The least smooth behaviour, as we have only approximate continuity, is when the switching is between $\mathbf{u}_{\text{slow}}$ and $\mathbf{u}_{\text{fast},2}$.

2.6.2 Rotating Winds

The wind has the following expression

$$\mathbf{w}(t) = W \begin{bmatrix} \cos \varOmega t & \sin \varOmega t & 0 \end{bmatrix}^T \tag{2.60}$$

[1] Sinusoidal wind simulation: <https://www.youtube.com/watch?v=fpV5KkmrrUc>

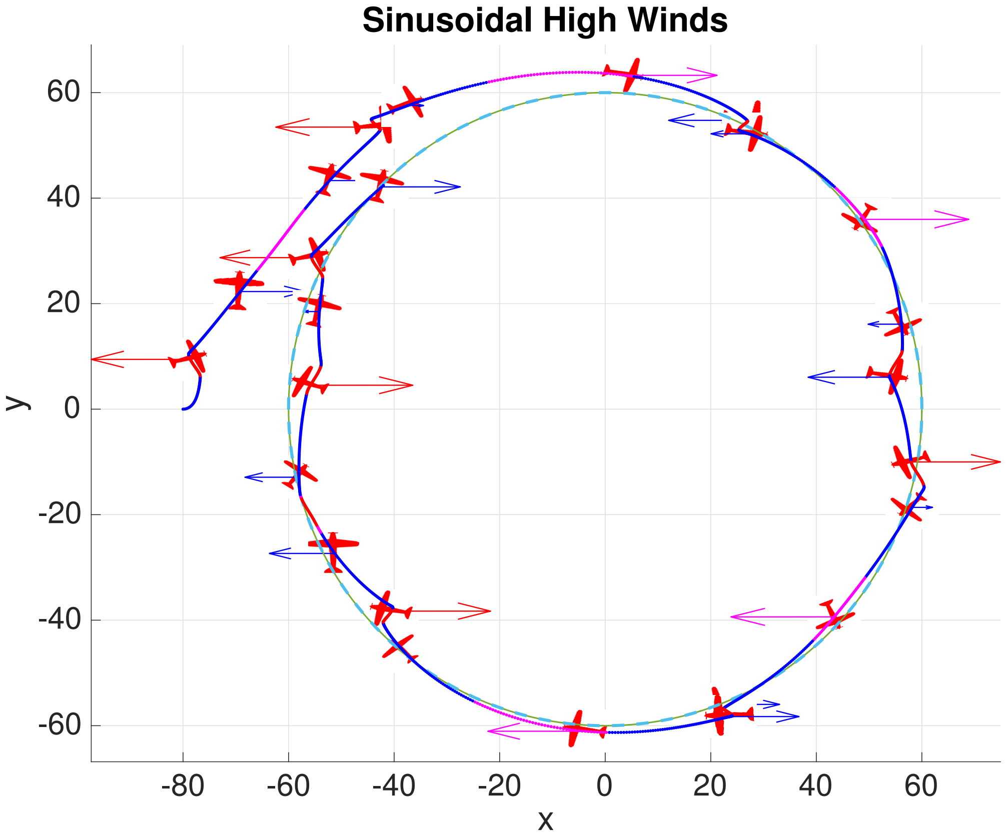

Figure 2.16: Sinusoidal winds. Blue: $\mathbf{u}_{\text{slow}}$ is applied. Magenta: $\mathbf{u}_{\text{fast},1}$ is applied. Red: $\mathbf{u}_{\text{fast},2}$ is applied

for some wind pulsation Ω and amplitude $W > v_M^\star$. When the direction becomes feasible, the vehicle exploits the wind so as to progress faster on the path while keeping the curvature. During the unfeasible direction parts, it tries to follow the progression of the circle. This also helps in making the required direction feasible again in a shorter time, as the wind rotates. This would be impossible with the old methods. A plot can be seen in Figure 2.17. An accompanying video is provided [2].

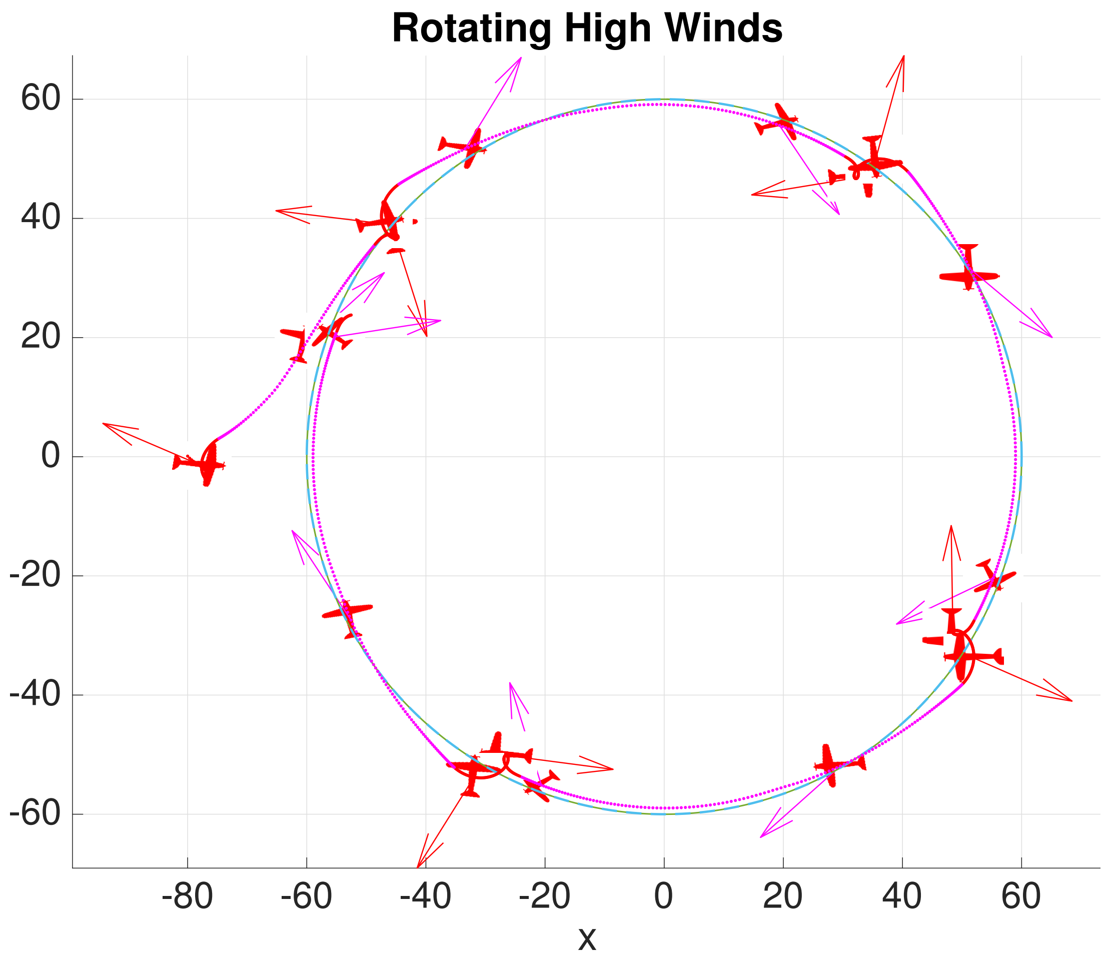

Figure 2.17: Rotating high winds. Magenta: faster wind, inside wind-cone. Red: faster wind, outside wind-cone

