POLITECNICO DI TORINO

Master Degree course in Computer Enginering

Master Degree Thesis

Design of an advanced control algorithm for Urbanloop capsules

Candidate
Letizia Licitra

Supervisors
Prof. Jérémie Kreiss
Prof. Romain Postoyan
Prof. Diego Regruto
Prof. Xavier Bombois

October 2024

Che le stelle mi guidino sempre
e la strada mi porti lontano.

Abstract

Urbanloop, founded in 2019, aims to develop an innovative, eco-friendly, fast mobility solution with minimal environmental impact. This advanced system features a fleet of self-driving, fully electric pods, designed to transport up to two passengers, operating on a rail-based circuit. The focus is on scaling the system to medium-sized cities to offer a new option for public transportation. This work aims to improve the performance of the control law used to move the pods along the circuit, focusing on enhancing speed and ensuring passenger comfort. First, a mathematical model based on physical laws will be presented. This model is then used to design an advanced control algorithm capable of minimizing travel time and improving performances while being robust to uncertainties. Specifically, this algorithm is developed by solving two key problems: the generation of a reference trajectory through an optimization problem, and the application of the Linear Quadratic Regulator (LQR) method to ensure the system precisely follows this trajectory. Finally, a comparison between the current heuristic control law and the newly proposed method is also presented, highlighting key differences and improvements.

Ringraziamenti

Contents

List of Tables

List of Figures

Acronyms

BLDC BrushLess Direct Current

CBTC Communication-Based Control Train

ANR Agence Nationale de la Recherche

CRAN Centre de Recherche en Automatique de Nancy

LORIA Laboratoire lorrain de Recherche en Informatique et ses Applications

CNAM Conservatoire National des Arts et Métiers

WP Work Package

ATO Automatic Train Operation

DP Dynamic Programming

MPC Model Predictive Control

LQR Linear Quadratic Regulator

PI Proportional-Integral

PID Proportional-Integral-Derivative

DC	Direct Current
FOC	Field Oriented Control
SIMO	Single Input Multiple Output
GPSS	Ground Power Supply System
OL	Open-Loop
CL	Closed-Loop
RMSE	Root Mean Squared Error
NL	Non-Linear
ZOH	Zero-Order-Hold
LTI	Linear Time-Invariant
LTV	Linear Time-Varying
COMMITS	COnverged coMMunication, control and scheduling Infrastructure for multi pods-based Transport Systems

Notation

$\mathbb{R}$	Set of real numbers.
$\mathbb{R}_{\geq 0}$	Set of non-negative real numbers including zero.
$\mathbb{R}_{>0}$	Set of positive real numbers (excluding zero).
$\mathbb{N}$	Set of natural numbers including zero.
$\mathbb{N}_{>0}$	Set of natural numbers without zero.
$\mathcal{PW}(\mathcal{A}, \mathcal{B})$	Set of piecewise constant functions from $\mathcal{A} \to \mathcal{B}$.
(a, b)	Open interval, which includes all x such that $a < x < b$ with $a, b \in \mathbb{R}$.
$[a, b]$	Closed interval, which includes all values of x such that $a \leq x \leq b$ with $a, b \in \mathbb{R}$.
$[a, b)$	Half-open interval, which includes all x such that $a \leq x < b$ with $a, b \in \mathbb{R}$.
$(a, b]$	Half-open interval, which includes all x such that $a < x \leq b$ with $a, b \in \mathbb{R}$.
$x \in \mathcal{S}$	x belongs to the set $\mathcal{S}$.
$\lVert x \rVert$	Euclidean norm of vector x.
$\lim\limits_{x \to a} f(x)$	Limit of continuous function f at a, where $f : \mathbb{R}^n \to \mathbb{R}^m$, $a \in (\mathbb{R} \cup \{-\infty, +\infty\})^n$, and $n, m \in \mathbb{N}_{>0}$.
$\max\limits_{x \in S} f(x)$	Maximum value of f, when it exists, over the set $S \subseteq \mathbb{R}^n$, where $f : \mathbb{R}^n \to \mathbb{R}$, and $n \in \mathbb{N}_{>0}$.
$\min\limits_{x \in S} f(x)$	Minimum value of f, when it exists, over the set $S \subseteq \mathbb{R}^n$, where $f : \mathbb{R}^n \to \mathbb{R}$, and $n \in \mathbb{N}_{>0}$.
$\left.\frac{\partial f(x,y)}{\partial x}\right\rvert_{(\bar{0},\bar{0})}$	Partial derivative of the differentiable function f with respect to its first argument, evaluated at the point $(\bar{0}, \bar{0})$, where $f : \mathbb{R}^n \to \mathbb{R}^m$, and $n, m \in \mathbb{N}_{>0}$.

Chapter 1

Introduction

Urban transportation in France faces ever-growing environmental, social, and economic challenges. Statistics show that over 50% of the French population relies on private cars daily due to a lack of viable alternatives[1]. To address this issue and offer a compelling choice for urban mobility, the Nancy metropolis, the Grand Est region, and the University of Lorraine are collaboratively developing a novel urban transport system: Urbanloop.

Urbanloop was initiated by Jean-Philippe Mangeot, who successfully gathered engineering students at the University of Lorraine in 2017 with the goal of creating an optimal mobility solution using electrical energy. The concept aims to combine the advantages of individual transport, such as reduced travel time and increased speed, and expand these benefits to public transport. The core idea involves using electric vehicles capable of transporting one or two people each, operating on a network of rails to eliminate waiting times and intermediate stops. In Fig 1.1, a virtual representation of the system is shown, where multiple vehicles, referred to as *pods* or *capsules*, move on rails.

For Urbanloop to achieve these objectives at a large scale, several fundamental challenges need to be addressed in terms of control, communication and security. In this context, the «Agence Nationale de la Recherche» (ANR) has funded in 2024 the grant COMMITS[2], which involves three research units, such as LORIA, CRAN, CNAM and of course Urbanloop. This internship is part of the COMMITS project and focuses on the automatic control of a single pod as explained in more detail in the following.

The following sections provide a more detailed overview of Urbanloop, the various objectives of the ANR COMMITS project, and finally, formalize the aims of this internship.

---

[1] https://www.francebleu.fr/infos/societe/sondage-municipales-les-francais-veulent-plus-de-place-pour-la-voiture-et-moins-de-travaux-en-ville-1581433718

[2] https://commits.loria.fr/fr/

Figure 1.1. Virtual representation of a station of Urbanloop [20].

1.1 Urbanloop

Urbanloop SAS[3] was founded in 2019 to develop an innovative, eco-friendly, and fast mobility solution aimed at minimizing environmental impact. The system employs a fleet of self-driving pods, as shown in Fig. 1.1, each capable of carrying up to two passengers, that operate on a network of interconnected loops as depicted in Fig. 1.2. This figure also shows that each station is a *bypass* of the main flow of pods.

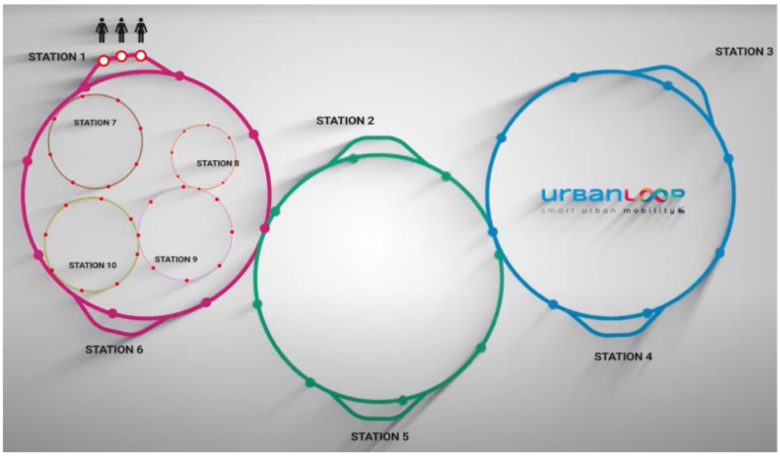


Figure 1.2. Circuit loops [31].

This cutting-edge system merges the benefits of rail transport, such as low energy consumption, with the convenience of on-demand services, thereby eliminating unnecessary stops and waiting times for passengers. The primary aim of Urbanloop is not to replace

---

[3] https://urbanloop.fr/

existing urban mobility solutions like metros in large cities, but rather to provide a viable alternative to car usage in medium-sized cities by offering an efficient, on-demand transportation option tailored to the needs of these urban areas.

On May 28 2021, under the supervision of Certifer, world leader in rail certification, Urbanloop set the world record for the lowest energy consumption per kilometer for an autonomous rail vehicle, with an impressive energy consumption of only 0.05 kWh/km (equivalent to 0.0047 cents)[4]. Given this significant achievement, Urbanloop's system has been selected by the French Minister of Transportation to connect the Olympic cycling venues in Saint-Quentin-en-Yvelines to Paris during the 2024 Olympic Games. For this event, ten pods operated on a two-kilometer single track loop and more than 6500[5] passengers used it during the event.

The company's challenge focus is on scaling the system to meet the increasing demands of urban areas while upholding its high standards of reliability and maintainability. Scheduled for deployment in Nancy in 2026, the Urbanloop system will feature a 4.4-kilometer loop with 40 pods and a relay connection. Scaling up the Urbanloop transport system to the size of a city raises several major scientific and technological challenges, that will be discussed in the next section.

1.2 ANR COMMITS

To facilitate the large-scale implementation of this urban transport model, the ANR COMMITS project concentrates on the development of efficient and resilient tools designed to manage the system's infrastructure. This research is crucial for ensuring safe operation while effectively addressing real-time passenger demands.

From a scientific and technological perspective, these tools will be designed to manage a network of rail pods that support personal and on-demand mobility. The expected features include predictive, adaptive, and convergent capabilities, as well as alignment with communication standards to ensure the safety of capsule control operations.

Objectives The COMMITS project aims to develop a unified communication, control, and scheduling infrastructure to build a cyber-physical system for managing the Urbanloop transport network at a large scale. The primary challenge lies in controlling the entire network while adhering to strict safety, security, and time constraints.

At the city scale, the cyber-physical system must be capable of coordinating a large number of pods, ranging from around 10 to 4,000, in real-time. The coordination must encompass several crucial elements. Firstly, passengers must encounter minimal waiting

---

[4] https://mines-nancy.univ-lorraine.fr/2021/06/02/urbanloop-bat-le-record-du-monde-la-plus-faible-consommation-energetique/

[5] https://www.estrepublicain.fr/transport/2024/08/20/les-capsules-urbanloop-ont-passe-un-premier-test-grandeur-nature-aux-jeux-olympiques

times, thereby enhancing the efficiency and convenience of the transport system. In addition, pods must travel directly to their destinations without making intermediate stops, ensuring a seamless and uninterrupted journey. Where necessary, the system should facilitate efficient and contiguous connections to ensure smooth transitions between routes. Lastly, the system must maintain a high average speed of 50 km/h, ensuring that travel is rapid and reliable.

Architecture of the system Regarding control systems, unlike conventional rail systems, Urbanloop requires extreme precision in controlling the trajectories of the capsules. Moreover, the physics of the pods differ significantly from those of a train. This necessitates the development of efficient and robust control techniques to enable scalability for a large number of pods. Control techniques developed for road traffic are also not directly applicable to Urbanloop due to its specific goals (e.g., minimization of waiting time) and unique characteristics (e.g., rail transport, inability to overtake, etc.). Additionally, the need to manage interactions with pedestrians presents a new challenge that requires the development of novel control strategies.

Compared to other pod-based systems, Urbanloop stands out as a unique on-demand mobility solution on rails. The closest comparable system is urban railway traffic, where Communication-Based Control Train (CBTC) [12] manages traffic automatically through continuous communication between the trains and the control system. However, implementing a safe and comfortable mechanism for Urbanloop requires special attention to the following details:

- solutions must be developed to prevent collisions between capsules and between pods and pedestrians at circuit intersections;
- a real-time scheduling program is essential for frequently recalculating optimal schedules and routes;
- the system must handle a large number of capsules with short intervals between them, necessitating effective management of chain braking mechanisms.

To address these challenges, the Urbanloop system employs a basic centralized CBTC architecture, where each pod communicates with a central unit. This setup requires building local controllers, one per pod, to execute the itinerary generated by the scheduler while adhering to speed and distance constraints relative to other pods.

These local controllers must manage safety and security properties to avoid collisions, as well as performance metrics such as waiting time and route length. Subsequently, to build a more reliable system, the centralized setup will gradually become more distributed, increasing the autonomy of the pods and giving the possibility to handle a large number of pods.

To enhance the organization of the work plan, the ANR COMMITS project is structured in different scientific Work Packages. Specifically, the first WP focuses on the development of the centralized architecture and is subdivided into the following steps:

WP1.1: Use-case requirements and centralized orchestration system specifications;

WP1.2: Design of centralized orchestration and scheduling;

WP1.3: Reporting on and prototyping the implementation of the local controllers.

In particular, this internship is within WP1.3 as explained in more detail in the next section.

1.3 Internship objectives

A central component of Urbanloop is the speed controller of each individual pod. The solution adopted today as deployed for the 2024 Olympic Games relies on a heuristic design. More precisely, a Proportional-Integral (PI) controller is implemented, whose gains vary depending on the reference velocity. The tuning of these gains has been done in an ad-hoc fashion. While this controller is fine for a small number of pods moving along a single loop, we foresee that its performance may not be compatible for larger scale scenarios.

In this context, the primary objective of this internship, which took place at CRAN, is to develop more performant feedback laws in a more systematic manner. We would like the controllers to be robust to uncertainties such as varying railroad slopes, differing pod weights (whether the pod is empty or occupied), and external disturbances, while being performant. By performance, we mean to minimize the travel time while meeting essential safety constraints for both passengers and the system.

To achieve these objectives, the plan is the following.

1. To construct a faithful *mathematical model* of the pod based on physical principles, ensuring it accurately represents the pod's real-world behavior.
2. To design local controllers, which minimize the travel time, while being robust to uncertainties.
3. To compare the performance of the existing heuristic control law with the proposed control law.

The other objectives in terms related to network issues (collision, platooning, (de) insertion etc.) are out of the scope of this internship and will be covered afterwards by the Agence Nationale de la Recherche (ANR) COMMITS.

1.4 Proposed Approach and Related Systems

To guide the development of our control framework, various classical control methods have been examined, focusing on automated rail-based systems. Early efforts in this domain concentrated on integrating automatic time-keeping capabilities into rail systems, with an emphasis on minimizing energy consumption while adhering to operational constraints [25].

Recent advancements in communication, control, and computer technologies [34] have enabled the integration of Automatic Train Operation (ATO) systems into urban rail networks, transitioning from manual to automatic train operation between stations. These systems dynamically adjust speed and braking based on real-time timetables and optimal speed profiles derived from sophisticated train behavior models. Before departure, control algorithms generate a *recommended train speed profile* that balances travel time, energy efficiency, speed limits, and safety. Once the train is in motion, a *speed controller* automatically adjusts commands for acceleration, coasting, cruising, or braking via a feedback loop, ensuring operational efficiency, passenger comfort, and compliance with speed and safety constraints. Control techniques like Proportional-Integral-Derivative (PID), Model Predictive Control (MPC) [9, 18, 33], and Linear Quadratic Regulator (LQR) [10, 21, 33], have been adopted to refine real-time control strategies.

Our approach divides the problem into two key components. First, we generate a reference tracking profile that meets punctuality requirements while adhering to speed and comfort constraints. Second, we develop an efficient speed control feedback method to track the designed trajectory. Specifically, the control method selected is based on the LQR technique, a classical method suitable for our scenario.

The rest of the report is structured as follows.

- *Chapter 2* presents a comprehensive description of the pod, including the construction of the mathematical model and studies conducted for its validation. Such a model was not developed by Urbanloop and is thus a first contribution.
- *Chapter 3* offers a comprehensive analysis of the implemented control strategy. It details the control specifications set by Urbanloop and provides an in-depth description of the reference trajectory generation process. Additionally, it discusses the proposed control law based on the LQR method, including specifics on tuning the weight matrices and the results obtained across various reference trajectories.
- *Chapter 4* examines the results achieved with the proposed control law applied to experimental data, and provides a comparative analysis between the heuristic control law and the proposed LQR approach.

Chapter 2

Modeling of the pod

This following chapter is organized into three sections. It starts with a comprehensive description of the Urbanloop system and its capsules, detailing the essential features and operational aspects relevant to understanding the system's dynamics.

Next, the chapter presents a mathematical model for the Urbanloop pod dynamics. This section includes the formulation of equations and the derivation of the model that describes the behavior and interactions within the system.

Finally, the chapter addresses the validation of the proposed mathematical model. It covers the methodologies and results used to verify the accuracy of the model using experimental data and simulations.

2.1 General description

As mentioned in Chapter 1, an Urbanloop network consists of several autonomous pods running on rails, capable of transporting one or two people each. The capsules are fully electric and move through interconnected loops, with energy provided by the rails. The Urbanloop network is designed for daily trips of 1 to 15 km.

The capsules move along a ring-shaped circuit designed for one-way traffic flow. Each vehicle operates autonomously without onboard personnel and travels in a single direction during normal operations. However, under specific conditions, capsules may be permitted to travel in both directions.

Although the capsule may go through multiple stations along the journey, it only stops at the destination selected by the user. This is made possible by the bypass positioning of the stations. This approach allows us to avoid stopping at every station, unlike traditional public transportation such as metros and trams.

Moreover, in contrast to a conventional railway system where a movable switch directs the train left or right, Urbanloop's rails are fixed with no moving parts, ensuring safe and reliable navigation. As illustrated in Fig 2.1, a pod is shown traversing the rails

Figure 2.1. View of the rails [24].

at a junction where the direction changes. Urbanloop has innovated with an on-board switch system for its vehicles, allowing convoys to be managed efficiently without the need to decelerate at junctions. This streamlined lane management is more resilient to hazards and demands less instrumentation. Direction changes are executed directly from the capsule, employing a system of tilters to navigate the vehicle into its designated lane.

2.2 Main Components

The pod, as shown in Fig. 2.2, is designed to transport one or two people and moves autonomously along the circuit, requiring no interaction from the user. These electrical capsules are powered by the rails, eliminating the need for on-board batteries. As a result, the pods can operate continuously, ensuring full-time availability. The following will comprehensively detail the power supply system, the energy conversion chain, and the position measurement of the pods along the circuit.

Power Supply System To minimize maintenance costs and environmental impact, Urbanloop vehicles are powered by very low voltage rails via a sliding contact. Power is supplied through direct contact with the rails, distributed by power containers positioned every 500 meters along the track, each providing several kilowatts. This *decentralized system*, managed by a peer-to-peer negotiation mechanism, reduces costs and improves reliability.

The power supply containers connect to the distribution network on one side and to the rails on the other. The rail power operates at a voltage level of 72V DC, with the two rails galvanically isolated from the electrical ground. Each vehicle is equipped with a component known as the *shoe*, a metallic element that conducts electricity and aligns with the positive and negative guidance tracks.

Figure 2.2. The pod [13].

