

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO



UPWIND - Controlo de seguimento de trajetória em Sistemas Aéreos de Energia Eólica

Conrado José Correia Guimarães Martins da Costa

MASTER IN ELECTRICAL AND COMPUTER ENGINEERING

Supervisor: Fernando A.C.C. Fontes

October 31, 2023

Resumo

Atualmente, a conversão da energia cinética do vento em energia eléctrica é amplamente conseguida através de turbinas eólicas que são colocadas em terra ou no mar. As tendências recentes visam aumentar a altura destas turbinas para aumentar a produção de energia, devido ao facto de a velocidade do vento começar a ser mais forte e mais estável a altitudes mais elevadas. Aumentar a altura das torres e a dimensão das pás dos aerogeradores apresenta-se como uma solução inviável uma vez que a construção destas estruturas envolve custos acrescidos. Por razões como estas, foram investigados e desenvolvidos sistemas inovadores de produção de energia eléctrica designados por sistemas aéreos de energia eólica.

O sistema é constituído por um dispositivo de asa rígida controlado amarrado por um cabo a um tambor de guincho que se enrola, permitindo a rotação do eixo da máquina eléctrica. Desenvolvimentos recentes retratam este dispositivo como sendo capaz de atingir elevadas altitudes com ventos mais fortes e mais estáveis, explorando a energia cinética do vento ao seguir trajectórias predefinidas que maximizam a produção de energia durante um ciclo de enrolamento/desenrolamento do cabo. Para que o dispositivo siga adequadamente o caminho desejado, é abordado o seu modelo dinâmico bem como algoritmos de seguimento de trajetórias que são utilizados para desenvolver um controlador de trajetória para o dispositivo se orientar autonomamente e convergir para o caminho desejado.

Esta dissertação estuda o problema do seguimento de trajectórias, em particular, o projeto de controladores avançados para diferentes perfis de trajetória implementado num modelo cinemático simplificado, bem como no modelo dinâmico do sistema aéro de energia eólica.

Abstract

Nowadays, the conversion of wind kinetic energy into electric energy is widely achieved through wind turbines that are placed on-shore or off-shore. Recent tendencies aim to increase their height in order to increase energy production, due to the fact that wind velocity starts to become stronger and more stable at higher altitudes. As more material is required to reach higher altitudes the whole device becomes less affordable. For such reasons, innovative electrical energy production systems designated as airborne wind energy systems have been researched and are under development.

The system comprises a rigid wing tethered device, a kite, linked to a winch drum that reels itself, allowing the shaft of the connected electric machine to rotate. Recent developments portray the kite as a device capable of reaching high altitudes with stronger and more stable winds, exploiting the wind kinetic energy while following predefined optimised paths that best maximise energy production during a two-phase cycle of reel-in/reel-out of the tether. For the kite to adequately follow the desired path, its dynamic model is addressed as well as guidance logic algorithms, with both being used to develop a trajectory controller for the device to steer autonomously and converge to the desired path.

This dissertation studies the path-following problem and state-of-the-art controllers that are to be implemented on a simplified kinematic model, for different path profiles. The dynamic model of the kite is assessed with respect to the best path-following controller.

Agradecimentos

Em primeiro, e sempre em primeiro, tenho gratidão para com a minha Família: à minha mãe Paula, ao meu pai Tomanel, aos meus irmãos Martim e Rosendo. Obrigado pela educação e pelos valores que sempre me transmitiram.

Em segundo, tenho gratidão para com os meus amigos da UBI: André L., André P., Jorge B., José F. e Tomás C..

Em terceiro, tenho gratidão para com o orientador da minha dissertação, o Professor Fernandes Fontes, pela ajuda e colaboração. Mas mais importante, por me integrar na equipa de excelência do projeto UPWind. Em particular, os membros: Gabriel F., Luís P., Luís R., Manuel F., Rui C., Sérgio V. e Thien N. que sempre se mostraram disponíveis para ajudar durante a elaboração da dissertação.

Conrado José Correia Guimarães Martins da Costa

Contents

1	Introduction	1
1.1	Context and Motivation	1
1.2	Goals	2
1.3	Dissertation Structure	3
2	Airborne Wind Energy Systems	5
2.1	Airborne Wind Energy Systems	5
2.2	Ground-Gen AWES	6
2.3	Fly-Gen AWES	9
2.4	Discussion	11
3	Path-Following Guidance	13
3.1	Path-Following and Trajectory Tracking	13
3.2	Path-Following Problem Formulation	14
3.2.1	Vehicle Kinematic Model	15
3.2.2	Path Parameterization and Specification	16
3.2.3	Path-Following Problem in 2D	18
3.2.4	Path-Following Methods for the 2D Problem	20
3.3	Discussion	27
4	2D Simulations for a Kinematic Car Model	29
4.1	2D Kinematic Car Model	29
4.2	Simulation: Distance to the closest point in the path with PID Controller	30
4.2.1	PID Controller	30
4.2.2	Simulation Parameters	31
4.2.3	Simulation results of the 8-Shape path	32
4.2.4	Simulation results of the Ellipse path	34
4.3	Simulation: Carrot Chase Method (L0 distance)	35
4.3.1	Simulation Parameters	35
4.3.2	Simulation results of the 8-Shape Path	36
4.3.3	Simulation results of the Ellipse Path	38
4.4	Simulation: Nonlinear Guidance Logic (L1 distance)	39
4.4.1	Simulation Parameters	39
4.4.2	Simulation results of the 8-Shape path	40
4.4.3	Simulation results of the Ellipse Path	42
4.5	Discussion	43

5	AWES Path-Following Guidance and Simulation	45
5.1	Coordinate Systems	45
5.2	Acting Forces and Dynamic Model	48
5.3	AWES Path-following Model	49
5.4	AWES Path-following Guidance Logic and Control	50
5.5	AWES Simulation	52
5.6	Discussion	56
6	Conclusions and Future Work	57
6.1	Conclusions	57
6.2	Future Work	58
A	Race Track Path-Following	59
A.1	Race Track Path-Following	59
	References	61

List of Figures

1.1	Wind speed velocity variation with altitude [1]	2
2.1	Most common AWES concepts and most renowned development entities [2]. . .	6
2.2	Ground-Gen AWES two-phase operation with electric machine located on the ground. Extension, or energy production, phase (left) and recovery, or energy consumption, phase (right) [3]	7
2.3	SkySails flexible wing kite [4]	8
2.4	Ampyx Power AP2 Prototype [5]	8
2.5	TwingTec TT100 AWE system [6]	8
2.6	Fly-Gen AWES [7]	9
2.7	Lighter than air: Altaeros prototype [8]	10
2.8	Makani M600 Prototype [9]	10
2.9	UPWIND Multiplex EasyStar II glider [1].	11
2.10	UPWIND Prototype with the Multiplex EasyStar II glider coupled to the ground-station [1].	12
3.1	Path-following illustration of a vehicle converging to a desired path.	14
3.2	Outer-loop Path-following illustration; $\mathbf{u}_d$: input to the vehicle's kinematics; $\mathbf{p}$: vehicle's position; η : vehicle's orientation [10].	15
3.3	3-DOF vehicle standard kinematic model representation. $\{G\} = \{x_G, y_G\}$: global inertial reference frame; $\{B\} = \{x_B, y_B\}$: body-fixed frame; ψ : heading angle; δ : steered angle; $\mathbf{V}$: velocity vector.	16
3.4	Example of two Lissajous Curves: circle and eight-shape figures.	17
3.5	Path-following 2D problem illustration.	18
3.6	Method 1 illustration: distance to the closest point in the path (straight line). . . .	20
3.7	Method 1 illustration: distance to the closest point in the path (curved).	21
3.8	Method 2 illustration: VTP ahead of the closest point on the path (straight line) to the vehicle Q	23
3.9	Method 2 illustration: virtual target point T ahead of the closest point on the path (curved) to the vehicle Q	24
3.10	Method 3 illustration: virtual target point T ahead of the vehicle's position P by a distance L_1	26
4.1	Car-like system geometry [11]	30
4.2	Method 1 8-Shape path simulation: cross-track error time variation.	32
4.3	Method 1 8-Shape path simulation: desired path and actual vehicle trajectory. . .	33
4.4	Method 1 8-Shape path simulation: heading angle time variation.	33
4.5	Method 1 Ellipse path simulation: desired path and actual vehicle trajectory. . . .	34
4.6	Method 1 Ellipse path simulation: cross-track error time variation.	34

4.7	Method 1 Ellipse path simulation: heading angle time variation.	35
4.8	Method 2 8-Shape path simulation: cross-track error time variation.	36
4.9	Method 2 8-Shape path simulation: desired path and actual vehicle trajectory. . .	37
4.10	Method 2 8-Shape path simulation: heading angle time variation.	37
4.11	Method 2 Ellipse path simulation: desired path and actual vehicle trajectory. . . .	38
4.12	Method 2 Ellipse path simulation: cross-track error time variation.	38
4.13	Method 2 Ellipse path simulation: heading angle time variation.	39
4.14	Method 3 8-Shape path simulation: desired path and actual vehicle trajectory. . .	40
4.15	Method 3 8-Shape path simulation: cross-track error time variation.	41
4.16	Method 3 8-Shape path simulation: heading angle time variation.	41
4.17	Method 3 Ellipse path simulation: desired path and actual vehicle trajectory. . . .	42
4.18	Method 3 Ellipse path simulation: cross-track error time variation.	42
4.19	Method 3 Ellipse path simulation: heading angle time variation.	43
5.1	Coordinate Systems [12].	46
5.2	Kite roll angle and turning dynamics [13].	47
5.3	Path-following Model [13]	50
5.4	Kite guidance logic [13]	52
5.5	L0 and L1 guidance logics [13]	52
5.6	Ellipse trajectory with a varying tether length $r \in [50, 250]$	54
5.7	Kite following the desired path in the (ϕ, β) space, with $T = 60s$ and $L_0 = 60^\circ$. . .	54
5.8	Cross-track error for $T = 60s$ and $L_0 = 60^\circ$	55
5.9	Simulated (r, ϕ, β) for $T = 60s$ and $L_0 = 60^\circ$	55
5.10	Generated Power (W) and Energy (Wh) for $T = 60s$ and $L_0 = 60^\circ$	56
A.1	Method 1 RCP track simulation: desired path and actual vehicle trajectory.	59
A.2	Method 1 RCP track simulation: cross-track error time variation.	60

List of Tables

4.1	Method 1: Simulation Parameters	32
4.2	Method 1: 8-Shape path simulation results, where $d_{(P_{x0},P_{y0})}$ is the initial distance to the path and t_0 the time the vehicle first crosses the path.	33
4.3	Method 1: Ellipse path simulation results, where $d_{(P_{x0},P_{y0})}$ is the initial distance to the path and t_0 is the time the vehicle first crosses the path.	35
4.4	Method 2: Simulation Parameters	36
4.5	Method 2: 8-Shape path simulation results, where $d_{(P_{x0},P_{y0})}$ is the initial distance to the path and t_0 the time the vehicle first crosses the path.	37
4.6	Method 2: Ellipse path simulation results, where $d_{(P_{x0},P_{y0})}$ is the initial distance to the path and t_0 the time the vehicle first crosses the path.	39
4.7	Method 3: Simulation Parameters	40
4.8	Method 3: 8-Shape path simulation results, where $d_{(P_{x0},P_{y0})}$ is the initial distance to the path and t_0 the time the vehicle first crosses the path.	41
4.9	Method 3: Ellipse path simulation results, where $d_{(P_{x0},P_{y0})}$ is the initial distance to the path and t_0 the time the vehicle first crosses the path.	43
5.1	Simulation parameters	53
5.2	Physical Simulation Parameters ([14],[13]).	54
A.1	Method 1 RCP track: simulation parameters	60

List of Acronyms

AWE	Airborne Wind Energy
AWES	Airborne Wind Energy System
LOS	Line of sight
PID	Proportional–integral–derivative controller
UAV	Unmanned aerial vehicle
UAV	Underwater autonomous vehicle
USV	Unmanned surface vessel
VTP	Virtual target point

Nomenclature

Chapter 3 - Path-Following Guidance:

Subsection (3.2) Path-following problem formulation:

$\mathbf{p}$	vehicle's position
η	vehicle's orientation
$\mathbf{u}_d$	input to the vehicle's kinematics

Subsection (3.2.1) Vehicle Kinematic Model:

c_x	external disturbance along x-axis
c_y	external disturbance along y-axis
r	heading rate
u	longitudinal speed
$\mathbf{V}$	velocity vector
v	lateral speed
δ	steering angle
ψ	heading angle

Subsection (3.2.3) Path-Following Problem in 2D and Subsection (3.2.4) Path-Following Methods for the 2D Problem:

$a_{s_{cmd}}$	centripetal acceleration
c	arc centre
D	total cross-track error
d	cross-track error
$\mathbf{e}$	position error vector
l	path length
k_p	proportional gain
L_1	distance between vehicle's centre of mass and virtual target point
$\mathbf{n}$	normal vector
P	vehicle's centre of mass position
$\mathbf{p}$	vehicle's position vector
Q	closest point on the path
$\mathbf{q}$	target position on the path vector
R	arc/circle radius

r	distance between vehicle's centre of mass position and virtual target point
s	along-track error
T	virtual target point
T_t	total time
U	total control effort
u_R	longitudinal reference speed
u_d	control input
W_A	path starting point
W_B	path ending point
z	vector between arc centre and vehicle's centre of mass position
$\tilde{z}$	normalized z
α	ahead angular distance on the path with respect to the closest point on the path
δ	ahead linear distance on the path with respect to the closest point on the path
θ_L	angle between straight-line path starting and ending points
θ_c	angle arc centre and vehicle's centre of mass position
η	angle between vehicle's velocity vector and line segment $\overline{PT}$
γ	path particle
λ	path variable
λ_1, λ_2	equation system variables
ψ	vehicle's heading angle
ψ_d	desired heading angle
$\mathcal{P}$	path

Chapter 4 - 2D Simulations for a Kinematic Car Model:

Subsection (4.1) 2D Kinematic Car Model

c	vehicle's curvature
l	distance between front and rear wheels
R	turning radius
U	control variables
u	longitudinal speed
X	state variables
δ	steering angle
ψ	yaw angle

Simulations subsections:

c	vehicle's curvature
D	total cross-track error
d	cross-track error
$\bar{d}$	average cross-track error
k_d	derivative gain
k_i	integral gain
k_P	proportional gain

L_0	path-following method 2 distance parameter
L_0	path-following method 3 distance parameter
R	turn radius
T	total simulation time
U	total control effort
u	longitudinal speed
αt	path-following method 2 angle distance parameter
Δt	time step
δt	path-following method 2 distance parameter
ψ	heading angle
ψ_d	desired heading angle

Chapter 5 - AWES Path-following Guidance and Simulation:

A	wing reference area of the kite
a_t	tether reel-out acceleration
a_l	kite lateral acceleration
c_D	aerodynamic drag coefficient
c_L	aerodynamic lift coefficient
d	cross-track error
$\bar{d}$	average cross-track error
$\vec{F}^{lift}$	aerodynamic lift force
$\vec{F}^{inert}$	inertial forces
$\vec{F}^{th}$	tether force
g	gravitational acceleration
L_0	distance ahead of the closest point to the vehicle in the path
L_1	distance ahead of the vehicle in the path
m	mass (kg)
p	kite position
$\dot{p}$	kite velocity
Q	closest point in the path to the vehicle
r	tether length
R	reference point in the path
T_{tether}	tether tension force
u	control vector
V	kite speed
v_a	apparent wind velocity
$\mathbf{V}_a$	kite aerodynamic apparent velocity vector
v_t	tether reel-out speed
v_w	wind velocity
x	state vector
α	angle of attack
ϕ	azimuth angle

β	elevation angle
ψ	roll angle
η	angle between the kite's heading and the heading to the reference target point
ρ	air density
ϱ	angle between the kite velocity and the path tangent

Chapter 1

Introduction

1.1 Context and Motivation

The current human needs in economic activities such as manufacturing and services, transportation systems, broad lifestyles and more, are tightly linked to a higher and rising electrical energy demand. Non-renewable resources, mainly fossil fuels, are primarily used to suffice the referred needs but, it is known the lack of sustainability associated with its exploitation, making climate change a reality.

Therefore, long-term usage of non-renewable resources becomes more and more restricted and, to achieve sustainability and overcome their drawbacks, governments and international organisations enact energy policies, initiatives, agreements, road maps and protocols. The majority of these thrive for CO_2 emissions reduction and limitation, energy-related technological developments and growth in the usage of renewable and sustainable energy sources [3].

Recent events show how fossil fuel shortage and over-dependency are able to disrupt the world's energy system. The energy transition to mostly renewable sources is required to urgently scale faster, allowing to overcome and mitigate climate change damages mainly through decarbonisation, carbon emission reductions, adequate availability of the renewable energy source and its exploitation at a reasonable price, but also, preferably, emerging renewable energy devices and systems [15, 3].

To achieve climate neutrality, the exploitation of several renewable energy sources has been and is still encouraged, one of them being wind energy. During the last decades wind energy systems, mainly wind turbines, have been highly developed and have contributed to a yearly increase in both the quantity of power installations and overall produced energy.

Nowadays, the conversion of wind kinetic energy into electric energy is widely achieved through wind turbines of several metres high (50 to 200 metres on average), placed on-shore or off-shore. Nonetheless, even though proper and adequate conversion and power control systems are designed for wind turbines, recent tendencies aim to increase their height in order to increase energy production, due to the fact that wind velocity starts to become stronger and more stable at higher altitudes - more kinetic power available [13].

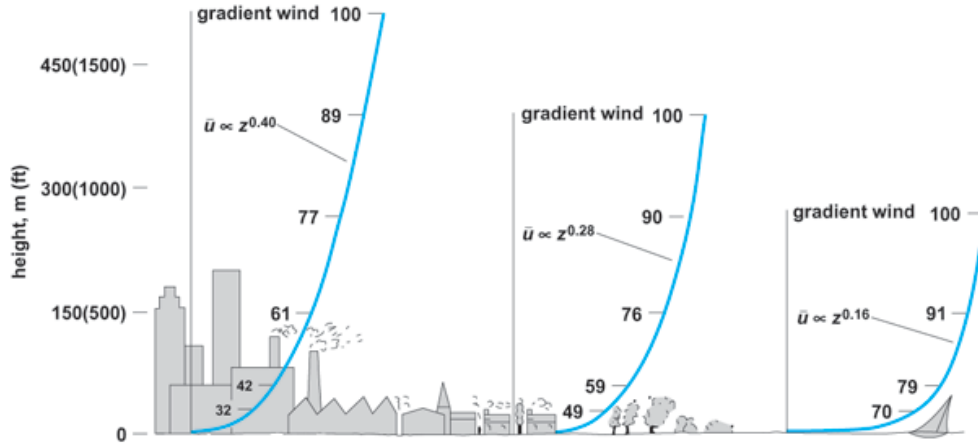


Figure 1.1: Wind speed velocity variation with altitude [1]

Wind turbine towers are not inexpensive and, as more material is required to reach higher altitudes the whole device becomes less affordable. Moreover, knowing that the swept area covered by the wind turbine blades increases the power output, increasing the blades length also increases the overall cost related to the deployment of this solution [16].

A growing community of research groups have been developing airborne wind energy systems, AWES. These systems are designed to be lightweight and inexpensive and able to exploit the kinetic energy from the high altitude winds. However, such devices require adequate airframe structures and guidance control algorithms to achieve autonomous flight under varying circumstances (e.g., wind gusts). Thus, several AWES thematic lines are being researched to make the designed solutions competitive [3].

1.2 Goals

Within the AWES community, the project UPWIND [1] aims to research and develop solutions concerning the AWES or inexpensive flying tethered devices, known as kites. The project has significantly contributed to a variety of challenging optimisation and control problems such as path-following guidance strategies, multiple kite systems layouts and automatic take-off/landing.

This dissertation is integrated into the UPWIND project and the main goals comprise the design and assessment of path-following controllers for different path profiles. It involves the literature review regarding state-of-the-art path-following controllers, the development of the controllers in a standard vehicle kinematic model and the simulation in several path profiles. In addition, considering the rigid-wing AWES concept, the designed and implemented path-following controller is to be simulated using the dynamic model researched within the referred project.

1.3 Dissertation Structure

Chapter 2 introduces the Airborne Wind Energy Systems, AWES, technology and the main categorisations of such systems, along with some examples. Then, the referred chapter ends with a wrap-up discussion bringing into context the technology used within the UPWind project that will be assessed in this dissertation.

Chapter 3 describes the path-following guidance. It starts by stating the differences between trajectory tracking and path-following. Next, the general path-following problem is formulated, a standard kinematic vehicle model is introduced and a method to build a path is presented. Afterwards, the 2D path-following problem is formulated and path-following algorithms are detailed.

Chapter 4 dwells into the simulation of a car-like model system in several path types using the path-following controllers devised in Chapter 3. The results are then discussed to assess the implemented controllers regarding the convergence of the vehicle to the path and the total control efforts.

With the implemented path-following controllers and their assessment for a standard kinematic model, Chapter 5 presents the Kite System Dynamic Model which represents fundamental knowledge required for the implementation and simulation of a path-following controller applied to an AWES solution. This chapter starts by covering the coordinate reference systems, the acting forces and the dynamic model. In addition, it characterizes the path-following problem applied to AWES, and does the linkage between the kite dynamic model and the algorithms presented in Chapter 3. Afterwards, the path-following controller with the best results for the car-like model system simulations is used for the AWES simulations, being the results discussed later in the chapter.

Finally, Chapter 6 ends with the conclusions and future work.