[2] Rotating wind simulation: <https://www.youtube.com/watch?v=AQWLYWPMvqk>

2.7 Roll Dynamics

The guidance algorithm was developed by considering the vehicle to behave as a point-mass, i.e. we assumed that each acceleration command from the nonlinear guidance could be matched perfectly and instantaneously by the vehicle. This is clearly not the case for a nonholonomic vehicle such as the fixed-wing UAV. In particular the nonlinear lateral dynamics of a fixed-wing UAV can be described in north-east coordinates as follows ([20]):

$$\begin{cases} \dot{n} = v_M^\star \cos\psi + w_n \\ \dot{e} = v_M^\star \sin\psi + w_e \\ \dot{\psi} = \frac{g\tan\phi}{v_M^\star} \\ \dot{\phi} = p \\ \dot{p} = b_0\phi_{\text{cmd}} - a_1 p - a_0\phi \end{cases} \tag{2.61}$$

where $\begin{pmatrix} w_e & w_n \end{pmatrix}^T = \mathbf{w}$ is the nonlinear wind effect, $g \approx 9.81$ is the acceleration of gravity, ψ is the heading of the aircraft (computed from the north-axis), ϕ is the roll angle that translates into ψ if we enforce the so called *coordinated turn hypothesis* that holds in steady-state flight ([21]), and the parameters a_0, a_1, b_0 describe the **roll-dynamics**, i.e. the dynamics between a roll command and the actual aircraft roll.

From the equations, we can deduce the transfer function between roll commands and actual roll:

$$\frac{\phi(s)}{\phi_{\text{cmd}}(s)} = \frac{b_0}{s^2 + a_1 s + a_0} \tag{2.62}$$

Higher order dynamics could be used, however it has been found in [20] through the identification procedures, that second-order fits appropriately the closed loop low-level autopilot attitude control response. The test-bed platform available in ASL-ETHZ is shown in Figure 2.18

Through the identification procedure outlined in [20], it was possible to identify the roll dynamics for the Techpod:

$$b_0 = 13.48 \qquad a_1 = 6.577 \qquad a_0 = 13.97 \tag{2.63}$$

In order to translate the acceleration $\mathbf{a}_M^N$ into a *roll command* ϕ_{cmd}, a simplified formulation was derived from the *coordinated turn hypothesis* ([21]):

$$\phi_{\text{cmd}} = \arctan\left(\frac{\|\gamma \mathbf{a}_N^M\|}{g}\right) \tag{2.64}$$

where g is the acceleration of gravity and γ is a parameter we can choose to achieve better performance. Usually $\gamma = 1$ is a proper choice.

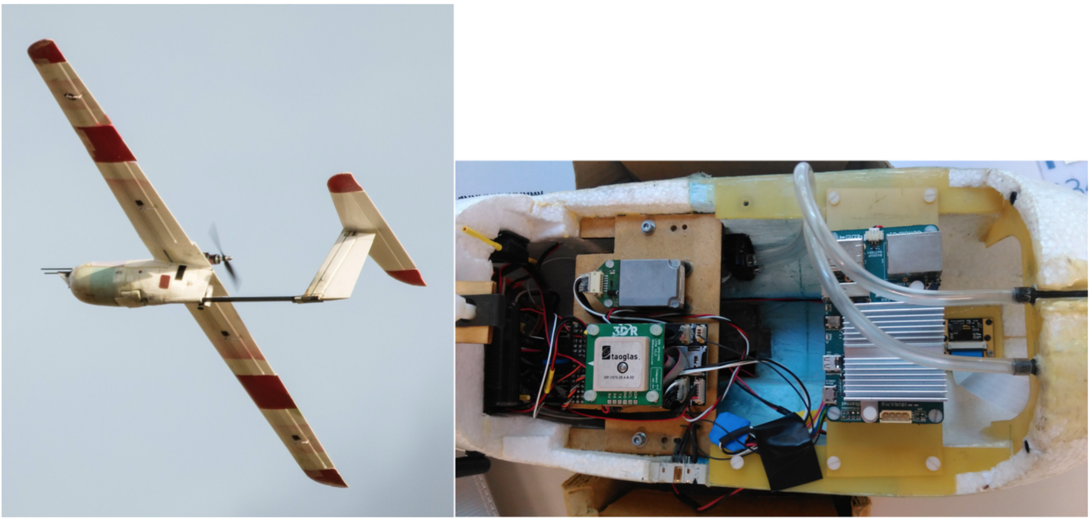

Figure 2.18: Fixed-Wing Test Platform: Techpod. Credits to [20]

With the real data at hand, some simulations were run to observe the performance of the nonlinear algorithm on the Techpod.
In figure 2.19, the wind is such that $w^\star < v_M^\star$, so we showed in 2.3.2 that the proposed guidance algorithm let the point-mass vehicle achieve the goals in 2.8, i.e. position/orientation/curvature convergence. As the algorithm does NOT consider the actual roll-dynamics directly, however, it is not possible to achieve perfect convergence. A delicate tuning of the parameters k and δ_{BL}, can partially take care of the roll-dynamics and make it possible to achieve very good performance. A simulation with the Techpod roll-dynamics and strong wind still such that $w^\star < v_M^\star$ is shown in Figure 2.19.
As to the case of $w^\star > v_M^\star$, the same considerations hold. Thanks to a fine tuning of the parameters it was possible to achieve good tracking performance and the objectives in 2.9. A simulation is shown in Figure 2.20.

2.8 Flight Results

The proposed algorithm was implemented on a Pixhawk Autopilot in C++, and thoroughly tested in HIL simulations. This was necessary to obtain a fine-tuning of the control parameters in the guidance law and be sure everything was right before actually flying. Subsequently, it was tested on a small fixed-wing UAV in high wind conditions. The Pixhawk Autopilot platform is shown in Figure 2.21. The test-bed platform aircraft for these flight tests was the so called "EasyGlider" (Figure 2.22, whose identification step gave the following minimum order roll-dynamics:

$$\frac{\phi(s)}{\phi_{\mathrm{cmd}}} = \frac{1.649}{s + 1.26} \tag{2.65}$$

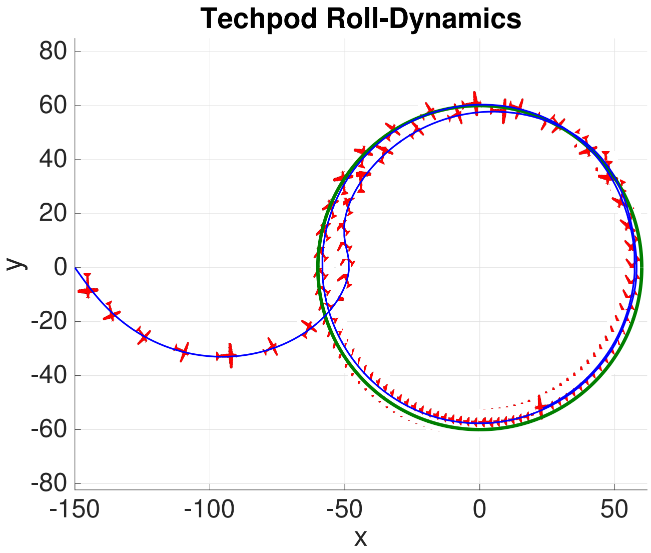

Figure 2.19: Wind is 10 m/s, flowing from left to right. Airspeed is 14 m/s. Techpod Roll Dynamics are considered.