To power the pods, Urbanloop adopts an innovative solution: the Ground Power Supply System (GPSS). This system features two aluminum profiles, one positive and one negative, fitted with stainless steel inserts. Aluminum high conductivity minimizes energy loss, ensuring efficient power delivery, while stainless steel enhances durability and prevents corrosion. The GPSS provides a consistent and reliable power feed to the vehicles. An extruded polymer sheath electrically insulates the poles from the environment, safeguarding against weather and corrosion and ensuring user safety by preventing direct contact. This system is highly versatile and can be installed on various terrains and surfaces. Additionally, supercapacitors are present to ensure continuous power supply in case of disconnection between the skids and the rails.

Energy conversion chain To facilitate the conversion of electrical energy into kinetic energy for vehicle movement, each pod is equipped with electric motors controlled by a torque controller. These motors generate torque that directly affects the capsule's speed and, consequently, its position. This rotational torque is then converted into propulsion force through the interaction between the tires and the rails.

The pod includes two 4kW BrushLess Direct Current (BLDC) motors from `QS Motor`[1], one for each rear wheel. The particularity of BLDC motors comes from the permanent magnet rotor generating a constant magnetic field [6]. On the stator side, three electrical phases geometrically spaced by 60° are used to create a rotating magnetic field using alternative current. The rotor is then rotated synchronously with the rotating field. BLDC motors belong to the class of synchronous motors.

---

[1] https://www.qsmotor.com/product/4000w-single-shaft-motor/

To generate the stator currents, the controller KellyKLS7222H[2] is used. It is composed of:

- a three-phase inverter linking the 72V coming out of the power supply capacitors to the stator phases;
- a Field Oriented Control (FOC) device [3,14] aiming at controlling the motor torque by acting on the switches of the inverter.

Given the measured rotor position and stator currents, the FOC relies on the Park transformation [27] on the three-phase frame of the stator toward a two-phase rotating frame, attached to the rotor. The overall current is then split on the two axes of the frame, leading to the so-called *direct current* and *quadratic current*. By moving the direct current, we can control the magnetic flux of the rotor whereas the quadratic current impacts the motor torque. Using PI controllers, the FOC controls precisely the currents and therefore the magnetic flux and motor torque. More details will be given when detailing the model in Section 2.3.

Position measurement To implement a system where numerous vehicles interact on the same network, the capsules must operate in close proximity. This requires extremely precise control of their speed and position on the track, as well as the ability to facilitate interactions with very low latency.

Each capsule is equipped with sensors capable of measuring its position and speed with high accuracy, ensuring an error probability [3] of less than 10^{-9}. The positioning of the capsules on the network is achieved through a patented system [4] by Urbanloop, which uses ferromagnetic markers of varying lengths placed along the track every 2 meters.

As a capsule passes over these markers, an inductive sensor, combined with an incremental sensor on the capsule's wheel, measures the length of each marker. These markers, functioning like a continuously scanned barcode, allow the system to read several successive markers to determine the precise position of the capsule on the track.

Using this system, it is possible to determine the absolute position of the pod along the circuit at every instant.

2.3 Model of the pod dynamics

In this section, we derive a state-space model of the capsule dynamics based on fundamental physics principles. We begin by introducing the assumptions made, the variables and parameters used to accurately describe the model, and then examine the forces acting on the system.

---

[2] https://media.kellycontroller.com/new/Kelly-KLS-HUserManualV1.2-100G-13.pdf

[3] https://urbanloop.fr/en/notre-technologie/

[4] https://bul.univ-lorraine.fr/index.php/apps/onlyoffice/s/ETiDxr9NHEZBAgn?fileId=106234105

2.3.1 Assumptions

Before proceeding with the analysis, a set of assumptions is established to simplify the description of the capsule's dynamic behavior while maintaining its key characteristics.

The capsule is modeled as a point mass, an approach frequently used in train transport systems [16]. This abstraction facilitates a more manageable mathematical treatment by concentrating exclusively on the capsule's translational motion. All four wheels are considered identical in diameter, material properties, and mechanical behavior.

The motors, located at the rear of the capsule, are assumed to be identical in terms of power output, control algorithms, and operating conditions. These motors are regulated by a uniform torque controller, effectively functioning as a single virtual motor. The total torque generated is the sum of the individual torques produced by each motor, a concept discussed in more detail in Section 2.3.3.

For the sake of simplification, the capsule's motion is assumed to be linear, following a straight trajectory. Consequently, track curvature and transverse inclination are considered to have negligible impact on the dynamics.

The interaction between the wheels and the track is simplified by assuming no sliding, meaning the wheels maintain perfect rolling contact with the track. The friction coefficient between the tires and rails is considered constant, directly proportional to the capsule's weight. Aerodynamic drag is modeled as proportional to the square of the capsule's speed, capturing the increased effect of air resistance at higher speeds [2, 4]. The coefficient of static friction, which governs the initiation of motion, remains constant regardless of whether the capsule is stationary or in motion.

These assumptions are summarized as follows.

1. The capsule is considered a point mass.
2. All four wheels of the capsule are identical in diameter and characteristics.
3. All motors have identical specifications and controllers, and they operate in the same manner.
4. For each pod, the two motors are assumed to generate a combined torque that represents the total contribution of both motors.
5. The capsule moves in a straight line, with turns being gentle enough not to introduce additional forces affecting its dynamics.
6. Neither the track's curvature nor its transverse inclination affects the capsule's dynamics.
7. The wheels maintain contact with the track without any sliding.
8. The friction between the tire and the rail is proportional to the capsule's mass.
9. The aerodynamic drag force is assumed to be proportional to the square of the speed.

10. The coefficient of dry friction remains constant regardless of whether the capsule is stationary or in motion.

2.3.2 Variables

To express equations and formulas clearly and concisely, it is necessary to introduce the mathematical notation that will be fundamental for understanding the concepts. First, we define several variables:

- $y : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is the position of the capsule on the track;
- $\dot{y} : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is its speed along the track axis;
- $i_q : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is the quadratic current of the motors;
- $\xi : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is the state of the integrator of the torque controller.

Thus, at time $t \in \mathbb{R}_{\geq 0}$, the position of the capsule is denoted by $y(t)$, its speed by $\dot{y}(t)$, the motor current by $i_q(t)$, and the state of the integrator by $\xi(t)$. Based on the assumptions formulated in Section 2.3.1, we will see further that we need just the following state variables y, $\dot{y}$, i_q, and ξ to capture the dynamics of the vehicle.

The position y affects the variable $\alpha : \mathbb{R} \to [-\frac{\pi}{2}; \frac{\pi}{2}]$ which represents the slope of the track in radians as a function of the position. The variable $\alpha(y)$ indicates the track inclination angle relative to the horizontal, with $\alpha(y)$ being positive when the vehicle is moving uphill. Typically, the slope angle is expressed as a percentage. Figure 2.3 provides an *illustrative example* of the pod traveling along a path where the slope angle varies with position y, showing an angle different from zero.

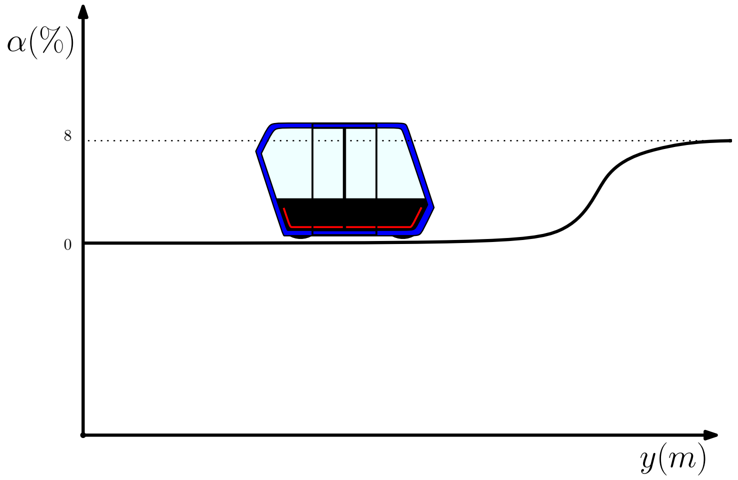


Figure 2.3. Example of a path with positive slope angle.

Since we consider a maximum inclination of 10-12%, in the Appendix A.2 it is possible to find a table to convert from percentage to radians to degrees. The parameters described in the following sections are detailed in Appendix A.1.

2.3.3 Forces Acting on the Pod

The dynamical model of the capsule is derived from the fundamental principle of dynamics. Therefore, we must first perform a balance of the forces in action.

Traction Force The traction force is produced by the wheel motors, which are permanent magnet synchronous machines controlled by a torque controller, as detailed in Section 2.2. These motors are managed in controlled jointly, effectively behaving as a single virtual motor with a combined torque that is the sum of each motor's torque. The electrical components of the motor with its torque controller are illustrated in Figure 2.4.

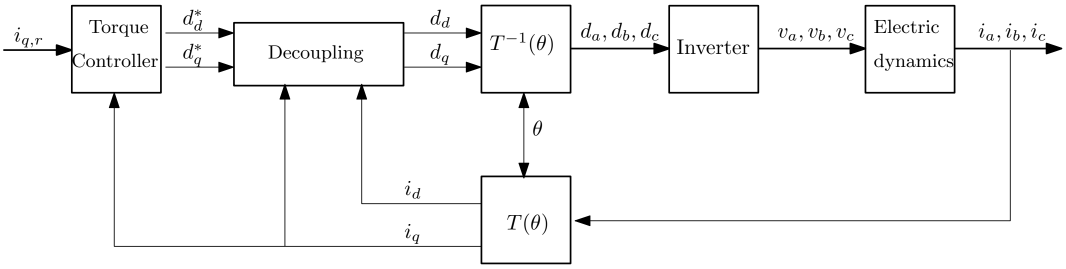


Figure 2.4. Torque Controller Scheme

The controller's input includes the reference quadratic current $i_{q,r}$ and the quadratic current i_q. The controller processes these inputs to generate reference duty cycles for the direct and quadratic axes, denoted as d_d^* and d_q^*. A decoupling block then separates the duty cycles into direct d_d and quadratic d_q components, using the axis currents i_d and i_q as feedback. The inverse Park transformation [27], $T^{-1}(\theta)$, where θ is the rotor angle, is employed to convert the signals d_q and d_d into three-phase duty cycles d_a, d_b, and d_c. These are used to control the three-phase inverter, which then drives the motor with voltage signals v_a, v_b, and v_c.

The total torque Γ is proportional to the quadratic current i_q, as expressed by:

$$\Gamma = K_\mu i_q,$$

where K_μ is the torque coefficient. For further details on this coefficient and its derivation, refer to Appendix A.1, which explains how the torques from each motor are aggregated into the final torque value. The quadratic current i_q depends on the three-phase currents i_a, i_b, and i_c and the rotor angle θ, through the Park transformation [27]. The dynamics of the quadratic current i_q are described by the following law.

$$\frac{\mathrm{d}i_q}{\mathrm{d}t} = -\frac{R}{L} i_q + \frac{E}{L} d_q,$$

---

[4] The values of K_μ, R, L, E, K_p, K_i, r, M, g, ρ, C_x, S, v_{wind} are available at Appendix A.1.

where R and L are the stator resistance and inductance, respectively, E is the supply voltage, and d_q is the quadratic duty cycle. Current regulation is achieved with a PI controller, described by:

$$\begin{cases} \dot{\xi} &= i_{q,r} - i_q \\ d_q &= K_p(i_{q,r} - i_q) + K_i \xi \end{cases} \tag{2.1}$$

where ξ is the integrator state, and K_p and K_i are the proportional and integral gains of the controller, respectively.

Therefore, we obtain

$$F_{\text{tract}} := \frac{\Gamma}{r}, \tag{2.2}$$

where r is the radius of the wheel, and Γ is the resulting torque from the two motors.

Weight The weight of the capsule affects its movement dynamics, particularly on a slope. The effect of weight is described by the following equation

$$P(y) = -Mg\sin(\alpha(y)) \quad \forall y \in \mathbb{R}, \tag{2.3}$$

where α is the slope angle described in Section 2.3.2, M is the mass of the pod and g is the gravitational constant.

Weight contributes to the capsule's movement only when α is non-zero. Figure 2.5 provides a graphical representation of the weight force as a function of the track's slope angle α.

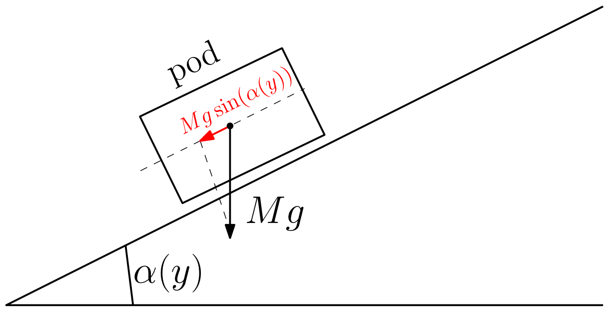


Figure 2.5. Weight force $P(y)$ as a function of the slope angle α

---

[4] The values of K_μ, R, L, E, K_p, K_i, r, M, g, ρ, C_x, S, v_{wind} are available at Appendix A.1.

Air Friction Force The air friction on the capsule is given by the following relationship

$$F_{\text{air}} = -\rho C_x S \left(\dot{y} - v_{\text{wind}}\right)^2, \tag{2.4}$$

where C_x is the drag coefficient of the capsule, ρ is the air density, S is the frontal area and v_{wind} is the speed of the wind component along the capsule axis of motion. The drag coefficient C_x ranges between 0 and 1, a value closer to 0 indicates a more aerodynamic vehicle.

Dry Friction Force To express dry friction, let us first denote F_{res} as the resultant force from all previously mentioned forces, i.e.,

$$F_{\text{res}} := F_{\text{tract}} + P(y) + F_{\text{air}}, \tag{2.5}$$

where F_{tract}, P and F_{air} are given by (2.2), (2.3) and (2.4) respectively.

Regarding the behavior of the dry friction force, it opposes external forces at zero velocity as long as it remains below the threshold given by

$$F_{\text{sec},0} := kMg \in \mathbb{R}_{\geq 0}, \tag{2.6}$$

where k is the dry friction coefficient and depends on the type of contact between the two moving surfaces. If F_{res} exceeds $F_{\text{sec},0}$, the capsule starts moving, and the dry friction force then opposes the movement with a constant amplitude $F_{\text{sec},0}$.

The definition of dry friction is as follows

$$F_{\text{sec}}(F_{\text{res}}, \dot{y}) := \begin{cases} F_{\text{sec},0} & \dot{y} < 0 \\ -\text{sat}_{[-F_{\text{sec},0}, F_{\text{sec},0}]}(F_{\text{res}}) & \dot{y} = 0 \\ -F_{\text{sec},0} & \dot{y} > 0 \end{cases} \tag{2.7}$$

where the saturation function sat is defined by

$$\text{sat}_{[a,b]}(x) = \begin{cases} a & x < a \\ x & x \in [a,b] \\ b & x > b, \end{cases} \qquad \forall x \in \mathbb{R}. \tag{2.8}$$

for $a, b \in \mathbb{R}$ such that $a \leq b$.

2.3.4 State Space Representation

With the forces influencing the capsule now analyzed, we can derive its dynamical model. Based on (2.2)-(2.7), the fundamental principle of dynamics along the capsule's motion axis is given by

---

[5] The value of k is available at Appendix A.1.

$$M\ddot{y} = \frac{\Gamma}{r} - Mg\sin(\alpha(y)) - \rho C_x S\left(\dot{y} - v_{\text{wind}}\right)^2 + F_{\text{sec}}(F_{\text{res}}, \dot{y}). \tag{2.9}$$

We define the input of the system as $u := i_{q,r}$, $u \in \mathbb{R}^{n_u}$, and the state vector $x \in \mathbb{R}^{n_x}$ as

$$x := \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} \xi \\ i_q \\ y \\ \dot{y} \end{bmatrix}, \tag{2.10}$$

and $z \in \mathbb{R}^{n_z}$ signifies the output. Specifically, $n_x := 4$, $n_u := 1$, and $n_z := 2$, making it a Single Input Multiple Output (SIMO) system.

We can represent the system as, given (2.1)-(2.7)

$$\begin{aligned} \dot{x} &= f(x, u, v_{wind}) \\ z &= h(x), \end{aligned} \tag{2.11}$$

where the function $f : \mathbb{R}^{n_x} \to \mathbb{R}$ defines the field vector, expressed as:

$$f(x, u, v_{wind}) := \begin{pmatrix} -x_2 \\ \frac{EK_i}{L}x_1 - \frac{EK_p+R}{L}x_2 \\ x_4 \\ \frac{K_\mu}{rM}x_2 - \nu(x_3, x_4, v_{wind}) \end{pmatrix} + \begin{pmatrix} 1 \\ \frac{EK_p}{L} \\ 0 \\ 0 \end{pmatrix} u. \tag{2.12}$$

The measured outputs are the velocity and position of the capsule, represented by x_3 and x_4 respectively. Thus, the output function $h : \mathbb{R}^2 \to \mathbb{R}$ is given by:

$$h(x) := \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} x. \tag{2.13}$$

System (2.11) is nonlinear due to the inclusion of trigonometric and quadratic terms. The function $\nu : \mathbb{R}^2 \to \mathbb{R}$ captures the nonlinear components in the last differential equation of (2.12):

$$\nu(x_3, x_4, v_{wind}) := g\sin(\alpha(x_3)) - \frac{F_{\text{sec}}(F_{\text{res}}, x_4)}{M} + \frac{\rho C_x S}{M}\left(x_4 - v_{\text{wind}}\right)^2. \tag{2.14}$$

Not only system (2.11) is nonlinear, it is also discontinuous because of F_{sec}, which justifies its writing as a differential inclusion. In this context, we consider solutions to (2.11) in the sense of Krasovskii [8].

2.3.5 Model Reduction

To proceed with the design of the controller in Chapter 3, we propose to simplify the system (2.11). In these paragraphs, we discuss the process of reducing the dynamical system described by the equations in (2.11)-(2.14). We exploit for this purpose the fact that system (2.11) exhibits several time-scales.

Considering the system in (2.11), given that the values of the parameters K_μ, R, L, E, K_p, K_i, r, M, g, ρ, C_x, S, and v_{wind} are provided in Appendix A.1, and since we assume a maximum slope of 12% for the angle $\alpha(x_3)$, as described in Section 2.3.2, we can approximate the system as follows.

$$f(x, u, v_{\text{wind}}) := \begin{pmatrix} -x_2 \\ \underbrace{\frac{EK_i}{L}}_{\approx 10^7} x_1 - \underbrace{\frac{EK_p + R}{L}}_{\approx 10^9} x_2 \\ x_4 \\ \underbrace{\frac{K_\mu}{rM}}_{\approx 10^{-2}} x_2 - \underbrace{g \sin(\alpha(x_3))}_{\approx 10^2} + \underbrace{\frac{F_{\text{sec}}(F_{\text{res}}, x_4)}{M}}_{\approx 10^{-1}} - \underbrace{\frac{\rho C_x S}{M}}_{\approx 10^{-3}} (x_4 - v_{\text{wind}})^2 \end{pmatrix} + \begin{pmatrix} 1 \\ \underbrace{\frac{EK_p}{L}}_{\approx 10^9} \\ 0 \\ 0 \end{pmatrix} u. \tag{2.15}$$

What we observe is that the equations describing the electrical behavior of the system, e.g. the first two equations, have significantly larger magnitudes compared to the equations related to the pod dynamics. Specifically, the factors in the second equation, highlighted in red, are at least seven orders of magnitude greater than those in the fourth equation, highlighted in blue.