Chapter 2

Airborne Wind Energy Systems

2.1 Airborne Wind Energy Systems

Airborne Wind Energy Systems are part of an innovative lightweight electrical energy production system focused on transforming wind kinetic energy into electrical energy. The system is fundamentally made up of autonomous tethered flying devices, commonly designated as kites, linked to a ground station by one or multiple tethers.

Pioneered by Miles Loyd [17] in the late 1970s and early 80s, since then new concepts of AWES have been developed through recent years. The majority of the designed AWES solutions require less material than tower-based wind turbines, have lower manufacturing costs and are able to reach high altitudes and stable winds [3], [7].

Therefore, AWES is currently portrayed as an emerging technology focused on wind energy harvesting that still maintains the general concept of energy conversion behind wind turbines, while having a different and inexpensive lightweight structural design approach.

These device developments have several problems and concerns that are yet to be considered and solved [3]. The variety and extent of the related problems are linked to the device concept and configuration and may range from the kite's structure to the implemented control strategies. Hence, within the growing community of researchers and manufacturing entities, there is still a lack of convergence towards the best design and control approach [7].

As the literature suggests, regarding the whole setup, AWES are buoyant or flying devices that primarily diverge on whether the conversion from mechanical to electrical power stage happens on the device or on its surroundings. Ground-Gen deployments have the conversion stage at ground level, while Fly-Gen has the conversion stage in the air [2]. Figure 2.1 portrays the common categorisation for the majority of the designs concerning AWES.

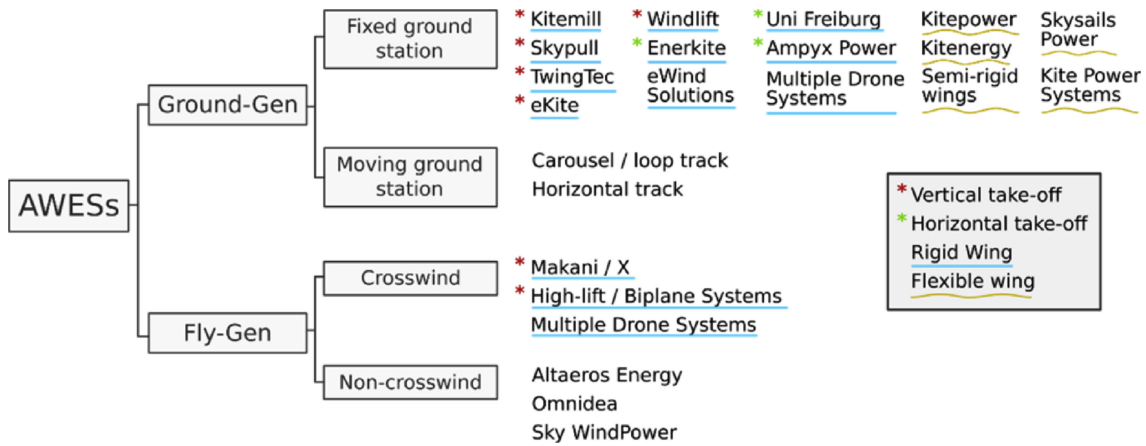


Figure 2.1: Most common AWES concepts and most renowned development entities [2].

2.2 Ground-Gen AWES

Ground-Gen concepts have the kite transferring mechanical energy through the tether to the ground station. A winch drum is coupled with the shaft of the electric machine that works as a generator or as a motor. For such systems, the complete production cycle has two phases as seen in Fig. 2.2.

During the energy production phase, designated as the traction phase, as the tether reels out with the stiff cable keeping adequate tension force, the cable keeps unwinding from the drum that actuates over the electric machine shaft and electric energy is generated. Desirably, at this phase the kite performs a periodic circular or 8-shaped path while maintaining mostly a crosswind flight, therefore having strong apparent wind velocity and withdrawing higher mechanical power from the wind. As the kite maintains the crosswind flight, the tether is forced to reel out and as a result the generator produces electricity.

When the tether reaches its maximum length, it must be retracted, therefore a recovery phase, or reel-in phase, is required. During this phase the movement of the kite must be such that energy consumption is minimised since, at this stage, the cable will be coiled again in the drum with the electric machine acting as a motor, therefore consuming energy [16]. This phase also requires some flight control to reduce the tension force on the tether and the lifting force over the kite.

Since the kite has a periodic predefined path to be followed, the overall cycle must have a positive energy balance so, during the extension stage it is desirable that the absolute value of the production power be significantly higher than the absolute value of the consumption power obtained during the recovery stage, in order for the system to be productive [13].

How well the movement is assessed in real-time under non-linear disturbances, such as wind dynamics, defines the maximisation of the energy production and, for that purpose, at first the desired trajectory must be defined and then a path-following controller must be designed [13].

As for real applications concerning the Ground-Gen group, it may be distinguished as having a fixed or movable ground station. Moreover, within the available solutions several types of AWES

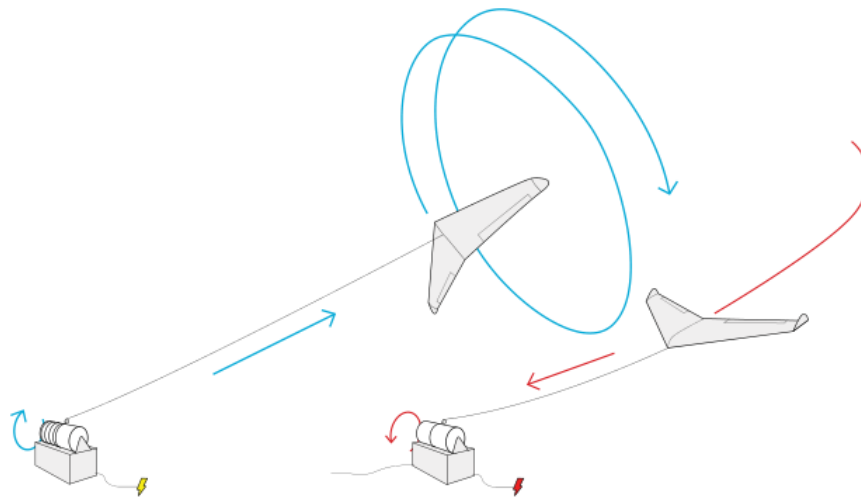


Figure 2.2: Ground-Gen AWES two-phase operation with electric machine located on the ground. Extension, or energy production, phase (left) and recovery, or energy consumption, phase (right) [3]

are possible considering the wing type and take-off method [7]. Focusing on the wing type, these are classified as flexible-wing or rigid-wing kites.

Regarding the flexible wing kites, some leading companies have full airborne solutions. Namely, SkySails developed during the last two decades systems for the usage of flexible wing kites as auxiliary propulsion systems for seagoing vessels, with the purpose of saving fuel [4]. Nonetheless, their business further evolved to the development of kite systems for energy production (see Figure 2.3). The company Kitepower [18] also has a full airborne solution that uses flexible wing kites with the main tether attached to an airborne control pod that allows steering control of the kite during flight.

Rigid wing kites are similar to an aircraft with most designs having the elevator, rudder, ailerons and one or multiple rotors. In addition, these systems still have the kite control unit, tether and ground station. Regarding such systems, several solutions are available on the market [7].

For example, the former company Ampyx Power developed rigid wing kite prototypes and solutions, one of these being displayed in Figure 2.4. The Ampyx Power AWES is a glider aircraft with an autopilot that allows the control of the device's manoeuvres during flight while performing repetitive crosswind patterns at high altitudes (200-450 metres) [19]. TwingTec is also on the market providing highly technological rigid wing kite solutions, with some innovative approaches regarding takeoff and landing. One of the prototypes designed by TwingTec is portrayed on Figure 2.5.



Figure 2.3: SkySails flexible wing kite [4]



Figure 2.4: Ampyx Power AP2 Prototype [5]



Figure 2.5: TwingTec TT100 AWE system [6]

2.3 Fly-Gen AWES

Fly-Gen concepts of AWES essentially have the generators mounted on the lifting device, with the resulting electrical power being transmitted through the tether (see Figure 2.6). For these systems, the tethers are usually thicker and heavier since additional conductive wires are required to transmit the electrical power to the ground [3].

Several designs for Fly-Gen AWES have been created throughout the last decade ranging from lighter-than-air systems to large aircrafts with generators on-board [8].

For lighter-than-air systems, some prototype designs by the company Altaeros exist with buoyant structures floating with a wind turbine inside (see Figure 2.7). As it moves in the air, the power of the high-altitude wind drives the inner turbine which in turn results in electrical power being transmitted through the tether.

Another onboard power generation solution was designed by the company Makani. The designed solution relies on several generators mounted in the airborne structure which flies in a crosswind motion, being the in-flight produced energy transported to the ground by the tether. With the end of the project in 2020, the company shared the technical reports, flight logs for the M600 Prototype (see Figure 2.8), code repositories containing avionics, flight controls, simulations and many other project artefacts [9].

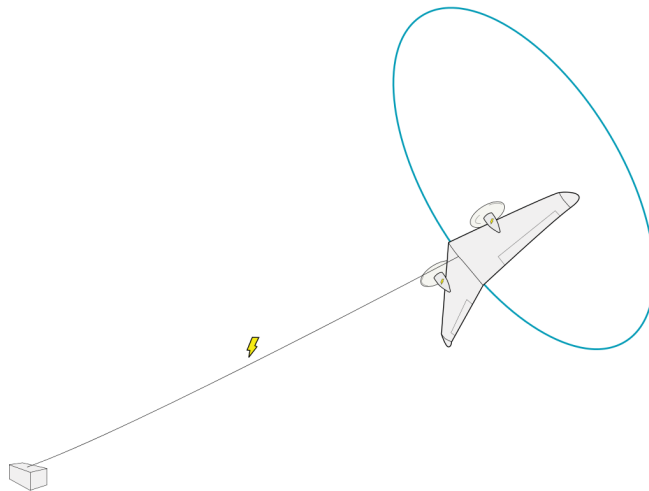


Figure 2.6: Fly-Gen AWES [7]



Figure 2.7: Lighter than air: Altaeros prototype [8]



Figure 2.8: Makani M600 Prototype [9]

2.4 Discussion

Throughout this chapter, a wide spectrum of Airborne Wind Energy Systems designs exist with the common purpose of exploiting higher altitude winds (see Figure 1.1). A considerable part of the industry pursues soft kites and tackles its issues, whereas others seek approaches with some similarity to current aircraft and unmanned aerial vehicles (UAV) technologies.

Some illustrated examples of flexible wing kites are either controlled by the ground station through several tethers, or just by a control pod below the airborne structure. In addition, relative to wind turbines, kite devices are in general lightweight, have low manufacturing costs and have stable flight behaviour. As for a scenario of a falling flexible and soft kite, it has some crash resistance and may not cause high damage [8].

On the other end of the spectrum, rigid-wing kites are highly similar to aircraft such as gliders. Instead of having the steering control on the ground station or on a hovering pod below the airborne structure, it is embedded in the airborne device. For such concepts, some standard aerodynamic concepts still apply, along with some common hardware and software tools. Some of these available tools are useful for simulation and developments regarding the flight controller.

The manoeuvre control of these airborne devices is a challenging problem since the kite's attitude and motion during flight have consequences on the generated electric power. Therefore, path-planning, control and optimisation methods are studied and implemented to maximise/minimise the energy production/consumption during the pumping cycles as the tether reels-out/reels-in [13], [16]. Furthermore, launching, landing and relaunching autonomously with reliability, robustness and safety during all operation phases, for extended periods of time, and for diverse weather conditions, is still a set of problems being evaluated by several research teams [8].

Within the UPWIND Project and concerning the scope of my dissertation, the kite to be considered is a rigid wing glider, namely a glider Multiplex EasyStar II (see Figure 2.9). Currently, the device is modified and has telemetry and autopilot capabilities, being able to fly autonomously with a path-following controller or manually through radio signals.



Figure 2.9: UPWIND Multiplex EasyStar II glider [1].



Figure 2.10: UPWIND Prototype with the Multiplex EasyStar II glider coupled to the ground-station [1].

Chapter 3

Path-Following Guidance

The following chapter details the path-following guidance problem required for adequate vehicle steering throughout a given spatial track, such that it reaches its target point while accomplishing the provided mission objectives. In Subsection 3.1, the distinction between path-following and trajectory tracking concerning time-dependency is clarified. In Subsection 3.2, the fundamental blocks concerning the path-following problem are inferred and portrayed from a control system standpoint, without ample concern for the inherent vehicle hardware block representation. Before the path specification and the guidance methods, in Subsection 3.2.1, the state-of-the-art kinematic model is exposed along with common simplifications. In Subsection 3.2.2, common spatial configurations that are found in the literature are described. Additionally, it provides one method to build them, ending with considerations and assumptions that are commonly contemplated. The path-following problem is then formulated in Subsection 3.2.3 which precedes the methods found in Subsection 3.2.4.

3.1 Path-Following and Trajectory Tracking

Path-following and trajectory tracking are both crucial tasks to be performed by a wide variety of autonomous vehicles under different application scenarios. Some examples of such devices are unmanned aerial vehicles (UAVs), unmanned surface vessels (USVs), underwater autonomous vehicles (UAVs), autonomous cars and many more.

Both tasks consist of steering a given vehicle through a pre-defined path known to lead to the accomplishment of the vehicle's mission. The vehicle's behaviour is linked to its motion dynamics, and to the path specification which may require tight tracking of complex curves and surrounding elements (e.g., obstacles) that may change the approach to the target. Hence, deviation from the path may occur, therefore it is required to implement a strategy to converge with the path but also to maintain it [20], [10], [21].

Regarding some mission constraints such as speed, time and maximum distance to the path, either path-following or trajectory tracking is chosen. Path-following is not directly parameterized by time, meaning that it is not required for the vehicle to be at specific positions of the path at

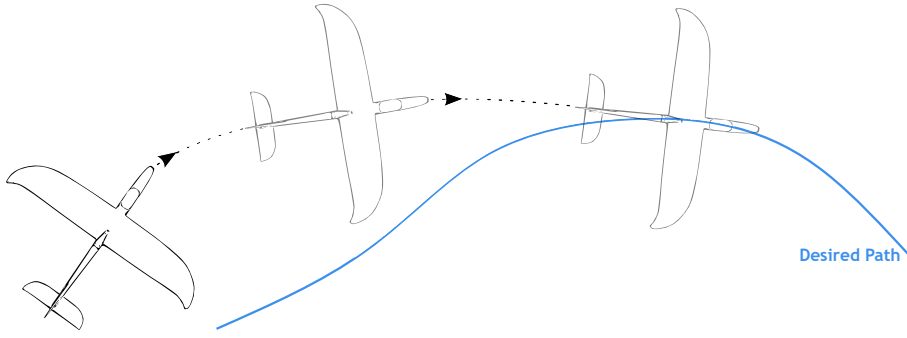


Figure 3.1: Path-following illustration of a vehicle converging to a desired path.

specific instants of time. Moreover, being non-time-dependent, path-following is described as a strategy that allows smoother manoeuvrability and convergence to the path, which may rely on the vehicle parameters (e.g., velocity, turn radius, *et cetera*) [10]. On the contrary, trajectory tracking differs due to the existence of time restrictions to arrive at certain waypoints. Figure 3.1 illustrates the spatial interpretation regarding the path-following problem.

The literature conveys a multitude of path-following methods that have been surveyed, reviewed, researched and applied during the past decades. The methods start to differ by virtue of the dynamic and kinematic models inherent to the vehicle being considered for the desired application. Considering such methods, the task of path-following requires the development of some control strategy that in fact solves the problem [10], [20], [13], [22].

Next, the path-following problem is formulated.

3.2 Path-Following Problem Formulation

The formulation is required to establish the theoretical background regarding path-following which is necessary to implement the control strategy.

In the literature, path-following is described as a problem of making an object (e.g., aircraft, ground vehicle, etc.) described by a set of kinematic constraints, converge and track a desired spatial path as the given mission progresses [10], [23], [24]. This geometric problem is primarily concerned with deriving a control law such that the object is driven with desired parameter profiles (e.g., speed) while minimizing the distance to the specified path.

Figure 3.1 shows a vehicle adjusting its heading angle which leads to a trajectory that converges with the desired path.

From a control system perspective, the literature has several examples which portray mainly the implemented controller designs for the application being described. However, the majority follow the approach of separating the vehicle guidance and control problems into the outer-loop and inner-loop controllers [20], [25]. The mentioned architecture is also suggested in [10] which has a more descriptive controller design regarding the inner-loop application.

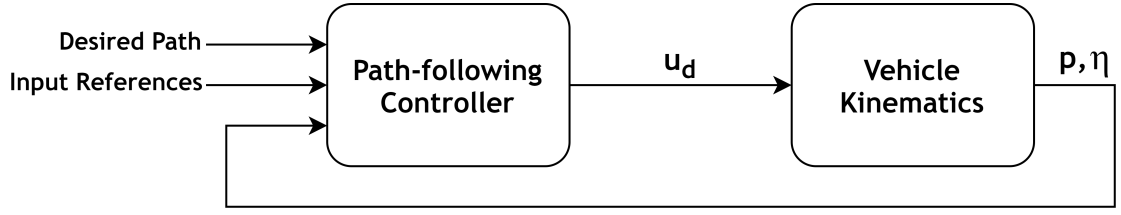


Figure 3.2: Outer-loop Path-following illustration; $\mathbf{u}_d$: input to the vehicle's kinematics; $\mathbf{p}$: vehicle's position; η : vehicle's orientation [10].

The outer-loop path-following controller is implemented whenever a guidance strategy is required. It focuses on capturing desired references that are essential to steer the vehicle to and through the path, though it may also have additional specifications (e.g., reference speed) [10], [23]. The simplified path-following system with the outer-loop path-following controller is depicted in Figure 3.2.

The outer-loop provides the reference commands to the inner-loop so that it controls the vehicle dynamics. In practice, those commands are inputs to the vehicle's autopilot which is the inner-loop path-following controller. With such commands, the necessary dynamics (e.g., forces) of the vehicle are assessed during the mission. In short, the inner-loop is usually implemented before the vehicle's kinematic block, being detailed in [10].

3.2.1 Vehicle Kinematic Model

The vehicle kinematic model is useful to implement a control law that gives a solution to the path-following problem. As the literature suggests, an abstract vehicle may be represented as shown in Figure 3.3 where the global inertial reference frame is $\{G\} = \{x_G, y_G\}$ with the vehicle being represented according to its body-fixed frame $\{B\} = \{x_B, y_B\}$. Considering $\{B\}$, the x-axis points in the vehicle's forward direction with the origin on its centre of mass. Its orientation with respect to $\{G\}$ describes the vehicle's heading angle, or yaw angle, denoted as ψ . In addition, δ represents the steered angle with respect to the body frame, and $\mathbf{v}$ is the velocity vector [26].

The most common vehicles perform longitudinal manoeuvres with ease, i.e. moving forward or backwards. As for lateral motion, not all vehicle designs require or even allow it to be immediate mostly due to the absence of actuators or elements of thrust that allow movement or rotation. Nonetheless, significant influences may exist and act as external disturbances to the vehicle's motion (e.g., wind).

Therefore, usually, the kinematic model of an abstract three degrees of freedom (3-DOF) vehicle is considered to have three main scenarios: under-actuated with no external disturbances; under-actuated with external disturbances; fully-actuated with external disturbances [10]. Considering such scenarios, in summary, a system with fewer actuators than degrees of freedom is said to be under-actuated [26].

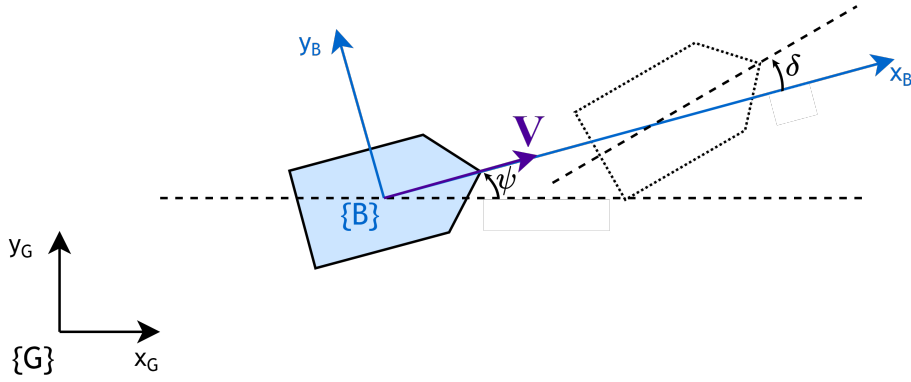


Figure 3.3: 3-DOF vehicle standard kinematic model representation. $\{G\} = \{x_G, y_G\}$: global inertial reference frame; $\{B\} = \{x_B, y_B\}$: body-fixed frame; ψ : heading angle; δ : steered angle; $\mathbf{V}$: velocity vector.

Commonly the under-actuated scenario is considered for the path-following implementation since the accurate model is regarded as being a more complex task [22]. In addition, this simplified model is sufficient to assess the vehicle's behaviour as it manoeuvres, without external disturbances (e.g., wind gusts), with a given path-following controller for a standard application scenario.

With respect to Figure 3.3, the complete kinematic model is as follows:

$$\begin{cases} \dot{x} = u \cos(\psi) - v \sin(\psi) + c_x \\ \dot{y} = u \sin(\psi) + v \cos(\psi) + c_y \\ \dot{\psi} = r \end{cases} \quad (3.1)$$

The under-actuated without external disturbances kinematic model is defined as:

$$\begin{cases} \dot{x} = u \cos(\psi) \\ \dot{y} = u \sin(\psi) \\ \dot{\psi} = r \end{cases} \quad (3.2)$$

where (x, y) describes the vehicle's position, ψ is the vehicle's heading angle, u and v are the longitudinal and lateral speeds respectively described the velocity vector $V = [u, v]$, r describes the angular speed or a given heading rate and, at last, (c_x, c_y) represents the effect of external unknown disturbances in $\{G\}$.

