TÉCNICO LISBOA

Sparse Predictive Control for Vehicle Motion

John Mark Vergara Batabat

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisor: Prof. João Manuel Lage de Miranda Lemos

Examination Committee

Chairperson: Prof. João Fernando Cardoso Silva Sequeira
Supervisor: Prof. João Manuel Lage de Miranda Lemos
Members of the Committee: Prof. José Manuel Prista do Valle Cardoso Igreja

September 2018

DECLARATION

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

John Mark Vergara Batabat

Date: July 25, 2018

“Behind every great achievement is a dreamer of great dreams.”

— Robert K. Greenleaf

Acknowledgments

I would like to express my deepest appreciation to all those who provided me with the possibility to complete this Master's Thesis. A special gratitude I give to my dissertation supervisor Prof. João Manuel Lage de Miranda Lemos for giving me the opportunity to work under his supervision. This thesis would not have been possible without his support, patience, knowledge, and time. Additionally, I take this opportunity to express my deep sense of gratitude to Prof. José Manuel Prista do Valle Cardoso Igreja for his contribution to this work.

I gratefully acknowledge the funding I received from the Erasmus Mundus Action 2 EXPERTS – SUSTAIN Project for the entire stay in Portugal and the entire study at Instituto Superior Técnico (IST). The project made all of these to happen.

Last but not the least, to all the European students I met during my entire stay in IST who were always so helpful in numerous ways, to the mobility coordinators who assisted me in all academic processes and to all of my professors in my courses in IST, thank you.

May the Almighty God richly bless all of you.

Part of this work has been performed in the framework of project SPARSIS, funded by FCT under contract PTDC/EEIPRO/0426/2014

FCT
Fundação para a Ciência e a Tecnologia
MINISTÉRIO DA CIÊNCIA, TECNOLOGIA E ENSINO SUPERIOR

Abstract

This work proposes the development of a control strategy based on the use of ℓ_1-norm penalty to the cost of Model Predictive Control (MPC) scheme for autonomous vehicles. The resulting control strategy will then be called sparse predictive control or sparse controller, because of the type of action considered. The vehicle is modeled using a discretized version of single integrator model expressed in Cartesian coordinates for a linear formulation, and a discretized version of unicycle model expressed in polar coordinates for a nonlinear formulation. Simulation results are provided to validate and evaluate the viability of the proposed control strategy in relation to the optimization parameters and constraints as applied in a known environment inhabited with given obstacles. A comparison analysis to quadratic or conventional cost function of MPC is also conducted. Based on the results from simulations, the feasibility of the proposed control strategy for vehicle motion control is shown as well as the effect of the selection of the cost weights that configure the controller. Furthermore, it can suggest that sparse controller is a better alternative to a conventional implementation of MPC, especially on the basis of minimizing the cost incurred during the process, resulting to a minimal fuel consumption.

Keywords

Sparsity, Model Predictive Control, Receding Horizon Control, Mobile Robot, Obstacle Avoidance.

Resumo

Este trabalho propõe o desenvolvimento de uma estratégia de controlo baseada na aplicação da norma de penalização ℓ_1 ao custo do esquema de Modelo de Controlo Preditivo para veículos autónomos. A estratégia de controlo resultante será denominada controlo preditivo esparso, ou controlador esparso, devido ao tipo de ação considerado. O veículo será modelado utilizando uma versão discretizada do modelo de um integrador simples, expresso em coordenadas cartesianas, bem como uma versão discretizada do modelo unicíclico, expressa em coordenadas polares, de modo a obter uma formulação não linear. Serão apresentados resultados de simulação, a fim de validar e avaliar a viabilidade da estratégia de controlo proposta, relativamente a parâmetros de otimização e restrições, aplicados num ambiente conhecido e preenchido por um dado conjunto de obstáculos. Será também levada a cabo uma análise comparativa relativamente à função de custo quadrática, ou convencional, do Modelo de Controlo Preditivo. Tendo por base os resultados das simulações, será demonstrada a viabilidade da estratégia de controlo proposta para o controlo do movimento de um veículo, bem como o efeito da seleção dos pesos de custo que configuram o controlador. Adicionalmente, a análise sugere que o controlador esparso poderá ser uma alternativa mais vantajosa, relativamente à implementação convencional do Modelo de Controlo Preditivo, especialmente no que toca à minimização do custo associado ao processo, resultando num consumo de combustível mínimo.