Moreover, system (2.11) is singularly perturbed [22, 26, 29], and the origin of the fast dynamics is globally exponentially stable. Consequently, we can consider the reduction of system (2.11) by assuming that the fast variables are constrained on the so-called slow manifold. The *slow manifold* technique [22] is employed to reduce the system. This method is particularly useful when a system has components evolving at different rates. The slow manifold represents a subspace of the state space where the fast variables are in equilibrium. Specifically, it consists of the set of points where the velocity of the fast variables is zero for a given slow state.

Applying the slow manifold technique allows us to reduce the complex system to a simpler model that captures the essential characteristics of the slow dynamics. Further details on the application of the slow manifold technique can be found in Appendix A.3.

Based on these observations, we can simplify the model to focus on the slow variables. The reduced model is represented by

$$f_{\text{red}}(\mathsf{x}, u, v_{\text{wind}}) := \begin{pmatrix} \mathsf{x}_2 \\ -\nu(\mathsf{x}_1, \mathsf{x}_2, v_{\text{wind}}) \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{K_\mu}{rM} \end{pmatrix} u, \tag{2.16}$$

where x_1 denotes the position of the capsule along the track, x_2 represents its velocity, and u is the reference quadratic current $i_{q,r}$.

The output function for the reduced model is also modified as follows

$$h_{\text{red}}(\mathsf{x}) := \mathsf{x}. \tag{2.17}$$

We can concisely represent the reduced model using the following notation

$$\Sigma : \begin{cases} \dot{\mathsf{x}} &= f_{\text{red}}(\mathsf{x}, u, v_{\text{wind}}) \\ \mathsf{z} &= h_{\text{red}}(\mathsf{x}). \end{cases} \tag{2.18}$$

The system described by (2.18) will be crucial for the following steps, including model validation and the development of the proposed control law.
From now on and for the remaining part of the report, we assume that the fast dynamics are static, so the dynamics of the pods follow the system described in (2.18).

2.4 Model Validation

The primary objective of this section is to evaluate how accurate the model is in the light of the available experimental recordings. Firstly, a brief overview of the experimental data will be provided, followed by a description of the heuristic control law employed for the tests. Next, the scheme used for model validation will be presented, and the section will be concluded with a discussion of the obtained results.

To validate the system (2.18), we consider the reduced one (2.16), where x_1 and x_2 represent respectively the position and speed of the system, while u denotes the control input associated with the reference quadratic current.

2.4.1 Experimental Data

To conduct various simulations, tests were carried out in Saint-Quentin-en-Yvelines. A total of 10 tests were performed on different days under varying weather conditions. A capsule was used on the circuit, with the number of occupants varying between 0, 1, and 2 individuals. In each test, multiple trajectories between stations were executed to evaluate the effectiveness of the heuristic control law proposed by Urbanloop in ensuring the system accurately followed the reference velocity trajectory.

For each test, different informations were collected: logs included the speed reference of the capsule v_{ref}, the pod speed v, the capsule's position relative to the circuit, the controller gain values, and flags indicating brake activation. These data were recorded at a sampling rate of one second. Since multiple trajectories were performed during each test, data cleaning was conducted to isolate a single test for each trajectory from station A to station B, resulting in 24 distinct tests. These tests are sorted and labelled from *Test 1* to *Test 24*.

However, data do not include information on wind speed along the route or the longitudinal slope angle. Therefore, the variables v_{wind} and $\alpha(\mathsf{x}_1)$ were set to zero for the entire model validation.

2.4.2 The Heuristic Control Law

Before proceeding with the model validation, it is important to reconstruct the control law used to perform the test. Specifically, this control law refers to the PI control method, which involves both a proportional and an integral gain. However, unlike the standard PI controller where the gains are fixed, in this approach, the gains vary according to the system speed v. Figure 2.6 offers a simplified overview of the controller used by Urbanloop.

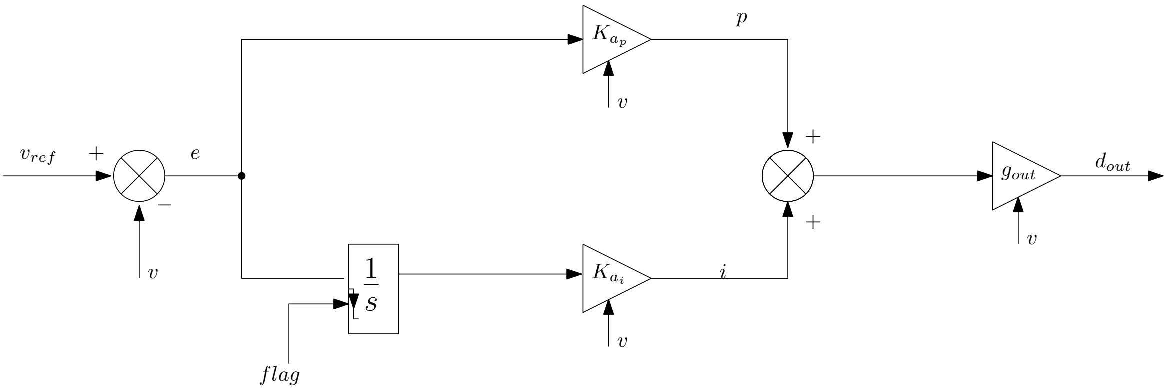


Figure 2.6. PI Urbanloop

The various signals used are as follows:

- v_{ref} is the reference speed;
- v is the generated speed of the system;
- e is the difference between the speed reference and the actual speed of the system. This value is used as input for both the integral and proportional components;
- p represents the actual value of the proportional component, calculated as $p := K_{a_p}e$, where K_{a_p} is the gain for the proportional part of the controller and varies with the speed of the system v;
- $flag$ is a boolean value initialized to zero; when the mechanical brake is activated, this value is set to 1, resetting the integral component to zero;
- i represents the value of the integral component. The error e is input to the integrator, which operates as an integrator with reset triggered on a falling edge. If the value of the $flag$ is equal to 1, the sum within the integrator is reset to zero. Similar to the proportional component, K_{a_i} represents the gain for the integral component;

- A gain g_{out}, that varies according to the speed of the system v, multiplies the sum of $p + i$;
- d_{out} represents the output value, calculated as $g_{out}(p + i)$. This output is a digital value between -100 and 100.

From now on, we will refer to this control law as the Urbanloop PI control law.

2.4.3 Assumptions and Unknown parameters

To validate the model, several challenges had to be addressed. Specifically, the data provides a digital value d_{out} as output, whereas the model requires a current value as input. Due to incomplete and non-reconstructible information about the electrical components, we opted to simplify the electrical aspects by treating the torque controller as a simple gain. This simplification was essential because the information related to the effective construction of the torque controller [6] is reserved and is essentially a black box, making it impossible to proceed otherwise.

The following scheme 2.7 illustrates the process.

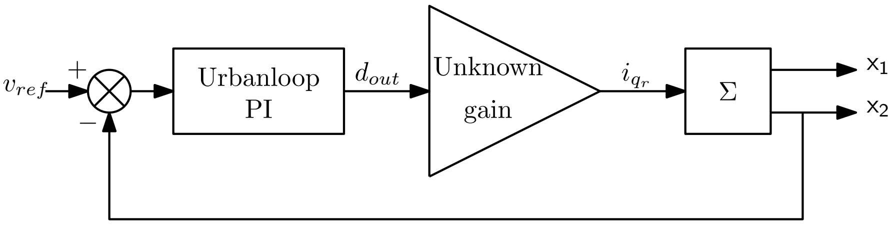


Figure 2.7. Closed-Loop Scheme

In Closed-Loop Scheme, we have the following signals:

- v_{ref} represents the reference speed value, collected in the data;
- *Urbanloop PI* represents the heuristic control law, explained in the previous section, with an output value d_{out}, a digital value ranging from $[-100, 100]$;
- *Unknown gain* represents the gain value between the control law and the model input;
- Σ represents the reduced model (2.18);
- x_1 and x_2 represent the position and speed of the system, respectively.

The value x_2 has been used in feedback to calculate the value of the error as $(v_{ref} - \mathrm{x}_2)$ and this last value has been used as input of the PI controller. The primary purpose of this approach represented in Fig. 2.7 is to determine the missing gain value. Finding this

---

[6] https://media.kellycontroller.com/new/Kelly-KLS-HUserManualV1.2-100G-13.pdf

gain is crucial to ensure that the model receives a current value as input rather than just a digital value. To determine the gain value, numerous simulations were performed using data from all 24 tests to achieve behavior similar to that observed in the experimental data. Below, we see the results of some tests in Fig. 2.8, 2.9:

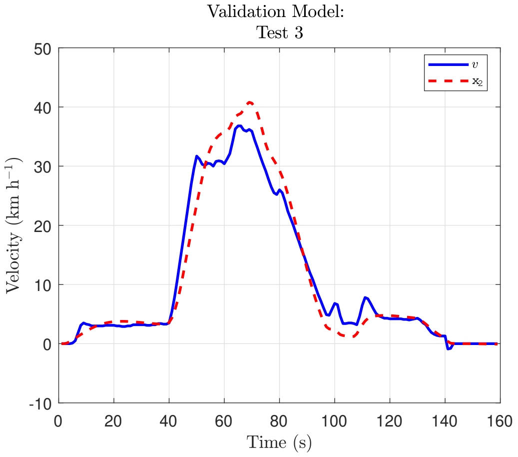


Figure 2.8. Closed-Loop Validation *Test 3*

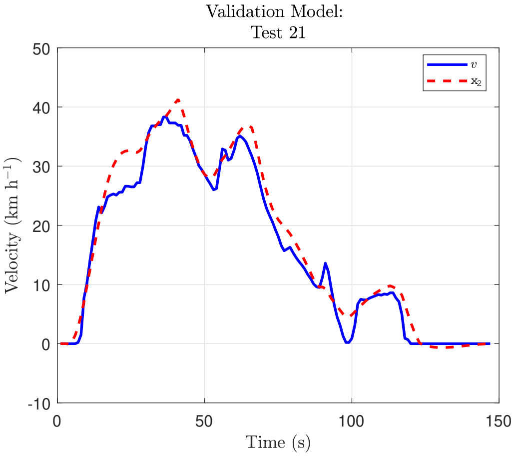


Figure 2.9. Closed-Loop Validation *Test 21*

The red curve represents the trajectory of the system speed profile x_2 generated by (2.16), while the blue curve shows the trajectory of the system speed collected from the test data v. To evaluate the mismatch between the graphs, we used the Root Mean Squared Error (RMSE) as the measurement index. The goal was to identify the gain that minimizes this value across all tests. For these simulations, we achieved RMSE values of 1.4907 m/s for *Test 3* and 2.5374 m/s for *Test 21*, using a gain of 1.3. These results are satisfactory, as the system's speed closely matches the recorded speed values, despite some minor discrepancies. The following table 2.1 presents the value of the RMSE obtained for all the tests.

Test	**RMSE [m/s]**	***Test***	**RMSE [m/s]**
1	1.4907	**13**	2.2165
2	1.4702	**14**	5.8384
3	2.6809	**15**	2.7706
4	2.6629	**16**	2.4236
5	1.7129	**17**	0.7927
6	2.3353	**18**	2.8910
7	2.1587	**19**	2.7706
8	2.5183	**20**	1.8282
9	3.1973	**21**	2.5374
10	2.2925	**22**	1.9586
11	2.0488	**23**	0.3534
12	2.6019	**24**	1.1932

Table 2.1. RMSE results for Closed-Loop Validation Tests

By analyzing the values reported in Table 2.1, we observe very similar results between all the tests provided.

Using this approach has been essential in ensuring a properly adapted reconstruction of the control scheme used, which allows for the correct interpretation of the data. The main goal was to efficiently reconstruct the control law employed by Urbanloop, with particular emphasis on determining the value of the *Unknown gain* that aligns with the chosen approach.

It is possible that the accuracy obtained in validating these tests may be attributed to the robustness of the PI controller used by Urbanloop, which appears capable of optimally correcting the error between the system's velocity and the reference speed. For this reason, it is necessary to utilize an Open-Loop (OL) scheme that does not rely on feedback from the system's velocity to correct errors. This is feasible because the contribution of the Urbanloop-designed controller is encapsulated in the value d_{out}, allowing us to effectively analyze the validation of our model.

2.4.4 Open-Loop Validation Tests

To properly validate the model, we analyze the system using experimental data without incorporating the system speed in Closed-Loop (CL). What we have is an OL system, as shown in the following figure 2.10:

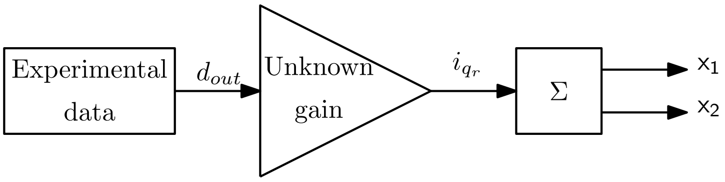


Figure 2.10. Open-Loop Scheme

The experimental data remain as previously documented. The *Unknown gain*, determined from earlier simulations, has been set to a value of 1.3. The model used is (2.18). The outputs consist of the system's speed and position.

Using this model to run the simulations, we obtain the following performances:

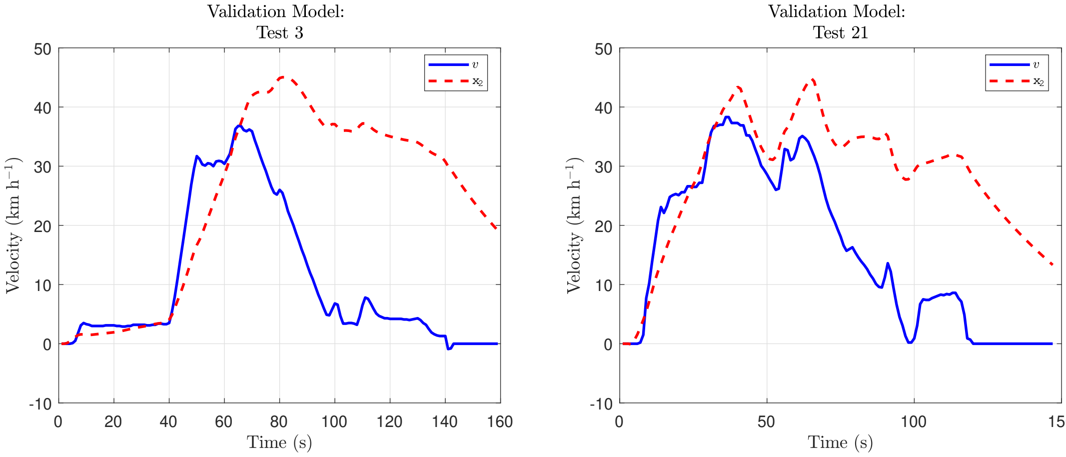


Figure 2.11. Open-Loop Validation *Test 3*

Figure 2.12. Open-Loop Validation *Test 21*

The red curve represents the system speed profile x_2, while the blue curve shows the system speed collected from the test data v. The tests yield RMSE values of 20.7261 m/s for *Test 3* and 16.4105 m/s for *Test 21*, indicating poorer performance compared to previous tests. For example, in *Test 3*, after 70 seconds, a noticeable offset appears between the two curves, leading to a divergence in their behavior. Similarly, in *Test 21*, after reaching peak speed, the system speed x_2 fails to closely follow the data speed v. The following table 2.2 presents the value of the RMSE obtained for all the tests.

Test	**RMSE [m/s]**	***Test***	**RMSE [m/s]**
1	21.1923	**13**	13.0134
2	6.9164	**14**	20.0848
3	20.7261	**15**	15.9893
4	25.8265	**16**	14.1000
5	12.8602	**17**	0.9595
6	22.1966	**18**	15.8211
7	24.3091	**19**	15.9893
8	7.1043	**20**	15.7190
9	16.1836	**21**	16.4105
10	16.2895	**22**	6.3374
11	4.3203	**23**	0.5056
12	14.8999	**24**	6.7729

Table 2.2. RMSE results for Open-Loop Validation Tests

In general, analyzing the different values presented in Table 2.2, we observe varying results that cannot be considered satisfactory across all tests. In particular, the behavior observed between the tests conducted in open-loop (OL) and closed-loop (CL) configurations differs significantly, with the OL tests exhibiting worse performance. A potential explanation for these results may be the neglect of wind power and slope inclination, which could introduce significant errors in the model's accuracy.

2.4.5 Final Considerations on Model Validation

The work done to accurately validate the model can still be improved; it is clear that there are some features to adjust. It is also important to highlight that the informations related to wind power and the longitudinal inclination of the pod along the track were missing, so these values have been set to zero, which may explain the mismatch observed in Fig 2.11 and 2.9.

In general, the variation of the angle can go up to 12% for the considered circuits, so having an angle different than zero, can significantly change the behaviour of the system. To illustrate this, simulations have been done by changing the value of α. Figure 2.13 shows the obtained results.

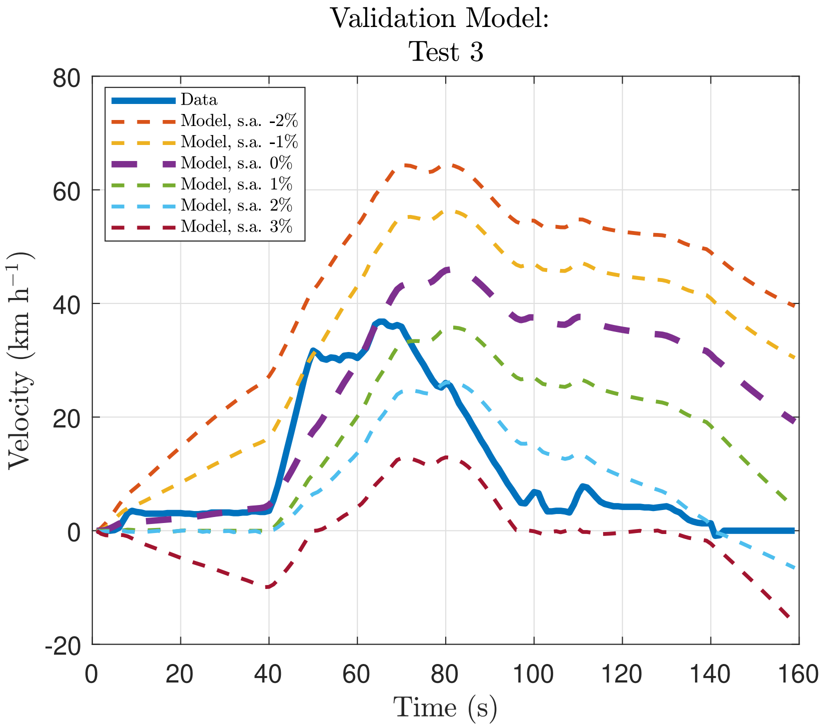


Figure 2.13. *Test 3* with angle slope variation

Using *Test 3* as the baseline, this plot illustrates how angle variation can affect the speed behavior. The blue curve represents the trajectory data speed v, while the other curves depict the trajectory system speed with different slope angles, (denoted as $s.a.$), ranging from -2% to 3%. The simulated angle variation is constant throughout the entire trajectory. However, in real scenarios, there are segments of the trajectory where the inclination may differ from zero. These plots demonstrate that the slope angle is a significant parameter that cannot be ignored during model validation. A worthwhile approach is to obtain and collect information related to the slope angle for the circuits under consideration.