After successful implementation, in Figure 2.23, 2.24, we show the results (taken directly from the stored data in the IMU of the aircraft) from the flight tests. The aircraft was commanded to follow a circular trajectory in counter-clockwise direction at a nominal airspeed of 8 m/s. The wind vector is represented in the figures using the following arrow, color scheme: $w^\star < v_M^\star$ (black), $w^\star > v_M^\star \cap (\hat{\mathbf{L}}_0$ feasible) (magenta), $w^\star > v_M^\star \cap (\hat{\mathbf{L}}_0$ infeasible) (red).

In Figure 2.23, the UAV can be seen to attempt curvature following despite the infeasible look-ahead direction until a point where the wind speed reduces and allows the start to convergence back to the path.

Figure 2.24 shows a wind-stabilized approach towards the trajectory until the point where simply pointing into the wind is the only option to reduce “runaway” from the track, recall the tracking direction is counter-clockwise.

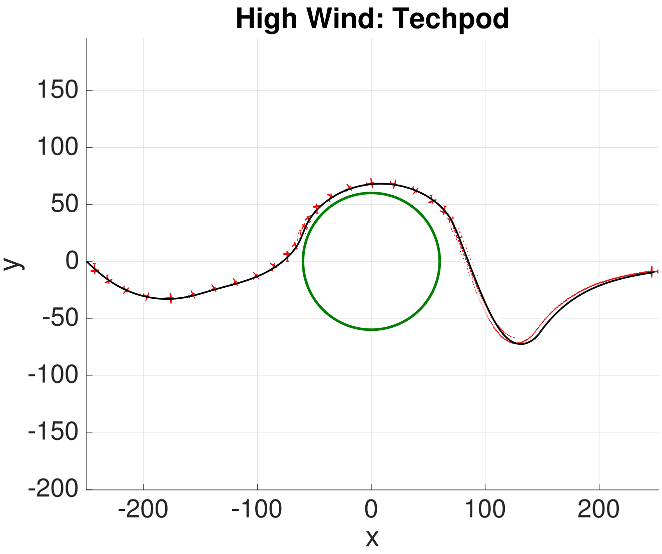

Figure 2.20: Wind is 15 m/s, flowing from left to right. Airspeed is 14 m/s. Techpod Roll Dynamics are considered.

Figure 2.21: The PixHawk PX4 platform. Credits to [3]

Figure 2.22: The EasyGlider PRO. Credits to http://www.green-eyes.it/Modellismo/EasyGlider.htm

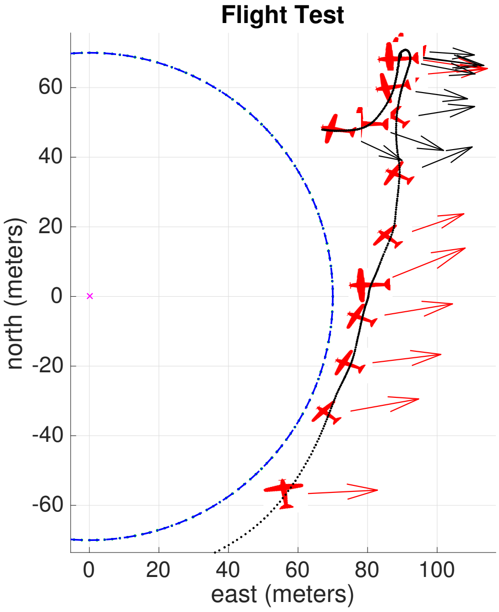

Figure 2.23: First moment from the flight test

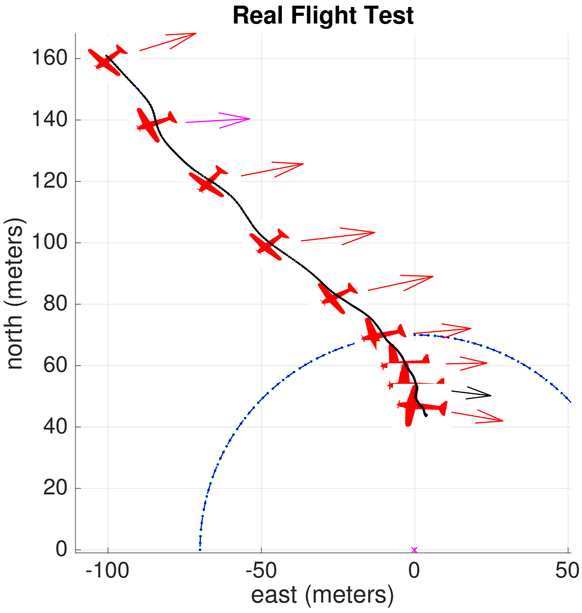

Figure 2.24: Second moment from the flight test

Chapter 3

A Guidance Approach Based On Model Predictive Control

In this chapter, we will investigate a completely different guidance approach. In chapter 2, we developed a novel nonlinear guidance algorithm able to cope with arbitrarily strong windfields: the solution is easy to implement and computationally very light, however the following should be noted.

- Having in mind the application to real fixed-wing UAVs, we cannot assume that the acceleration commands from the nonlinear guidance law will be matched perfectly by the (nonholonomic) vehicle. This will result in the necessity of fine-tuning the control parameters very carefully for each different aircraft and for each path, depending on the wind profile, which is doable but inconvenient in many situations.

- The tradeoff between performance and safety does not consider the minimization of a cost function, and relies on the particular choice we make for the mapping function 2.35. This means the tradeoff is built upon commonsense, and cannot be the mathematically best choice in every situation. We are giving up on optimality in exchange for simplicity and computational lightness.

For these reasons, it seemed natural to compare the performance of the proposed nonlinear algorithm with a controller based on an optimizer. As the model of the aircraft we can obtain is not perfect and the wind profile can vary in time, we chose to implement a Model Predictive Controller (MPC).

The MPC control framework is briefly outlined in 3.1. The application to fixed-wing UAVs is described in 3.2. After showing some preliminary simulation results in 3.3, the reshaping of the cost function to envision the possibility of winds such that $w^{\star} > v_{M^{\star}}$ and obtain the safety objectives in 2.9 is described in 3.4. Finally, simulation results are shown in 3.5.

3.1 The (Nonlinear) MPC Framework

The introduction about MPC given in this section is partially quoted from the Preface of the book [22], as it is a very clear introduction to the problem.

Dynamic optimization has become a standard tool for decision making in a wide range of areas. The basis for these dynamic optimization problems is a dynamic model in the form

$$\dot{x}(t) = f(x(t), u(t)), \quad x(0) = x_0 \tag{3.1}$$

that describes the evolution of state $x(t)$ in time as it is influenced by a control input $u(t)$. In general, function $g(\cdot)$ is a nonlinear function of its arguments. The goal of the dynamic optimization is to find, at each time τ, that control input $u([\tau, \tau + N])^\star$ (where $[\tau, \tau + N]$ indicates the time interval between τ and $\tau + N$) such that some objective function is optimized over some time horizon $[\tau, \tau + N]$, for some chosen N.
Usually, the minimization of the objective function takes the following form:

$$\min_{u(\tau,\tau+N)} \int_{\tau}^{\tau+N} q(x(t), u(t))dt \quad + \quad p(x(\tau + N)) \tag{3.2}$$

The terms $q(x, u)$ and and $p(x)$ are referred to as the *stage cost* and *terminal cost*. Many realistic problems can be put into this framework and a lot of algorithms and software packages are available to come up with the optimal input $u([\tau, \tau + N])^\star$. Even large problems described by complex models and involving many degrees of freedom can be solved efficiently and reliably.
To this point, it seems that finding the proper $u([\tau, \tau + N])^\star$ is everything we need: however, this structure still lacks some feedback to compensate for model imprecision and external disturbances, which combined could make the system deviate from its predicted future behaviour. This would ruin the procedure and make it unreliable.
For this reason, it is common practice to measure the state after some time period, say one time step, and to solve the dynamic optimization problem again, starting from the measured state x$(\tau{+}T_s t)$ (where T_s is the time needed for two consecutive sensing steps) as the new initial condition. This feedback of the measurement information to the optimization endows the whole procedure with a robustness typical of closed-loop systems.