3.2.2 Path Parameterization and Specification

Depending on the given mission, the desired path is not unique. Regardless of the variety of application scenarios, the path often includes straight lines and curves. Therefore, the literature commonly evaluates the path-following algorithms on a set of spatial paths: straight lines, orbits or ellipses, circles and eight-shape. The latter includes both lines and semi-circles and represents a figure that is recurrently found in AWES applications.

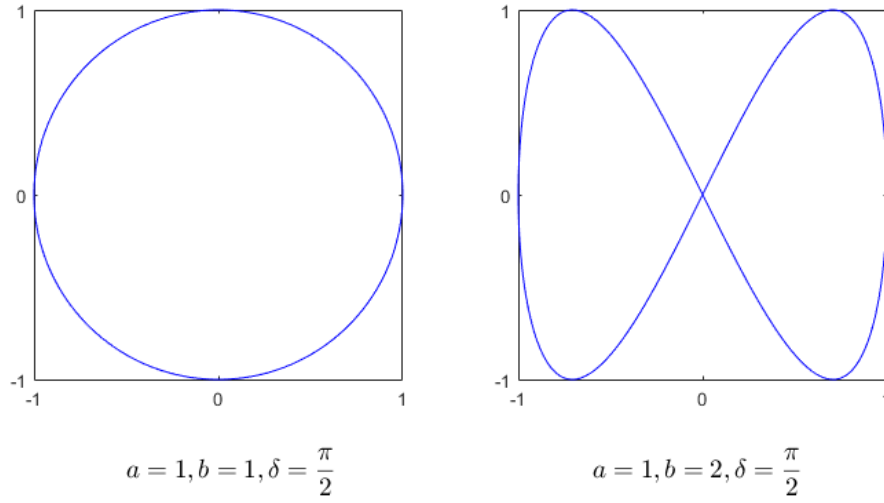


Figure 3.4: Example of two Lissajous Curves: circle and eight-shape figures.

For future reference, the mentioned shapes are usually built as *Lissajous curves* expressed mathematically as:

$$\begin{cases} x = A\sin(at + \delta) + C \\ y = B\sin(bt + \delta) + D \end{cases} \quad (3.3)$$

In example, the circle has parameters $a = b$, $A = B$ and $\delta = (2n + 1)\frac{\pi}{2}$ where $A, B, n \in \mathbb{Z}$ and the eight-shape the parameters are $a = \frac{b}{2}$ and $\delta = k\frac{\pi}{2}$ with $k \in \mathbb{Z}$. Figure 3.4 portrays both cases.

During the progress of the vehicle's mission, such paths are usually defined on a xy plane at a constant altitude and speed [22].

Nevertheless, regarding the path frames, the literature sometimes mentions specifically the one being considered. The chosen path frame is important to characterize the position error between the vehicle and the waypoint on the path. Depending on the position of the vehicle and the curvature of the path the perception (or sign) of the error varies. In [10] the two most common path frames are characterized and compared under specific circumstances (e.g., the existence of inflection points on the path).

The specified and assigned mission path may still hold particular circumstances, in addition to complex curves. With respect to the eight-shape, the intersection point may hold a problem for some path-following guidance algorithms. Thus, to keep track of the correct path direction some alternatives are considered such as implementing a looking ahead strategy [22] or a state-machine.

It is important to note that the assigned path is often the result of some optimization process which imposes a set of constraints. This result may relate to the mission's overall requirements and be linked with the implemented system. Thus, finding and evaluating the optimal path would be extensive work which is not the current focus of this document. Hence, focusing on the previously referred path profiles, they are fed to the vehicle as a look-up table. In practice, this avoids the allocation of computational processing to compute the trajectory waypoints. Concerning standard

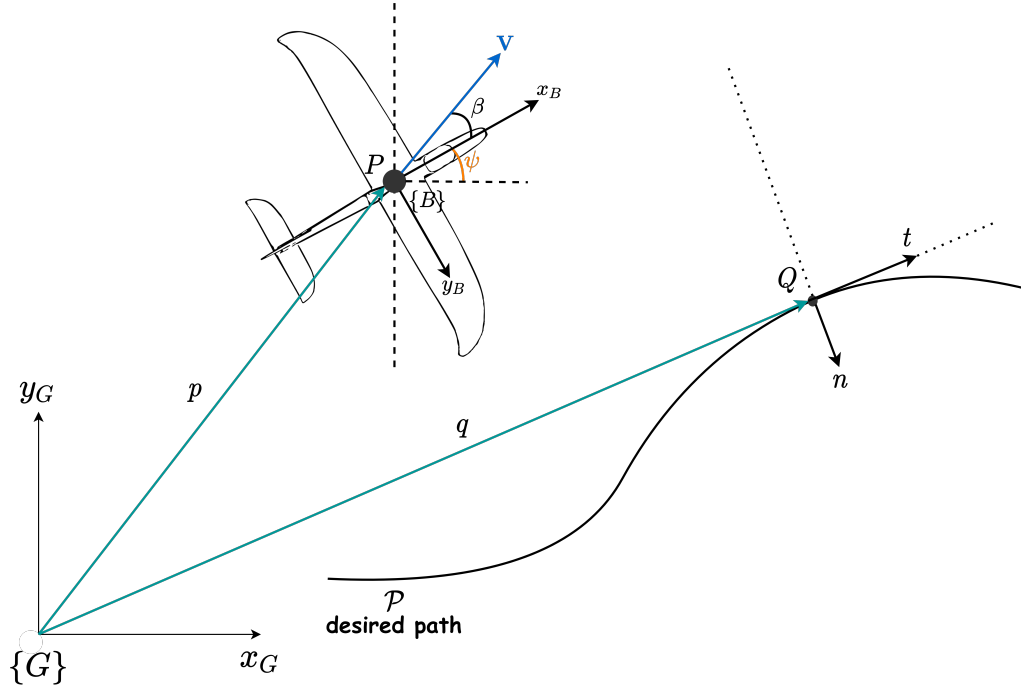


Figure 3.5: Path-following 2D problem illustration.

applications, it simplifies the path-following problem, being our focus on the control strategy over the vehicle's kinematics and dynamics. This set of scenarios encourages the evaluation of the path-following algorithms which will be described in Subsection 3.2.4.

3.2.3 Path-Following Problem in 2D

The path-following problem in 2D is portrayed in Figure 3.5. In this figure, the inertial global frame is denoted as $\{G\} = \{x_G, y_G\}$ and the vehicle's body-fixed frame is denoted as $\{B\} = \{x_B, y_B\}$.

The position vector of the vehicle is denoted as $\mathbf{p} = [x, y]^T \in \mathbb{R}^2$ with its origin at $\{B\}$, the vehicle heading angle as $\psi = \angle(x_G, x_B)$ and the sideslip as $\beta = \angle(x_B, \mathbf{V})$. The latter concerns the deviation between the vehicle's route and the actual track being followed due to the surrounding fluid (e.g., water or air) behaviour.

In addition, Q is the target position on the desired path $\mathcal{P}$ of length l parameterized by $\gamma(\lambda) \in \mathbb{R}^2$ with $\lambda \in [0, l]$ [27]. Thus, as in [28], the geometric path is expressed as:

$$\mathcal{P} = \{\mathbf{q} \in \mathbb{R}^2 \mid \mathbf{q} = \gamma(\lambda) \forall \lambda \in [0, l]\} \quad (3.4)$$

Figure 3.5 illustrates the 2D path-following problem as it is commonly found in the literature [28], [10], [22]. In particular, it displays the position vector of the vehicle ($\mathbf{p}$), the position vector of the point on the path ($\mathbf{q}$) and the normal vector ($\mathbf{n}$) known to be orthogonal to the tangent to the path at each waypoint to be tracked.

As mentioned earlier, the guidance strategies regarding path-following focus on making the vehicle converge and follow the desired path, without time constraints. Nevertheless, as the simplified control scheme in Figure 3.2 suggests, a given reference input is usually given such as the reference speed. Therefore, the vehicle keeps track of both the desired path and the desired input reference.

It is, therefore, implied that the path-following problem has two major tasks as firstly described in [28] and later revised in [10]. The tasks are the following:

- (i) Geometric Task: make the vehicle converge and follow the desired spatial path so that the position error, defined as the vector difference $\mathbf{e} := \mathbf{p} - \mathbf{q}$, evolves along time and converges to 0. Mathematically:

$$\lim_{t \rightarrow 0} \mathbf{e}(t) = 0 \quad (3.5)$$

- (ii) Dynamic Task: make the vehicle track a given input reference. Assuming that it is desirable to track the reference longitudinal speed u_R then it yields:

$$\lim_{t \rightarrow 0} u(t) - u_R(t) = 0 \quad (3.6)$$

The position error vector $\mathbf{e} = [s, d]$ with $\mathbf{e} \in \mathbb{R}^2$ components describe the along-track error and the cross-track error, respectively, the cross-track error d is mostly used as a comparison metric between several path-following algorithms as it defines the distance between the vehicle and the shortest point on the track [24]. This assumption is valid whenever Q is chosen such that it coincides with the orthogonal projection of the centre of mass of the vehicle P [10]. Therefore, with the along-track error component being zero, methods that employ this assumption for a set of guidance laws evaluate the error as the total cross-track error given by the following expression:

$$D = \sum_{t=0}^{T_t} d(t) \quad (3.7)$$

where $d(t)$ denotes the cross-track error at an instant of time t , being T_t the total time (e.g., simulation time).

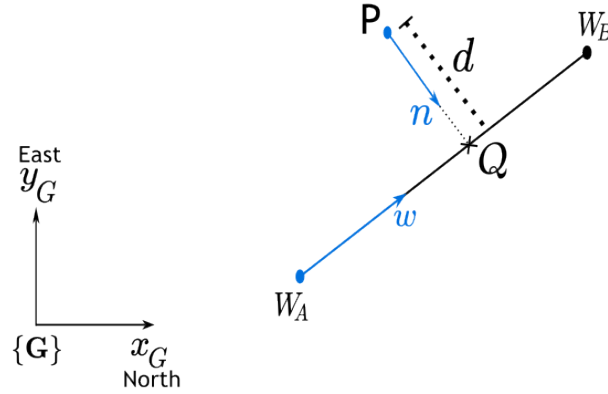


Figure 3.6: Method 1 illustration: distance to the closest point in the path (straight line).

Moreover, commonly the total control effort is also evaluated as in [24]. It is computed by the following expression:

$$U = \sum_{t=0}^{T_f} u_d(t)^2 \quad (3.8)$$

where $u_d(t)$ is the control input to the vehicle's kinematics.

3.2.4 Path-Following Methods for the 2D Problem

3.2.4.1 Method 1: distance to the closest point in the path

For this method, first, it is considered a straight line that is a segment of the path to be followed. It is delimited by two waypoints defined as $W_A = [W_{Ax}, W_{Ay}]^T \in \mathbb{R}^2$ and $W_B = [W_{Bx}, W_{By}]^T \in \mathbb{R}^2$, with the orientation of the line being described by the vector $\mathbf{w}$.

In addition, the vehicle's centre of mass P is projected onto the line and coincides with the reference point Q , also designated as the point which is closest to P . This orthogonal projection is characterized by the normal vector $\mathbf{n}$. Therefore, it holds that the error to minimize given the path-following problem is the cross-track error d . Figure 3.6 depicts the geometric interpretation for this scenario.

Vector $\mathbf{w}$ is expressed as:

$$\mathbf{w} = [w_x, w_y]^T = \frac{1}{\|W_A W_B\|} \overrightarrow{W_A W_B} = \begin{bmatrix} \frac{1}{\|W_A W_B\|} (W_{Bx} - W_{Ax}) \\ \frac{1}{\|W_A W_B\|} (W_{By} - W_{Ay}) \end{bmatrix} \quad (3.9)$$

The normal vector $\mathbf{n}$:

$$\mathbf{n} = \begin{bmatrix} n_x \\ n_y \end{bmatrix} = \begin{bmatrix} -w_y \\ w_x \end{bmatrix} \quad (3.10)$$

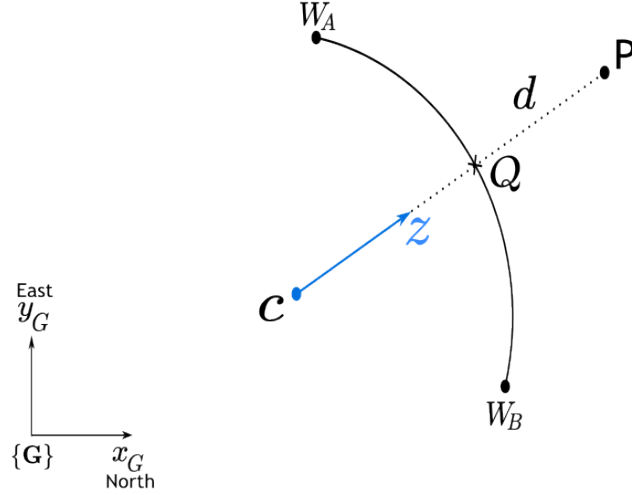


Figure 3.7: Method 1 illustration: distance to the closest point in the path (curved).

The reference point Q is described by the equality:

$$Q = W_A + \lambda_1 \mathbf{w} = P + \lambda_2 \mathbf{n} \quad (3.11)$$

where the Q coordinates are obtained when the following equations are solved with λ_1 and λ_2 determined:

$$\begin{cases} W_{Ax} + \lambda_1 w_x = P_x + \lambda_2 n_x \\ W_{Ay} + \lambda_1 w_y = P_y + \lambda_2 n_y \end{cases} \quad (3.12)$$

With the segments being perpendicular then:

$$\overline{W_A W_B} \perp \overline{PQ} \Rightarrow \angle(\overline{W_A W_B}, \overline{PQ}) = \theta = \frac{\pi}{2} \text{ rad} \quad (3.13)$$

which could be verified with $\arctan(\frac{m_1 - m_2}{1 + m_1 m_2})$ being the m_1 and m_2 the slopes of the line segments $\overline{W_A W_B}$ and $\overline{PQ}$.

At last, regarding the straight line scenario, the cross-track error d is given by the Euclidean distance:

$$d = \sqrt{(Q_x - P_x)^2 + (Q_y - P_y)^2} \quad (3.14)$$

If the path is defined as a curve, the above strategy does not fully apply. Figure 3.7 describes one possible approach which consists of defining the arc centre (c) that passes in Q and is oriented to P by the vector $z = [z_x, z_y]^T \in \mathbb{R}^2$.

To find the closest point Q the factor $\frac{R}{\|z\|}$ is used to normalize so that the magnitude of z is equal

to the radius R . Thus

$$\tilde{z} = \begin{bmatrix} \tilde{z}_x \\ \tilde{z}_y \end{bmatrix} = \begin{bmatrix} z_x \frac{R}{\|z\|} \\ z_y \frac{R}{\|z\|} \end{bmatrix} \quad (3.15)$$

where $\|z\| = \sqrt{z_x^2 + z_y^2}$.

Now, adding the curve centre coordinates, the closest point to P on the path is given by the following expression:

$$Q = \begin{bmatrix} Q_x \\ Q_y \end{bmatrix} = \begin{bmatrix} c_x + \tilde{z}_x \\ c_y + \tilde{z}_y \end{bmatrix} \quad (3.16)$$

Finally, the cross-track error is also obtained with the expression 3.14.

In summary, with Algorithm 1 it is possible to implement the method described above, for both the straight line and curved trajectories.

Algorithm 1 Method 1: distance to the closest point in the path.

Parameterize the path $\mathcal{P}$ as $\gamma(\lambda) \in \mathbb{R}^2$

Initialize the vehicle's initial position $P = (P_x, P_y)$

for every instant of time $0 < t \leq T$ **do**

 Compute $\lambda_p(t) = \underset{\lambda < L}{\operatorname{argmin}} |p(t) - \gamma(\lambda)|$;

Straight Line:

 Compute $\mathbf{w}$ (3.9) and $\mathbf{n}$ (3.10);

 Solve for λ_1 and λ_2 the equation (3.12);

 Compute Q (3.11);

 Compute d (3.14);

 (optional) validate $\theta = \arctan(\frac{m_1 - m_2}{1 + m_1 m_2}) = \frac{\pi}{2} \text{rad}$ (3.13);

Curve:

 Compute $\|z\| = \|P - C\|$ with $P = [P_x, P_y] \in \mathbb{R}^2$ and $c = [c_x, c_y] \in \mathbb{R}^2$;

 Compute $\tilde{z}$ (3.15);

 Compute Q (3.16);

 Compute d (3.14);

 Apply the control input $u_d = k_p d$;

end for

3.2.4.2 Method 2: Carrot Chase

The Carrot Chase method is widely described in the literature, being in general the first to be presented for a given application and further compared with other methods. Such applications are described in the articles [27] and [25], while [24] is prior to these and reports several methods which are evaluated and compared in two generic applications: straight line and circular (loiter) paths.

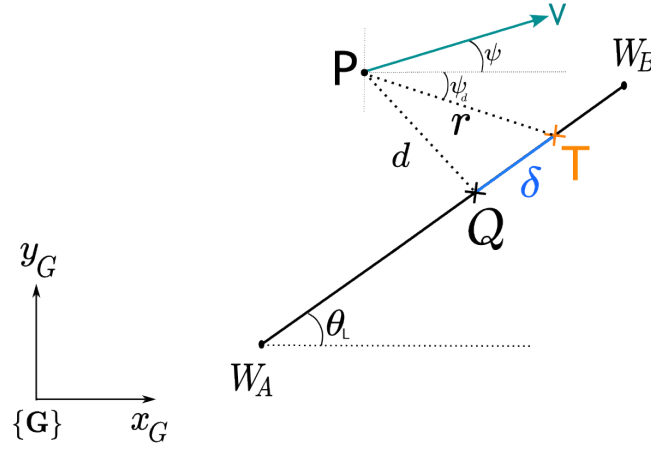


Figure 3.8: Method 2 illustration: VTP ahead of the closest point on the path (straight line) to the vehicle Q .

The majority of the mathematical expressions presented in the first method 3.2.4.1 still hold. For this method, it is assumed the existence of a virtual target point (VTP) which slides along the path. That point denoted as T is ahead of the closest point to the path Q by a distance of δ if the path corresponds to a straight line. On the contrary, for curved paths such distance would be described by an angle. Hence, one must also parameterize the look-ahead distance and evaluate the strategy for the path geometry variations.

Regarding the previous method, in addition, it is required to specify the coordinates of T . One possibility is to solve:

$$\begin{cases} d(T, Q) = \delta \\ d(T, P) = r \end{cases} \Leftrightarrow \begin{cases} \sqrt{(T_x - Q_x)^2 + (T_y - Q_y)^2} = \delta \\ \sqrt{(T_x - P_x)^2 + (T_y - P_y)^2} = r \end{cases} \quad (3.17)$$

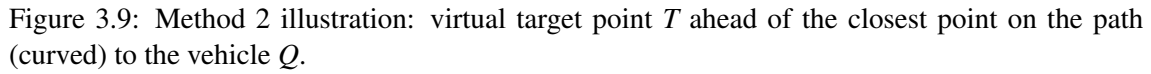
where T_x and T_y are the desired coordinates of the virtual target point on the path. The distance r may be computed with:

$$r = \sqrt{(d)^2 + (\delta)^2} \quad (3.18)$$

However, there may exist more than one solution. Therefore, usually, the coordinates are obtained with the angle $\theta_L = \angle(\overline{W_A W_B}, y_G)$ computed with the four-quadrant inverse tangent, denoted as atan2 , as follows:

$$\theta_L = \text{atan2}(W_{By} - W_{Ay}, W_{Bx} - W_{Ax}) \quad (3.19)$$

This function is widely used in programming and allows one to compute the angle in all four quadrants, without division by zero errors.


$$\begin{cases} T_x = (d(W_A, Q) + \delta) \cos \theta_L \\ T_y = (d(W_A, Q) + \delta) \sin \theta_L \end{cases} \quad (3.20)$$
$$\psi_d = \text{atan2}(T_y - P_y, T_x - P_x) \quad (3.21)$$

For a path that is curved parameterized with centre c and radius R , as in Figure 3.9, the virtual target point is ahead on the path by an angle α which is a design parameter. Hence, similarly to 3.19:

The target point has the following coordinates:

$$\begin{cases} T_x = C_x + R \sin(\theta_c + \alpha) \\ T_y = C_y R \cos(\theta_c + \alpha) \end{cases} \quad (3.23)$$

In summary, with Algorithm 2 it is possible to implement the method 2.