Another factor that can significantly affect the behaviour of the trajectory is wind speed. Similar to the slope angle, simulations have been conducted in *Test 3* to observe the impact of wind, as shown in Fig 2.14.

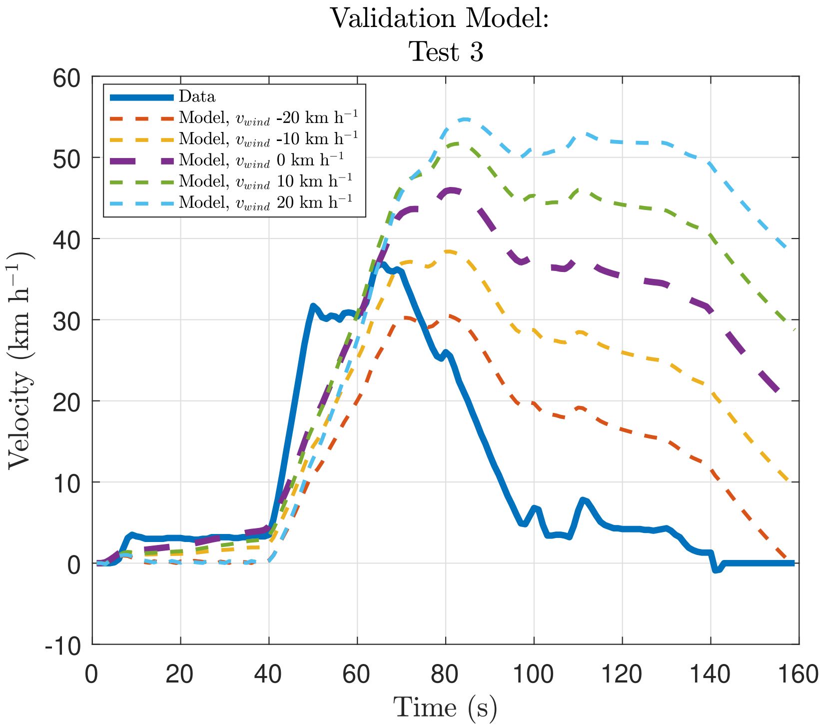


Figure 2.14. *Test 3* with wind variation

The impact of air friction force must be considered. As for the slope angle, simulations were conducted assuming a constant wind speed along the entire trajectory, ranging from -20 km/h to 20 km/h. However, this parameter can vary based on weather conditions and the direction of the pod along the circuit.

To resume, further work is needed to accurately represent the mathematical model. Simulations with varying wind speed and slope angle parameters show significant differences in the system's behavior, highlighting their substantial impact on performance. To address this, it would be beneficial to obtain detailed information about the slope angle from the circuit used in the tests or to redo the tests while accounting for variations in wind speed to properly adjust the proposed mathematical model.

In summary, this chapter has provided a detailed description of the constructed model, validation tests, and the corresponding results. Looking ahead, it will be important to gather data on the slope angle of the test circuit and wind speed to further assess the model's accuracy.

A promising approach could involve validating the model using the OL scheme. If metrics like RMSE approach those obtained with the CL scheme, it would suggest that the model reliably captures the pod's system dynamics.

Despite some minor discrepancies, the model's structure is generally well-suited for accurately representing the pod's behavior. Additional tests, particularly involving changes in slope angle or wind speed, may necessitate further refinements based on the outcomes.

The subsequent chapter will center on formulating the control strategy. The proposed algorithm will leverage the mathematical model to generate a reference trajectory tailored to the system under study. Furthermore, a closed-loop control method will be devised to guarantee that the system accurately follows this trajectory.

Chapter 3

Control Design

In this chapter, the specifications set by Urbanloop are formalized as control objectives for system (2.18).

3.1 Objective and strategy

In this MSc project, we focus on a single pod operating on a loop, with the following objective.

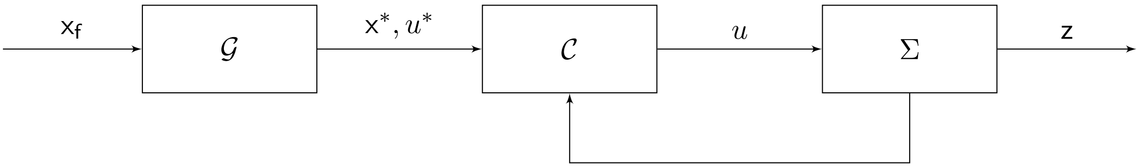
Control problem Automatically move the pod from an initial resting position to a desired location in the minimum time, while ensuring comfort and security.

Constraints related to the comfort and the security will be detailed in Section 3.2.

Our approach to solve the control problem is twofold:

1. As a first step, we formalize and solve a *minimal time control* problem to generate an optimal reference trajectory that is compatible with the pod's dynamics (2.18) as well as additional constraints we will detail in the following. We therefore obtain a feedforward input which we will refer to as the *reference input*.
2. Subsequently, given the importance of ensuring consistent performance despite potential disturbances or uncertainties, we aim to design an output feedback law to ensure the system's speed closely follows the reference trajectory.

To provide a clearer understanding of the approach, we refer to the block diagram shown in Figure 3.1 where x_{f} denotes the final state that the pod aims to achieve. The block $\mathcal{G}$ generates the optimal input trajectory u^* for the pod to follow, as well as the optimal reference trajectory, denoted as x^*. Furthermore, the block $\mathcal{C}$ operates as a feedback controller, interacting with the system to adjust the action, thereby influencing the input vector u. Following this, the input to the block Σ (Plant) yields the output vector z, in accordance with dynamics (2.11). The plant dynamics Σ are then linked back to the block $\mathcal{C}$ to refine the controller's action based on the pod's position and velocity information.



$\mathcal{G}$: Reference trajectory generator
$\mathcal{C}$: Controller
Σ : Plant
x^* : Reference trajectory
u^* : Reference input
x_f : Final condition
z : Output State vector

Figure 3.1. Block diagram of the control strategy.

As previously mentioned, through this modular strategy, changes to one of the blocks $\mathcal{G}$ or $\mathcal{C}$ do not require changes to the other block: this enables us to address both problems independently.

3.2 Reference Trajectory Generation

References Various scientific papers and books have been reviewed for the generation of the speed reference trajectory. Since transportation systems most similar to Urbanloop are urban railway traffic systems, it has been useful to search for information related to handling similar ones, such as rail systems. The paper [32] addresses a similar problem: optimizing a train's trajectory to minimize energy consumption while considering constraints related to the train's speed. Additionally, to adhere to the train's planned timetable, it is necessary to minimize delays and respect speed constraints. The problem is formulated as a multi-phase optimal control problem and solved using the pseudospectral method.

Conversely, the paper [1] analyzes a different system, but the optimization strategy is particularly interesting due to its similarity to ours. In particular, a convex programming algorithm is presented to compute the numerical solution for the minimum fuel powered descent guidance problem associated with Mars pinpoint landing. Specifically, the problem involves reformulating the trajectory optimization problem, which has nonconvex control constraints, as a finite-dimensional convex optimization problem, resulting in a second-order cone programming problem. Even though, we did not employ convexification techniques as in [1], this reference has been an inspiring example for the total discretization method. Further, [19] extensively examines various optimization methods for generating the optimal trajectory of a train-based system. The authors propose energy-efficient control strategies founded on numerical algorithms to compute the optimal trajectory.

In summary, these papers provide valuable insights, particularly regarding the formulation of constraints and the optimization problem. However, it is important to note that, in our case, the primary objective is to minimize travel time between the start and end points, rather than focusing on power consumption, as seen in the references analyzed. Additionally, unlike conventional railway traffic systems, there is no predefined timetable. Instead, the goal is to find a solution that minimizes arrival time as much as possible, while accounting for constraints on speed, acceleration, and input, ensuring adherence to limits and maintaining passenger comfort aboard the pod.

Given the initial and final conditions for the pod, our objective is to generate the reference trajectory x^* corresponding to the reference input u^*. The aim is to minimize the travel time for the pod to reach the final destination at a given velocity. First, we need to clearly define the initial and final conditions that the system must meet, along with the various constraints concerning the system's dynamics, input, and state. Additionally, we will introduce the cost function associated with this problem. Subsequently, we will present the solution obtained using the discretization method, accompanied by the results of simulations conducted in Matlab.

3.2.1 Constraints

In this section, we outline the constraints the solutions (2.18) must satisfy.

Initial condition We define $t_0 = 0$ as the initial instant of time and x_0 as the initial value of the state vector. Assuming that the pod begins its journey from the starting point y_0 with a given velocity $v_0 \in \mathbb{R}$, we can express the initial conditions as

$$\mathsf{x}(0) = \mathsf{x}_0 := \begin{pmatrix} y_0 \\ v_0 \end{pmatrix}, \tag{3.1}$$

with $y_0, v_0 \in \mathbb{R}$.

Final condition We denote the desired final value of the state vector as $\mathsf{x}_\mathsf{f} \in \mathbb{R}^{n_x}$. It is essential that the pod arrives at the endpoint $y_f \in \mathbb{R}$ with a given velocity $v_f \in \mathbb{R}$. We denote the final conditions as follows

$$\mathsf{x}_\mathsf{f} := \begin{pmatrix} y_f \\ v_f \end{pmatrix}, \tag{3.2}$$

with $y_f, v_f \in \mathbb{R}$.

It is important to note that the initial and final velocities of the pod are not assumed to be zero. This requirement ensures that the optimization problem accounts for the optimal trajectory not only between the initial and final stations but also when the initial or final position involves a switch for changing the direction on the rail.

Speed Constraints The pods must adhere to strict velocity and acceleration constraints for safety reasons and comfort. Specifically, we must satisfy

$$V_{\min}(\mathsf{x}_1(\mathsf{t})) \leq \mathsf{x}_2(\mathsf{t}) \leq V_{\max}(\mathsf{x}_1(\mathsf{t})), \quad \forall t \in \mathbb{R}_{\geq 0}. \tag{3.3}$$

where $V_{\min}, V_{\max} \in \mathcal{PW}(\mathbb{R}, \mathbb{R})$, where $\mathcal{PW}(\mathbb{A}, \mathbb{B})$ is the set of piecewise constant functions from $\mathbb{A}$ to $\mathbb{B}$. The vehicle's speed is generally limited to 50 km/h along the entire route, regardless of the physical conditions. However, when approaching curves, the speed must be reduced to 20 km/h to prevent damage to the railway switches. Additionally, there may be areas where the maximum speed should be further restricted (to 30 km/h for istance), such as highly populated areas or regions close to the urban center.

These speed limitations are directly linked to the pod's position within the capsule, denoted by x_1. Therefore, speed adjustments should be made with consideration of the pod's exact location on the track.

Acceleration Constraints For acceleration, it is crucial to ensure that its value remains bounded throughout the entire trajectory

$$A_{\min} \leq \mathsf{f}_{\mathsf{red}_2}(\mathsf{x}, u, v_{\text{wind}}) \leq A_{\max}. \tag{3.4}$$

Here, $\mathsf{f}_{\mathsf{red}_2}(\mathsf{x}, u, v_{\text{wind}}) := \frac{rK_\mu}{M}u - \nu(\mathsf{x}_1, \mathsf{x}_2, v_{\text{wind}})$ is the second component of the function $\mathsf{f}_{\mathsf{red}}$ as defined in (2.16). The parameters $A_{\min}$ and $A_{\max}$ represent the acceleration constraints, which are essential for meeting the requirements set by Urbanloop and for maintaining passenger comfort. The values of $A_{\min}$ and $A_{\max}$ can be found in Appendix A.1.

Current constraints Regarding the reference quadratic current u, it must be bounded to prevent overloading the system's actuators.

$$I_{q,r_{\min}} \leq u(t) \leq I_{q,r_{\max}} \quad \forall t \in \mathbb{R}_{\geq 0}. \tag{3.5}$$

The values $I_{q,r_{\min}}$ and $I_{q,r_{\max}}$ are defined in Appendix A.1.

Dynamic Constraints The system will move according to the dynamics described by the state equation (2.18). Given this dynamic behavior, we have

$$\dot{\mathsf{x}} = \mathsf{f}_{\mathsf{red}}(\mathsf{x}, u, v_{\text{wind}}) \tag{3.6}$$

where $\mathsf{f}_{\mathsf{red}}$ is given in (2.16).

3.2.2 Cost Function

To formulate the optimization problem, it is necessary to define the cost function to be minimized. The objective is to minimize the time to bring the capsule from x_0 to x_f as already mentioned. The classical way to deal with time minimization is to define the cost function as follows [7], for any initial condition x_0 of the form (3.1), any final condition x_f of the form (3.2), and any measurable control input u in the sense of Lebesgue

$$C(\mathsf{x}_0, \mathsf{x}_\mathsf{f}, u) = \int_0^{t_f} 1\, dt, \tag{3.7}$$

$$\text{subject to } \begin{cases} \dot{\mathsf{x}} = \mathsf{f}_{\mathsf{red}}(\mathsf{x}, u, v_{\text{wind}}) \\ \mathsf{x}(t_f) = \mathsf{x}_\mathsf{f} \end{cases}$$

where $t_f \in \mathbb{R}_{\geq 0}$.

3.2.3 Optimal Control Problem

In light of Sections 3.2.1 - 3.2.2, our goal is to address the optimal control problem outlined by

$$(u^*, \mathsf{x}^*, t_f^*) \in \underset{u, t_f}{\arg\min} \quad C(\mathsf{x}_0, \mathsf{x}_\mathsf{f}, u) = \int_0^{t_f} 1\, dt \tag{3.8}$$

$$\text{subject to } \begin{cases} \dot{\mathsf{x}} = \mathsf{f}_{\mathsf{red}}(\mathsf{x}, u, v_{\text{wind}}) & \text{(Dynamic Constraint)} \\ V_{\min}(\mathsf{x}_1) \leq \mathsf{x}_2(t) \leq V_{\max}(\mathsf{x}_1) & \text{(Speed Constraint)} \\ A_{\min} \leq \mathsf{f}_{\mathsf{red}_2}(x, u, v_{\text{wind}}) \leq A_{\max} & \text{(Acceleration Constraint)} \\ I_{q, r_{\min}} \leq u(t) \leq I_{q, r_{\max}} & \text{(Current Constraint)} \\ \mathsf{x}(t_f) = x_f & \text{(Final Constraint)} \end{cases}$$

3.2.4 Discretization method

Due to the fact that the state model is non-linear, the optimization problem (3.8) may be hard to solve analytically. It is therefore possible to use a numeric approach by the so-called total discretization method [30, Chapter 9] to obtain an approximate solution to (3.8).

The method consists in employing a numerical integration method to solve the differential equations. We discretize time using a period $T \in \mathbb{R}_{>0}$ applying the direct Euler method:

$$\mathsf{x}_{i+1} = \mathsf{x}_i + T \mathsf{f}_{\mathsf{red}}(\mathsf{x}_i, u_i, v_{\text{wind}}), \quad \forall i \in \mathbb{N}, \tag{3.9}$$

where x_i and u_i are defined as $\mathsf{x}_i := \mathsf{x}(iT)$ and $u_i := u(iT)$. To handle the discontinuity in the control input, we employ Zero-Order-Hold (ZOH) devices. This approach ensures that the control input $u(t)$ remains constant between sampling intervals, i.e., $u(t) = u(iT)$ for $t \in [iT, (i+1)T[$. As a result, the input is discretized, and the system evolves with piecewise constant control values within each time step, effectively handling the discontinuity by holding the control signal steady until the next sampling instant. We discretize all the constraints described in (3.8) as follows

$$\begin{aligned} &V_{\min}(\mathsf{x}_{i,1}) \leq \mathsf{x}_{i,2} \leq V_{\max}(\mathsf{x}_{i,1}) \quad \forall i \in \mathbb{N}, \\ &A_{\min} \leq \mathsf{f}_{\mathsf{red}_2}(\mathsf{x}_i, u_i, v_{\text{wind}}) \leq A_{\max} \quad \forall i \in \mathbb{N}, \\ &I_{q,r_{\min}} \leq u_i \leq I_{q,r_{\max}} \quad \forall i \in \mathbb{N}. \end{aligned} \tag{3.10}$$

The cost function in (3.7) becomes

$$C(\mathsf{x}_0, u, \mathsf{x}_i) = \sum_{i=1}^{k} \int_{t_i}^{t_{i+1}} 1 \, dt = \sum_{i=1}^{k} T = kT, \tag{3.11}$$

where $k \in \mathbb{N}$. Thus, the problem in (3.8) is modified as

$$(\{u_i^*\}_{0 \leq i \leq k^*}, k^*) \in \underset{\{u_i\}_{0 \leq i \leq k}, k}{\arg\min} \quad C(x_0, u_i, x_i) = kT \tag{3.12}$$

$$\text{subject to} \begin{cases} \mathsf{x}_{i+1} = \mathsf{x}_i + T\mathsf{f}_{\mathsf{red}}(\mathsf{x}_i, u_i, v_{\text{wind}}) & \text{(Dynamic Constraint)} \\ V_{\min}(\mathsf{x}_{i,1}) \leq \mathsf{x}_{i,2} \leq V_{\max}(\mathsf{x}_{i,1}) & \text{(Speed Constraint)} \\ A_{\min} \leq \mathsf{f}_{\mathsf{red}_2}(t_i, \mathsf{x}_i, u_i, v_{\text{wind}}) \leq A_{\max} & \text{(Acceleration Constraint)} \\ I_{q,r_{\min}} \leq u_i \leq I_{q,r_{\max}} & \text{(Input Constraint)} \\ \mathsf{x}(t_k) = \mathsf{x}_f & \text{(Final Constraint)} \end{cases}$$

3.2.5 Optimal solution

The optimal solution to the trajectory generation problem, as formulated in (3.12), is the optimal discrete iteration k^* as well as the sequence of inputs $\{u_i^*\}_{0 \leq i \leq k^*}$ that minimizes the cost function C, subject to the dynamic constraints (3.9) and the speed constraints (3.10). The optimal trajectory $\mathsf{x}^*(t)$ is then obtained by applying $u^*(t)$ to the system (2.16) starting from the initial condition x_0. Therefore, the optimal input corresponds to

$$u^*(t) = u_i^*, \quad \forall t \in [iT, (i+1)T[. \tag{3.13}$$

To numerically solve the problem using the direct Euler method and perform simulations, the example provided in [30, p. 214] has been particularly insightful, as it has been adapted to align with the specific state dynamics and constraints of our system, allowing us to derive the desired optimal trajectory.

A key aspect of this adaptation is that N, representing the number of intervals, corresponds to the parameter k used earlier in (3.12). The objective is to find the optimal k, denoted as $k^* = N^*$, that effectively solves the problem.