Palavras Chave

Esparsidade, Controlo Preditivo baseado em Modelos, Controlo de Horizonte Recidivo, Robô Móvel, Evitar Obstáculos.

Contents

List of Figures

List of Tables

Listings

Acronyms

MPC	Model Predictive Control
GPC	Generalized Predictive Control
NMPC	Nonlinear Model Predictive Control
RHC	Receding Horizon Control
MHC	Moving Horizon Control
AV	Autonomous Vehicles
LQR	Linear Quadratic Regulator
LQG	Linear Quadratic Gaussian
MAC	Model Algorithmic Control
DMC	Dynamic Matrix Control
MATLAB	MATrix LABoratory
LAD	Least Absolute Deviations
LAE	Least Absolute Errors
CARIMA	Controlled Auto-Regressive and Integrated Moving Average
ARIMA	Auto-Regressive and Integrated Moving Average
LTI	Linear Time-Invariant

1

Introduction

Contents

Control systems are everywhere in our everyday lives. Nowadays, people are highly dependent on those things in one form or another. Applications of control systems have significantly increased through advances in computer technology and development of materials, that provide unique opportunities for highly efficient actuation and sensing [1]. Control systems have been applied by aerospace engineers to guidance, navigation, and control of spacecrafts for numerous space missions [2]. Chemical engineers have used control system processes in a chemical plants such as to control the temperature of steam, operating a jacketed reactor, maintaining a set ratio of reactants, and controlling the level of fluid in a tank [3]. Mechanical engineers have used the application of control system to control industrial machines [4], and electrical engineers have exploited control theory for many areas such as power electronics [5].

The purpose of this thesis is to develop a control strategy based on the application of Model Predictive Control (MPC) and sparsity optimization concepts for vehicles. The controller developed will then be studied to comply with the given constraints on the system and its environment.

This thesis begins with an understanding of MPC in a systematic way. A discussion is presented on the various concepts necessary for the optimality of the control design such as the cost function, mathematical model, and constraints. Then, sparsity is introduced in the cost function utilizing the ℓ_1-norm penalty in the regularization term. Furthermore, algorithms based on the ℓ_2-norm are implemented for comparison. After the development of the controller, the simulation is performed using the MATrix LABoratory (MATLAB) software. The MPC controller is coded in MATLAB script files utilizing several functions that deemed necessary. Various types of scenarios are simulated to validate controller performance. Different stationary obstacles are also being given in the operating environment.

In this chapter, the context and introductory background on the MPC is presented in the following section. After which, it will be followed by the literature review on this matter. The main contributions, the structure of the thesis, and some of the notations used are also included.

1.1 Context and Motivation

There is a rapid increase of interest on Autonomous Vehicles (AV), that includes airborne, space borne, ground, and submersible. As vehicles are becoming more and more autonomous, it will not be too distant that fully autonomous ones are around the corner. In the case of ground vehicles, many of the automotive companies are already into the testing of fully autonomous cars. A report from the consulting firm Navigant ranks the major players of the emerging driverless car industry. Navigant analysts see General Motors and Waymo as the clear industry leaders [6]. Waymo, formerly the Google Self Driving Car Project, is vying to become the first company to put autonomous cars without safety drivers onto the streets of California [7]. Only time will tell which companies to follow this progress. This will then be the mark of another era that will revolutionize the way driving is regulated.