Algorithm 2 Method 2: Carrot Chase

```

Parameterize path  $\mathcal{P}$  as  $\gamma(\lambda) \in \mathbb{R}^2$ 
Initialize the vehicle's initial position  $P = (P_x, P_y)$ 
for every instant of time  $0 < t \leq T$  do
  Compute  $\lambda_p(t) = \underset{\lambda < L}{\operatorname{argmin}} |p(t) - \gamma(\lambda)|$ ;
  Straight Line:
    Initialize  $\delta$ 
    Compute  $\mathbf{w}$  (3.9) and  $\mathbf{n}$  (3.10);
    Solve for  $\lambda_1$  and  $\lambda_2$  the equation (3.12);
    Compute  $Q$  (3.11),  $d$  (3.14),  $r$  (3.18) and  $\theta$  (3.19);
     $(Tx, Ty) \leftarrow ([d(W_A, Q) + \delta] \cos \theta_L, [d(W_A, Q) + \delta] \sin \theta_L)$  (3.20);
  Curve:
    Initialize  $\alpha$ 
    Compute  $\|z\| = \|P - C\|$  with  $P = [P_x, P_y] \in \mathbb{R}^2$  and  $c = [c_x, c_y] \in \mathbb{R}^2$ ;
    Compute  $\tilde{z}$  (3.15),  $Q$  (3.16) and  $d$  (3.14);
     $(Tx, Ty) \leftarrow (C_x + R \cos(\theta_c + \alpha), C_y + R \sin(\theta_c + \alpha))$  (3.23);
    Compute  $\psi_d$  (3.21);
    Compute the required heading angle of  $\psi_d - \psi$ ;
    Apply the control input  $u_d = k_P(\psi_d - \psi)$ 
end for

```

3.2.4.3 Method 3: Nonlinear Guidance Logic (L1 distance)

The nonlinear guidance logic, often referred to as the L_1 distance method, is firstly described in [20] as being a method that generates the lateral acceleration command required to steer the vehicle to engage the target on the path. The idea is similar to the previous method: the virtual target point (T) is defined ahead on the path based on a design parameter (in this case distance L_1). However, the approach to compute the required steering angle and acceleration is simpler and broad as it applies to straight and curved paths.

Figure 3.10 portrays the L_1 distance method. The vehicle has at a given instant of time position P , velocity $\mathbf{V}$ and heading angle $\psi = \angle(x_B, x_G)$ without sideslip ($\beta = 0$). First, the acceleration is considered to be pointing to the centre of a circle of radius R . This circle intercepts at least one point of the path designated the VTP, T , at a distance L_1 from the vehicle's position. For this scenario, just T is considered, nevertheless for multiple interceptions, one may add a decision strategy such as choosing the VTP which points on a given direction.

With these concepts, it yields the expression for the L_1 distance:

$$\sin(\eta) = \frac{L/2}{R} \Leftrightarrow L = 2R \sin(\eta) \quad (3.24)$$

where $\eta = \angle(\mathbf{V}, \overline{PT})$.

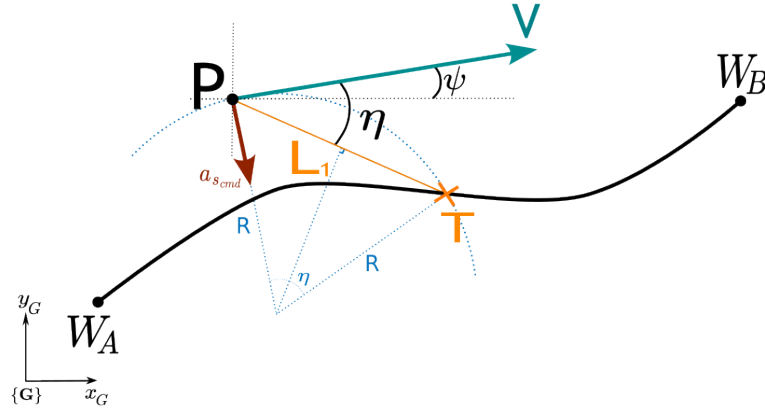


Figure 3.10: Method 3 illustration: virtual target point T ahead of the vehicle's position P by a distance L_1 .

Denoting $a_{s_{cmd}}$ the centripetal acceleration, with the expression 3.24 it results:

$$a_{s_{cmd}} = \frac{V^2}{R} = \frac{2V^2 \sin(\eta)}{L_1} \quad (3.25)$$

with $R = \frac{L_1}{2\sin(\eta)}$.

It is possible to easily compute the η angle if the target coordinates are known:

$$\eta = \psi_d + \psi \quad (3.26)$$

where ψ_d is given similarly to 3.21.

The coordinates of T may be found by considering a circle of radius L_1 with its centre in P . This circle will intersect the path at the points that are the solution of the following circle equation:

$$L_1^2 = (x_r - P_x)^2 + (y_r - P_y)^2 \quad (3.27)$$

While implementing this method, as the above expressions show, the L_1 distance is a constant design parameter and therefore it is not required to compute R . The values of L_1 must be defined to achieve the required vehicle performance. Thus, further increase or decrease of this parameter changes how far the target point is on the path, hence changing the engagement to the path and consequently the cross-track error.

In summary, with Algorithm 3 it is possible to implement Method 3.

Algorithm 3 Method 3: L1 distance

```

Parameterize the path  $\mathcal{P}$  as  $\gamma(\lambda) \in \mathbb{R}^2$ 
Initialize the vehicle's initial position  $P = (P_x, P_y)$ 
Initialize the L1 distance
for every instant of time  $0 < t \leq T$  do
    Compute the target  $T$  coordinates 3.27;
     $\psi_d = \text{atan2}(T_y - P_y, T_x - P_x)$ ;
     $\eta = \psi_d + \psi$  (3.26);
    Compute the centripetal acceleration  $a_{scmd}$  (3.25);
    Apply the control input  $u_d = a_{scmd}$ ;
end for

```

3.3 Discussion

Throughout this chapter, the path-following guidance has been described. As the illustration depicted in Figure 3.2 suggests, the desired path is provided to the system that in practice is a look-up table when considering methods 1, 2 and 3 that were characterized in Subsection 3.2.4.

Recent trends outline other methods which require real-time processing for the path to be followed, some of them being hybrid topologies combining the look-up table with real-time waypoint computation. For this dissertation, the path (e.g., ellipse or 8-shape) is fixed and given to the system with the assumption that is the optimal path to be followed. With that, the focus dwells on the design of a control strategy to assess the behaviour of the vehicle along its mission, something to be accomplished using state-of-the-art guidance algorithms which are known to hold adequate control demand for the most common autonomous vehicle's autopilot systems.

The three referred methods were detailed in this chapter. Method 1 requires the evaluation of the path profile, i.e. either straight or curved, and does not consider the angles formed between the existing elements of the global or body frames. Therefore, it is implied that this method uses directly the cross-track error on the control input. Regarding Algorithm 1, the control input does not fully specify the required turning angle of the vehicle based on the cross-track error, since depending on the vehicle model being considered the expression may differ. Often, the mentioned resulting expression is related to the vehicle's curvature equation and uses parameter λ_2 to assess the above or under the path circumstances.

Method 2 is similar to the previous one as it also evaluates the path profile, but scrutinizes the existence of angles and uses the VTP concept. From the literature, it is clear that the approach to determine the VTP coordinates and the required heading angle is not unique. Despite this, as algorithm 2 suggests, the fundamental instructions of the previous method were used. With the adequate angle description, it holds that the control input is given by an angle and a proportional gain.

Method 3 also employs the VTP and is commonly used in autonomous vehicles being available in some autopilot solutions (e.g., ardupilot). It has the advantage of being similar to the majority of path profiles and, in addition, the control input is defined as a relation of the vehicle's velocity and a design parameter (L1 distance). The performance of the method is determined by this parameter L1 distance and should be properly chosen such that the overall cross-track error along the entire path is minimised.

Chapter 4

2D Simulations for a Kinematic Car Model

This chapter covers simulations using the vehicle kinematic model addressed in Chapter 3. Section 4.1 defines the variables to be considered when simulating a car-like vehicle with the referred kinematic model. Sections 4.2, 4.3 and 4.4 describe the simulation environment, i.e. important considerations and simulation parameters, for the simulation of the path-following guidance methods detailed in 3.2.4. The methods are compared with the metrics mathematically expressed in equations 3.7 and 3.8.

4.1 2D Kinematic Car Model

The 3-DOF wheeled vehicle motion to be simulated in this chapter has the kinematic model identical to the one described in Chapter 3 (equation 3.2) with its position and orientation represented by the body coordinate frame $\{B\}$ as in Figure 3.3.

The car-like system to be considered for the simulations is illustrated in Figure 4.1 and its kinematic model is described as follows:

$$\begin{cases} \dot{x} = u(t)\cos(\psi(t)) \\ \dot{y} = u(t)\sin(\psi(t)) \\ \dot{\psi} = u(t)c(t) \end{cases} \quad (4.1)$$

The vehicle's configuration evolves as the control inputs change. Therefore, for some instant of time t , we have that $\psi(t)$ is the yaw angle, $\delta(t)$ is the steering angle, $(x(t), y(t))$ is the mid-point of the axle, $c(t)$ is the curvature of the path driven by the vehicle and l is the distance between the front and rear wheels. These last two parameters are related through the turning radius R by the following expressions:

$$\begin{cases} c(t) = \frac{\tan(\delta(t))}{l} \\ R_{min} = \frac{1}{|c_{max}|} \end{cases} \quad (4.2)$$

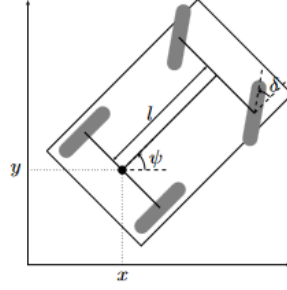


Figure 4.1: Car-like system geometry [11]

At last, for some t , in seconds, we have that

$$X(t) = (x(t), y(t), \psi(t)) \quad (4.3)$$

$$U(t) = (u(t), c(t)) \quad (4.4)$$

The generalized kinematic model described in Chapter 3 (equation 3.2) is changed to a system of equations 4.1 so that some of the car-like parameters and constraints are considered. Mainly, the curvature is known to saturate during the simulations as the control inputs are given. With $\psi = 0$ the curvature is zero and this often occurs in straight line paths, with the maximum curvature defined by $\psi \in [-\psi_m, \psi_m]$.

In this Chapter, this model will be used throughout the simulations with the various path-following algorithms described in Chapter 3 being implemented.

4.2 Simulation: Distance to the closest point in the path with PID Controller

Algorithm 1 does not define extensively the computation of the required turning angle to steer the vehicle to the path, as stated and explained in the discussion of Chapter 3. Considering model 4.1, the required angle is related to the vehicle's curvature c along the path and is associated with the steering angle δ .

4.2.1 PID Controller

This method is simulated with a PID controller. Theoretically, this controller continuously tries to minimise the error $e(t)$ over time by adjusting a control variable $u(t)$. It holds that $e(t)$ is the difference between a reference $r(t)$ and the process variable $y(t)$.

The PID controller has the mathematical expression:

$$u(t) = K_p \cdot e(t) + K_i \cdot \int_0^t e(\tau) d\tau + K_d \cdot \frac{de(t)}{dt} \quad (4.5)$$

where $u(t)$ is the control action of the PID controller at time t , K_P is the proportional gain, K_I is the integral gain and K_D is the derivative gain.

The Euler's method is to be considered to compute the derivative term at each discrete time step. Hence:

$$e'(t_i) \approx \frac{e(t_i + h) - e(t_i)}{\Delta t} \quad (4.6)$$

where $i = 0, 1, \dots, T$ being T the simulation time.

4.2.2 Simulation Parameters

Regarding the Method 1 simulation, the used parameters for both the 8-shape and ellipse figures are defined in table A.1. It is important to note that, the vehicle velocity is considered to be constant, whereas the curvature is controlled with the mentioned PID controller. Therefore, the curvature at a given simulation time step is computed with the expression:

$$c = K_p \cdot d(t) + K_i \cdot \int_0^t d(\tau) d\tau + K_d \cdot \dot{d} \quad (4.7)$$

where d denotes the cross-track error distance. The derivative term is computed with:

$$\dot{d} \approx \frac{d_{i+1} - d_i}{\Delta t} \quad (4.8)$$

being Δt the simulation time step, d_{i+1} and d_i the current and previous cross-track error distances respectively.

The PID Gains were partially tuned with the Ziegler-Nichols tuning method and rearranged with trial and error. At first, the integral and derivative gains are set to zero, except for the proportional gain which starts with an arbitrarily low value (e.g., equal to zero). Then, the system's response is evaluated as the proportional gain is increased, with this procedure ending when constant oscillations at the output are observed. In the end, the obtained gain and oscillation period are used to compute the remaining two gains, with the aid of reference tables.

The specified path has profile variations, i.e. is either curved or straight, therefore setting the constant gains initially for the whole simulation often leads to undesirable behaviour in some path segments. Thus, those values were adjusted by trial and error as more simulations were carried out.

The proportional control steers heavier the further the current position is from the desired trajectory, i.e. whenever the cross-track error is considerably high. The overall performance rises when the gain increases, however, it may result that depending on the distance from the trajectory the vehicle starts to drift and have periodic circular motions. With just this control, in scenarios where the vehicle is considered to be deflected to the path, oscillations along it still occur and, therefore, overshoot still exists.

Table 4.1: Method 1: Simulation Parameters

Parameter	8-Shape Path	Ellipse Path
Simulation Time T	160	160
Total Path Length	143.23	142.83
Starting Point (P_{x_0}, P_{y_0})	(22,8)	(15,28)
Velocity (u)	1	1
Turn Radius (R)	10	10
Starting Heading Angle (ψ)	0°	0°
PID Gains	$K_P = 17$	$K_P = 19$
	$K_I = 0.11$	$K_I = 0.075$
	$K_D = 27.3$	$K_D = 25$

Adding a derivative term, the cross-track error rate may be evaluated, allowing the movement in a perpendicular direction, with respect to the desired trajectory, to be checked. Hence, whenever the path is perfectly being followed the rate is zero. Therefore, tuning the derivative gain changes how the vehicle moves towards the path, i.e. if it moves faster or slower. Lower values do not counteract the oscillations and higher values may result in a longer response time to correct the distance offsets.

If there exists misalignment of the vehicle while facing surrounding disturbances (e.g., cross-wind, water swat, ground unevenness), the changes caused may not be perceived, thus this additional offset persists. This is described as being a steady state error which could be minimised by adding an integral term. This term is defined as a sum of the cross-track error and indicates if the vehicle has been moving more on one side of the trajectory rather than the other. Higher values of the integral gain may contribute to instability as small deviations from the path are exaggerated. On the contrary, small values determine a slower response to the changes.

Hence, the gains were used and adjusted to acknowledge these scenarios and provide a resulting trajectory as close as possible to the reference path.

4.2.3 Simulation results of the 8-Shape path

The parameters of the simulation of the 8-shape path are summarised in table A.1.

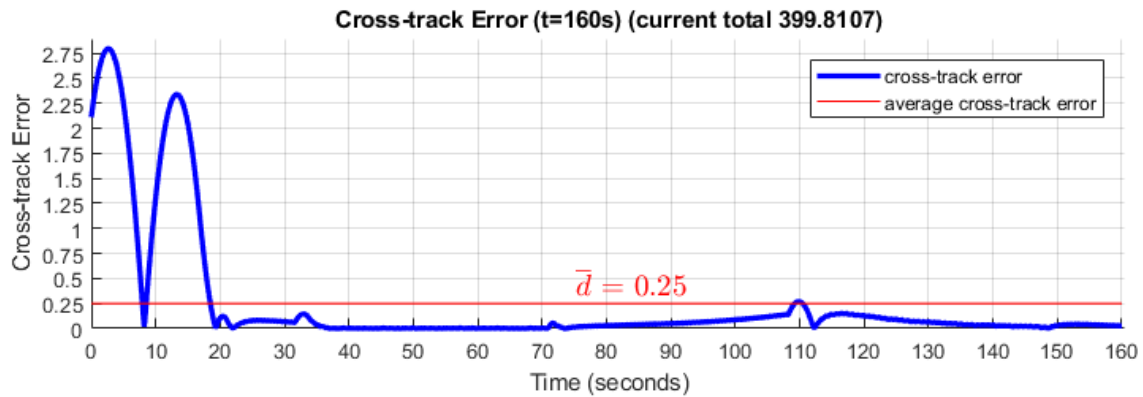


Figure 4.2: Method 1 8-Shape path simulation: cross-track error time variation.

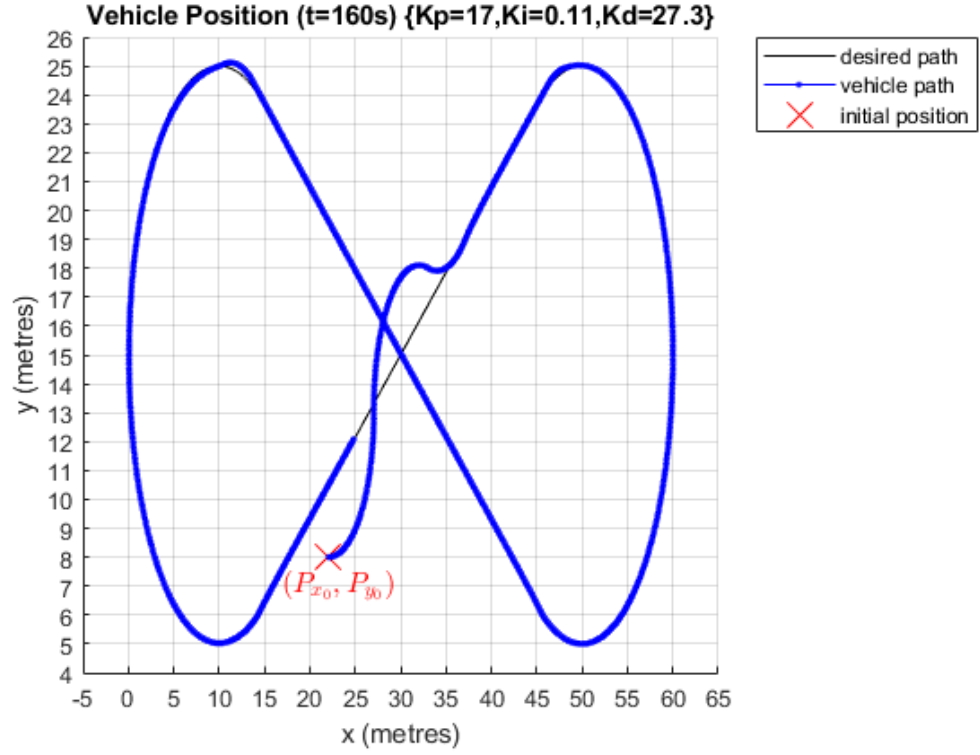


Figure 4.3: Method 1 8-Shape path simulation: desired path and actual vehicle trajectory.

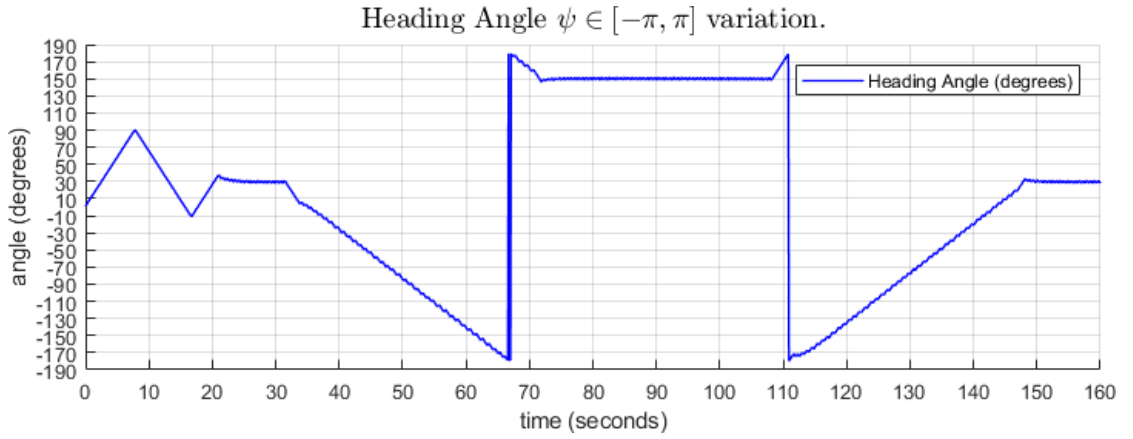


Figure 4.4: Method 1 8-Shape path simulation: heading angle time variation.

Table 4.2: Method 1: 8-Shape path simulation results, where $d_{(P_{x0}, P_{y0})}$ is the initial distance to the path and t_0 the time the vehicle first crosses the path.

Parameter	$d_{(x_0, y_0)}$	t_0	D_{Total} $t \in [0, T]$	$\bar{d}$, $t \in [0, T]$	D_{Total} $t \in [t_0, T]$	$\bar{d}$ $t \in [t_0, T]$	U_{Total} $t \in [0, T]$	U_{Total} $t \in [t_0, T]$
Result	2.11	8.2 s	399.81	0.25	299.5	0.15	59.57	56.34

4.2.4 Simulation results of the Ellipse path

The parameters of the simulation of the ellipse path are summarised in table A.1.