Looking more closely at the example [30, p. 214], the problem involves the direct discretization (using *fmincon.m* in Matlab) of a minimal time problem, described by the following system:

$$\begin{cases} \dot{x_1} = x_2, \\ \dot{x_2} = u, \\ |u| \leq 1, \end{cases}$$

where the objective is to minimize the time required to move from the point $(0,0)$ to $(0,-1)$. This problem has been casted as:

$$\begin{cases} \dot{x_1} = x_2, \\ \dot{x_2} = v, \end{cases}$$

where v is defined as:

$$v := bu + \nu(x_1, x_2, v_{\text{wind}}), \quad b := \frac{K_\mu}{rM}.$$

In this formulation, unknown parameters and disturbances, such as v_{wind} and α, are set to zero. Various constraints, including limits on current, speed, and acceleration, are incorporated into the discretized constraint script. The initial and final conditions were also modified according to the different scenarios.

3.2.6 Numerical results

Various simulations and tests were conducted using Matlab. Depending on the dynamic constraints and the number of interval steps, feasible solutions to the problem can be achieved. Specifically, three distinct CASES are analyzed in the following sections, each highlighting different types of speed constraints and distances to be covered. As mentioned in Section 3.2.1, the speed limits could vary depending on the path to be traversed. Table 3.1 summarizes the key characteristics of these cases, emphasizing the resulting performance metrics, such as final time and average velocity over the entire path.

Case	**Distance [m]**	**Speed Constraints**	T_{final} **[s]**	V_{mean} **[km/h]**
1	1000	-	85.23	42.20
2	1000	1 turn under 30 km/h	106.53	33.79
3	1500	2 turns under 30 km/h	171.85	31.42

Table 3.1. Summary of CASES related to the Generation of the Reference Trajectory.

Case 1 In the simplest scenario, the speed constraints are assumed to be constant throughout the entire path:

$$V_{\min} \le \mathsf{x}_2(\mathsf{t}) \le V_{\max}, \quad \forall t \in \mathbb{R}_{\ge 0}.$$

In this case, the speed is constrained between $V_{\min} = 0$ km/h and $V_{\max} = 50$ km/h $\approx$ 13.89 m/s.

To analyze this case, the path was discretized into $N^* = 100$ steps, starting from the initial point $(0, 0)$ and ending at $(1000, 0)$. The results are illustrated in Fig. 3.2, which shows the trajectories for position, velocity, acceleration, and control input, along with their respective constraints. The optimal time achieved was 85.23 seconds, resulting in a mean velocity of

$$v_{\text{mean}} = \frac{1000\,\text{m}}{85.23\,\text{s}} \approx 11.73\,\text{m/s} \approx 42.24\,\text{km/h}.$$

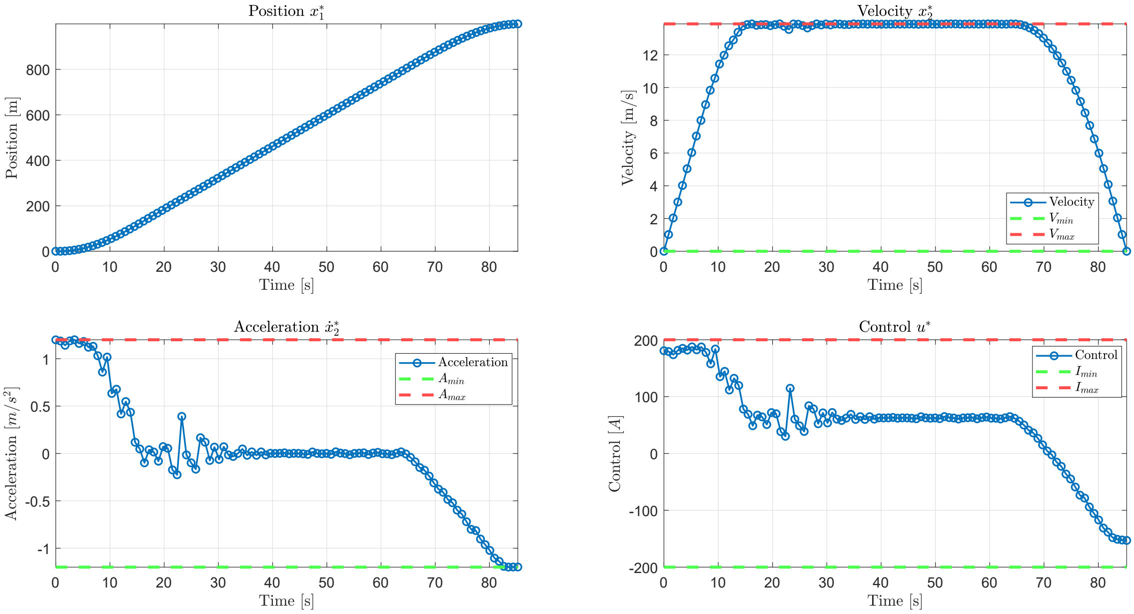


Figure 3.2. Reference Trajectory *Case 1*

Case 2 This case extends the constraints established in CASE 1 by incorporating a turn along the path. During this turn, the maximum speed is constrained to 30 km/h within a designated area of 250 m. Specifically, the speed limits are defined as follows

- A maximum speed of 50 km/h between 0 and 750 m;
- A reduced maximum speed of 30 km/h $\approx$ 8.33 m/s between 750 and 1000 m.

The results of this simulation are illustrated in Fig. 3.3. Using a discretization parameter $N^* = 100$ for the entire path, including the segment where the turn occurs, the optimal time achieved is 106.53 seconds. The simulation begins at the initial point $(0, 0)$ and ends at $(1000, 0)$, resulting in a mean velocity of

$$v_{\text{mean}} = \frac{1000\,\text{m}}{106.53\,\text{s}} \approx 9.39\,\text{m/s} \approx 33.79\,\text{km/h}.$$

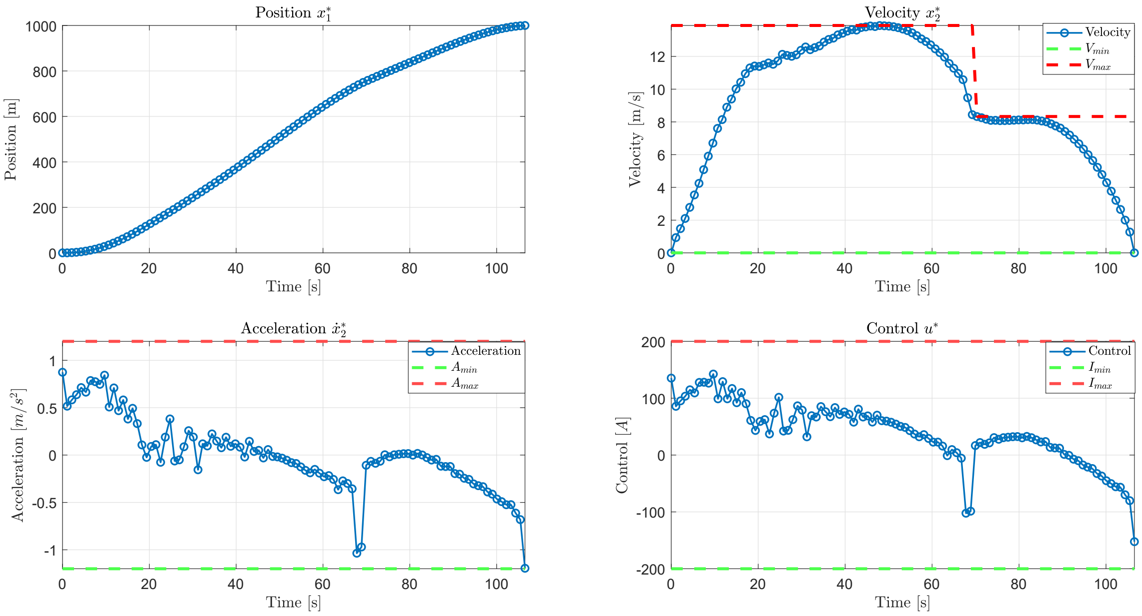


Figure 3.3. Reference Trajectory *Case 2*

Case 3 This case introduces two turns along the path, each with a speed limit of 30 km/h over a distance of 150 m. The speed constraints are defined as follows

- A maximum speed of 50 km/h from 0 to 100 m;
- A reduced maximum speed of 30 km/h from 100 to 250 m;
- A maximum speed of 50 km/h from 250 to 1200 m;
- A reduced maximum speed of 30 km/h from 1200 to 1350 m;
- A maximum speed of 50 km/h from 1350 to 1500 m.

The results are summarized in Fig. 3.4. With a discretization parameter $N^* = 100$, the optimal time achieved is 171.85 seconds. The trajectory optimization begins at the initial point $(0, 0)$ and ends at $(1500, 0)$, resulting in a mean velocity of

$$v_{\text{mean}} = \frac{1500\,\text{m}}{171.85\,\text{s}} \approx 8.73\,\text{m/s} \approx 31.42\,\text{km/h}.$$

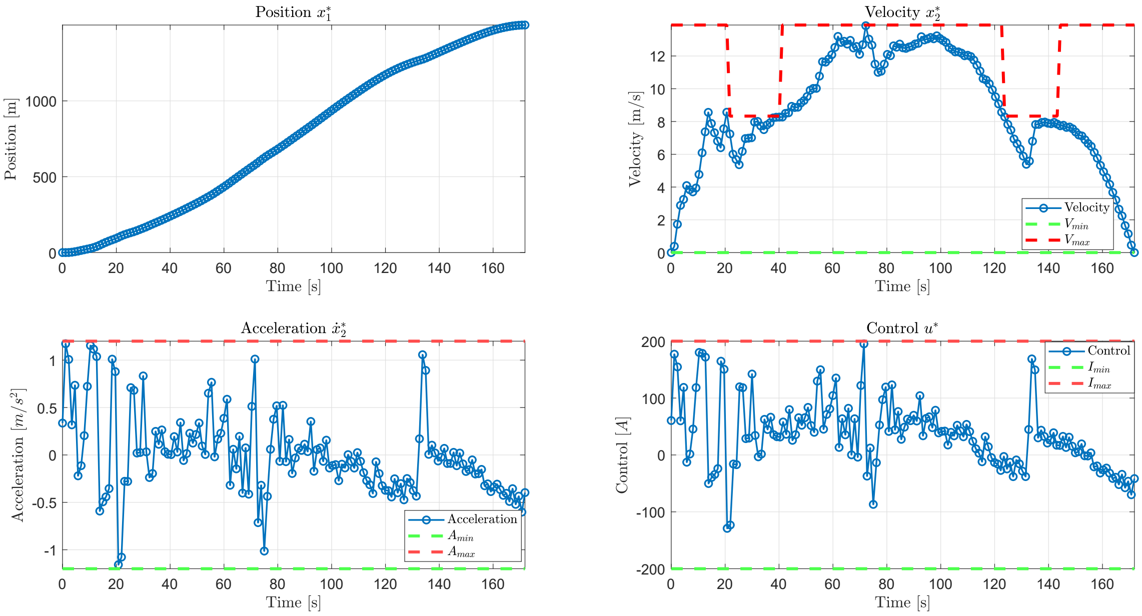


Figure 3.4. Reference Trajectory *Case 3*

The generation of the reference trajectory profile yielded promising results. However, determining the optimal number of steps for trajectory simulation can be both challenging and time-consuming. Variations in the number of steps, N, can significantly impact the results. Generally, the choice of N^*, and thus identifying the best value for a feasible trajectory, depends on the total distance to be traversed and the specific speed constraints. Consequently, N^* must be carefully evaluated for each case to ensure accurate and effective trajectory simulation.

The graphs reveal that although the problem is feasible with the given constraints, the acceleration graph often exhibits chattering effect. This phenomena may affect the comfort of passengers onboard. To mitigate this issue, it might be useful to impose constraints on the acceleration variation or to use a smoothing function.

In practical applications, it would be particularly valuable to track the path of interest with varying curves and speed constraints to determine if a feasible solution can be obtained with the given resolution. The code is designed to be flexible and can be adapted to generate real-world paths, incorporating the same turns and constraints along the track.

3.3 Design of the Control Strategy

Now that we have seen how to generate the reference trajectory and input, we focus on the design of a Closed-Loop strategy to track the reference trajectory.

3.3.1 Control Algorithm

In this context, it is crucial to recognize that the proposed mathematical model differs from the actual system. Consequently, relying solely on an open-loop trajectory, such as the one discussed in Section 3.2, will not suffice for achieving the desired reference. To address this, it is essential to implement a closed-loop control law to effectively counteract any disturbances or modeling errors the system may encounter.

Different sources have been reviewed [10, 21, 33], where the LQR method has been employed to control various systems. In particular, the reference [33] provides an overview of the application of different methods in controlling intelligent vehicle lateral dynamics. For this purpose, several control techniques, including LQR, were developed based on the path tracking error model. Through simulations conducted in Matlab/Simulink, the performance of these methods was analyzed.

Our approach adopts a similar strategy by first constructing an error model through linearization of the system around the reference trajectory. We then apply the proposed LQR method to this linearized system. This approach is justified by the observation that, at the initial time, the state of the capsule is closely aligned with the starting point x_0 of the reference trajectory. Consequently, it is reasonable to utilize linearization techniques in the neighborhood of the reference trajectory.

The schema 3.5 provides further details about the implementation of this control method.

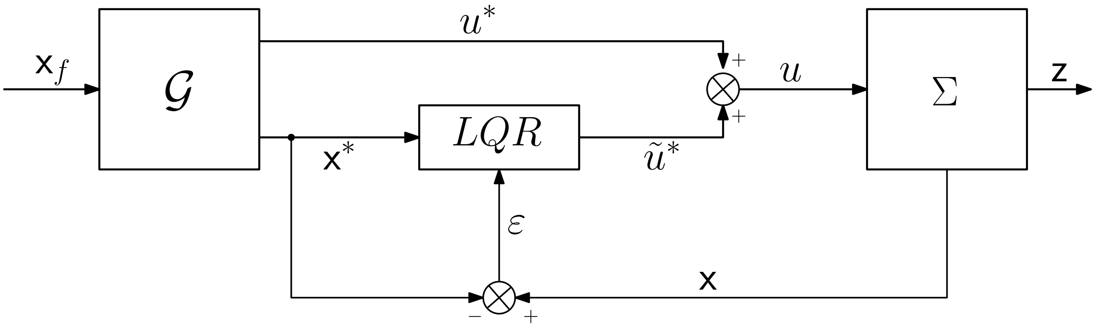


Figure 3.5. Control Algorithm scheme

The control input u is determined as $u := u^* + \tilde{u}^*$, where u^* represents the optimal control input generated by the reference trajectory, and $\tilde{u}^*$ is an additional control input designed to enhance stability and performance. The LQR method is employed to design $\tilde{u}^*$. The input u is applied to the system (2.18), denoted as Σ, which provides feedback related to the state vector x. This feedback signal is crucial for constructing the error dynamics ε, defined as the difference between x and the reference state x^*. The desired performance is that, at the final time specified by the reference trajectory, the difference between the reference and the system output should approach zero.

To effectively address the reference tracking problem, we analyzed the error dynamics of the system given by (2.16), resulting in a linearized model. We then studied the controllability and observability of the linearized system before applying the method to the model under study.

3.3.2 Linearization of the error dynamics

As a first step, we aim to transform the reference tracking problem into a problem of driving the error dynamics to zero. To this end, let us define

$$\begin{aligned} \varepsilon &:= \mathsf{x} - \mathsf{x}^* \\ \tilde{u} &:= u - u^*, \end{aligned} \tag{3.14}$$

where ε represents the difference between the state vector and the reference trajectory, and $\tilde{u}$ represents the difference between the input and the optimal input state.

Using (3.14) in (2.11), we obtain the dynamics of ε

$$\dot{\varepsilon} := \begin{pmatrix} -\varepsilon_2 \\ -\nu(\varepsilon_1 + \mathsf{x}_1^*, \varepsilon_2 + \mathsf{x}_2^*, v_{\text{wind}}) + \nu(\mathsf{x}_1^*, \mathsf{x}_2^*, v_{\text{wind}}) \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{K_\mu}{rM} \end{pmatrix} \tilde{u}, \tag{3.15}$$

where ν is defined as in (2.14). Given that we are considering small oscillations (i.e., less than 10°) of the slope angle that characterizes the weight force, we can approximate $\sin(\alpha(x_1)) \approx \alpha(x_1)$, Thus (2.14) can be considered as:

$$\nu(\mathsf{x}_1, \mathsf{x}_2, v_{\text{wind}}) \approx g\alpha(\mathsf{x}_1) - \frac{F_{\text{sec}}(F_{\text{res}}, \mathsf{x}_2)}{M} + \frac{\rho C_x S}{M}\left(\mathsf{x}_2 - v_{\text{wind}}\right)^2. \tag{3.16}$$

It is important to note that in the second component of (3.15), it is not possible to substitute $\nu(\varepsilon_1, \varepsilon_2, v_{\text{wind}})$ instead of $\nu(\varepsilon_1 + \mathsf{x}_1^*, \varepsilon_1 + \mathsf{x}_1^*, v_{\text{wind}}) - \nu(\mathsf{x}_1^*, \mathsf{x}_2^*, v_{\text{wind}})$ since the function ν is a nonlinear function.

To solve this problem, we will study the linearized dynamics of (3.15) at the origin of the systems defined as $(\varepsilon, \tilde{u}) := (\bar{0},0)$.

Linearizing (3.15) around $(\bar{0},0)$ gives

$$\dot{\varepsilon} = A(\mathsf{x}_2^\star)\varepsilon + B\tilde{u}, \tag{3.17}$$

where the matrix A is computed as the Jacobian matrix of the function $f(\varepsilon, \tilde{u})$ with respect to ε, evaluated at the equilibrium point $(\bar{0},0)$, that means

$$A := \left.\frac{\partial f(\varepsilon, \tilde{u}, v_{\text{wind}})}{\partial \varepsilon}\right|_{(\bar{0},0)} = \begin{pmatrix} 0 & 1 \\ \left.\frac{\partial \nu(\varepsilon,\tilde{u},v_{\text{wind}})}{\partial \varepsilon_1}\right|_{(\bar{0},0)} & \left.\frac{\partial \nu(\varepsilon,\tilde{u},v_{\text{wind}})}{\partial \varepsilon_2}\right|_{(\bar{0},0)} \end{pmatrix} \tag{3.18}$$

and the partial derivative of the function $\nu(\varepsilon, \tilde{u})$ with respect to ε_3, evaluated at the equilibrium point gives

$$\left.\frac{\partial \nu(\varepsilon, \tilde{u}, v_{\text{wind}})}{\partial \varepsilon_1}\right|_{(\bar{0},0)} = g\dot{\alpha}(\varepsilon_1 + \mathsf{x}_1^\star) - g\dot{\alpha}(\mathsf{x}_1^\star) = g\ddot{\alpha}(w_1)\varepsilon_1, \quad w_1 \in [\varepsilon_1 + \mathsf{x}_1^\star, \mathsf{x}_1^\star]. \tag{3.19}$$

By applying the Mean Value Theorem [11], the difference $\dot{\alpha}(\varepsilon_1+\mathsf{x}_1^\star)-\dot{\alpha}(\mathsf{x}_1^\star)$ can be written as $\ddot{\alpha}(w_1)\varepsilon_1$, where w_1 lies between $\varepsilon_1+\mathsf{x}_1^\star$ and $\mathsf{x}_1^\star$. Since ε_1 is small, the term $g\ddot{\alpha}(w_1)\varepsilon_1$ becomes negligible. In addition, if the second derivative of the function $\alpha(\mathsf{x})$ is small near $\mathsf{x}_1^\star$, the curvature $\ddot{\alpha}(w_1)$ is also small, further diminishing the magnitude of this term. Therefore, the expression (3.19) can be approximated as zero.