A big limitation of MPC is that running the optimization procedure online at each time step requires a huge amount of time and computational resources. However, as explained in [22], "today, fast computational platforms together with advances in the field of operation research and optimal control have enlarged in a very significant way the scope of applicability of MPC to fast-sampled applications. One approach is to use tailored optimization routines which exploit both the

structure of the MPC problem and the architecture of the embedded computing platform to implement MPC in the order of milliseconds.
The second approach is to have the result of the optimization pre-computed and stored for each state in the form of a look-up table or as an algebraic function $u(t) = g(x(t))$ which can be easily evaluated. Whatever approach we may choose, the optimization guidance is not easy to implement and generally require a great amount of on-board computational power."

3.2 The MPC Guidance Approach For Fixed-wing Vehicles

Several control schemes based on Model Predictive Control were developed in the literature. In [23], a nonlinear model predictive controller (NMPC) is built on the kinematic fixed-wing model of the UAV, in order to track a desired line, and a stability analysis is performed. In [24], this approach is extended to tackle the case of multiple line segments with obstacle avoidance. In [25], the problems inherent to real-time implementation are tackled: an adaptive-horizon approach is proposed based on the curvature of the target path. An alternative approach based on model predictive control and backstepping techniques is proposed in [26].
The case of windy scenarios is directly tackled in [27], but the wind is supposed to be slower than the aircraft airspeed. We are indeed interested in tackling the case of arbitrary windspeeds.

To test the algorithm which will be proposed in the following, we took the framework described in [20]. There, the formulation and implementation of a Nonlinear MPC controller for general high-level fixed-wing lateral-directional trajectory tracking was developed. The ACADO Toolkit [28] for generation of a fast C code based nonlinear solver was used.

The optimization problem takes the following form

$$\begin{aligned}\min_{u,x} \int_{t=0}^{T} & \Big((y(t) - y_{\text{ref}}(t))^T Q_y \, (y(t) - y_{\text{ref}}(t)) + (u(t) - u_{\text{ref}}(t))^T R_u \, (u(t) - u_{\text{ref}}(t)) \Big) \, dt \\ & + (y(T) - y_{\text{ref}}(T))^T P \, (y(T) - y_{\text{ref}}(T)) \\ & \text{subject to} \quad \dot{x} = f(x, u) \\ & \qquad\qquad\quad\; y = h(x, u) \\ & \qquad\qquad\quad\; u(t) \in \mathbb{U} \\ & \qquad\qquad\quad\; x(0) = x_(t_0)\end{aligned} \tag{3.3}$$

where $h(x, u)$ is the selection function that maps states and inputs into the *penalization variables* that appear into the objective function. The choice of the proper penalization variables for the application at hand is very important, as we will see in what follows. Considering the aircraft dynamics in 2.61, we come up with the state vector $x = \begin{bmatrix} n & e & \psi & \phi & p \end{bmatrix}^T$ and the control input $u = \phi_{\text{cmd}}$. As the set of achievable rolls is bound, then $\mathbb{U}$ denotes the (convex) set of possible inputs, which is directly taken into account into the optimization problem.

The standard approach is to create the following error variables

$$\begin{aligned} e_t &= \mathbf{e} \cdot \hat{\mathbf{N}}_P \\ e_\chi &= \chi_d - \chi \end{aligned} \tag{3.4}$$

where $\mathbf{e}$, $\hat{\mathbf{N}}_P$ are the same as defined in 2.2, $\chi_d = \arctan 2\left(\hat{\mathbf{T}}_{P_e}, \hat{\mathbf{T}}_{P_n}\right)$, $\chi = \arctan 2(\mathbf{v}_{G_e}, \mathbf{v}_{G_n})$. So, e_t represents the *track-error* and e_χ represents the *course error*, i.e. how much the groundspeed direction is different from the tangent direction to the path at its closest point.

Having defined these error variables, the usual choice for the penalization variables vector is the following:

$$y = \begin{bmatrix} e_t & e_\chi & \phi & p & \phi_r \end{bmatrix}^T \tag{3.5}$$

By a proper choice of the weight matrices Q_y, R_u, P it is then possible to achieve convergence to arbitrarily shaped paths, provided the maximum curvature does not exceeds the minimum turn-radius of the aircraft at its maximum speed (which is when the wind is in favour of the aircraft).

However, the ratio of the weights inside matrix Q_y is tricky to be chosen, and depends on both the path shape and the wind profile.

Taking inspiration from the Look-Ahead method in [1], in order to have a simpler tuning, we created the following penalization variable:

$$e_{\hat{\mathbf{L}}} = \arctan 2\left(\hat{\mathbf{L}}_e, \hat{\mathbf{L}}_n\right) - \chi \tag{3.6}$$

where $\hat{\mathbf{L}}$ is chosen as in 2.13, and considered the following penalization variables vector:

$$y = \begin{bmatrix} e_{\hat{\mathbf{L}}} & \phi & p & \phi_r \end{bmatrix}^T \tag{3.7}$$

By doing so, we are actually dividing the tuning step in two parts:

- Tune the parameters k, δ_{BL} for the Look-Ahead guidance, in order to define the desired behaviour we want to see the aircraft perform.
- Tune matrices Q_y, R_u, P, but this is much easier than before as we don't have to find the correct ratio between cross-track and course errors.

So, now the MPC control problem is set. The wind is considered to be constant for all the prediction horizon. In what follows, we investigate the performance of the MPC controller we have set on the Techpod aircraft, for a loitering mission, for different windfields

3.3 Simulation Results: A First Attempt

First, the wind is considered to be slower than the airspeed ($w^\star < v_M^\star$). We considered a time-horizon $N = 50$, where each step is $0.1s$ long. This means the optimizer considers a time-frame of 5 seconds in the future, which is reasonable for fixed-wing UAVs as their dynamics are quite slow.
The result of the simulation is shown in Figure 3.1.

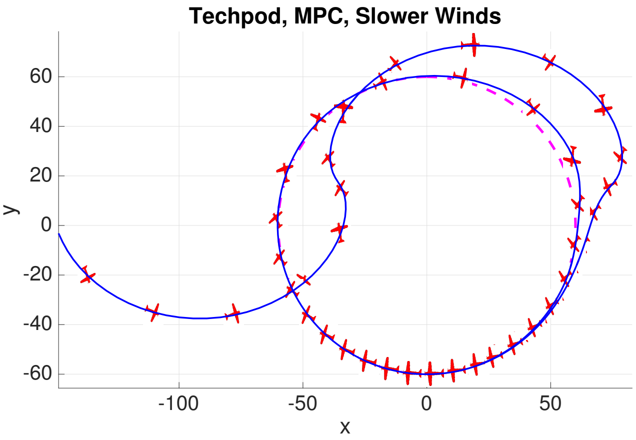

Figure 3.1: Wind is $8m/s$, directed from left to right, airspeed is $14m/s$

Almost perfect convergence to the path is achieved by the MPC controller. When it is not the case, it simply means that it's physically impossible for the vehicle to keep the curvature, as the minimum turn-radius is lower bounded by the input set $\mathbb{U}$. The behaviour is also generally very smooth.