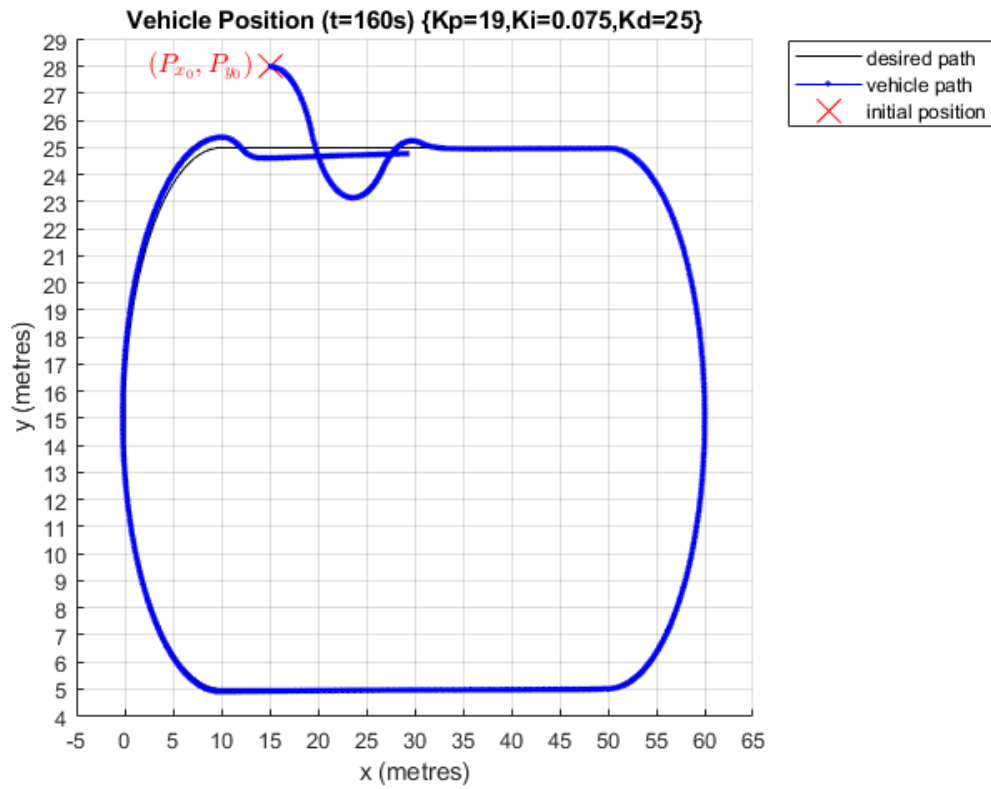


Figure 4.5: Method 1 Ellipse path simulation: desired path and actual vehicle trajectory.

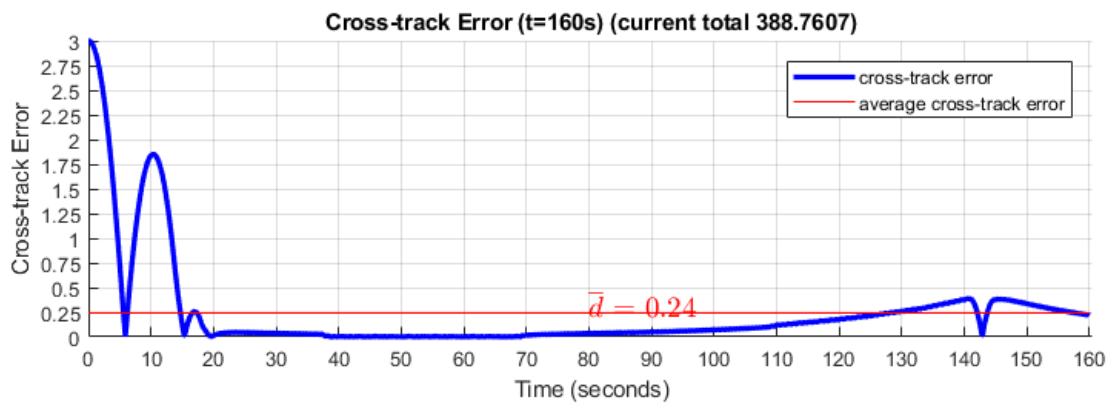


Figure 4.6: Method 1 Ellipse path simulation: cross-track error time variation.

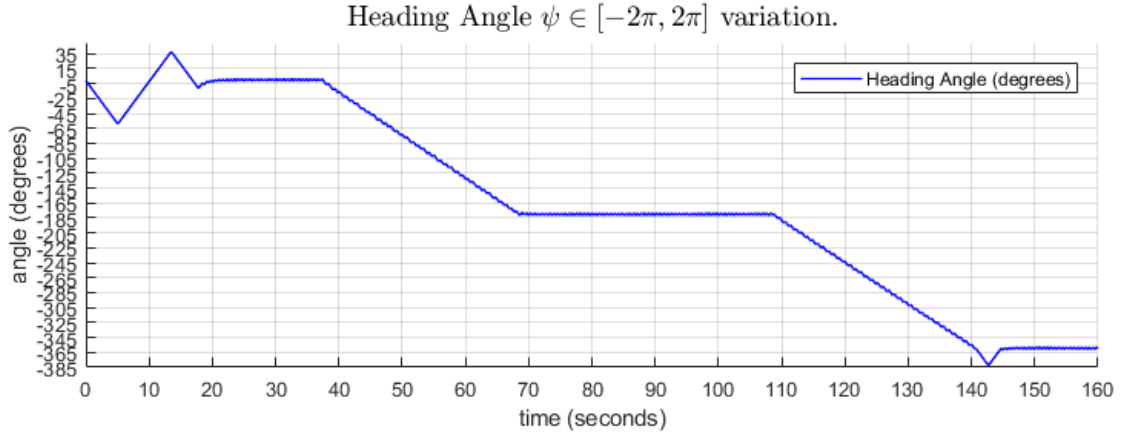


Figure 4.7: Method 1 Ellipse path simulation: heading angle time variation.

Table 4.3: Method 1: Ellipse path simulation results, where $d_{(P_{x0}, P_{y0})}$ is the initial distance to the path and t_0 is the time the vehicle first crosses the path.

Parameter	$d_{(x_0, y_0)}$	t_0	D_{Total} $t \in [0, T]$	$\bar{d}$ $t \in [0, T]$	D_{Total} $t \in [t_0, T]$	$\bar{d}$ $t \in [t_0, T]$	U_{Total} $t \in [0, T]$	U_{Total} $t \in [t_0, T]$
Result	3	5.9 s	388.76	0.24	272.45	0.18	59.77	57.49

4.3 Simulation: Carrot Chase Method (L0 distance)

Method 2 is described in Chapter 3 and is commonly referred to as the L0 distance method. This algorithm uses the VTP concept, being this point ahead on the path with respect to the nearest point Q concept. Thus, the required parameters contrary to the previous method are the $L0$ distance (or δ distance) for straight paths and the α angle for curved paths. This method provides the required computations for the desired angle to adequately steer the vehicle. It is common to add a proportional gain K_P to enhance the system's response, with a behaviour similar to the PID gains description in the Subsection 4.2.1.

4.3.1 Simulation Parameters

Regarding the simulation of Method 2, the used parameters for both the 8-shape and ellipse paths are defined in Table 4.4. It is important to note that the vehicle velocity is considered to be constant, with the heading angle being adjusted with the control $u = K_p(\psi_d - \psi)$. When implementing Algorithm 2 it is noticeable the sequence of computations to determine the parameters for this method, hence next some considerations concerning these parameters are made.

The L0 distance, or δ , is defined first as it is easily restricted by the path length. Starting with small values, the distance of the target point T is close to the nearest point on the path Q , thus the vehicle is impelled to move abruptly towards the path. With such values, this behaviour occurs several times with the vehicle oscillating as it moves along the reference path.

Table 4.4: Method 2: Simulation Parameters

Parameter	8-Shape Path	Ellipse Path
Simulation Time T	38	38
Total Path Length	143.23	142.83
Starting Point (P_{x_0}, P_{y_0})	(22,8)	(15,28)
Velocity (u)	4	4
L0 Distance	4	4
α angle	$-5^\circ = -0.087rad$	$-5^\circ = -0.087rad$
Starting Heading Angle (ψ)	0°	0°
Proportional Gain	$K_P = 2.55$	$K_P = 19$

On the contrary, for high values, the VTP is extremely ahead and important waypoints are often missed, i.e. the vehicle does not acknowledge path profile changes. Also, the vehicle requires more time to converge with the path and, therefore, it yields a high cross-track error cumulative sum.

Even with the L0 distance defined, it is also important to adjust the proportional gain K_P as it allows softer steering. With $K_P = 1$ the required heading angle is given as the difference of the desired heading angle with the current one. Hence, starting at this value, the gain is increased or decreased as smoother or harsher steering manoeuvres are required.

4.3.2 Simulation results of the 8-Shape Path

The parameters of the simulation of the 8-shape path are summarised in Table 4.4.

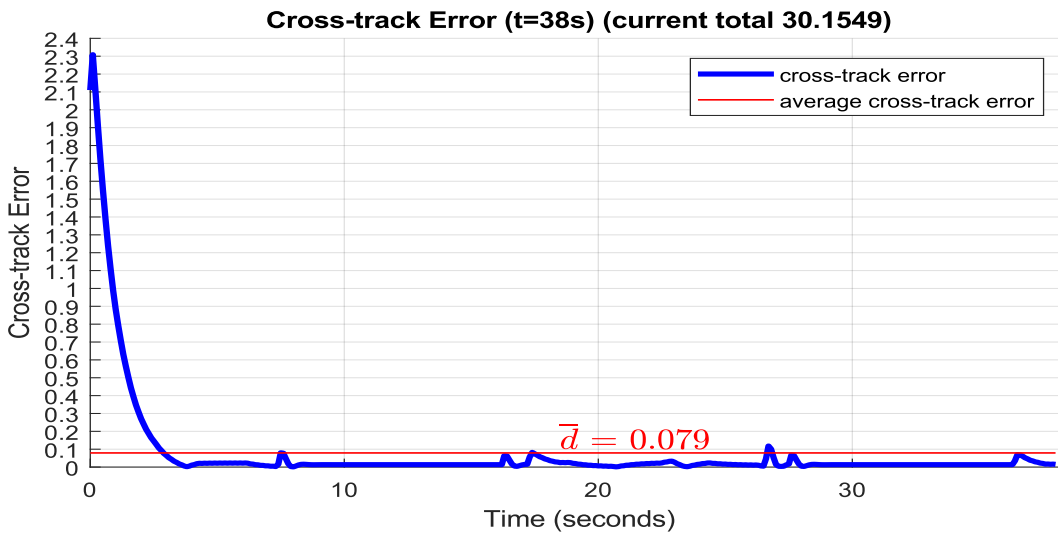


Figure 4.8: Method 2 8-Shape path simulation: cross-track error time variation.

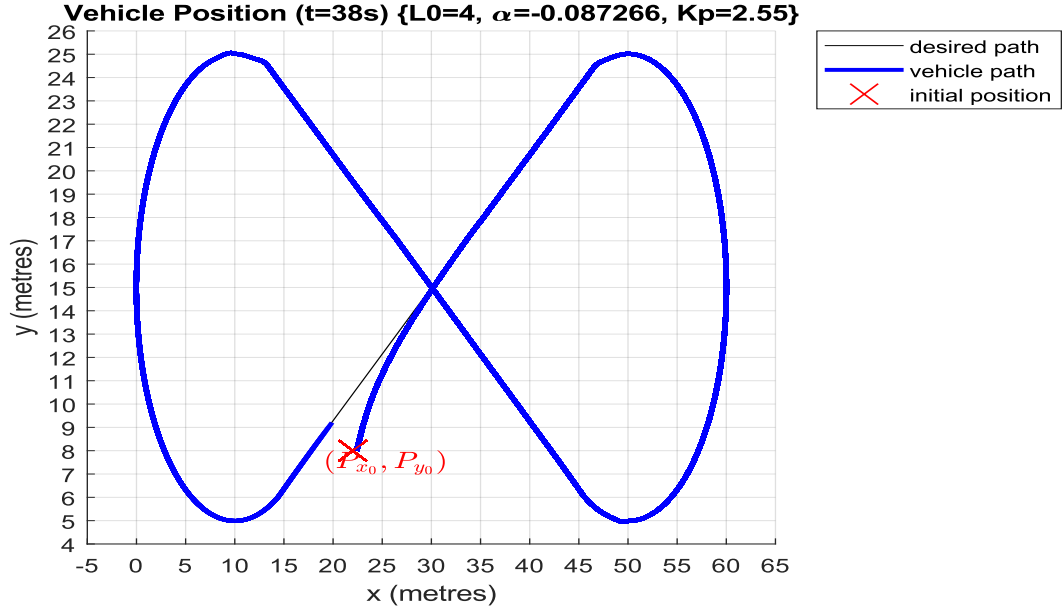


Figure 4.9: Method 2 8-Shape path simulation: desired path and actual vehicle trajectory.

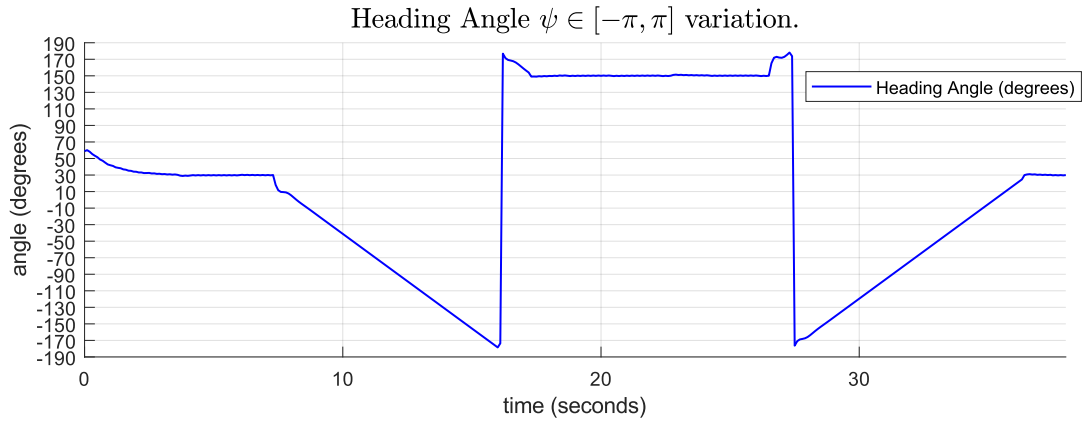


Figure 4.10: Method 2 8-Shape path simulation: heading angle time variation.

Table 4.5: Method 2: 8-Shape path simulation results, where $d_{(P_{x0}, P_{y0})}$ is the initial distance to the path and t_0 the time the vehicle first crosses the path.

Parameter	$d_{(x_0, y_0)}$	t_0	D_{Total} $t \in [0, T]$	$\bar{d}$ $t \in [0, T]$	D_{Total} $t \in [t_0, T]$	$\bar{d}$ $t \in [t_0, T]$	U_{Total} $t \in [0, T]$	U_{Total} $t \in [t_0, T]$
Result	2.11	3.2 s	30.15	0.079	6.39	0.018	0.21	0.20

4.3.3 Simulation results of the Ellipse Path

The parameters of the simulation of the ellipse path are summarised in table 4.4.

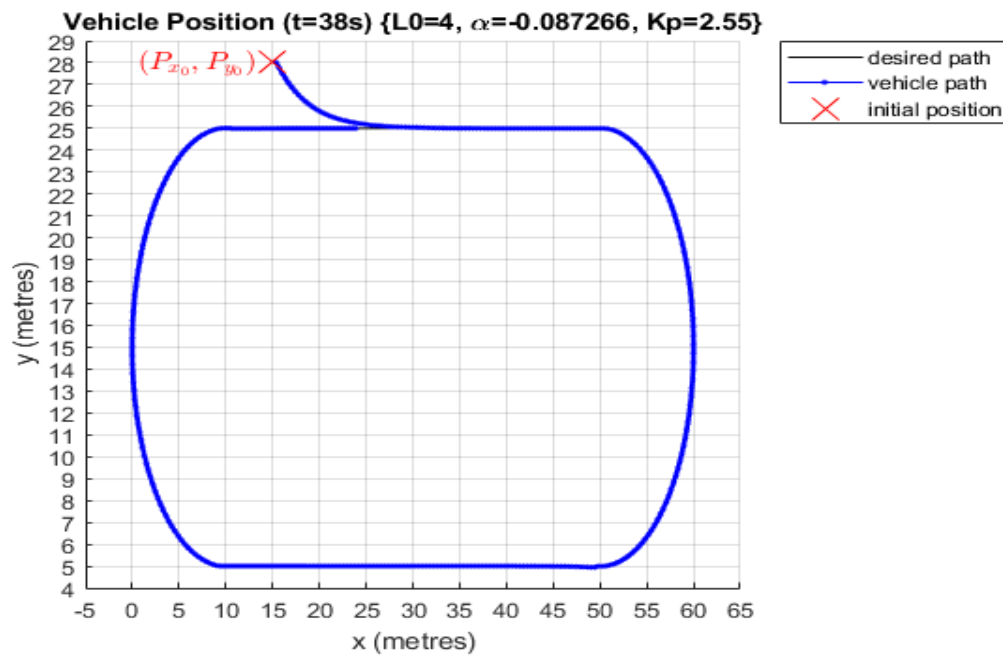


Figure 4.11: Method 2 Ellipse path simulation: desired path and actual vehicle trajectory.

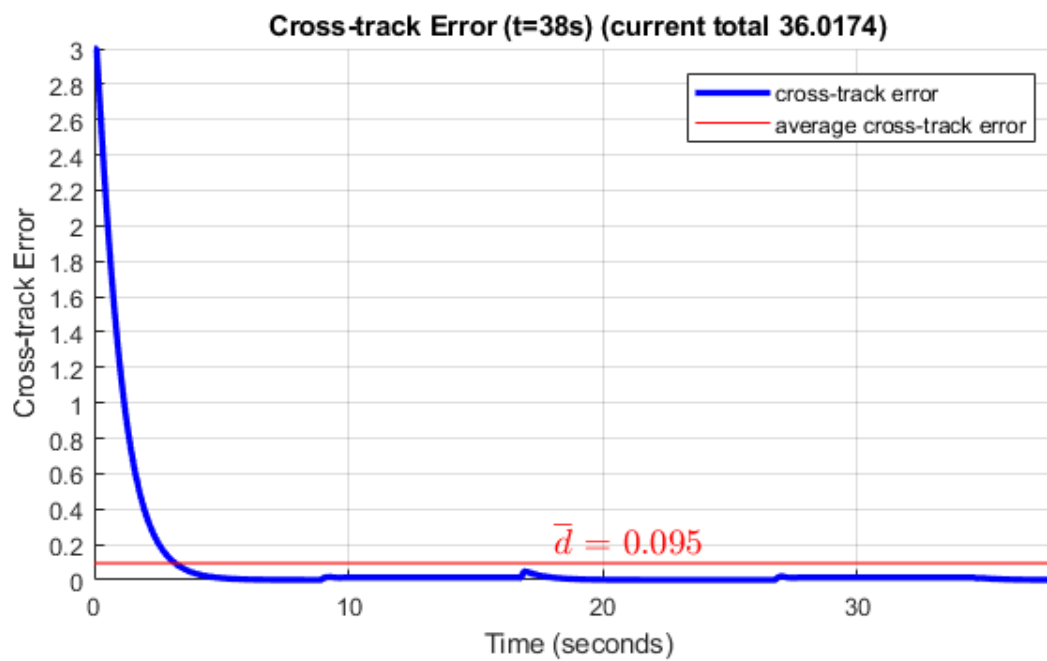


Figure 4.12: Method 2 Ellipse path simulation: cross-track error time variation.

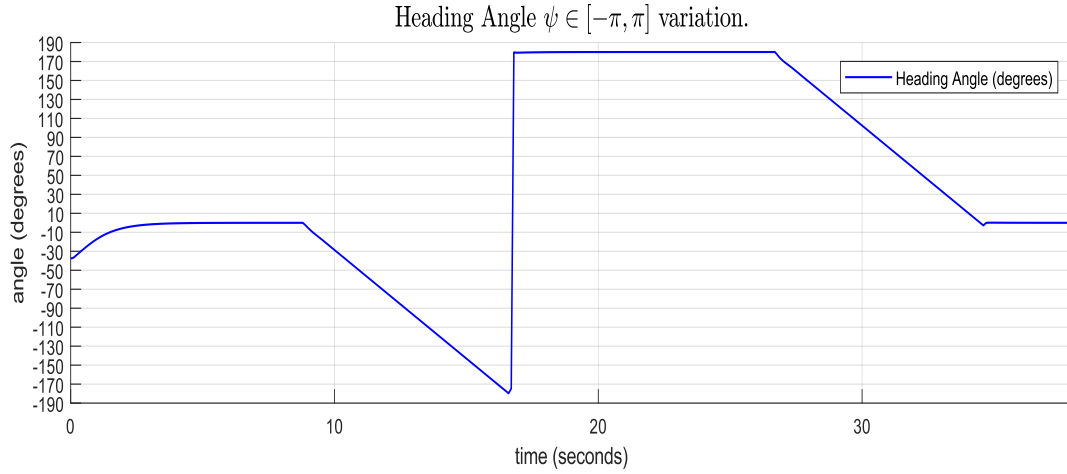


Figure 4.13: Method 2 Ellipse path simulation: heading angle time variation.

Table 4.6: Method 2: Ellipse path simulation results, where $d_{(P_{x0}, P_{y0})}$ is the initial distance to the path and t_0 the time the vehicle first crosses the path.

Parameter	$d_{(x_0, y_0)}$	t_0	D_{Total}	$\bar{d}$	D_{Total}	$\bar{d}$	U_{Total}	U_{Total}
			$t \in [0, T]$	$t \in [0, T]$	$t \in [t_0, T]$	$t \in [t_0, T]$	$t \in [0, T]$	$t \in [t_0, T]$
Result	3	3.8 s	36.02	0.095	3.13	0.009	0.103	0.102

4.4 Simulation: Nonlinear Guidance Logic (L1 distance)

Method 3 is described in Chapter 3. Contrary to the previous two methods, the nonlinear guidance logic does not depend on the path profile to do specific computations that influence the control, i.e. both straight and curved paths are treated the same way. Nevertheless, the control depends on just one design parameter (L1 distance), thus since the velocity is considered constant to simplify the evaluation of the algorithms, the vehicle manoeuvres depend on this constant and the L1 distance. Moreover, the given control holds the centripetal acceleration command which in more specific applications is useful.

4.4.1 Simulation Parameters

Regarding Method 3 simulation, the used parameters for both the 8-shape and ellipse trajectories are defined in Table 4.7.