For what concerns the partial derivative of the function $\nu(\varepsilon, \tilde{u}, v_{\text{wind}})$ with respect to ε_2, evaluated at the equilibrium point we have

$$\left.\frac{\partial\nu(\varepsilon,\tilde{u},v_{\text{wind}})}{\partial\varepsilon_2}\right|_{(\bar{0},0)} = \frac{-1}{M}\frac{\partial F_{sec}(F_{res},\varepsilon_2+\mathsf{x}_2^\star)}{\partial\varepsilon_2} + \frac{2\rho C_x S}{M}(v_{wind}-\mathsf{x}_2^\star) = \frac{2\rho C_x S}{M}(V_{wind}-\mathsf{x}_2^\star) \tag{3.20}$$

where the term $\frac{\partial F_{sec}(F_{res},\varepsilon_2+\mathsf{x}_2^\star)}{\partial\varepsilon_2}$, since we are assuming to have strictly positive values $\varepsilon_2+\mathsf{x}_2^\star$ along the defined interval and the function $F_{\text{sec}}(F_{\text{res}},\dot{y})$ (2.7) gives the constant term $-F_{sec,0}$ for positive values of speed, is equal to 0.

After these substitutions (3.19) and (3.20), we can re-write (3.18) as

$$A = \begin{pmatrix} 0 & 1 \\ 0 & \dfrac{2\rho C_x S}{M}(V_{\text{wind}}-\mathsf{x}_2{}^*(t)) \end{pmatrix} \tag{3.21}$$

. The matrix B is defined as $B := \left.\frac{\partial f(\varepsilon,\tilde{u},v_{\text{wind}})}{\partial\tilde{u}}\right|_{(0,0)}$ and assumes the following values

$$B = \begin{pmatrix} 0 \\ \dfrac{K_\mu}{rM} \end{pmatrix} \tag{3.22}$$

At this point, it is extremely important to remark that our linearized system (3.17) is a Linear Time-Varying (LTV) system, since the matrix A contains a parameter that varies with respect to the time, i.e. $\mathsf{x}_2^*(t)$.

3.3.3 Controllability and Observability Properties

Controllability is a fundamental concept in control system design, ensuring that the system's output state can be guided to any desired position in the output space through appropriate manipulation of the input variables [5].

Observability, on the other hand, refers to the ability to infer the complete internal state of a dynamic system based solely on its measurable inputs and outputs over time. If a system is observable, it means that by analyzing the outputs generated in response to the inputs, one can accurately reconstruct the entire state of the system at any given moment [22].

In order to study the controllability and observability of the linearized system, we consider the time-varying parameter x_2^* as constant. Therefore, we have

$$A = \begin{pmatrix} 0 & 1 \\ 0 & \dfrac{2\rho C_x S}{M}(V_{\text{wind}}-\mathsf{x}_2{}^*) \end{pmatrix}, \quad B = \begin{pmatrix} 0 \\ \dfrac{K_\mu}{rM} \end{pmatrix}, \quad C := \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{3.23}$$

Controllability Analysis According to the Kalman criterion [28], a system is controllable if the controllability matrix M_c, defined as

$$M_c := \begin{pmatrix} B & AB \end{pmatrix}, \tag{3.24}$$

has full rank, which is equal to the order of the system, i.e., $n_x = 2$.

We first compute the product AB as follows

$$AB = \begin{pmatrix} 0 & 1 \\ 0 & \frac{2\rho C_x S}{M}(V_{\text{wind}} - x_2^*) \end{pmatrix} \cdot \begin{pmatrix} 0 \\ \frac{K_\mu}{rM} \end{pmatrix} = \frac{K_\mu}{rM} \begin{pmatrix} 1 \\ \frac{2\rho C_x S}{M}(V_{\text{wind}} - \mathsf{x}_2{}^*) \end{pmatrix}.$$

Thus, the controllability matrix M_c is

$$M_c = \begin{pmatrix} B & AB \end{pmatrix} = \frac{K_\mu}{rM} \begin{pmatrix} 0 & 1 \\ 1 & \frac{2\rho C_x S}{M}(V_{\text{wind}} - \mathsf{x}_2{}^*) \end{pmatrix}.$$

The rank of M_c can be determined by computing its determinant:

$$\det(M_c) = \left(\frac{K_\mu}{rM}\right)^2 \det \begin{pmatrix} 0 & 1 \\ 1 & \frac{2\rho C_x S}{M}(V_{\text{wind}} - \mathsf{x}_2{}^*) \end{pmatrix} = -\left(\frac{K_\mu}{rM}\right)^2.$$

Since $\frac{K_\mu}{rM} \neq 0$, it follows that $\det(M_c) \neq 0$. This implies that M_c has full rank, confirming that the system is controllable.

Observability Analysis According to the Kalman criterion [28], a system is observable if the observability matrix M_o, defined as

$$M_o := \begin{pmatrix} C \\ CA \end{pmatrix}, \tag{3.25}$$

has rank equal to the order of the system, which is $n_x = 2$.

To compute CA, we have:

$$CA = C \cdot A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 \\ 0 & \frac{2\rho C_x S}{M}(V_{\text{wind}} - \mathsf{x}_2{}^*) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & \frac{2\rho C_x S}{M}(V_{\text{wind}} - \mathsf{x}_2{}^*) \end{pmatrix}.$$

The observability matrix M_o is then:

$$M_o = \begin{pmatrix} C \\ CA \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & \frac{2\rho C_x S}{M}(V_{\text{wind}} - \mathsf{x}_2{}^*) \end{pmatrix}.$$

It is straightforward to verify that the rank of M_o is 2. In particular, by examining the 2×2 submatrix:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

which is clearly of full rank, we can conclude that M_o also has full rank. Therefore, the system is observable.

The analysis of controllability and observability for the complete model described by equation (2.12) is provided in Appendix A.4.

3.3.4 The LQR method

The LQR method is a classic optimal control technique used to minimize a quadratic performance index for linear dynamic systems [23]. This method seeks to minimize (or maximize) a quadratic cost function that incorporates both system states and control inputs, weighted appropriately during the design phase according to the given specifications.

In our scenario, the system to be controlled is described by the state equations

$$\dot{\varepsilon}(t) = A(t)\varepsilon(t) + B(t)\tilde{u}(t), \tag{3.26}$$

where $\varepsilon(t)$ represents the state vector, $A(t)$ and $B(t)$ are time-dependent matrices characterizing the system dynamics, and $\tilde{u}(t)$ is the control input.

The performance measure to be minimized is defined by the cost function:

$$J = \frac{1}{2}\varepsilon^T(t_f)H\varepsilon(t_f) + \int_{t_0}^{t_f} \frac{1}{2}\left[\varepsilon^T(t)Q(t)\varepsilon(t) + \tilde{u}^T(t)R(t)\tilde{u}(t)\right] dt, \tag{3.27}$$

where H and $Q(t)$ are real symmetric positive semi-definite matrices, $R(t)$ is a real symmetric positive definite matrix, and t_0 and t_f are the initial and final times, respectively.

The solution to this optimal control problem is given by

$$\tilde{u}^*(t) = -R(t)^{-1}B^T(t)K(t)\varepsilon(t), \tag{3.28}$$

where the matrix $K(t)$ is determined by numerically integrating the Riccati differential equation:

$$\dot{K}(t) = -Q(t) + K(t)B(t)R^{-1}(t)B^T(t)K(t) - K(t)A(t) - A^T(t)K(t), \tag{3.29}$$

with the boundary condition:

$$K(t_f) = H. \tag{3.30}$$

To enhance the performance of the LQR method, adjustments to the weights of the matrices Q and R are required. These matrices have been considered constant across all trajectories and represent the cost of the state variables and control input, respectively. Furthermore, the weight matrix H is essential for solving the Riccati equation and determining the matrix $K(t)$.

3.3.5 LQR Method: Implementation and Results

The implementation of this approach in the given system necessitates solving the time-varying Riccati differential equation through numerical techniques. In this context, the MATLAB solver `ode15s` was utilized to integrate the Riccati equation (3.29). The output of this integration is the time-varying matrix $K(t)$, which is essential in formulating the optimal control law (3.28). By fine-tuning the weighting matrices Q, R, and H in accordance with the system's performance criteria, the LQR controller proficiently minimizes the cost function while maintaining stability and responsiveness throughout the variation of the system's dynamics.

Performance criteria It is essential to highlight the key performance criteria used in selecting the weights for the matrices Q, R, and H. The primary objective is for the system to closely follow the reference trajectory from the initial point, x_0, to the final point, x_f. Given that the final time represents the arrival at the destination, it is crucial for the error at this moment, $\varepsilon(t_f)$, to be as close to zero as possible. Additionally, minimizing the error, $\varepsilon(t)$, throughout the entire trajectory is important to ensure high precision in following the reference path. Furthermore, it is important to ensure that the system adheres to constraints related to speed and control inputs to guarantee safety and prevent the motors from operating under excessive load.

Tuning The parameters were adjusted to best meet the performance criteria presented before. Additionally, since the control input $\tilde{u}^*$ only becomes non-zero in the presence of disturbances and varying parameters, it was crucial to run simulations under these conditions to thoroughly test the performance of the LQR method. Simulations included perturbations, such as wind speed v_{wind}, and variations in typically uncertain parameters, as the slope angle α, and passenger mass M_{add}, to evaluate the robustness of the controller against both disturbances and changes in unknown parameters.

The matrices Q, R, and H were chosen after multiple trials, aiming to achieve an error close to zero at the final time, $\varepsilon(t_f)$, ensuring that the system states closely follow the reference states. Specifically, the following values have been selected.

$$Q = \begin{bmatrix} 1\times 10^{-7} & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1, \quad H = \begin{bmatrix} 0.002 & 0 \\ 0 & 1 \end{bmatrix}.$$

To select the weights of the matrices accurately, they were initially set to one and subsequently adjusted based on the results of various simulations. Specifically, it was necessary to reduce the first weight of the matrix Q to decrease the error related to the position of the system. Further adjustments were made by reducing the first component of the matrix H, which improved the results, and then slightly increasing the first weight of Q. Setting this value too low risks not meeting input constraints. Additionally, performance can be improved by decreasing the weight related to the speed error state in Q; however, this results in a smaller reduction in error. Therefore, reducing the positional component of Q appears to be the optimal choice in this case, as it effectively reduces both position and speed-related errors.

What is particularly unusual is that, unlike the typical use of the LQR method — where it is usually necessary to increase the weights of matrix Q to penalize states and thus reduce state errors — it was necessary in this case to significantly decrease those weights. Indeed, several simulations conducted using only the linearized model conform to the behavior predicted by general LQR theory. However, when applying the model corresponding to system (2.18), it became necessary to reduce these values. A possible explanation could be that the simplifications made to derive the linearized model introduce substantial differences between the two models. As the nonlinear contributions are introduced and varied in the system, it becomes necessary to decrease the weights in matrix Q to achieve better state error control. Furthermore, it is important to highlight that the states of position and velocity differ significantly in magnitude: the position state ranges from 10^2 to 10^3, whereas the velocity state does not exceed 10^1. This disparity may also elucidate the necessity for a substantial reduction of the error position weight of matrix Q.

In particular, to perform this tuning, the CASES related to trajectory generation previously developed in Section 3.2.6 were considered. Simulations were conducted by varying the previously mentioned parameters.

Results within the Trajectory Cases The results obtained from applying the LQR method have been analyzed for each CASE described before in Section 3.2.6, using the matrices values specified in Section 3.3.5. The following table 3.2 presents the results obtained.

Case	**Input Constraints**	**Speed Constraints**	**RMSE ϵ_1 [m]**	**RMSE ϵ_2 [m/s]**	**$\epsilon_1(t_f)$ [m]**	**$\epsilon_2(t_f)$ [m/s]**
1	Respected	Respected	12.4102	0.1872	9.8883	-0.6019
2	Respected	Respected	7.7580	0.1372	5.2622	-0.6199
3	Respected	Respected	9.0035	0.2441	5.1731	-0.1032

Table 3.2. LQR Application Results for the three *Cases*

The table provides a summary of the results for the three CASES, indicating whether the input and speed constraints were satisfied. It also includes the RMSE values for the position error ε_1 and speed error ε_2, along with the final position error $\epsilon_1(t_f)$ and speed error $\epsilon_2(t_f)$.

In general, the input and speed constraints were met across all cases. For the RMSE values related to the position error along the trajectory, ε_1, a value below 15 meters can be considered acceptable. The RMSE for the speed error consistently remained low in all cases, indicating minimal discrepancy between the system's speed and the reference trajectory speed. Regarding the final positional error, $\epsilon_1(t_f)$, a value greater than zero indicates that the pod stops beyond the final station, while a value less than zero means it stops before reaching the station. Ideally, values between -5 and 5 meters are preferred, as this range corresponds to the vicinity of the station. As for the final speed error, $\epsilon_2(t_f)$, an acceptable value would be below 5 km/h $\approx$ 1.4 m/s, as the system is equipped with a mechanical braking mechanism capable of fully stopping the pod if its speed falls below 5 km/h.

To summarize, these results can be considered satisfactory for the different cases. However, to fully assess the effectiveness of the controller for the proposed model, it is necessary to introduce perturbations, such as variations in wind speed v_{wind}, or modify uncertain parameters like the passenger mass M_{add} or the slope angle α.

Perturbation Analysis and Robustness Testing To evaluate the robustness of the control law, several tests were conducted for Case 3. Specifically, this case simulates two turns along a path of 1500 meters, with a total duration of 172 seconds. Various simulations were carried out under the following conditions:

- One passenger (75 kg) on board;
- Two passengers (150 kg) on board;
- Wind speed of +10 km/h applied between 30 seconds and 70 seconds;
- Wind speed of -10 km/h applied between 40 and 70 seconds;
- Slope angle of +5% applied between 500 meters and 650 meters;
- Slope angle of -5% applied between 500 meters and 650 meters.

Table 3.3 summarizes the results of these simulations for this case.

Disturbances	**Input Constraints**	**Speed Constraints**	**RMSE ε_1 [m]**	**RMSE ε_2 [m/s]**	$\varepsilon_1(t_f)$ **[m]**	$\varepsilon_2(t_f)$ **[m/s]**
$M_{\text{add}} = 75$ **kg**	Respected	Respected	6.7028	0.3820	16.3543	0.3908
$M_{\text{add}} = 150$ **kg**	Respected	Respected	19.6656	0.6840	18.6608	1.0345
$v_{\text{wind}} = 10$ **km/h**	Respected	**NOT respected (51.5 km/h)**	29.8393	0.6684	-3.0673	-0.6549
$v_{\text{wind}} = -10$ **km/h**	Respected	Respected	15.0586	0.6193	11.3411	0.3733
$\alpha = 5\%$	Respected	Respected	59.6996	2.1388	-23.6948	2.1962
$\alpha = -5\%$	Respected	**NOT respected (60 km/h)**	47.3496	1.1755	4.1618	-1.1954

Table 3.3. Simulation Results with disturbances/varying parameters on Case 3.

Overall, the input constraints are respected for all the different simulations. However, the speed constraints were violated in some instances, such as when wind speed was set to 10 km/h and in the case of a -5% slope angle, where a maximum speed of 60 km/h was recorded, exceeding the imposed limit of 50 km/h. For position and speed errors along the trajectory (RMSE ε_1 and ε_2), higher values were observed, particularly for the slope cases. For the final time position error $\varepsilon_1(t_f)$, large deviations were noted, especially with passengers on board. Additionally, a significant discrepancy in the final speed error $\varepsilon_2(t_f)$ was observed in the case of $\alpha = 5\%$. The most concerning scenarios are those involving slope angle variations during the path. This is due to the significant influence of this parameter on the model compared to the other parameters. For example, by examining the system in (2.15), it is evident that the weight force 2.3.3 contributes more significantly compared to the other forces in play. These simulation results underscore certain deficiencies in the current control method when subjected to disturbances and variations in unknown parameters.

To conclude, this chapter has documented the strategy employed for controlling the capsule, focusing primarily on the approach used for generating the reference speed trajectory and the feedback control method applied using the LQR method on the linearized system.

The results related to trajectory generation are particularly noteworthy, as it was possible to construct scenarios that simulate real-world conditions, such as turns along the path, where speed constraints must be further limited.

Concerning the control method's performance, the results show good behavior under nominal conditions. However, when disturbances are introduced or unknown parameters vary, the controller's performance degrades. In some cases, this leads to significant deviations, with the pod failing to precisely reach the final station.

A potential solution to this issue involves generating reference trajectories that account for slope variations. In the current method, the slope angle was set to zero during the trajectory generation process. By incorporating these slope changes, the system could improve its overall accuracy and robustness in real-world scenarios. This adjustment would allow the algorithm to better reflect actual operating conditions, leading to more reliable trajectory tracking.

The next chapter will focus on reconstructing the reference speed trajectory using experimental data previously employed for model validation. This process will involve generating the reference trajectory while ensuring compliance with the constraints applied in different test scenarios. The LQR control method will then be implemented to assess its performance, allowing for a comparison between the Urbanloop PI control law and the proposed control strategy.

Chapter 4

Simulation Results

In this chapter, we will demonstrate the application of the proposed control strategy, as outlined in Chapter 3, to the experimental data used for model validation. The objective is to compare the heuristic control law developed by Urbanloop with the proposed method. We begin by reconstructing the paths taken by the pod based on the experimental data, incorporating the associated speed constraints. These constraints will be used to generate the reference trajectory, as described in Section 3.2. Subsequently, the LQR method detailed in Section 3.3.4 will be applied to the linearized system. Additionally, a performance comparison between the heuristic control law utilized by Urbanloop and the proposed control method will be presented. The chapter concludes with simulation results to account for disturbances and variations in unknown parameters.

4.1 Implementation of the Advanced Control Algorithm

To evaluate the proposed control method, a thorough analysis of the received data is essential for accurately understanding the pod's trajectory and reconstructing the corresponding speed constraints. For this purpose, four out of the twenty-four tests were selected. Using the *logs* from the experimental data presented in Section 2.4.2, the positions where the pod made directional changes, such as right and left turns, were analyzed to reconstruct its trajectory.

These logs offer detailed information about the locations of turns along the path. Generally, the speed is restricted to a maximum of 50 km/h throughout the trajectory. However, during turns, specific speed constraints are imposed, limiting the maximum speed to 20 km/h. The segments with these constraints typically range from 100 to 150 meters in length.