Now, the wind is considered to be faster than the airspeed ($w^\star > v_M^\star$). as before, we considered $N = 50$. In this case, as the vehicle is bound to leave the path behind, we obtained bad behaviours by considering the penalization variables in 3.6. An example of such behaviours is shown in Figure 3.2.
The explanation for such behaviour, is that the $\hat{\mathbf{L}}$ vector changes its direction depending on the vehicle position, even as we are going far away from the path.

As the only goal for the optimizer is to align the groundspeed to the look ahead vector, the best choice seems to try and follow the $\hat{\mathbf{L}}$ "swinging" by letting the vehicle "swing" in return. As the prediction horizon is limited, the swing direction changes from time to time.

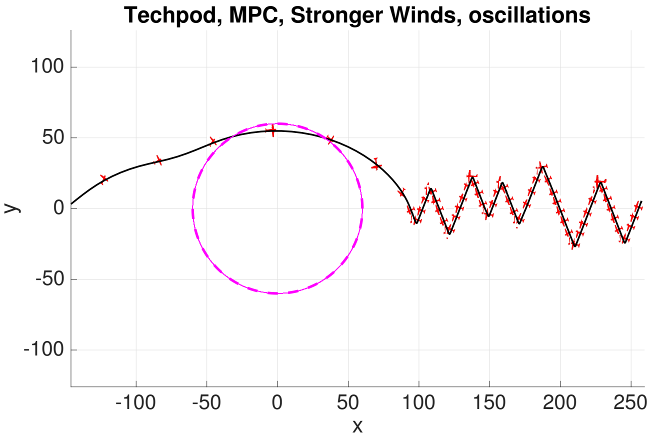

Figure 3.2: Wind is $15m/s$, directed from left to right, airspeed is $14m/s$

The first solution that comes to mind in order to avoid this unpredictable and undesirable behaviour, is to insert again the *cross-track* error e_t into the penalization variables vector, and slightly penalize it through matrix Q_y. In this case we obtain the behaviour highlighted in Figure 3.3
As it is easy to notice, the behaviour is quite "sketchy" as the vehicle gets away from the path following the wind. This is due to the fact that the e_t variable brings in more weight as the distance to the path grows, so until its value is not large enough the $e_{\hat{\mathbf{L}}}$ error variable penalization will try to enforce the oscillating behaviour we observed in Figure 3.2.

The conclusion is that the *look-ahead error* as it was defined in 3.6 is not suitable for the case of $w^{\star} > v_M^{\star}$. In what follows we reshape the error variable to be penalized in order to obtain satisfactory behaviour as the vehicle flows away with the wind.

3.4 Reshaping Of The Penalization Variable

As we would like to achieve the **asymptotic safety** objective in 2.9, the idea to obtain this behaviour through MPC was to also penalize the norm of the groundspeed, and give up on following the $\hat{\mathbf{L}}$ when the vehicle is flowing away from the

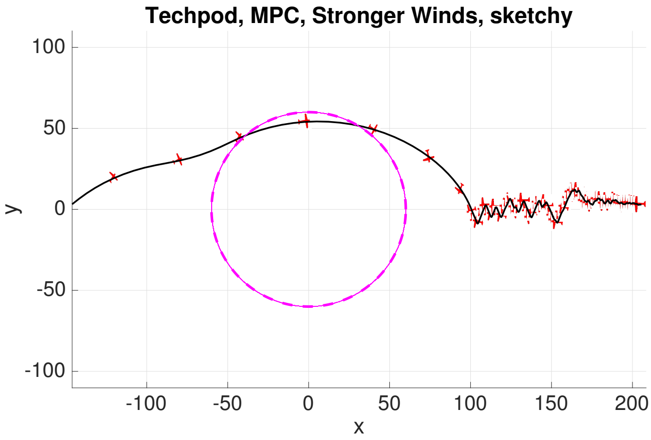

Figure 3.3: Wind is $15m/s$, directed from left to right, airspeed is $14m/s$

path together with the wind.

As with an MPC approach it is difficult to define a complex behaviour only through the shaping of cost functions and penalization variables, this task was tricky to tackle. In particular, it was necessary to define a new penalization variable $e^1_{\hat{\mathbf{L}}}$ as follows

$$e^1_{\hat{\mathbf{L}}} = \begin{cases} (1 - \text{sat}\left(\frac{\|\mathbf{e}\|}{\delta_w}\right))e_{\hat{\mathbf{L}}} + \\ + \text{sat}\left(\frac{\|\mathbf{e}\|}{\delta_w}\right)(e_{\|\mathbf{v_G}\|}), & \text{if } w^\star > v^\star_M \text{ and } (\mathbf{r}_M - \mathbf{r}_C) \cdot \mathbf{w} > 0 \\ \\ \\ e_{\hat{\mathbf{L}}}, & \text{otherwise} \end{cases} \tag{3.8}$$

where $\mathbf{r}_C$ indicates the vector to the center of the circle, δ_w is a boundary parameter to be shaped and $e_{\|\mathbf{v_G}\|} = \|\mathbf{v}_G\| - w^\star + v^\star_M$. In case we are not dealing with a circular path, $\mathbf{r}_C$ indicates the center of the smallest circle that encloses the path.
To say it in words, the old look-ahead error variable $e_{\hat{\mathbf{L}}}$ is modified when the following two conditions hold together:

- The wind is stronger than the airspeed $w^\star > v^\star_M$.
- The vehicle is leaving the path behind, with respect to the wind direction ($(\mathbf{r}_M - \mathbf{r}_C) \cdot \mathbf{w} > 0$). As the wind is considered to be non-stopping, this

means that from now on the vehicle is bound to flow away with the wind indefinitely by leaving the path behind it.

When these two conditions are verified, we need to shape the cost function so that the minimization leads to the desired behaviour of **asymptotic safety** described in 2.9. The way to obtain this is to penalize the norm of the groundspeed ($e_{\|\mathbf{v}_G\|}$) and gradually stop considering the $\hat{\mathbf{L}}$. In order to obtain the third objective in 2.9, we also need to slightly penalize the cross-track error e_t. If the vehicle is near the path, though, we still want to preserve tracking performance. In order for it to happen, we define $e^1_{\hat{\mathbf{L}}}$ as a tradeoff between $e_{\hat{\mathbf{L}}}$ and $e_{\|\mathbf{v}_G\|}$ as in the first part of 3.8: the more the distance grows, the more we stop considering tracking performance and try to reach **asymptotic safety**. If the distance grows over a boundary parameter δ_w, whose choice is a degree of freedom, we stop to consider $e_{\hat{\mathbf{L}}}$ completely.

To sum it up, the penalization variables vector will be defined as follows

$$y = \begin{bmatrix} e_t & e^1_{\hat{\mathbf{L}}} & \phi & p & \phi_r \end{bmatrix}^T \tag{3.9}$$

Notice that the vector defined in 3.9 is the same as the one in 3.7 if $w^\star < v^\star_M$ and we keep the part of Q_y that acts on e_t to 0. Also notice that if the part of Q_y that acts on e_t is kept to a very low value different from 0, the behaviour will almost be exactly the same for the case of $w^\star < v^\star_M$: the cross-track error e_t gets to 0 as soon as the vehicle reaches the path, and will keep being almost 0 forever. This is the reason why using the penalization variables vector as in 3.9, with Q_y kept small in the part that acts on e_t, is the correct choice even in the case of changing winds.