Concerning the path length, the L1 distance should be chosen such that there is at least one point on the path that is the solution of equation 3.27. Often, this distance is such that initially the path is reachable, facilitating the convergence. However, the VTP is not always within the vehicle's reach, hence these scenarios should be evaluated. Having a moderate L1 distance with respect to the path length is a possibility to avoid similar problems. The L1 distance choice for the

Table 4.7: Method 3: Simulation Parameters

Parameter	8-Shape Path	Ellipse Path
Simulation Time T	160	160
Total Path Length	143.23	142.83
Starting Point (P_{x_0}, P_{y_0})	(22,8)	(15,28)
Velocity (u)	1	1
L1 Distance	7	7
Starting Heading Angle (ψ)	0°	0°

following simulations has these ideas in consideration and, with this value the vehicle converges with the path initially and after path profile transitions, i.e. between straight lines and curves.

As for the velocity, it is set as a small constant value since higher values would make the vehicle hop faster between the different path segments and, therefore, accumulate high cross-track error. This is due to the fact that the vehicle would not start to steer properly at the right times, with this behaviour deprecating as the target point is always ahead.

4.4.2 Simulation results of the 8-Shape path

The parameters of the simulation of the 8-shape path are summarised in table 4.7.

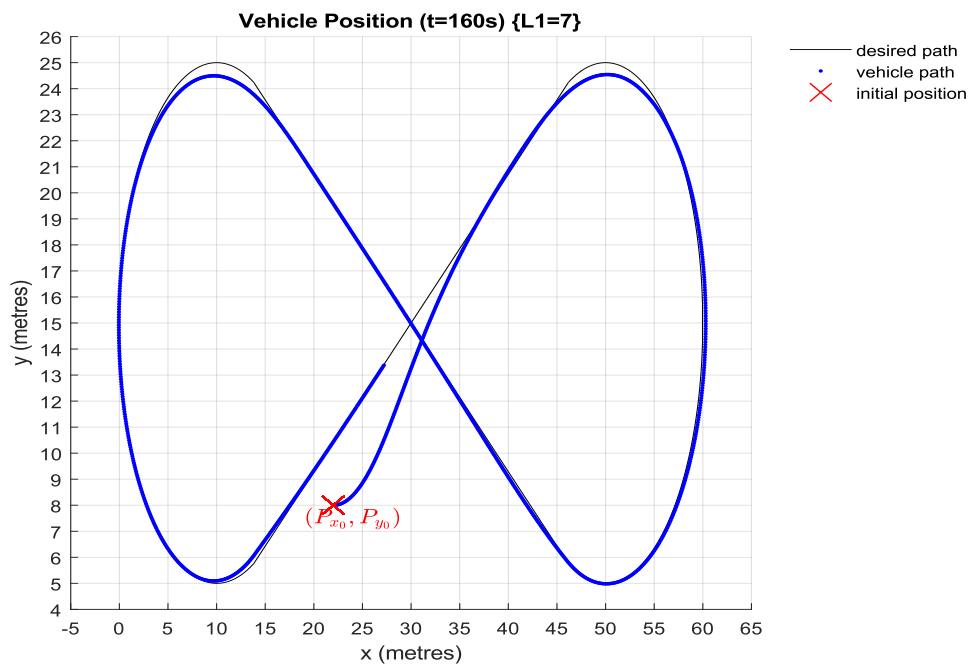


Figure 4.14: Method 3 8-Shape path simulation: desired path and actual vehicle trajectory.

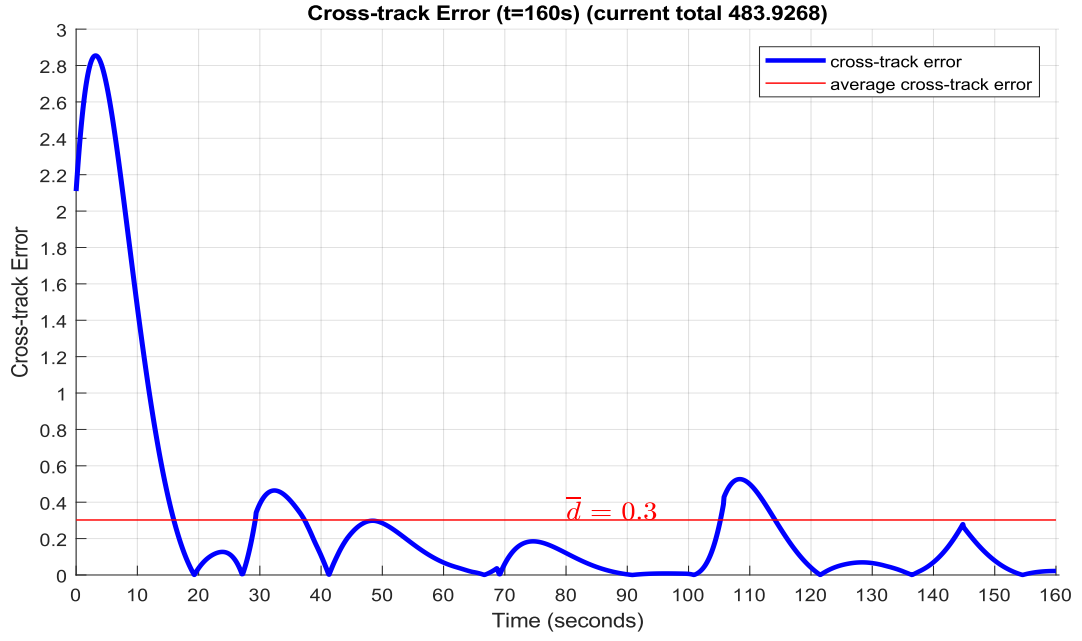


Figure 4.15: Method 3 8-Shape path simulation: cross-track error time variation.

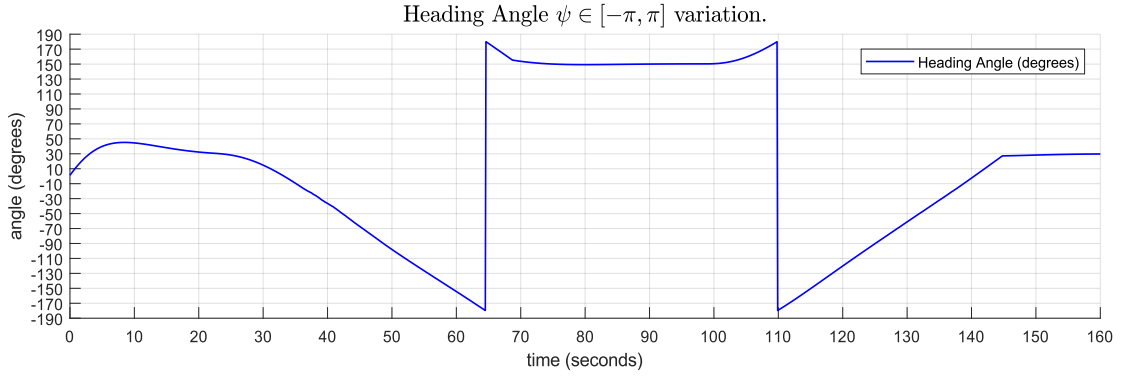


Figure 4.16: Method 3 8-Shape path simulation: heading angle time variation.

Table 4.8: Method 3: 8-Shape path simulation results, where $d_{(P_{x0}, P_{y0})}$ is the initial distance to the path and t_0 the time the vehicle first crosses the path.

Parameter	$d_{(x_0, y_0)}$	t_0	D_{Total} $t \in [0, T]$	$\bar{d}$ $t \in [0, T]$	D_{Total} $t \in [t_0, T]$	$\bar{d}$ $t \in [t_0, T]$	U_{Total} $t \in [0, T]$	U_{Total} $t \in [t_0, T]$
Result	2.11	18.6 s	483.93	0.302	189.93	0.134	9.19	8.07

4.4.3 Simulation results of the Ellipse Path

The parameters of the simulation of the ellipse path are summarised in table 4.7.

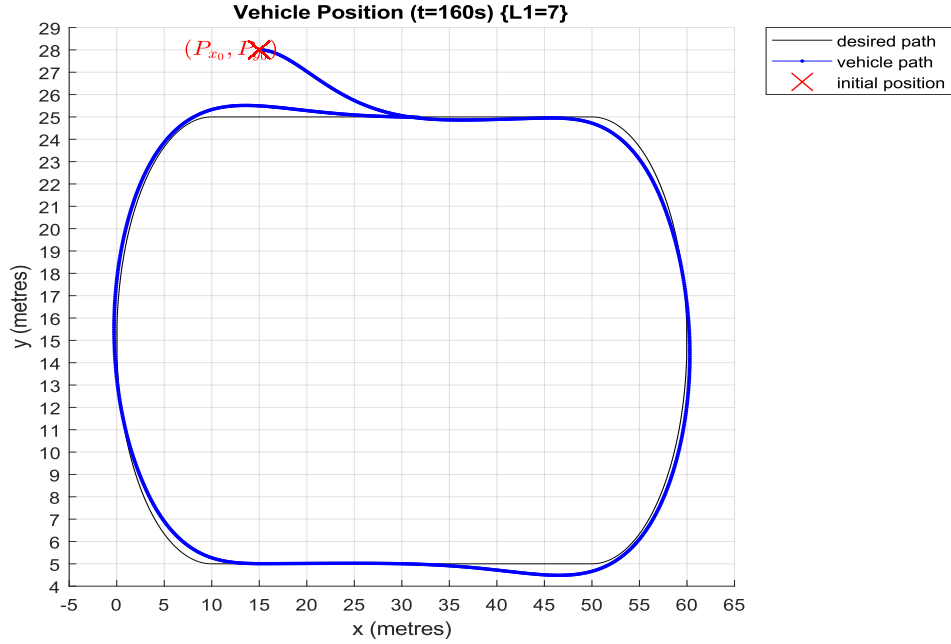


Figure 4.17: Method 3 Ellipse path simulation: desired path and actual vehicle trajectory.

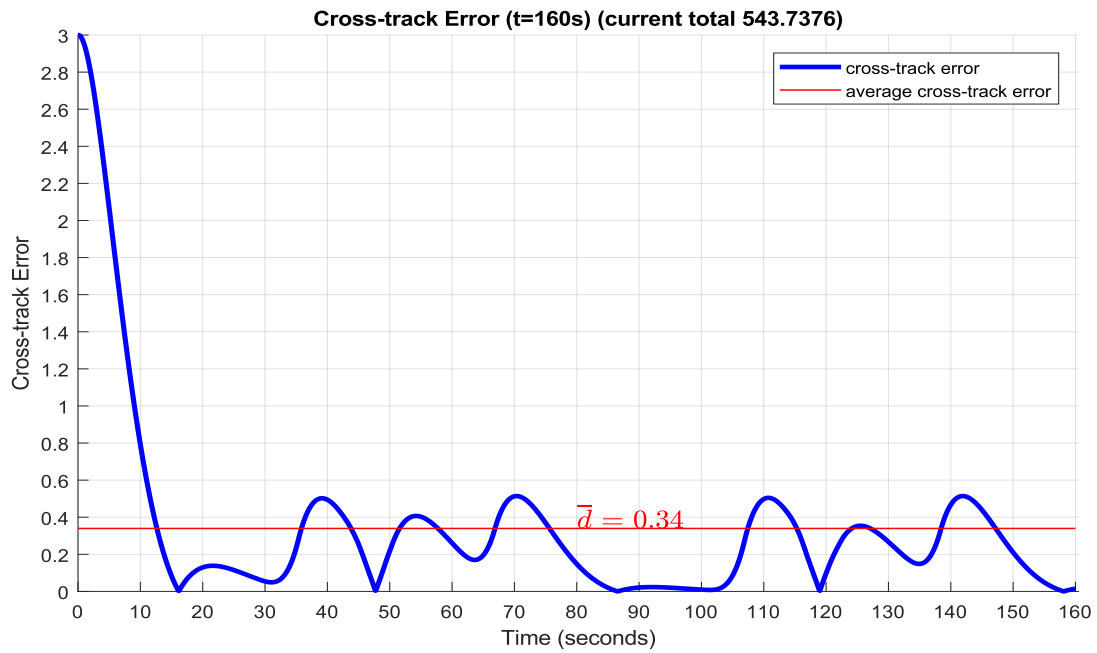


Figure 4.18: Method 3 Ellipse path simulation: cross-track error time variation.

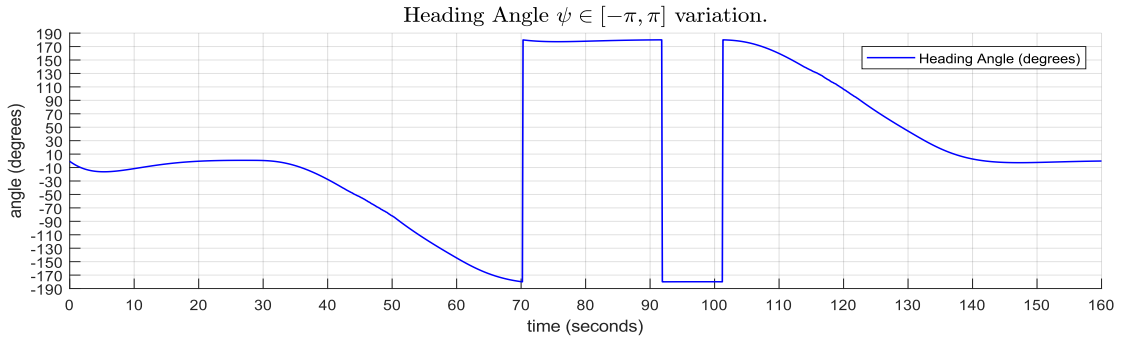


Figure 4.19: Method 3 Ellipse path simulation: heading angle time variation.

Table 4.9: Method 3: Ellipse path simulation results, where $d_{(P_{x0}, P_{y0})}$ is the initial distance to the path and t_0 the time the vehicle first crosses the path.

Parameter	$d_{(x_0, y_0)}$	t_0	D_{Total} $t \in [0, T]$	$\bar{d}$, $t \in [0, T]$	D_{Total} $t \in [t_0, T]$	$\bar{d}$ $t \in [t_0, T]$	U_{Total} $t \in [0, T]$	U_{Total} $t \in [t_0, T]$
Result	3	15.5 s	543.74	0.34	320.42	0.222	6.03	5.76

4.5 Discussion

Throughout this chapter the path-following algorithms presented in Chapter 3 were simulated in two pre-defined paths: 8-shape and ellipse figures. These paths allow the evaluation of the different algorithms in trajectories that may be assigned to the vehicle's mission. The scheme with both straight and curved profiles is useful to assess how the different implemented controllers handle the situations where after convergence there is a change in the path profile. The simulations were performed with the control over the car-like system curvature or heading angle. In addition, the velocity was kept constant but also tested, as some algorithms would increase or decrease the overall cross-track error according to the combination of other simulation parameters (e.g. gains, distance parameters, initial position and orientation).

Considering the simulations of the 8-shape path, the results for the three algorithms are presented in Tables 4.2, 4.5 and 4.8. As for the ellipse path, the results are in Tables 4.3, 4.6 and 4.9.

These tables contain the starting distance to the path is given by $d_{x0, y0}$, the instant of time t_0 that indicates when the vehicle first crossed the path, i.e. first time is zero cross-track error, the sum of the cross-track error D_{Total} , the average cross-track error $\bar{d}$ and the total control effort U . These last three parameters are shown for both intervals $t \in [0, T]$ and $t \in [t_0, T]$ with T being the simulation time.

Both the 8-shape and ellipse paths were simulated considering that the vehicle would move along the path segments while executing a finite state machine. This strategy allows for the sequential follow of the segments, even when there exists any nearby closest point of a future, but not the next, state.

As for the results of the simulation, the first method takes some time to converge and oscillates until it does. It employs the PID controller to which acquiring the best gains is not trivial and, although the cross-track error is minimised with a small average value, of the three methods it has the highest control effort. Thus, the controller keeps correcting the heading angle of the vehicle, with very small oscillations that could be imperceptible.

The second method performs better than the other two. The VTP ahead on the path, the proportional gain, the L0 distance and the evaluation of the path profile are advantageous. The L0 distance with moderate values allows for a smooth approximation to the path, instead of oscillating which happens for low values. With the evaluation of the path profile, it is possible to specify the VTP and have, therefore, accurate path transitions. As with the tuning of the proportional gain, it becomes possible to fix abrupt manoeuvres. Nevertheless, this method holds the lowest result values of all three, except for the initial distance.

The third method gives intermediate results when compared to the other two. The path length specified limits the maximum L1 distance and, therefore, only a small range of values is possible. The higher the distance, the more ahead the target is, thus there exists a tendency for a smooth initial convergence. This has the cost of accumulating high cross-track until it finally converges. If low values are considered, they may not be viable since there could be no point on the path within reach.

Chapter 5

AWES Path-Following Guidance and Simulation

In a system with an AWES of the type rigid wing, the device is set to follow a time-independent 3D predefined path with a fast crosswind motion performing 8-shaped or elliptical periodic trajectories [13]. The attained energy production deprecates if the current kite movement and followed path deviates significantly from the desired and optimal trajectory. This happens due to the production and consumption phases that are inherent to the rigid wing AWES implementation that is being considered.

In this chapter the 3D mass-point kite power system model [29] that has been used in 3D Simulations within the UPWIND Project is described. The dynamic model of the rigid-wing tethered kite is necessary for the interpretation of the kite motion and to adequately design the control system for its guidance. Section 5.1 presents the different coordinate systems necessary to describe the kite motion, and Section 5.2 details the acting forces over the kite along with the system dynamics.

This Chapter also describes the path-following guidance which has been applied in the UPWIND project. In Section 5.3 the idea of the path-following implemented within the project is characterized. In section 5.4 the guidance methods previously seen in Chapter 3 are linked to the AWES application. In Section 5.5, some results concerning the implementation of the L0 distance guidance algorithm in the kite's model are illustrated.

5.1 Coordinate Systems

The model of the kite is defined through the standard definition of the considered reference frames and coordinate systems: body coordinate system (coupled to the kite body), local coordinate system and global coordinate system [13], [16] depicted in Figure 5.1.

The Global Coordinate System, G, is an inertial cartesian coordinate system (x,y,z) with the origin on the ground. Usually, it coincides with the ground station where one of the ends of the

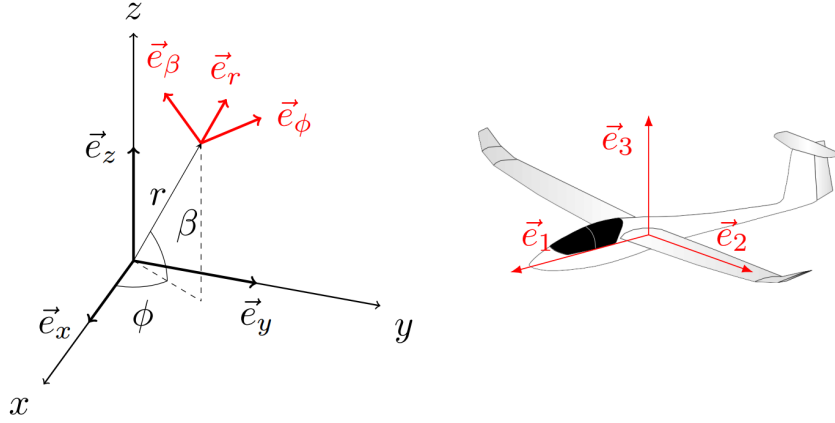


Figure 5.1: Coordinate Systems [12].

tether is attached. The basis of this system is defined as $(\vec{e}_x, \vec{e}_y, \vec{e}_z)$ with the x -axis pointing towards the main wind direction.

The Local Coordinate System, L, is a non-inertial spherical coordinate system (r, ϕ, β) with basis $(\vec{e}_r, \vec{e}_\phi, \vec{e}_\beta)$.

The Body Coordinate System, B, is a non-inertial cartesian coordinate system with basis $(\vec{e}_1, \vec{e}_3, \vec{e}_2)$ and origin at the centre of gravity of the kite body. Considering a glider type AWES, the x -axis points through the nose, the y -axis points through the right-hand side wing and the z -axis points down.

Considering the coordinate systems defined above, the position of the kite is given by

$$\mathbf{p} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}_G = \begin{bmatrix} r \cos(\beta) \cos(\phi) \\ r \cos(\beta) \sin(\phi) \\ r \sin(\beta) \end{bmatrix}_G \quad (5.1)$$

with $\mathbf{p} = [r, \phi, \beta]^T_L$.