Table 4.1 summarizes the relevant data for each test, including the total distance, the reconstructed speed constraints, the final time, and the mean velocity achieved along the trajectory.

Test	**Distance [m]**	**Speed Constraints**	T_{final} [s]	V_{mean} **[km/h]**
9	432	2 turns under 20 km/h	107	14.50
10	703	2 turns under 20 km/h	159	15.9
12	650	1 turn under 20 km/h	131	17.86
13	464	2 turns under 20 km/h	159	10.5

Table 4.1. Informations about trajectories with Speed Constraints for *Tests 9, 10, 12* and *13.*

4.1.1 Reference Trajectory Generation

Following the reconstruction of the path, the method described in Section 3.2 was employed to generate the reference speed trajectory for each test. For all tests, the maximum speed is set to 50 km/h, except in segments with turns, where the speed limit is constrained to 20 km/h.

The following table 4.2 summarizes the application of the reference trajectory generation method for *Tests 9, 10, 12*, and *13*, providing details on the optimal number of steps N^*, the final time, the mean velocity along the trajectory, and the percentage improvements. These improvements are calculated based on the difference between the mean velocity of the trajectories related to the experimental data, shown in Table 4.1, and the mean velocity achieved using the generated reference trajectory.

Test	N^*	T_{final} [s]	V_{mean} **[km/h]**	**Improvements** [%]
9	30	69.52	22.37	54
10	60	85.07	28.04	76
12	60	80.36	29.12	63
13	30	60.89	27.43	161

Table 4.2. Informations about the Reference Trajectory Generation for *Tests 9, 10, 12*, and *13.*

By comparing Tables 4.1 and 4.2, the results are quite promising. For each test, the approach described in Section 3.2 successfully generated feasible trajectories, respecting the constraints related to speed, input, and acceleration. The results demonstrate that the generated reference trajectories achieve a shorter final time, allowing the pod to maintain a higher average speed, compared to the original scenario.

As an example, the graph corresponding to the trajectory of *Test 10* will be presented in detail.

For this particular test, which covers a distance of 703 meters, the following speed constraints are considered:

- A turn with a speed limit of 20 km/h $\approx$ 5.56 m/s between 1 and 42 meters;

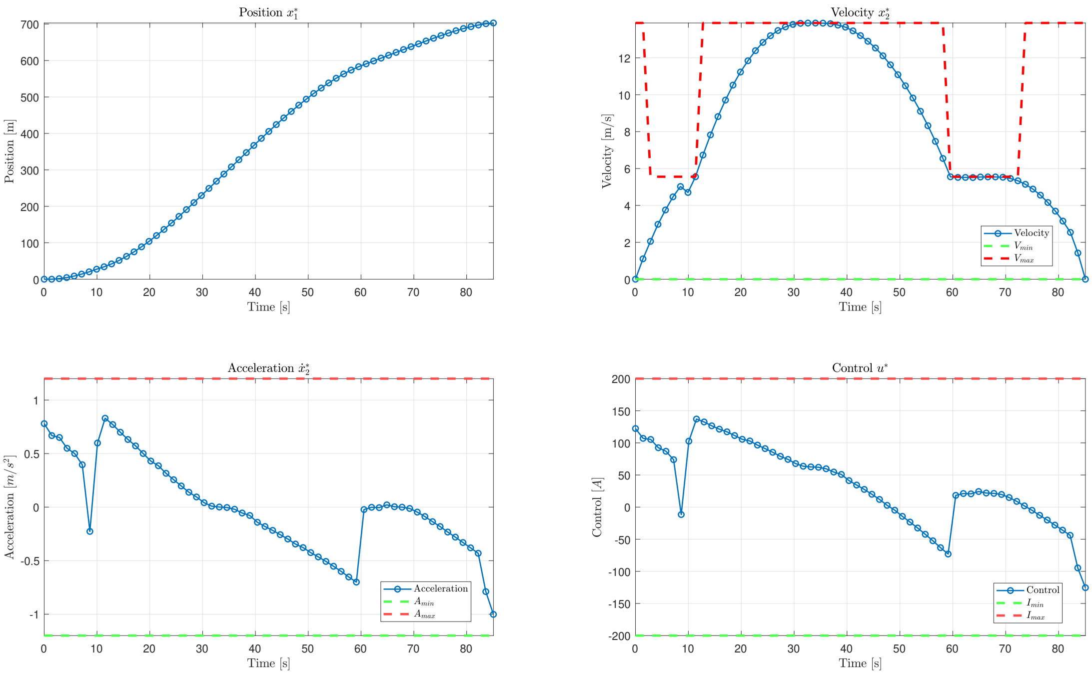


Figure 4.1. Reference Trajectory Generated for *Test 10*

- A turn with a speed limit of 20 km/h between 580 and 660 meters;
- A maximum speed limit of 50 km/h for all other sections.

Figure 4.1 illustrates the position, velocity, acceleration, and control input profiles, along with the corresponding constraints. Finding an optimal N^* that allowed the generation of a feasible trajectory for each test proved to be time-consuming and required a trial-and-error approach. This may be attributed to inefficiencies in the solver used.

With the reference trajectory now generated, the next step is to apply the LQR method, as explained in Section 3.3.4 to assess whether the system can accurately follow the reference path.

4.1.2 LQR application

For the application of the LQR method, the following values have been used, as detailed in Chapter 3.

$$Q = \begin{bmatrix} 1 \times 10^{-7} & 0 \\ 0 & 1 \end{bmatrix}, \quad R = 1, \quad H = \begin{bmatrix} 0.002 & 0 \\ 0 & 1 \end{bmatrix}.$$

The following table presents the results obtained by applying the LQR method to the linearized system across the four *Tests*.

Test	**Input Constraints**	**Speed Constraints**	**RMSE ϵ_1 [m]**	**RMSE ϵ_2 [m/s]**	$\epsilon_1(t_f)$ [m]	$\epsilon_2(t_f)$ [m/s]
9	respected	respected	6.8136	0.9927	0.7097	-1.5554
10	respected	respected	13.6894	0.3716	10.0975	-0.6942
12	respected	respected	8.6185	0.3244	6.1818	-0.7948
13	respected	respected	7.4412	0.5543	-3.3351	-1.2505

Table 4.3. LQR Application on *Tests 9, 10, 12*, and *13*.

The table provides a summary of the results for the four tests, indicating whether the input and speed constraints were satisfied. It includes the RMSE values for the position error ε_1 and speed error ε_2, along with the final position error $\epsilon_1(t_f)$ and speed error $\epsilon_2(t_f)$.

As we can observe, the criteria have been largely met: the constraints on both the input and velocity were respected. The RMSE values for position and velocity are consistent, and the final time errors can be considered acceptable across all tests.

We now examine the application of the LQR method in more detail for *Test 10*, shown in Fig. 4.2. In the illustrated figures, we observe the profiles of position, velocity, the associated errors for both, displayed on two appropriate metric scales for their respective magnitudes, and the control input applied to the system.

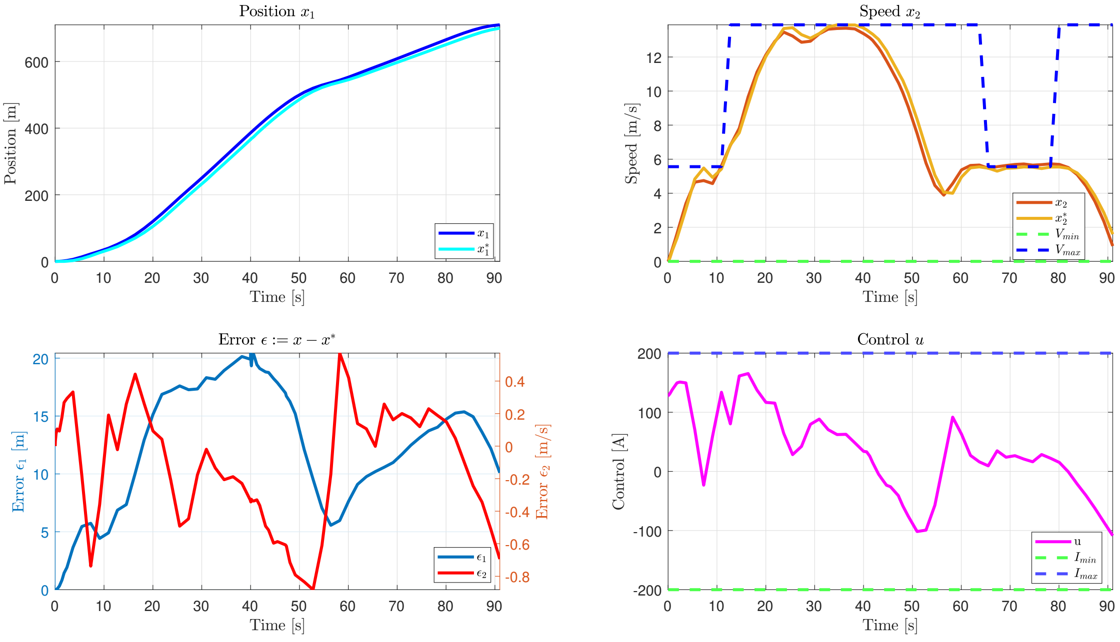


Figure 4.2. Application of LQR on *Test 10*

Specifically, in the top-left graph, we see the reference trajectory for the position, shown in light blue as x_1^*, and the actual pod position in dark blue denoted as x_1, which closely follows the reference trajectory with minimal error. Regarding the system's velocity, displayed in the top-right graph, the reference trajectory for velocity x_2^* is shown in orange, and the pod's actual speed is in red, denoted as x_2. Additionally, the minimum and maximum velocity constraints are shown in green and blue, respectively. In particular, the maximum velocity reflects the dynamic constraints related to the presence of turns in the zones between 0-42 meters and 580-660 meters.

In the bottom-left graph, we plot the errors along the trajectory, with position errors in blue and velocity errors in red. Lastly, in the bottom-right graph, we observe the control input, which consists of the sum of the optimal reference trajectory control $\mathtt{u}^*$ and the control derived from the LQR method, $\tilde{u}^*$. We can see that the input constraints are also respected.

For *Test 10*, we will see in detail the comparison between the Urbanloop PI control law and the proposed one.

4.2 Comparison between PI Urbanloop and the Proposed Control Algorithm

To effectively compare the two methods, it is important to recognize that the reference trajectories differ. For Urbanloop's PI controller, the trajectory is predefined, resulting in a final time of 159 seconds for *Test 10.* In contrast, the proposed method reaches the final station in 90.24 seconds. Consequently, a direct comparison based solely on time would not be appropriate. Instead, the comparison focuses on the system's speed relative to the distance traveled.
Figure 4.3 illustrates this relationship for both methods.

In the figure, the trajectory of the pod using the PI controller is shown in yellow, while the trajectory using the LQR method is depicted in blue. The minimum speed constraint is represented in green, and the maximum speed constraints are shown in red, indicated in km/h. Dynamic speed constraints corresponding to the turns are also illustrated, specifically between 0 and 42 meters and between 580 and 660 meters.

Notably, the area covered by the red trajectory is larger than that of the blue trajectory, indicating that the LQR method achieves a higher average speed compared to the PI method. Specifically, the LQR method yields an average speed of 28 km/h, whereas the PI method results in an average speed of 16 km/h.

This demonstrates that the objectives are effectively achieved with this control strategy. Specifically, the capsule is managed within a reduced time frame while adhering to a range of constraints, including speed limits, input limitations, and passenger comfort requirements. The control strategy not only optimizes performance but also ensures that operational constraints are consistently met, highlighting its efficacy in managing the capsule efficiently.

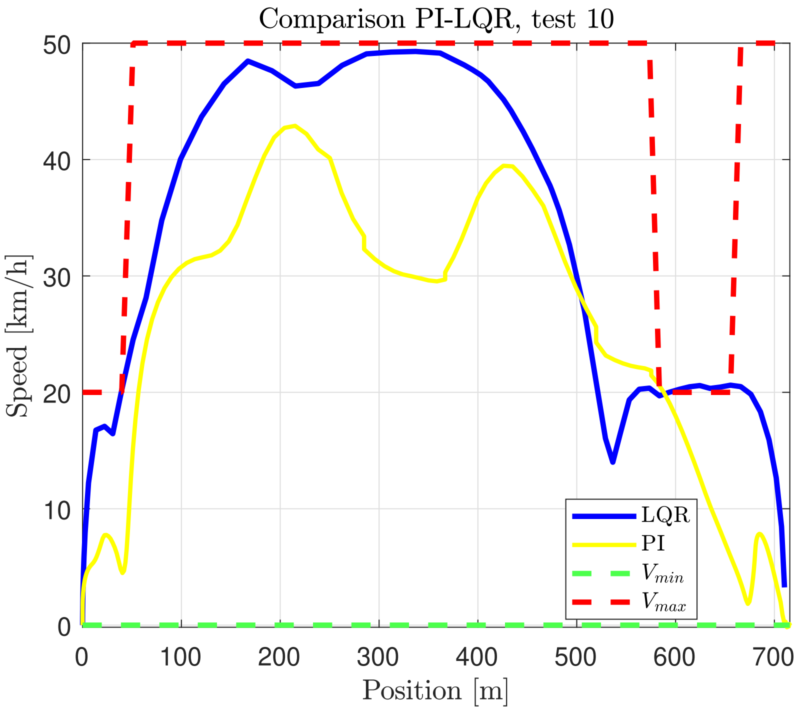


Figure 4.3. Comparison between LQR and PI on *Test 10*

4.3 Analysis of Disturbance Effects and Parameter Variations

To assess the effectiveness of the LQR method in practical scenarios, such as *Test 10*, simulations with varying parameters and disturbances were performed. This test, which covered a 703-meter path and had a duration of 91 seconds, was evaluated through multiple simulations in different conditions:

- One passenger (75 kg) on board;
- Two passengers (150 kg) on board;
- Wind speed of +10 km/h applied from 40 to 70 seconds;
- Wind speed of -10 km/h applied from 40 to 70 seconds;
- Slope angle of +5% applied from 200 to 350 meters;
- Slope angle of -5% applied from 200 to 350 meters.

The results of these simulations are summarized in Table 4.4.

Disturbances	**Input Constraints**	**Speed Constraints**	**RMSE** ϵ_1 **[m]**	**RMSE** ϵ_2 **[m/s]**	$\epsilon_1(t_f)$ **[m]**	$\epsilon_2(t_f)$ **[m/s]**
$M_{\text{add}} = 75$ **kg**	respected	respected	4.9219	0.5769	6.4972	-0.1099
$M_{\text{add}} = 150$ **kg**	respected	respected	15.2494	0.9104	-0.9898	0.3735
$v_{\text{wind}} = 10$ **km/h**	respected	NOT respected (52 km/h)	34.5758	0.6922	63.2987	-0.5219
$v_{\text{wind}} = -10$ **km/h**	respected	respected	22.8380	1.1403	-50.2185	-0.9257
$\alpha = 5$ %	respected	respected	61.0191	2.0345	-100.5502	-0.1471
$\alpha = -5$ %	respected	NOT respected (61.31 km/h)	48.5509	1.0231	64.1057	-1.2205

Table 4.4. Simulation Results with disturbances/varying parameters on *Test 10.*

Analyzing Table 4.4, it is evident that the input constraints were consistently satisfied across all cases. However, this was not the case for speed constraints. Specifically, when introducing wind at 10 km/h or a slope of -5%, the maximum speed values exceeded 50 km/h.

For position and speed errors along the trajectory (RMSE ε_1 and ε_2), higher values were observed, particularly for the slope and wind simulations. The final time position error $\varepsilon_1(t_f)$ showed significant deviations, especially in scenarios involving wind and slope. These deviations are concerning because positive errors indicate that the capsule will overshoot the final station, while negative errors suggest that it will not reach the final destination. On the other hand, the final speed error $\varepsilon_2(t_f)$ was considered acceptable across all simulations.

The most critical scenarios involve variations in slope angle and wind power during the trajectory. In the study case, ensuring that the pod reaches the final destination as specified by the reference trajectory is essential. The observed errors, particularly the final time position error $\varepsilon_1(t_f)$, indicate that the current control method may not be satisfactory under these perturbations. For managing slope angles, it would be beneficial to accurately track the path the capsule must follow. This approach could allow for modifications to the reference trajectory to account for slope variations, improving overall control effectiveness. Regarding the introduction of wind power, the results indicate a notable deficiency in reaching the final destination.

In conclusion, this chapter has applied the proposed control strategy to real-world scenarios, such as the ones in the experimental data, comparing it directly with the PI controller used by Urbanloop to assess potential improvements. Specifically, the proposed control method demonstrates significant enhancements over the PI controller, allowing the capsule to achieve an average speed between $22 - 30$ km/h along the route. However, the results are less impressive when disturbances or unknown parameter variations are introduced.

Conclusions

In conclusion, this work has made notable progress towards enhancing the control law for guiding the pod along the circuit, with a focus on improving speed and ensuring passenger comfort. The model study and validation have yielded interesting results, highlighting areas where further modifications are needed. Acquiring more detailed information on slope variations and wind power along the route is crucial for validating the assumptions of the constructed model. Additional tests that adjust these factors may reveal the need for further refinements based on the results obtained.

The control algorithm developed relies on this model, incorporating a reference trajectory generation capable of addressing various constraints, such as speed, input limits, and acceleration, to ensure passenger comfort while minimizing the arrival time at the final destination. The LQR method is employed on the linearized system around the reference trajectory to ensure the system follows this trajectory effectively.

A comparative study was conducted between the proposed algorithm and the existing Urbanloop PI controller to evaluate performance differences. The results demonstrate that the proposed control method significantly enhances performance compared to the PI controller, allowing the capsule to achieve an average speed of $22 - 30$ km/h along the route.

Regarding the control method's performance, the results indicate favorable behavior under nominal conditions. However, when disturbances are introduced or unknown parameters vary, the performance of the controller deteriorates, leading to substantial deviations and, in some instances, failure to precisely reach the final destination.

Future Improvements

The current work highlights several avenues for improvement and further development. A key step involves conducting additional tests to account for slope variations and wind power, which will help validate the model under more realistic conditions and refine necessary adjustments.

The control algorithm, initially developed for the reduced model, should also be tested with the complete model that incorporates the system's electrical behavior. It is important to note that the current control strategy is effective primarily under nominal conditions. An alternative approach could involve adjusting the reference trajectory to account for variations in slope angle and then applying the LQR method to this modified trajectory. However, this approach may be costly and might not be scalable for different scenarios. Therefore, exploring alternative control methods, such as MPC or H_∞ control, could provide enhanced robustness to these variations.

Effective testing of these control algorithms will require simulations with the pod on the actual circuit. A practical approach involves first implementing the reference trajectory generation and then replacing the heuristic PI controller with the proposed control algorithm.

Furthermore, expanding beyond single-pod scenarios is crucial. When multiple pods operate on the same circuit, additional challenges such as collision avoidance and intersection management must be addressed. Investigating cooperative strategies, potentially utilizing communication between autonomous vehicles, will be essential for effectively managing multi-pod operations.

Appendix

A.1 Parameters

The table A.5 presents the values and the description of different parameters that characterize the mathematical model.