3.5 Simulation Results: Tackling Strong Winds

We report here some simulation results with the penalization variables defined as in 3.4.
In Figure 3.4, the aircraft started already heading against the wind. As performing a considerable turn would result in huge groundspeed, that would last for quite some time as the roll-dynamics of the Techpod are quite slow, the optimizer understands that the best thing to do to achieve tracking performance is performing small and slow turns and follow the path by going backwards.
This kind of behaviour was not possible to obtain with the nonlinear-geometric approach, as the model was not considered in the equations (so performing huge turns was not considered to be a problem by the guidance controller), and the continuity constraints on changing winds were not consistent with a behaviour similar to the one shown in Figure 3.4.

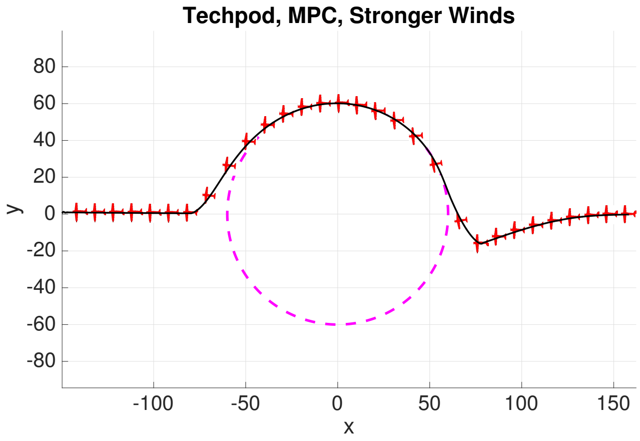

Figure 3.4: Wind is $15m/s$, directed from left to right, airspeed is $14m/s$

A similar behaviour is shown in Figure 3.5, the only difference being that the aircraft is starting headed down. Even though the aircraft is forced to perform a huge turn while the roll-dynamics are very slow, the optimizer still manages to understand that the best thing to do is to follow the circle backwards.

In Figure 3.6, however this was not the case: getting the airplane to turn by 180 degrees in order to follow the path backwards was not convenient, as that would have resulted in a oddly-shaped and suboptimal vehicle path. So, the optimizer way of reasoning is still consistent.

These simulations show some problems inherent to the MPC approach:

- The resulting behaviour is extremely difficult to predict. This is because we are trying to enforce a complex behaviour by only acting on penalization variables, without knowing what the optimizer will actually decide in each situation. We are left with a lot of subcases to be taken care of one by one.
- We might lose smoothness of the behaviour if the wind is changing between $w^\star < v_M^\star$ and $w^\star > v_M^\star$. In particular, as we've seen in Figures 3.4 and 3.5, the optimizer might choose to let the aircraft follow the path backwards when $w^\star > v_M^\star$. As soon as $w^\star < v_M^\star$, going backwards is not an option anymore: this will result in the optimizer being forced to ask the aircraft to abruptly change its heading, resulting in non-smooth behaviour to changing winds.

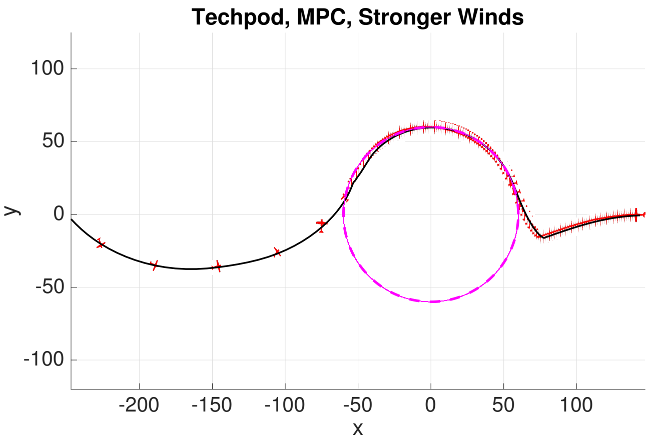

Figure 3.5: Wind is $15m/s$, directed from left to right, airspeed is $14m/s$

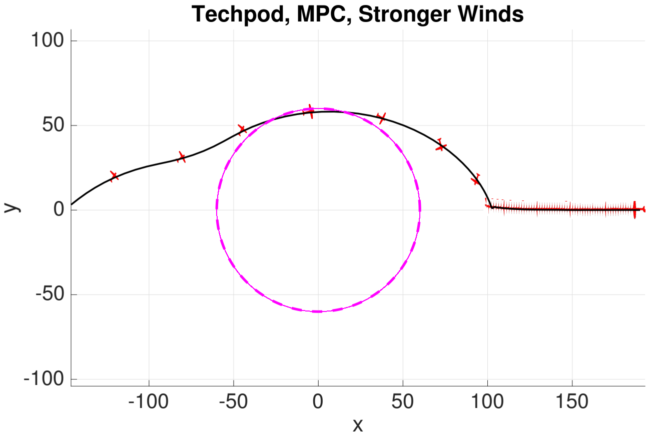

Figure 3.6: Wind is $15m/s$, directed from left to right, airspeed is $14m/s$

Chapter 4

Comparison Of Performance and Conclusions

Purpose of this chapter is to draw some conclusions about the advantages and disadvantages of the two guidance approaches presented in chapters 2 and 3. Even though some of them were already highlighted through this work, it is worth trying to compare quantitatively the performance: this is the goal of the next section. In the last section, future work will be discussed.

4.1 Spatial Performance

As the main application for fixed-wing UAVs in windy scenarios is surveillance and inspection of large areas, the mission to be accomplished is defined "spatially": our main concern is to cover the given area as precisely as possible, by tight tracking of the given path. In other words, the *vehicle path* should correspond to the *target path* as precisely as possible, without defining time-constraints. This means we are not concerned about following the track at a given inertial speed, as we just need to use the on-board camera to photograph certain areas from the correct viewpoints.

The most natural way to compare the performances of the nonlinear-geometric approach and the MPC approach could seemingly be to compute the following *time-wise average distance*:

$$\mu_{\text{avg,time-wise}} = \frac{\int_0^{T_f} \|[\mathbf{r}_M(t) - \mathbf{r}_P(t)]\| dt}{T_f} \tag{4.1}$$

where T_f is the final time for the flight, $\mathbf{r}_M$ and $\mathbf{r}_P$ defined as in 2.2. We can compare the results for this parameter for the nonlinear-geometric and MPC algorithms: a lower value would indicate a better tracking performance.
As explained above, however, we are not concerned about following the path at a

given speed, so the changing $\|\mathbf{v}_G(t)\|$ over time would influence the computation and lead to deceiving results.

Instead, the following *space-wise average distance* parameter was considered:

$$\mu_{\text{avg,space-wise}} = \frac{\int_0^L \|[\mathbf{r}_M(l) - \mathbf{r}_P(l)]\| dl}{L} \tag{4.2}$$

where $L = \int_0^{T_f} \|\mathbf{v}_G\|(t)dt$ is the length of the vehicle path. In particular, notice that $dl = \|\mathbf{v}_G\|(t)dt$.
The parameter in 4.2 is better suited to compare the spatial performance of the two algorithms.

We can see in Figure 4.1 the comparison of performance between the MPC and the NL/Geometric Guidance applied on the Techpod, and lastly the performance of the NL/Geometric Guidance applied on a point mass.