The associated rotation matrix from the Local coordinate system to the Global coordinate system:

$$R_{LG} = \begin{bmatrix} \vec{e}_r & \vec{e}_\phi & \vec{e}_\beta \end{bmatrix} = \begin{bmatrix} \cos(\beta) \cos(\phi) & -\sin(\phi) & -\sin(\beta) \cos(\phi) \\ \cos(\beta) \sin(\phi) & \cos(\phi) & -\sin(\beta) \sin(\phi) \\ \sin(\beta) & 0 & \cos(\beta) \end{bmatrix} \quad (5.2)$$

The associated rotation matrix from the Global coordinate system to the Local coordinate

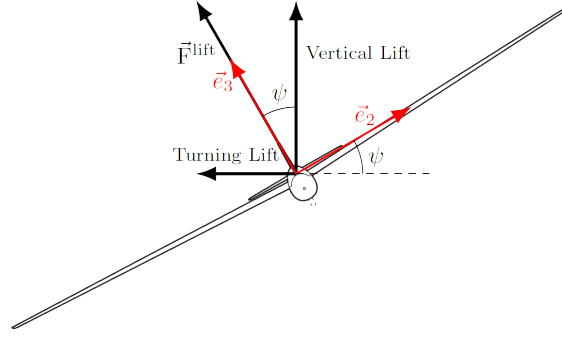


Figure 5.2: Kite roll angle and turning dynamics [13].

system:

$$R_{GL} = R_{LG}^{-1} = R_{LG}^T = \begin{bmatrix} \vec{e}_r^T \\ \vec{e}_\phi^T \\ \vec{e}_\beta^T \end{bmatrix} = \begin{bmatrix} \cos(\beta)\cos(\phi) & \cos(\beta)\sin(\phi) & \sin(\beta) \\ -\sin(\phi) & \cos(\phi) & 0 \\ -\sin(\beta)\cos(\phi) & -\sin(\beta)\sin(\phi) & \cos(\beta) \end{bmatrix} \quad (5.3)$$

Being the kite an aerodynamic lifting device, the apparent wind velocity vector quantifies the relation between the wind velocity vector, $\mathbf{v}_w$ and the flight velocity vector, $\mathbf{v}_k$, relative to the stationary ground station. The apparent wind velocity, $\mathbf{v}_a$, is defined as:

$$\mathbf{v}_a = \mathbf{v}_w - \mathbf{v}_k = \mathbf{v}_w - \dot{\mathbf{p}} \quad (5.4)$$

Assuming the longitudinal axis of the body of the kite aligned with the apparent wind velocity results

$$\vec{e}_1 = \frac{-\mathbf{v}_a}{\|\mathbf{v}_a\|} \quad (5.5)$$

The basis of the Body Cartesian coordinate system $(\vec{e}_1, \vec{e}_3, \vec{e}_2)$ for a rigid-wind kite has its unit vectors $\vec{e}_1$ aligned with the longitudinal axis and pointing forward, $\vec{e}_2$ pointing towards the left wing and $\vec{e}_3$ pointing upwards. The $\vec{e}_2$ and $\vec{e}_3$ are both portrayed on Fig.5.2.

With ψ being the roll angle around the longitudinal axis $\vec{e}_1$, assuming that initially the roll angle is $\psi = 0$, then let $\tilde{\mathbf{e}}_2 = \vec{e}_2$. With $\tilde{\mathbf{e}}_2$ contained in the plane τ tangent to a sphere centered at the origin which contains the axis $\vec{e}_\beta$ and $\vec{e}_\phi$ it results that $\tilde{\mathbf{e}}_2 \perp \vec{e}_r$ and $\tilde{\mathbf{e}}_2 \perp \vec{e}_1$ [29].

Therefore, $\tilde{\mathbf{e}}_2$ is given as

$$\tilde{\mathbf{e}}_2 = \frac{\vec{e}_r \times \vec{e}_1}{\|\vec{e}_r \times \vec{e}_1\|} \quad (5.6)$$

Assuming that the roll angle ψ is directly controlled, $\tilde{\mathbf{e}}_2$ may be rotated ψ along the axis $\vec{e}_1$

using the following Rodrigues' formula:

$$\vec{e}_2 = \tilde{e} \cos \psi + (\vec{e}_1 \times \tilde{e}_2) \sin \psi + \vec{e}_1 (\vec{e}_1 \cdot \tilde{e}_2) (1 - \cos \psi) \quad (5.7)$$

with $\vec{e}_3$ resulting as the cross product

$$\vec{e}_3 = \vec{e}_1 \times \vec{e}_2. \quad (5.8)$$

5.2 Acting Forces and Dynamic Model

The rigid wing of the kite acts as an aerodynamic control surface, thus changing the aerodynamic forces and moments that adjust its flight motion [30]. Therefore, the aerodynamic forces must be considered for an adequate dynamic model of the kite. Moreover, considering the energy harvesting purpose of the device, tether length variations, during the two-phase generation cycle, and respective tension force must also be considered. The complete model allows the assessment of the kite's flight path and the harvested energy.

Newton's second law of motion equation applied to the kite system is as follows

$$m\ddot{\mathbf{p}} = \vec{F}^{th} + \vec{F}^{grav} + \vec{F}^{aer}(\alpha) \quad (5.9)$$

The above expression relates the kite mass and acceleration with the tether force that acts on the kite, $\vec{F}^{th}$, the gravity force, $\vec{F}^{grav}$, and the resultant aerodynamic force, $\vec{F}^{aer}(\alpha)$.

Considering the tether to be inelastic and massless, the tether force is expressed as:

$$\vec{F}^{th} = -T_{tether} \vec{e}_r = \begin{bmatrix} -T_{tether} \\ 0 \\ 0 \end{bmatrix}_L \quad (5.10)$$

being the tension force, T_{tether} , commonly measured on the ground station.

As for the gravity force:

$$\vec{F}^{grav} = -mg \vec{e}_z = \begin{bmatrix} 0 \\ 0 \\ -mg \end{bmatrix}_G = \begin{bmatrix} -mg \sin(\beta) \\ 0 \\ -mg \cos(\beta) \end{bmatrix}_L \quad (5.11)$$

for a given kite with mass m

Concerning the aerodynamic force applied on the kite, it is just given by the sum of the drag and lift forces:

$$\vec{F}^{aer}(\alpha) = \vec{F}^{lift}(\alpha) + \vec{F}^{drag}(\alpha) \quad (5.12)$$

with the Lift Force, $\vec{F}^{lift}(\alpha)$, aligned with axis $\vec{e}_3$, and the Drag Force, $\vec{F}^{drag}(\alpha)$, aligned with the axis $\vec{e}_1$. In addition, both lift and drag forces depend on the airfoil parameters and design, having respectively lift and drag coefficients expressed as $c_L(\alpha)$ and as $c_D(\alpha)$ [13].

The referred lift and drag forces are defined as

$$\vec{F}^{lift}(\alpha) = \frac{1}{2} \rho A c_L(\alpha) \|\mathbf{v}_a\|^2 \vec{e}_3 \quad (5.13)$$

$$\vec{F}^{drag}(\alpha) = -\frac{1}{2} \rho A c_D(\alpha) \|\mathbf{v}_a\|^2 \vec{e}_1 \quad (5.14)$$

Thus results that

$$\vec{F}^{aer}(\alpha) = \frac{1}{2} \rho A \|\mathbf{v}_a\|^2 (c_L(\alpha) \vec{e}_3 - c_D(\alpha) \vec{e}_1) \quad (5.15)$$

depending on the angle of attack, α , of the wing of the kite and the square of the apparent wind velocity.

For the spherical coordinate system (r, ϕ, β) , according to [29],[13] we have that $m\ddot{\mathbf{p}}$ may be rewritten as:

$$m\ddot{\mathbf{p}} = m \begin{bmatrix} \ddot{r} \\ r\ddot{\phi}\cos(\beta) \\ r\ddot{\beta} \end{bmatrix}_L = \vec{F}^{th} + \vec{F}^{grav} + \vec{F}^{aer}(\alpha) + \vec{F}^{inertial} \quad (5.16)$$

with the inertial force being $\vec{F}^{inertial} = \vec{F}^{centrifugal} + \vec{F}^{Coriolis}$ in the non-inertial coordinate system.

Finally, the state-space model used in simulations of the kite trajectory is given with the following dynamic equation [13]:

$$\dot{\mathbf{x}} = f(\mathbf{x}(t), \mathbf{u}(t)) = \frac{d}{dt} \begin{bmatrix} r \\ \phi \\ \beta \\ \dot{r} \\ \dot{\phi} \\ \dot{\beta} \end{bmatrix} = \begin{bmatrix} \dot{r} \\ \dot{\phi} \\ \dot{\beta} \\ a_t \\ \frac{1}{mr\cos(\beta)} F_\phi(\alpha, \psi) \\ \frac{1}{mr} F_\beta(\alpha, \psi) \end{bmatrix} \quad (5.17)$$

being the state $\mathbf{x} = (r, \phi, \beta, \dot{r}, \dot{\phi}, \dot{\beta})$ and control $\mathbf{u} = (a_t, \alpha, \psi)$, where a_t is the direct control of the tether acceleration, $\ddot{r}$, exerted by the winch located at the ground station upon the reel-in and reel-out cycle phases, α the angle of attack and ψ the roll angle.

5.3 AWES Path-following Model

Considering the coordinate systems portrayed in Figure 5.1, for a given AWE system with a tether with constant length r , the kite moves on the surface of a sphere of radius equal to the tether length

r . Moreover, the position of the kite is defined according to the azimuth and elevation angles as $\mathbf{p}(\phi, \beta)$. Thus, the 3D desired path is reduced to a 2D path parameterized in the (ϕ, β) space, usually periodic with an elliptical shape or figure-of-eight shape [31], [12].

In order to maintain the trajectory of the kite as close as possible to the time-independent predefined path in the (ϕ, β) space, its heading angle must be controlled. If the roll angle, ψ , around the longitudinal axis of the kite, is considered to be directly controllable, then changing the roll angle provides a lateral component to the lift force designated as Turning Lift (see Figure 5.2).

Hence, this turning lift is associated with the lateral acceleration, a_l , responsible for steering the kite within the (ϕ, β) plane [31]. This lateral acceleration is given by

$$a_l = \frac{1}{m} F^{lift} \sin(\psi) \quad (5.18)$$

where m is mass, F^{lift} the lift force and ψ the roll angle.

For a given path, the path-following controller steers the kite by changing its roll angle. While the airborne device moves, the distance between its position $\mathbf{p}(\phi, \beta)$ and the nearest point, Q , on the reference path is designated as cross-track error, d . In addition, ϱ is the angle between the kite velocity vector, $\mathbf{V} = \dot{\mathbf{p}}$, with the tangent to the path.

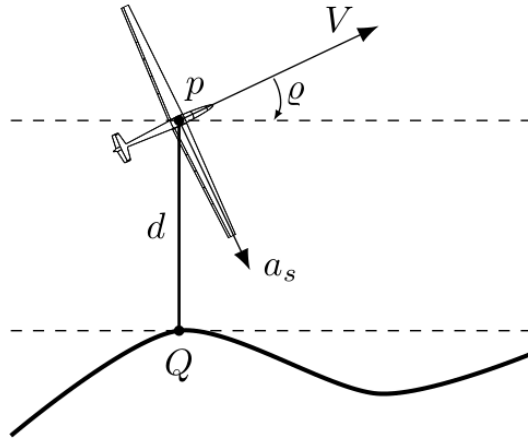


Figure 5.3: Path-following Model [13]

5.4 AWES Path-following Guidance Logic and Control

Chapter 3 details three guidance methods to steer the vehicle through a pre-defined path. The first one describes the distance to the closest point in the path obtained through the path geometrical relations. However, it does not have a specific formulation to compute the required steering angle which is required. In addition, for the simplified kinematic model, the control effort is considerably high.

Hence, considering that it is possible to steer the kite through lateral acceleration commands, it is convenient to use the nonlinear guidance logic, or L1 distance, along with the carrot chase (L0 distance) method. In practice, these two algorithms have been applied to autonomous vehicles for path-following missions, in particular the L1 distance with the autopilot system Ardupilot [32].

Considering Figure 5.4, the centripetal acceleration of the kite is adjusted over time with the main purpose of making the kite's position converge with the desired path. For that, some reference point on the desired path, located ahead of the kite's current position, is firstly considered, and then the controller adjusts accordingly the acceleration and roll angle that grants the needed lifting force. The distance between the vehicle and the reference point is the mentioned L1 distance which is a design parameter that allows to adjust how steep it is the convergence to the path [13], [16].

As already detailed in Section 3.2.4.3, with the designated reference point, the required lateral acceleration, a_s , for the vehicle to move towards its target while following a curved trajectory is given by the expression:

$$a_{s_{cmd}} = 2 \frac{V^2}{L_1} \sin(\eta) \quad (5.19)$$

The above control input was used for the car-like model simulation in Chapter 4. It was discussed the problem of the non-existence of any target points ahead on the path that would be at an L1 distance from the vehicle. Thus, to avoid having coupled strategies to solve the regions where this could happen, the L0 distance is used instead.

The L0 controller is suggested in [13] for the AWES application. This controller applies the base idea of the path-following model seen in Section 3.2.4.2 and Section 5.3. The main difference compared to the L1 logic is the fact that the distance to be considered is the distance between the nearest point on the path, Q , and the target reference point on the path that is ahead, R . Also, the L0 distance is a design parameter to be defined for the controller of the kite, instead of the L1 distance. Both L1 and L0 guidance logics are depicted in Figure 5.5.

Therefore, it yields that L1 is computed by the following expression:

$$L1 = \sqrt{d^2 - L_1^2} \quad (5.20)$$

being d the distance between the current position, P , and the nearest point on the path, Q .

Knowing the L1 distance, it is possible to solve equations 5.18 and 5.19 to compute the roll angle:

$$\psi_{ref} = \arcsin\left(\frac{2mV^2 \sin(\eta)}{F^{lift} L_1}\right) \quad (5.21)$$

Similarly to Algorithm 2, having the required roll angle it is possible to apply the control input:

$$\dot{\psi} = \frac{1}{I} K_P (\psi - \psi_{ref}) \quad (5.22)$$

with K_p being the proportional gain. Since it is being considered the roll angle along the kite's longitudinal axis, the moment of inertia I is added to the expression [13].

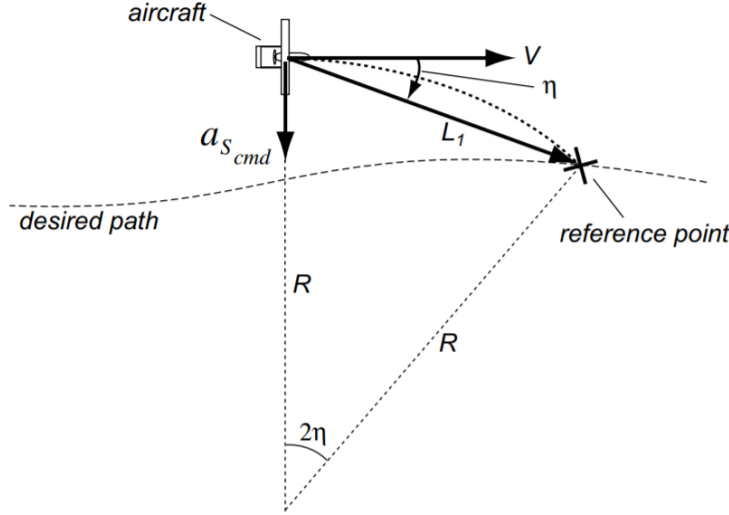


Figure 5.4: Kite guidance logic [13]

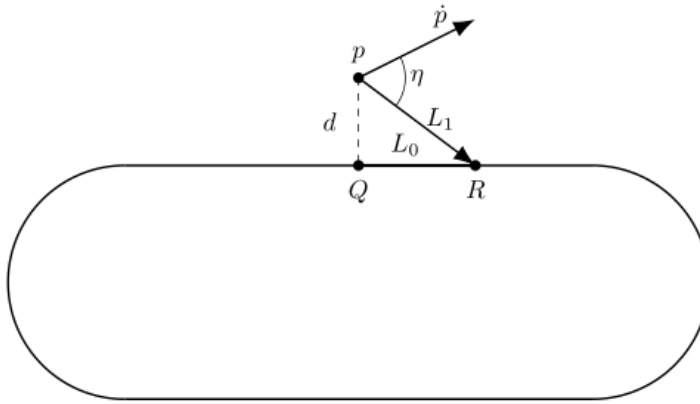


Figure 5.5: L0 and L1 guidance logics [13]

5.5 AWES Simulation

This section comprises simulations made with the software already developed previously by the UPWIND Team. It implements the dynamic model seen in Section 5.2 and the control strategy mentioned earlier. However, the path specification changes when compared to the car-like simulations. In the car-like simulations, the ellipse was defined in xy-plane with the arcs that would sweep from $-\pi$ to π with an offset. For the AWES simulation, the reference trajectory is a curve

parameterized by (ϕ, β) space on the surface of the sphere. Thus, considering the ellipse, the curve is closed with two straight lines and two arcs defined in radians, or degrees.

With both the dynamical model and chosen guidance algorithm, through Simulink it is possible to test the system. Mainly, test steep approximations to the desired path, deviations and oscillations along the path, loss of vehicle orientation, overshooting, error accumulation with an increase of the distance from the vehicle to the path (cross track error) and even the reversal of the flight direction [13].

Regarding the simulation, the parameters are described in the Table 5.1. The vehicle wing reference area, the aerodynamic coefficients and the fluid values used are summarized in Table 5.2.

Table 5.1: Simulation parameters

Parameter	Ellipse Figure
Simulation Time	60 s
ϕ	$\phi \in [-30^\circ, 30^\circ]$
β	$\beta \in [-15^\circ, 15^\circ]$
Starting Point (ϕ_0, β_0)	$(30^\circ, 25^\circ)$
Tether Length	$r \in [50, 250] m$
L0	60°

The Kite starts with the tether reeled out at 50 metres and starts to move towards the desired elliptical trajectory. The 3D trajectory that the kite has during this traction phase is portrayed in Figure 5.6. As the kite increases in altitude, it is controlled to track the desired path in the (ϕ, β) space on the surface of the sphere, as depicted in figure 5.7.

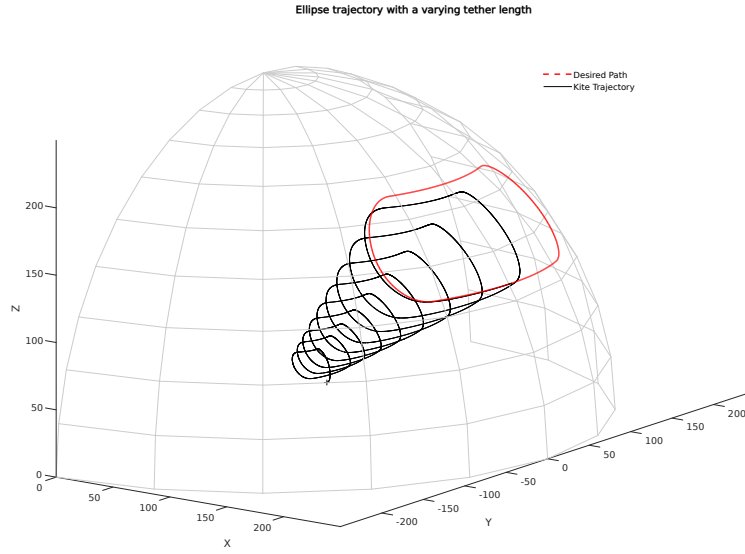
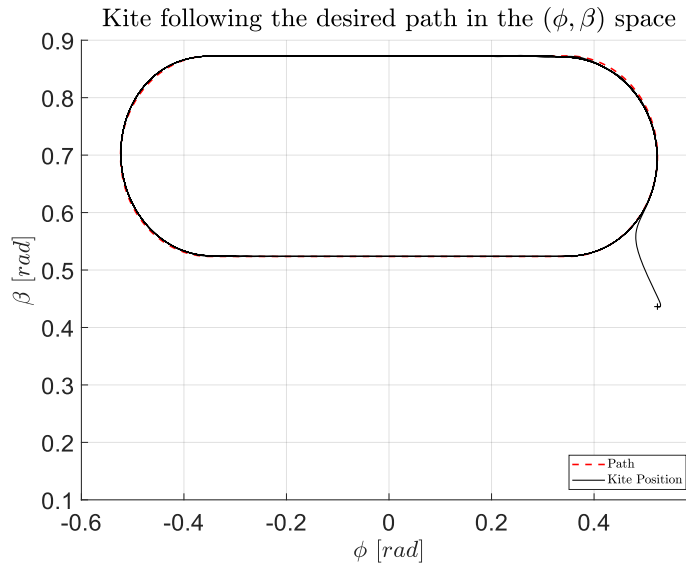
In Section 5.2, equation 5.17, the state-space model used within the current simulation is mentioned. The state vector $\mathbf{x}$ variables (r, ϕ, β) are illustrated in Figure 5.9 and describe the kite's behaviour during the simulation. The reeled-out tether length keeps increasing until it reaches near its maximum admissible length, which takes approximately 60 seconds. This variation describes the increase in altitude of the kite's body.