M	Mass of the vehicle with no passengers	950	kg
S	Frontal area of the capsule	1.8	m^2
C_x	Drag coefficient of the capsule	0.8	[-]
ρ	Density of air	1.2	$\mathrm{kg.m}^{-3}$
E	Motor Supply Voltage	72	V
g	Gravitational constant	9.81	$\mathrm{m.s}^{-2}$
r	Wheel radius	0.24	m
k	Coefficient of dry friction	0.01	[-]
R	Stator resistance of the motor	2.1	$\mathrm{m}\Omega$
L	Stator inductance of the motor	58.2	$\mu\mathrm{H}$
n_p	Number of pole pairs of the motor	16	[-]
K_Φ	Electromagnetic flux coefficient	34.1	mWb
K_μ	Torque coefficient	1.64	Wb
K_p	Proportional gain of the current controller	500	[-]
K_i	Integral gain of the current controller	10	[-]
V_{min}	Minimum speed of the capsule	0	$\mathrm{m.s}^{-1}$
A_{min}	Minimum acceleration of the capsule	-1.2	$\mathrm{m.s}^{-2}$
A_{max}	Maximum acceleration of the capsule	1.2	$\mathrm{m.s}^{-2}$
$I_{\mathrm{q,r_{min}}}$	Minimum reference quadrature current	-200	A
$I_{\mathrm{q,r_{max}}}$	Maximum reference quadrature current	200	A
M_{add}	Mass of the passengers	$-$	kg
v_{wind}	Wind speed along the direction of capsule movement	$-$	$\mathrm{m.s}^{-1}$

Table A.5. Parameters of the model.

The values of the coefficients K_p and K_i have been chosen by looking at the Kelly Controller interface. Since it is programmable, these values can be changed.

The value of K_μ, denoted as the torque coefficient, is obtained using the formula

$$K_\mu = \frac{3}{2} n_p 2 K_\phi,$$

where:

- the constant $\frac{3}{2}$ originates from the equations derived by Park equations [27];
- n_p denotes the number of poles in the motor;
- the factor 2 arises we are considering the combined torque from two motors in each pod;
- K_ϕ represents the electromagnetic flux coefficient.

It is important to include an additional term for the mass of the pod, represented by the variable M_{add}. Specifically, an individual is assumed to have a mass of 75 kg, and a maximum of 2 individuals can be hosted per pod.

The variable v_{wind} represents the wind speed in the direction of the capsule's movement. Positive values indicate that the wind is opposing the capsule's direction. This parameter varies depending on weather conditions and the direction axis of the pod along the track.

A.2 Conversion table for the slop angle

To convert from degrees to percentage, the following formula can be used:

$$\text{Percentage} = \tan(\alpha) \cdot 100$$

where α is the angle in degrees.

The following table A.6 provides a comprehensive overview of the typical values used for the case study of the circuits deployed for Urbanloop.

Percentage [%]	**Radians**	**Degrees [°]**
1	0.01	0.57
2	0.02	1.15
3	0.03	1.72
4	0.04	2.29
5	0.05	2.86
6	0.06	3.43
7	0.07	4.00
8	0.08	4.57
9	0.09	5.14
10	0.1	5.71
11	0.11	6.28
12	0.119	6.84

Table A.6. Conversion table from percentage to radians and degrees.

A.3 Application of the Slow Manifold Technique

In this appendix, we present the application of the slow manifold technique to simplify the dynamics of the capsule system of Section 2.3.5.

Firstly, we recall that the input and the state variables of the system are the following:

- $u := i_{q,r}$ is the reference quadratic current,
- $x_1 := \xi$ is the integrator state,
- $x_2 := i_q$ is the quadratic current,
- $x_3 := y$ is the position of the capsule along the track,
- $x_4 := \dot{y}$ is the velocity of the capsule.

The dynamics of the system can be expressed as:

$$\begin{cases} \dot{x}_1 & = -x_2 + u, \\ \dot{x}_2 & = \frac{EK_i}{L} x_1 - \frac{(EK_p + R)}{L} x_2 + \frac{EK_p}{L} u, \\ \dot{x}_3 & = x_4, \\ \dot{x}_4 & = \frac{K_\mu}{rM} x_2 - g\sin(\alpha(x_3)) + \frac{F_{\text{sec}}(F_{\text{res}}, x_4)}{M} - \frac{\rho C_x S}{M} \left(x_4 - v_{\text{wind}}\right)^2 . \end{cases} \tag{1}$$

To simplify this system, we apply the slow manifold technique [15], which is a method used to analyze and reduce systems characterized by multiple time scales. The slow manifold technique identifies a lower-dimensional subspace where the fast variables equilibrate, allowing us to focus on the slow dynamics.

We first address the fast dynamics by setting $\dot{x}_2 = 0$. Solving for x_2, we obtain:

$$\dot{x}_2 = 0 \Rightarrow x_2 = \left(K_p + \frac{R}{E}\right)^{-1} \left(K_i x_1 + K_p u\right). \tag{2}$$

Substituting x_2 from (2) into the original system (1) and simplifying using:

$$\begin{aligned} a &:= \left(K_p + \frac{R}{E}\right)^{-1} K_i, \\ b &:= \left(1 - \left(K_p + \frac{R}{E}\right)^{-1} K_p\right) u. \end{aligned}$$

We obtain the reduced system:

$$\begin{cases} \dot{x}_1 & = -a x_1 + b u, \\ \dot{x}_3 & = x_4, \\ \dot{x}_4 & = \frac{K_\mu}{rM}[a x_1 + (1-b)u] - g\sin(\alpha(x_3)) + \frac{F_{\text{sec}}(F_{\text{res}}, x_4)}{M} - \frac{\rho C_x S}{M} \left(x_4 - v_{\text{wind}}\right)^2 . \end{cases} \tag{3}$$

Next, we simplify by setting $\dot{x}_1 = 0$, leading to:

$$\dot{x}_1 = 0 \Rightarrow x_1 = \frac{b}{a}u. \tag{4}$$

Substituting x_1 from (4) into the reduced system (3), we obtain the further simplified system:

$$\begin{cases} \dot{x}_3 & = x_4, \\ \dot{x}_4 & = \frac{K_\mu}{rM}u - g\sin(\alpha(x_3)) + \frac{F_{\text{sec}}(F_{\text{res}}, x_4)}{M} - \frac{\rho C_x S}{M}\left(x_4 - v_{\text{wind}}\right)^2. \end{cases} \tag{5}$$

By applying the slow manifold technique, we have successfully reduced the complexity of the original system while retaining its essential slow dynamics. The resulting system, described by (5), focuses on the slower dynamics of position and velocity, thereby simplifying the analysis and control design.

A.4 Controllability and Observability Analysis

In order to study the controllability and observability of the linearized system, we consider the time-varying parameter x_4^* as constant.

Therefore, we have

$$A := \begin{pmatrix} 0 & -1 & 0 & 0 \\ \frac{EK_i}{L} & -\frac{EK_p+R}{L} & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{K_\mu}{rM} & 0 & \frac{2\rho C_x S}{M}(V_{\text{wind}} - x_4^*) \end{pmatrix}, \quad B := \begin{pmatrix} 1 \\ \frac{EK_p}{L} \\ 0 \\ 0 \end{pmatrix}, \quad C := \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{6}$$

with $E, L, K_i, K_p, R, M, r, K_\mu \in \mathbb{R}_{>0}$.

Controllability analysis Since controllability is invariant under state-feedback, let us study the pair (A_F, B), where $A_F := A + BF$ with $F = \begin{pmatrix} 0 & 1 & 0 & 0 \end{pmatrix}$ reads

$$A_F = \begin{pmatrix} 0 & 0 & 0 & 0 \\ \frac{EK_i}{L} & -\frac{R}{L} & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{K_\mu}{rM} & 0 & \frac{2\rho C_x S}{M}(V_{\text{wind}} - x_4^*) \end{pmatrix}.$$

The Hautus criteria [17] points out that (A_F, B) is controllable if and only if $\forall\lambda \in \sigma(A_F)$, rank $\begin{pmatrix} A_F - \lambda I & B \end{pmatrix} = n$ where $n = 4$ is the state dimension.

Lemma 1. *The following statements are equivalent*

- $K_i \neq \frac{R}{L}K_p$ *and* $K_i \neq -\frac{2\rho C_x S}{M}(V_{wind} - x_4^*)K_p$*;*
- *the pair* (A, B) *is controllable.*

Proof. Clearly, $\sigma(A_F) = \left\{0, -\frac{R}{L}, 0, \frac{2\rho C_x S}{M}(V_{\text{wind}} - x_4^*)\right\}$ so that we need to study the three following cases:

1. $(\lambda = 0)$:

$$\begin{pmatrix} A_F - \lambda I & B \end{pmatrix} = \begin{pmatrix} A_F & B \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 1 \\ \frac{EK_i}{L} & -\frac{R}{L} & 0 & \frac{EK_p}{L} \\ 0 & 0 & 1 & 0 \\ 0 & \frac{K_\mu}{rM} & \frac{2\rho C_x S}{M}(V_{\text{wind}} - x_4^*) & 0 \end{pmatrix}$$

Simple operations on the lines of the above matrix ($l_1 \to l_4 \to l_2 \to l_1$) lead to the following condition

$$\text{rank} \begin{pmatrix} \frac{EK_i}{L} & -\frac{R}{L} & 0 & \frac{EK_p}{L} \\ 0 & \frac{K_\mu}{rM} & \frac{2\rho C_x S}{M}(V_{\text{wind}} - x_4^*) & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = 4$$

since the above matrix is upper triangular with non-zero diagonal elements.

2. $\left(\lambda = -\frac{R}{L}\right)$:

$$\begin{pmatrix} A_F - \lambda I & B \end{pmatrix} = \begin{pmatrix} \frac{R}{L} & 0 & 0 & 0 & 1 \\ \frac{EK_i}{L} & 0 & 0 & 0 & \frac{EK_p}{L} \\ 0 & 0 & \frac{R}{L} & 1 & 0 \\ 0 & \frac{K_\mu}{rM} & 0 & \frac{2\rho C_x S}{M}(V_{\text{wind}} - x_4^*) + \frac{R}{L} & 0 \end{pmatrix}$$

which is rank 4 if and only if $K_i \neq \frac{R}{L}K_p$ since $E, L, K_i, K_p, R, M, r, K_\mu \in \mathbb{R}_{>0}$.

3. $\left(\lambda = \frac{2\rho C_x S}{M}(V_{\text{wind}} - x_4^*)\right)$:

$$\begin{pmatrix} A_F - \lambda I & B \end{pmatrix} = \begin{pmatrix} -\lambda & 0 & 0 & 0 & 1 \\ \frac{EK_i}{L} & -\lambda - \frac{R}{L} & 0 & 0 & \frac{EK_p}{L} \\ 0 & 0 & -\lambda & 1 & 0 \\ 0 & \frac{K_\mu}{rM} & 0 & 0 & 0 \end{pmatrix}$$

which is rank 4 if and only if $K_i \neq -\frac{2\rho C_x S}{M}(V_{\text{wind}} - x_4^*)K_p$, since $E, L, K_i, K_p, R, M, r, K_\mu \in \mathbb{R}_{>0}$.

□

In conclusion, we can assert that the pair (A, B) is controllable if the following condition both hold:

- $K_i \neq \frac{R}{L}K_p$;
- $K_i \neq -\frac{2\rho C_x S}{M}(V_{\text{wind}} - x_4^*)K_p$.

The values presented above are not particularly restrictive in our case. Specifically, the first scenario can occur if the parameters K_p and K_i are selected accordingly. The second scenario is also probable, but since the system will remain in that configuration only briefly, it may be uncontrollable only for a short period. Consequently, the system's configuration will already change by then, implying that these values do not pose any issues regarding the controllability property of the system.

Observability Analysis According to the Kalman criterion [28], the system is observable if the matrix M_o defined as

$$M_o := \begin{pmatrix} C \\ CA \\ CA^2 \\ CA^3 \end{pmatrix} \tag{7}$$

has rank equal to the order of the system, i.e., $n_x = 4$.

Lemma 2. *The observability matrix M_o, as defined in (7), is a square matrix of size 4×4 and has full rank, i.e.,* $\text{rank}(M_o) = 4$.

Proof. To simplify the algebraic manipulation, we reparameterize the matrix A using the parameters $a, b, c, d \in \mathbb{R}$ as follows:

$$A := \begin{pmatrix} 0 & -1 & 0 & 0 \\ a & b & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & c & 0 & d \end{pmatrix} \tag{8}$$

where $a := \frac{EK_i}{L}$, $b := -\frac{EK_p+R}{L}$, $c := \frac{K_\mu}{rM}$, and $d := \frac{2\rho C_x S}{M}(V_{\text{wind}} - x_4^*)$.

Therefore, we obtain

$$M_o = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & c & 0 & d \\ 0 & c & 0 & d \\ ac & bc+cd & 0 & d^2 \\ ac & bc+cd & 0 & d^2 \\ abc+acd & d(bc+cd)-c(-b^2+a) & 0 & d^3 \end{pmatrix}. \tag{9}$$

It is easy to see that the columns of M_o are linearly independent since none of them can be expressed as a linear combination of the others. Thus, the rank of M_o is equal to 4 for $E, L, K_i, K_p, R, M, r, K_\mu \in \mathbb{R}_{>0}$. ☐

Based on these conclusions, the observability analysis results are satisfactory, as there are no identified limitations.

Bibliography

[1] Behcet Acikmese and Scott R. Ploen. Convex Programming Approach to Powered Descent Guidance for Mars Landing. 30(5):1353–1366, 2007.

[2] F. Andriaminahy, A. Amamou, S. Kelouwani, N. Zioui, A. Ghobadpour, and K. Agbossou. Comparative Study of Vehicle Aerodynamic and Rolling Resistance Coefficients Estimation Methods. In *2019 IEEE Vehicle Power and Propulsion Conference (VPPC)*, pages 1–5, 2019.

[3] Felix Blaschke. The principle of field orientation as applied to the new transvector closed-loop control system for rotating-field machine. 34:217–220, 1972.

[4] Liviu Sevastian Bocîi and Ioan Bele. The Influence of Tunnel Factor upon Aerodynamics Resistance at Advancing of High Speed Trains . *Annals of the University of Petroșani, Mechanical Engineering*, 2007.

[5] O. Bosgra, H. K. Kwakernaak, and G. Meinsma. *Design Methods for Control Systems*. Springer, 2007.

[6] Harrison D. Brailsford. Commutatorless direct current motor, 1955.

[7] G.C. Calafiore and L.E. Ghaoui. *Optimization Models*. Cambridge University Press, 2014.

[8] F.H. Clarke, Yu.S. Ledyaev, and R.J. Stern. Asymptotic Stability and Smooth Lyapunov Functions. *Journal of Differential Equations*, 149(1):69–114, October 1998.

[9] Cristián E. Cortés, Doris Sáez, Freddy Milla, Alfredo Núñez, and Marcela Riquelme. Hybrid Predictive Control for Real-Time Optimization of Public Transport Systems: Operations Based on Evolutionary Multi-Objective Optimization. *Transportation Research Part C: Emerging Technologies*, 18(5):757–769, 2010.

[10] R. Craig Coulter. Automatic steering methods for autonomous automobile path tracking. Technical Report CMU-RI-TR-92-01, Robotics Institute, Carnegie Mellon University, 1992.

[11] Richard Courant and Fritz John. *Introduction to Calculus and Analysis*. Classics in Mathematics. Springer Berlin Heidelberg, Berlin, Heidelberg, 1999.

[12] Jahanzeb Farooq and José Soler. Radio Communication for Communications-Based Train Control (CBTC): A Tutorial and Survey. 19(3):1377–1402, 2017.

[13] France TV Info. "On monte à deux adultes : l'Urbanloop, un nouveau mode de transport va être expérimenté pour Paris 2024 à Saint-Quentin-en-Yvelines", 2024.

[14] Rupprecht Gabriel, Werner Leonhard, and Craig J. Nordby. Field-Oriented Control of a Standard AC Motor Using Microprocessors. IA-16(2):186–192, 1980.

[15] C. W. Gear, T. J. Kaper, I. G. Kevrekidis, and A. Zagaris. Projecting to a Slow Manifold: Singularly Perturbed Systems and Legacy Codes. 4(3):711–732, 2005.

[16] Donald T. Greenwood. *Principles of Dynamics*. Prentice Hall, Englewood Cliffs, NJ, 2nd edition, 1988.

[17] Malo LJ Hautus. Controllability and Observability Conditions of Linear Autonomous Systems. In *Indagationes Mathematicae (Proceedings)*, volume 72, pages 443–448, 1969.

[18] András Hegyi. *Model predictive control for integrating traffic control measures*. Netherlands TRAIL Research School, 2004.

[19] Philip G. Howlett and Peter J. Pudney. *Energy-Efficient Train Control*. Advances in Industrial Control. Springer London, 1995.

[20] Keolis Innovation. Sustainable Transit: Keolis Nancy Partner of the Urbanloop Light Rail Pods, 2024.

[21] Hojin Jung, Dohyun Jung, and Seibum B. Choi. Lqr control of an all-wheel drive vehicle considering variable input constraint. *IEEE Transactions on Control Systems Technology*, 30(1):85–96, 2022.

[22] Hassan K. Khalil. *Nonlinear Systems*. Prentice Hall, Upper Saddle River, NJ, 3rd edition, 2002.

[23] D.E. Kirk. *Optimal Control Theory: An Introduction*. Dover Books on Electrical Engineering Series. Dover Publications, 2004.

[24] Le Monde. Les petites capsules autonomes d'Urbanloop profitent de l'effet JO. 2024.

[25] Ian P Milroy. *Aspects of automatic train control*. PhD thesis, Loughborough University, 1980.

[26] John O'Reilly. *Observers for linear systems*, volume 170. Academic press, 1983.

[27] R. H. Park. Two-reaction theory of synchronous machines generalized method of analysis-part i. *Transactions of the American Institute of Electrical Engineers*, 48(3):716–727, 1929.

[28] Sigurd Skogestad and Ian Postlethwaite. *Multivariable Feedback Control: Analysis and Design*. Wiley, Chichester, UK, 2nd edition, 2005.

[29] Gang Tao and Petar V Kokotovic. *Adaptive control of systems with actuator and sensor nonlinearities*. John Wiley & Sons, Inc., 1996.

[30] Emmanuel Trélat. *Contrôle optimal: Théorie & applications*. Vuibert, Paris, 2012.

[31] Université de Lorraine, Télécom Nancy. Urbanloop Project - Projets et Plateformes de Formation, 2024.

[32] Pengling Wang and Rob M.P. Goverde. Multiple-phase train trajectory optimization with signalling and operational constraints. 69:255–275, 2016.

[33] Tao Yang, Ziwen Bai, Zhiqiang Li, Nenglian Feng, and Liqing Chen. Intelligent vehicle lateral control method based on feedforward + predictive lqr algorithm. *Actuators*, 10(9), 2021.

[34] Jiateng Yin, Tao Tang, Lixing Yang, Jing Xun, Yeran Huang, and Ziyou Gao. Research and Development of Automatic Train Operation for Railway Transportation Systems: A Survey. *Transportation Research Part C: Emerging Technologies*, 85:548–572, 2017.