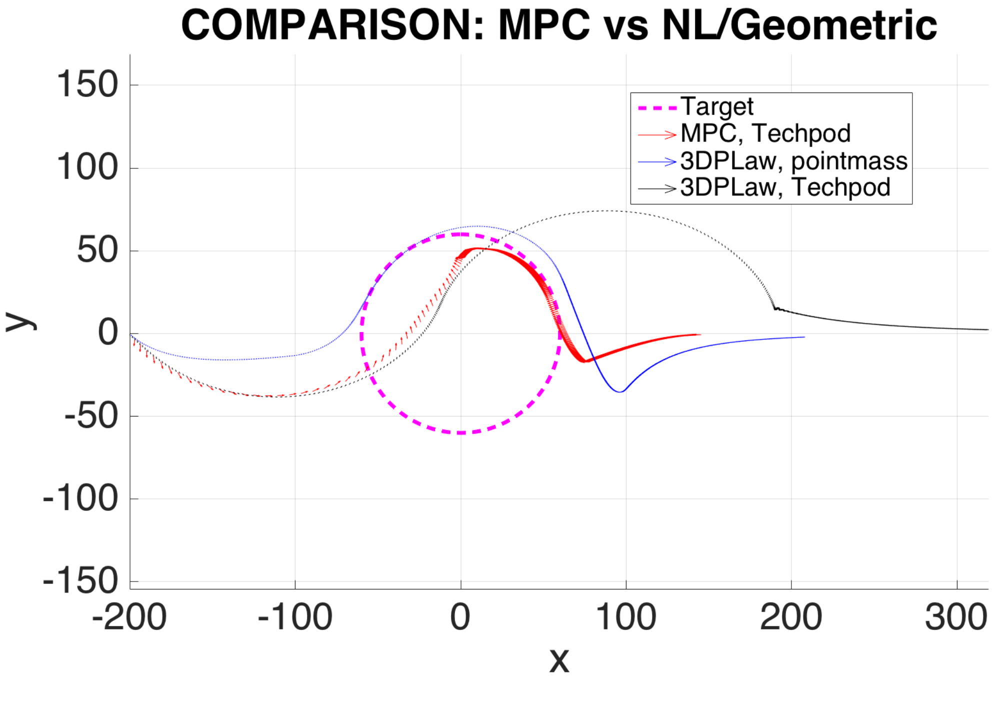

Figure 4.1: Comparison of Performance: MPC and NL/Geometric law on Techpod, NL/Geometric law on PointMass. Windspeed is $16m/s$, airspeed is $14m/s$.

As it was easy to predict, the NL/Geometric approach does not take the roll-dynamics of the Techpod into account, resulting in lower tracking performance.

The MPC understands that the best thing to do is to follow the circle backwards. The performance for the NL/Geometric approach applied on a point mass is pretty good, as the discrepancy caused by the roll-dynamics is not present.

The *space-wise average distance* parameter $\mu_{\text{avg,space-wise}}$ was computed in this case for the three vehicle trajectories. To do it numerically without having to solve the integral in 4.2, the procedure was to sample the distance from the vehicle path to the circle from a large number of equally spaced points on the vehicle path. This is highlighted in Figure 4.2, together with the obtained values for the $\mu_{\text{avg,space-wise}}$

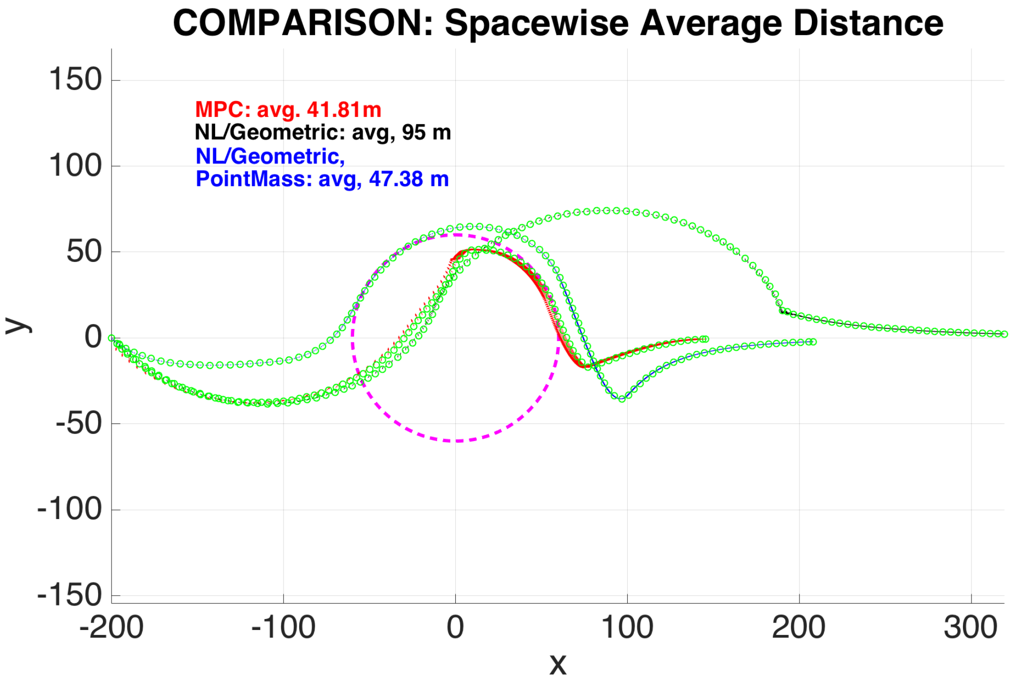

Figure 4.2: Comparison of Performance: MPC and NL/Geometric law on Techpod, NL/Geometric law on PointMass. Windspeed is $16m/s$, airspeed is $14m/s$. The trajectories are the same as those in Figure 4.1. The green circles, equally spaced, indicate the points from where the distance from the circle was sampled.

Some observations:

- Tracking performance is way better achieved with an MPC approach, as the roll-dynamics of the aircraft are directly taken into account through an optimization procedure.

- The performance for MPC is similar to the NL/Geometric guidance performance, in case the pointmass dynamics are considered. This means that the advantage of the optimization almost completely lies in the model compensation. The logical tradeoff between safety and performance is then achieved through the NL/Geometric guidance.

4.2 Conclusions

In chapter 2 a novel nonlinear guidance method, taking arbitrarily strong wind into account, was developed. The approach showed good performance and coherent behaviour, and was also validated through real flight tests in windy scenarios. The main **advantages** inherent to the approach are:

- Convergence to path, i.e. fulfillment of the objectives in 2.8, whenever $w^\star < v_M^\star$, when the roll-dynamics are neglected.
- Good tracking performance, until it's possible, when $w^\star > v_M^\star$. Convergence to a safe configuration (i.e. **asymptotic safety** as formulated in 2.9) and smooth behaviour.
- Continuity to varying winds, in every situation, as shown in 2.6.
- Computationally light. A single list of computations is required, in order to come up with the control input to the system. The tradeoff is found without any optimization procedure.

That said, there are some disadvantages:

- The tuning of the parameters, especially when the roll-dynamics are considered, is very tricky and delicate if we want to achieve good performance. This is because we have to adapt to the particular path shape, wind profile, and roll-dynamics of the vehicle at hand.
- The fact that we can't take the roll-dynamics into consideration directly (but only through the choice of parameters k and δ_{BL}), generally results in poor tracking performance in the case $w^\star > v_M^\star$.

As to the MPC Guidance approach, the good sides are:

- Convergence to path when $w^\star < v_M^\star$, even when the roll-dynamics are considered, as far as it is physically possible.
- Good tracking performance, until it's possible, when $w^\star > v_M^\star$, and convergence to a safe configuration (2.9).
- Cost function tuning is easy. No need to fine tune with respect to any particular wind profile.