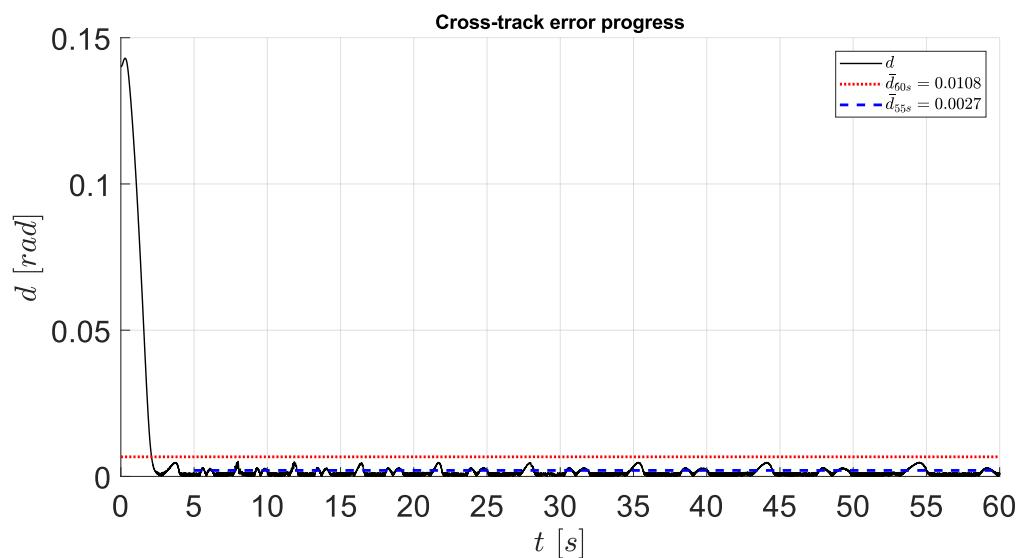
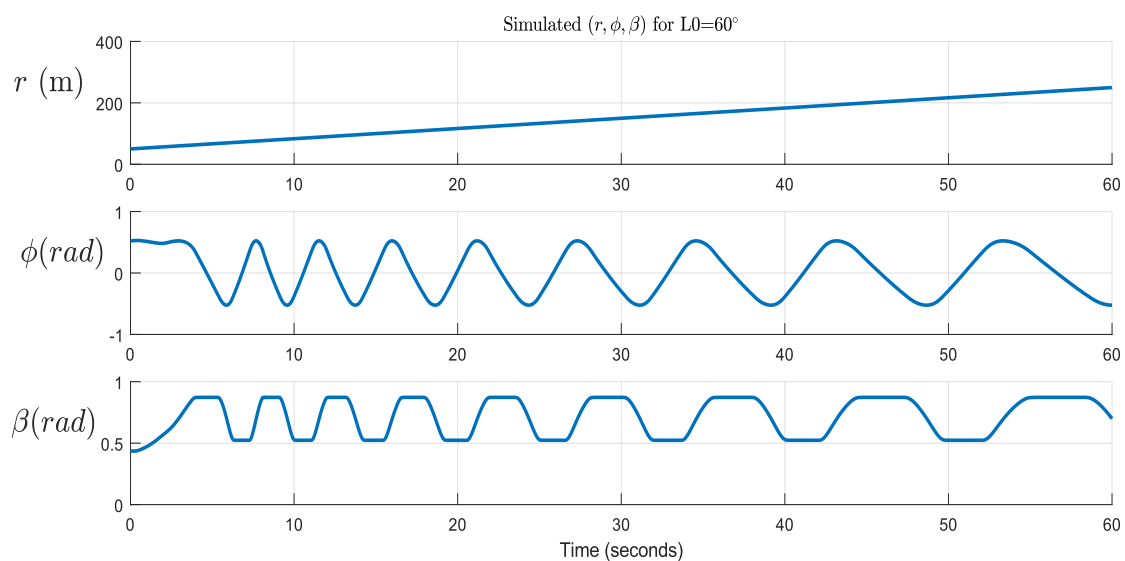
The kite tracks the path with change in the ϕ and β state variables, diminishing the cross-track error. The variations of these variables become periodic as the kite effectively converges with the path. These variations are illustrated in Figure 5.9.

The system illustrated is being simulated with the L0 design parameter set to 60° , which defines the cross-track error. With this value, the kite has an overall good performance with the average cross-track error after the first 5 seconds being $\bar{d}_{t \geq 5s} = 0.0027$. If lower values for the L0 are used, the kite has steeper approximations to the target point ahead on the path, hence overshoot, i.e. oscillations, may happen most of the time. This often leads to higher accumulated cross-track error and higher control effort, which is undesirable.

Table 5.2: Physical Simulation Parameters ([14],[13]).

Parameter	Value
Wing Reference Area A	0.28 m^2
Acceleration of Gravity g	9.8 ms
Kite Mass m	0.7 kg
Fluid Velocity v_w	10 ms
Fluid Density ρ	1.2 kgm^3

Figure 5.6: Ellipse trajectory with a varying tether length $r \in [50, 250]$.Figure 5.7: Kite following the desired path in the (ϕ, β) space, with $T = 60 \text{ s}$ and $L0 = 60^\circ$.

Figure 5.8: Cross-track error for $T = 60s$ and $L0 = 60^\circ$ Figure 5.9: Simulated (r, ϕ, β) for $T = 60s$ and $L0 = 60^\circ$

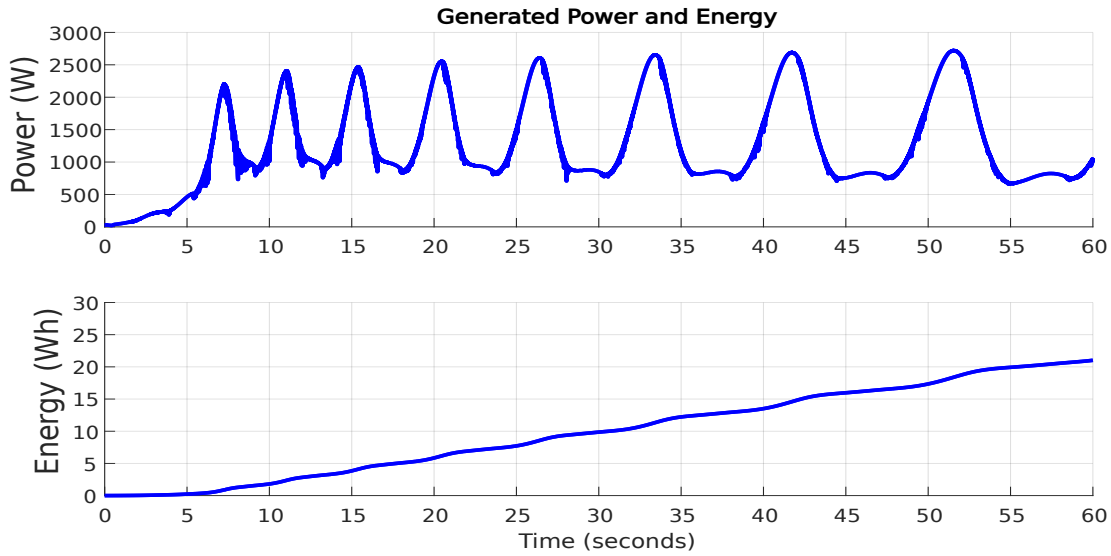


Figure 5.10: Generated Power (W) and Energy (Wh) for $T = 60s$ and $L_0 = 60^\circ$.

5.6 Discussion

The evaluation of the kite model, considering the path-following problem, is highly similar to the majority of the vehicles. The L1 distance method holds the same problem as the car-like system: the possibility of losing track of the path, i.e. when the vehicle is not in the neighbourhood of the desired trajectory. It is possible to solve this problem with additional coupled control strategies, however, in such scenarios, other methods are implemented.

Hence, the L0 distance method, highly identical to the carrot chase method previously described, is used as an alternative that guarantees convergence and stability. The total cross-track error achieved is very low, still hardly reaching and stabilizing at zero. Nonetheless, the control effort is fairly low and the computations are often quick.

With the kite properly following the path and flying in a crosswind direction with its angle of attack controlled, i.e. direction perpendicular to the direction of the wind, a high lift is produced, leading to a high tether force. As the tether reels out, the electric machine produces electric power.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

The path-following controllers developed in this dissertation do not consider time restrictions for the vehicle to arrive at the pre-defined path waypoints. It mainly consists of a geometrical problem that has the advantage of not requiring real-time references. Hence, with offline data, i.e. look-up tables (e.g., desired path), the system continuously computes the vehicle dynamics without additional computation processing dedicated to, for example, the path waypoints. Thus, these controllers focus mainly on the steering control of the vehicle which allows its convergence to the referred waypoints.

The three algorithms presented were then tested on a car-like kinematic model. The model is straightforward and, therefore, it yields the advantage of allowing the comparison of different guidance methods, without additional restrictions and assumptions. In addition, it is commonly the starting point to develop more complex algorithms for autonomous vehicles, before delving into complex dynamics.

The simplified standard kinematic model was simulated with the developed controllers, allowing the assessment of the vehicle's manoeuvrability along two different paths: ellipse and 8-shape. All of the implemented control strategies allow the convergence of the vehicle to the path, however, results show that the L0 distance is more adequate if the the path length is known and some path variations are acceptable. In addition, this method gives the centripetal acceleration command which for more complex vehicle kinematics is useful.

As simulation results show, it is a challenge to tune the best design parameters. Nevertheless, the simulations that were carried out for the car-like model hold interesting results. Overall, the obtained results are sufficient as no specific restrictions were defined, other than some limitation of some parameters due to saturation.

With the devised controllers for the simplified kinematic model, the implementation of the L0 to the AWES shows good results. With the kite dynamic model, it is possible to find the required lateral acceleration command required to steer and converge the kite's position with the VTP on the path. The AWES simulation shows how at first the algorithms may be developed for

a standard kinematic model and then applied to a more specific application, maintaining the same overall path-following control strategy.

6.2 Future Work

From the Literature, cyclical phases and repetitive movements bring to discussion iterative methods that keep adjusting the parameters of the system, while taking into consideration information from previous iterations [3]. The flight control steering parameters references are computed by solving optimisation problems that must be solved in real-time within a fixed sampling time, usually having high computational resources demand [31]. This sometimes may not be ideal, hence hybrid approaches have been surveyed.

Under the theme of combined control architectures, Model Predictive Control has been added on top of the path-following basis controller. As stated previously, the kite has a pre-defined periodic path to be followed while it completes each of the two phases portrayed in Fig. 1.1, repeating the cycle thereafter.

These aerial devices hold path-following problems that are still being researched. Guaranteeing that they operate considerably well under diverse unpredictable scenarios is still a challenge. It is possible to simulate these models for fixed parameters and for varying and shifting behaviours, approximating simulated models and controllers to real applications. These possibilities are important to assess wind flow dynamics and respective disturbances over the airfoil and airframe of the AWES. However, such methods are often computationally sophisticated, and may be, at first, simulated for simple models.

Thus, it results that, this dissertation provides a starting point for more challenging and newly state-of-the-art trajectory controllers to steer the vehicle under disturbances while being autonomous. This could require the design of an architecture where the vehicle is able to assess surrounding circumstances and land or take-off as required and safely. These scenarios should be considered in the future (e.g. existence of wind gusts within a reliable automatic take-off and landing scheme) which may require a path-following algorithm parameterized differently and more robust to changes.

Appendix A

Race Track Path-Following

A.1 Race Track Path-Following

This appendix shows the application of the PID controller seen in Chapter 4 applied to a more complex path: a race track. This type of track has been used to test autonomous racing as described in [33].

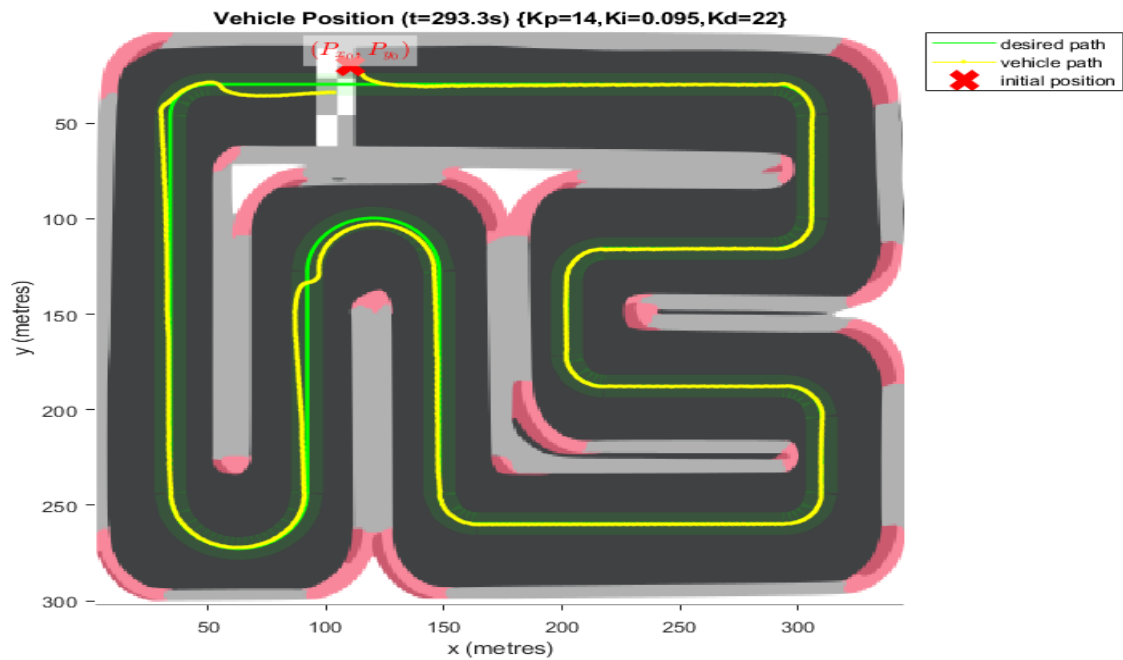


Figure A.1: Method 1 RCP track simulation: desired path and actual vehicle trajectory.

The developed codes are available at the [Github page](#).

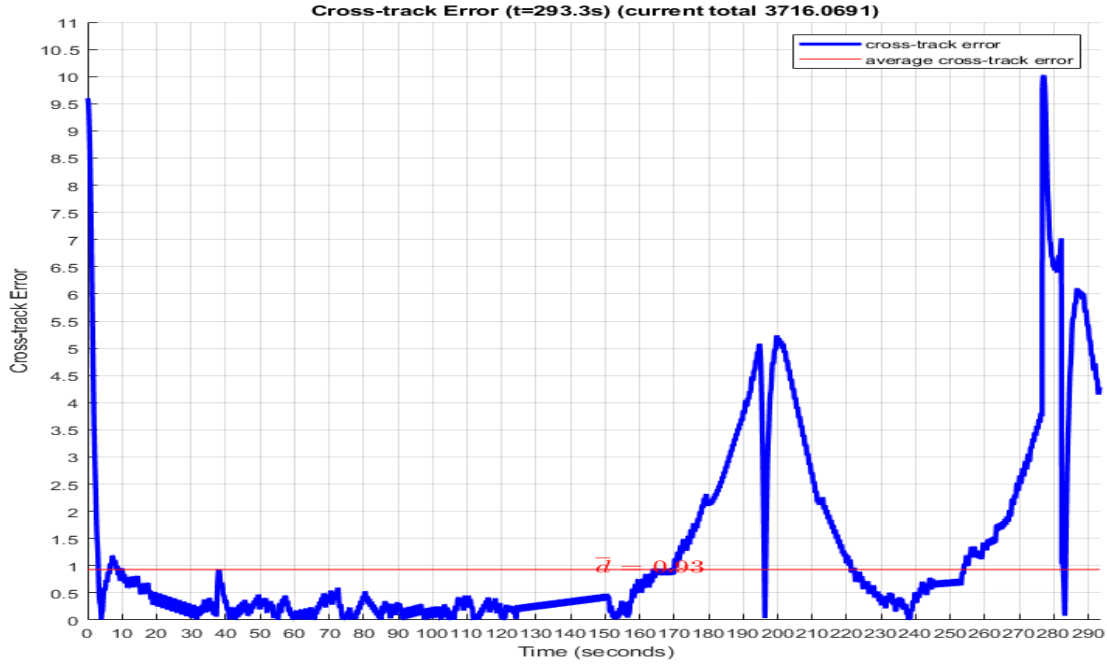


Figure A.2: Method 1 RCP track simulation: cross-track error time variation.

Table A.1: Method 1 RCP track: simulation parameters

Parameter	RCP Track
Simulation Time T	293.3
Starting Point (P_{x_0}, P_{y_0})	(110, 20)
Velocity (u)	5
Turn Radius (R)	10
Starting Heading Angle (ψ)	0°
PID Gains	$K_P = 14$
	$K_I = 0.095$
	$K_D = 22$

References

- [1] LTPaiva FACCFontes. UP WIND Project [online]. URL: <http://www.upwind.pt>.
- [2] Simon Watson, Alberto Moro, Vera Antunes dos Reis, Charalampos Baniotopoulos, Stephan Barth, Gianni Bartoli, Florian Bauer, Elisa Boelman, Dennis Bosse, Antonello Cherubini, Alessandro Croce, and Fagiano. Future emerging technologies in the wind power sector: A European perspective. *Renewable and Sustainable Energy Reviews*, 113, October 2019.
- [3] Roland Schmehl, editor. *Airborne Wind Energy: Advances in Technology Development and Research*. Green Energy and Technology. Springer Singapore, Singapore, 2018.
- [4] Michael Erhard and Hans Strauch. *Automatic Control of Pumping Cycles for the SkySails Prototype in Airborne Wind Energy*. April 2018. Journal Abbreviation: Green Energy and Technology.
- [5] Elena Malz, Jonas Koenemann, S. Sieberling, and Sebastien Gros. *A reference model for airborne wind energy systems for optimization and control*. December 2018.
- [6] Rolf Luchsinger, Damian Aregger, Florian Bezdard, Dino Costa, Cédric Galliot, Flavio Gohl, Jannis Heilmann, Henrik Hesse, Corey Houle, Tony A. Wood, and Roy S. Smith. *Pumping Cycle Kite Power with Twings*. Springer Singapore, 2018.
- [7] Udo Zillmann, Kristian Petrick, and Stefanie Thoms. Introduction to Airborne Wind Energy.
- [8] Yashank Gupta. *Magnus Based Airborne Wind Energy Systems*. PhD thesis, November 2018.
- [9] Makani [online]. URL: <https://x.company/projects/makani/>.
- [10] Nguyen Hung, Francisco Rego, Joao Quintas, Joao Cruz, Marcelo Jacinto, David Souto, Andre Potes, Luis Sebastiao, and Antonio Pascoal. A review of path following control strategies for autonomous robotic vehicles: theory, simulations, and experiments, April 2022.
- [11] Luís Paiva and Fernando Fontes. Adaptive time-mesh refinement in optimal control problems with state constraints. *Discrete and Continuous Dynamical Systems*, September 2015.
- [12] Gonçalo B. Silva, Luís Tiago Paiva, and Fernando A.C.C. Fontes. A Path-following Guidance Method for Airborne Wind Energy Systems with Large Domain of Attraction. In *2019 American Control Conference (ACC)*, July 2019.
- [13] Manuel C. R. M. Fernandes, Sérgio Vinha, Luís Tiago Paiva, and Fernando A. C. C. Fontes. L0 and L1 Guidance and Path-Following Control for Airborne Wind Energy Systems. *Energies*, February 2022.
- [14] CLARK Y AIRFOIL (clarky-il) [online]. URL: <http://airfoiltools.com/airfoil/details?airfoil=clarky-il>.

- [15] European Commission. Directorate General for Research and Innovation. *Research and innovation to REPower the EU*. Publications Office, 2022.
- [16] Manuel Côrte-Real de Matos Fernandes. Airborne Wind Energy Systems: Modelling, Simulation and Economic Analysis, July 2018.
- [17] Miles L. Loyd. Crosswind kite power (for large-scale wind power production). *Journal of Energy*, May 1980. Publisher: American Institute of Aeronautics and Astronautics.
- [18] Kitepower - Airborne Wind Energy [online]. URL: <https://thekitepower.com/>.
- [19] Michiel Kruijff and Richard Ruitkamp. A Roadmap Towards Airborne Wind Energy in the Utility Sector. In *Green Energy and Technology*. April 2018.
- [20] Sanghyuk Park, John Deyst, and Jonathan How. A New Nonlinear Guidance Logic for Trajectory Tracking. August 2004.
- [21] Andrzej Stateczny, Paweł Burdziakowski, Klaudia Najdecka, and Beata Domagalska-Stateczna. Accuracy of Trajectory Tracking Based on Nonlinear Guidance Logic for Hydrographic Unmanned Surface Vessels. *Sensors*, February 2020.
- [22] Jorge Estrela Da Silva and Joao Borges De Sousa. A dynamic programming approach for the control of autonomous vehicles on planar motion. June 2010.
- [23] A. Pedro Aguiar, Dragan B. Dačić, João P. Hespanha, and Petar Kokotović. Path-following or reference tracking? *IFAC Proceedings Volumes*, July 2004.
- [24] P.B. Sujit, Srikanth Saripalli, and J.B. Sousa. An evaluation of UAV path following algorithms. July 2013.
- [25] Ehab Safwat, Weiguo Zhang, Ahmed Mohsen, and Mohamed Kassem. Design and Analysis of a Robust UAV Flight Guidance and Control System Based on a Modified Nonlinear Dynamic Inversion. *Applied Sciences*, January 2019.
- [26] Peter Corke, Witold Jachimczyk, and Remo Pillat. *Robotics, Vision and Control: Fundamental Algorithms in MATLAB®*. Springer Tracts in Advanced Robotics. Springer International Publishing, 2023.
- [27] Hector Perez-Leon, Jose Joaquin Acevedo, Jose A. Millan-Romera, Alejandro Castillejo-Calle, Ivan Maza, and Anibal Ollero. An Aerial Robot Path Follower Based on the ‘Carrot Chasing’ Algorithm. In Manuel F. Silva, José Luís Lima, Luís Paulo Reis, Alberto Sanfeliu, and Danilo Tardioli, editors, *Robot 2019: Fourth Iberian Robotics Conference*. Springer International Publishing, 2020.
- [28] M. Breivik and T.I. Fossen. Principles of Guidance-Based Path Following in 2D and 3D. 2005.
- [29] Luís Tiago Paiva and Fernando A. C. C. Fontes. Optimal Control Algorithms with Adaptive Time-Mesh Refinement for Kite Power Systems. *Energies*, (3), March 2018.
- [30] U. Fechner and R. Schmehl. Flight path planning in a turbulent wind environment. *Airborne Wind Energy*, 2018.

- [31] Fernando A. C. C. Fontes, Mcrm Fernandes, and L. T. Paiva. A Model Predictive Control Scheme to Improve Performance of a Path-following Controller for Airborne Wind Energy. July 2020.
- [32] Navigation Tuning — Plane documentation [online]. URL: <https://ardupilot.org/plane/docs/navigation-tuning.html>.
- [33] Alexander Liniger, Alexander Domahidi, and Manfred Morari. Optimization-Based Autonomous Racing of 1:43 Scale RC Cars. *Optimal Control Applications and Methods*, September 2015.