In an AV, it is appropriate to ask what will be the way to control a vehicle for a given obstacle avoidance situation. This assessment can be done in the context of MPC, one of the modern control methods. The development of these modern control concepts can be traced back to the works of Kalman in the early 1960's [8,9]. Kalman formulated in his two ground-breaking papers a precise design for the optimal least squares transient control problem and later on became to be known as Linear Quadratic Regulator (LQR) / Linear Quadratic Gaussian (LQG). The LQR was designed to minimize an unconstrained quadratic objective function of state and inputs with no noise in the system and the full state is observed. The LQR algorithm had powerful stabilizing properties because of the infinite horizon [10]. The LQG, on the other hand, is an extension of LQR with the consideration of the presence of noise and the full system state now may not be directly observed. These works became a standard approach to solve control problems in a wide range of application areas [11]. But it made a less significant impact on control technology development in the process industries. These failures can be traced to:

1. its incapacity to deal with constraints
2. its incapacity to solve process nonlinearities
3. its uncertainty in modeling
4. uniqueness of its cost function
5. cultural reasons.

Because of the LQG shortcomings, it led to the development of a more advanced control method. In the late 1970's, various papers were released showing applications of MPC in the industry. Two of the most well-known papers were from the works of Richalet et al. [12] presenting MPC as Model Algorithmic Control (MAC) and of the Shell Oil company engineers Cutler and Ramakter [13] for their work on Dynamic Matrix Control (DMC) initially intended for its use in petroleum refineries. The underlying idea of both the theories was to use a dynamic model of the process to handle multivariable control systems and predict the effect of future control actions, which were determined by minimizing the predicted error subject to operational restrictions. The optimization is repeated at each sampling instant with updated information from the process. However, the earliest patent appears to be granted to Martin-Sanchez in 1976, who called his method simply Adaptive Predictive Control. From the name itself, it implies the exploitation of the presence of the internal model to obtain adaptive control, by adapting the model and relying on the optimization to obtain the appropriate control signals [14]. An adaptive MPC technique approach developed by Clarke et al., Generalized Predictive Control (GPC) [15, 16], has also received considerable attention. The term "predictive" appears in the name of GPC because of the application of minimum variance in predicted values on the finite future time. Additionally, the term "predictive" also

appears in MPC since the objective function is given in predictive values on the finite future time that can be computed by using the model [17].

In recent years the MPC landscape has changed drastically, with a large increase in the number of reported applications, significant improvements in technical capability, and mergers between several of the vendor companies [11]. Although some other options are considered, such as the so called Economic MPC [18], currently, the typical MPC formulations are commonly structured with quadratic cost function, both in the input and output cost. The choice is mainly due to its simplicity and to the possibility of making use of the properties from the LQR. Even though MPC have already been implemented in many ways and widely studied for AV, it is also of great interest to develop such controllers that will guarantee desirable path planning while considering fuel efficiency or minimality in the control effort. This proposition could be achieved by implementing the sparsity on its cost function which will penalize the number of nonzero elements in the control, thus it is the leading motivation of this work.

1.2 State of the Art

In recent years, sparsity has become a significant topic of researchers in various fields. It has been successfully applied to some real-world applications, mostly in the fields of image processing and computer vision [19].

In the field of controls, some literature has recommended the ℓ_1-norm regularization for control methods involving complex requirements on the control signals, such as in the tracking problems as proposed in [20]. In this paper, the trajectory generation is dealt with by proposing a formulation to obtain a control signal with compact representation and that changes only when needed, something often wanted in tracking. This formulation takes the shape of a constrained least-squares problem with sum-of-norms regularization, a generalization of the ℓ1-regularization.

Another interesting literature adapts tools from the area of compressed sensing and propose to design control packets via on-line minimization of a suitable ℓ_1 and ℓ_2 cost functions [21]. The authors investigated a networked control architecture for LTI plant models with a scalar input. A numerical example illustrates that sparsity reduces bit-rates, thereby making the paper proposal suited to control over unreliable and bit-rate limited networks.