The downsides inherent to MPC are:

- Computationally very heavy. A successful real-time implementation requires sufficient computational power onboard (which is not always the case) and tailored efficient implementation of the optimization procedure. Complex implementation.
- It is difficult to impose any complex behaviour in extreme scenarios, such as when $w^\star > v_M^\star$. In order to achieve the goals in 2.9 it was necessary to define a complex penalization variable $e^1_{\hat{\mathbf{L}}}$.
- As we are not defining mathematically the control inputs with respect to time, it is difficult to predict in advance the actual behaviour of the vehicle. The more the penalization variables are complex, the more we will incur in unpredicted behaviours: consider for example the different behaviour shown in Figures 3.5 and 3.6.
- With our particular MPC implementation, we may lose continuity to changing winds: the option of “following the path backwards” is not available when $w^\star < v_M^\star$, so if the wind changes intensity we might experience a sudden aircraft turn.

4.3 Future Work

Several extensions are possible for future work.

As one of the main innovations introduced by the non-windy guidance algorithm shown in [1], which we took inspiration from, was the extension to three-dimensional paths, it will be necessary to consider three-dimensional paths in the windy scenario as well. Not only that, the wind could also have a vertical component (ascensional currents) which we might need to take care of for acrobatic maneuvers and tight tracking of complex 3D paths.

A mathematical proof of convergence to the goals in 2.8 for the windy algorithm in case $w^\star < v_M^\star$ should be provided.

As one of the main drawbacks of the NL/Geometric approach was the delicate tuning, an automatic procedure to come up adaptively with parameters k and δ_{BL} should be thought of, at least in an approximate form.

As the normal-acceleration commands are at the present state translated into roll/pitch commands by taking advantage of approximate and simplified formulations that neglect dynamic couplings, better performance could be achieved by implementing a *meta-optimization* procedure that finds the most proper roll/pitch commands by considering the full nonlinear aircraft dynamics

An in-depth analysis of the stability and the hidden unpredicted behaviours inherent to the MPC approach should be enforced. This would allow to come up with a simpler formulation for the penalization variables to obtain the desired behaviours.

Bibliography

[1] C. Namhoon, K. Youdan, and P. Sanghyuk, “Three-dimensional nonlinear differential geometric path-following guidance law,” *Journal of Guidance, Control, and Dynamics*, 2015.

[2] L. Furieri, T. Stastny, L. Marconi, R. Siegwart, and I. Gilitschenski, “Gone with the wind: Nonlinear guidance for small fixed-wing aircrafts in arbitrarily strong windfields,” *arXiv*, 2016.

[3] Pixhawk Autopilot, 2015, http://pixhawk.org/.

[4] P. Oettershagen, T. Stastny, T. Mantel, A. Melzer, K. Rudin, P. Gohl, G. Agamennoni, K. Alexis, and R. Siegwart, *Long-Endurance Sensing and Mapping Using a Hand-Launchable Solar-Powered UAV.* Cham: Springer International Publishing, 2016, pp. 441–454.

[5] P. Oettershagen, A. Melzer, T. Mantel, K. Rudin, T. Stastny, B. Wawrzacz, T. Hinzmann, S. Leutenegger, K. Alexis, and R. Siegwart, “Design of small hand-launched solar-powered UAVs: From concept study to a multi-day world endurance record flight,” *Journal of Field Robotics*, 2016.

[6] N. Yoshitani, “Flight trajectory control based on required acceleration for fixed-wing aircraft,” *Proceedings of the 27th International Congress of the Aeronautical Sciences*, 2010.

[7] I. Rhee, S. Park, and C. Ryoo, “A tight path following algorithm of an uas based on pid control,” *Proceedings of the SICE Annual Conference*, 2010.

[8] A. Ratnoo, P. Sujit, and M. Kothari, “Adaptive optimal path following for high wind flights,” *Proceedings of the 18th IFAC World Congress*, 2011.

[9] L. Cunjia, M. Owen, and C. Wen-Hua, “Path-following control for small fixed-wing unmanned aerial vehicles under wind disturbances,” *International Journal of Robust and Nonlinear Control*, 2012.

[10] T. Yamasaki, S. Balakrishnan, and H. Takano, “Separate-channel integrated guidance and autopilot for a path-following uav via high- order sliding modes,” *AIAA Guidance, Navigation, and Control Conference*, 2012.

[11] P. Sanghyuk, D. John, and H. Jonathan, “Performance and lyapunov stability of a nonlinear path-following guidance method,” *JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS*, 2007.

[12] ——, “A new nonlinear guidance logic for trajectory tracking,” *AIAA Guidance, Navigation, and Control Conference and Exhibit, Guidance, Navigation, and Control and Co-located Conferences*, 2004.

[13] O. John and R. Rolf, “Waypoint guidance for small uavs in wind,” *American Institute of Aeronautics and Astronautics*, 2005.

[14] N. Derek, B. Blake, M. Timothy, and B. Randal, “Vector field path following for miniature air vehicles,” *Robotics, IEEE Transactions on*, 2007.

[15] M. Timothy and H. Karl, “Path planning and control for multiple point surveillance by an unmanned aircraft in wind,” *2006 American Control Conference*, 2006.

[16] R. Rolf, “Unmanned aerial vehicle path following for target observation in wind,” *Journal of Guidance, Control, and Dynamics*, 2006.

[17] B. Alexandru, E. Tadeo, C. Pedro, and L. Rogelio, “Adaptive trajectory following for a fixed-wing uav in presence of crosswind,” *Journal of Intelligent and Robotic Systems*, 2012.

[18] R. Beard and T. McLain, “Implementing dubins airplane paths on fixed-wing uavs,” *Contributed Chapter to the Springer Handbook for Unmanned Aerial Vehicles*, 2013.

[19] J. Zhang, L. Cheng, and B. Liang, “Path-following control for fixed-wing unmanned aerial vehicles based on a virtual target,” *Journal of AEROSPACE ENGINEERING*, 2012.

[20] M. Kamel, T. Stastny, K. Alexis, and R. Siegwart, “Model predictive control for trajectory tracking of unmanned aerial vehicles using ros,” *Springer Book on Robot Operating System (ROS) – The Complete Reference (Volume 2)*, 1981.

[21] R. Chen, “Kinematic properties of rotary-wing and fixed-wing aircraft in steady coordinated high-g turns,” *Atmospheric Flight Mechanics Conference; Aug. 19-21, 1981; Albuquerque, NM*, 1981.

[22] F. Borrelli, A. Bemporad, and M. Morari, “Predictive control for linear and hybrid systems,” 2015.

[23] Y. Kang and J. K. Hedrick, "Linear tracking for a fixed-wing uav using nonlinear model predictive control," *IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, VOL. 17, NO. 5*, 2009.

[24] ——, "Design of nonlinear model predictive controller for a small fixed-wing unmanned aerial vehicle," *AIAA Guidance, Navigation, and Control Conference and Exhibit, Keystone, Colorado*, 2006.

[25] K. Yang, S. Sukkarieh, and Y. Kang, "Adaptive nonlinear model predictive path tracking control for a fixed-wing unmanned aerial vehicle," *AIAA Guidance, Navigation, and Control Conference, Chicago, Illinois*, 2009.

[26] F. Gavilan, R. Vazquez, and S. Esteban, "Trajectory tracking for fixed-wing uav using model predictive control and adaptive backstepping," *IFAC-PapersOnLine, Volume 48*, 2015.

[27] A. Rucco, A. Aguiar, F. Pereira, and J. Borges-de Sousa, "A predictive path-following approach for fixed-wing unmanned aerial vehicles in presence of wind disturbances," *Robot 2015: Second Iberian Robotics Conference*, 2015.

[28] B. Houska, H. Ferreau, M. Vukov, and R. Quirynen, "Acado toolkit user's manual," *http://www.acadotoolkit.org*, 2009-2013.