Another literature to consider proves that the proper cost function to use involves the ℓ_1-norm, and not the popular quadratic forms, to minimize the amount of fuel consumed by spacecraft guidance and control algorithms [22]. The author shows that the penalty for not using the ℓ_1-norm is significantly more propellant consumption and undesirable continuous thrusting.

In recent works, modifications of conventional quadratic MPC methods have been developed to obtain spatially or temporally sparse control signals. One paper presented two stabilizing ℓ_1-regularized

MPC controllers for redundantly-actuated Linear Time-Invariant (LTI) systems, by means of terminal constraint set and terminal cost [23]. However, the paper only focuses on the investigation of theoretical basis in enhancing optimality, enlarging the domain of attraction, and minimizing conservativeness of the candidate terminal controllers. Another literature examines the effectiveness of sum-of-norms regularized with a numerical example for controlling a drone MPC [24]. Though the paper gives a good modification of the conventional cost function of MPC, it emphasizes only the use of the euclidean norm to sparsify the solution.

1.3 General Problem Formulation

The focus of this thesis is to develop, innovate, and test a control strategy for vehicle motion control. The application of sparsity in the control strategy is explored and will be later called the sparse predictive control or sparse controller. After the development process, the controller is tested for autonomous vehicle-like in a defined environment through simulations. The operating environment is subject to different stationary obstacles that the vehicle will have to avoid. It allows to execute preventive and collision avoidance actions of the vehicle. Then, it follows the performance verification of the controller through simulations in MATLAB. Some concepts which will be dealt with in this thesis are:

- The controller must be able to perform efficiently on a given vehicle model.
- Control, obstacles, and state constraints must be included in the control formulation.
- The vehicle must be able to navigate in the environment without colliding with any obstacle or other vehicle while attaining the objective.
- The robustness of the software for the controllers is to be tested.
- The tuning of parameters for linear and nonlinear formulations and its effects to the cost and performance of the algorithms.
- Consideration of the addition of terminal cost in the nonlinear formulation.

1.4 Contribution

This thesis presents an approach to study the application of sparsity to MPC control strategy for a self-governing vehicle in the essence of a given obstacle avoidance situation. The use of ℓ_1-norm regularization in the sparsity formulation is typically known to cause many elements in the control set to be equal to zero. As a result, it takes a little effort in control (that is, more fuel-efficient for vehicles) by a substantial margin compare to the other regularization terms.

The objective of this thesis is to develop a control strategy and study the implementation of the ℓ_1-norm, as well as the ℓ_2-norm in the regularization term or the control input cost. Then, a comparison of the performance of both implementations will be performed, enough to explain and deduce a conclusion on the sparsity idea. Both controllers are also simulated in an environment containing obstacles with different shapes to examine the extent of their capacities to handle hard constraints.

1.5 Thesis Outline

This thesis is organized as follows. **Chapter** 1 provides a short introductory discussions relevant to this thesis. **Chapter** 2 gives a background on optimal control and how it helps to the development of the MPC scheme. Moreover, the elements that appear in any MPC formulation, as well as its algorithm are detailed. The strategy of linear and nonlinear MPC are also presented. **Chapter** 3 focuses on the concept of sparsity and its different representations.

In **Chapter** 4, a step by step approach is presented leading to the development of the linear sparse predictive control algorithm. This chapter begins by providing the vehicle model used to represent the plant. Next, the design of the MPC is presented as well as the formulation of the sparse predictive control. Then, the obstacles constraints are given to better understand the equations used behind the forms of obstacle in the defined environment. After which, the solvers to optimize the problem are provided. Finally, simulations are made to analyze and compare the performance of the sparse controller to the conventional controller, and the respective results are discussed.

Chapter 5 establishes the nonlinear sparse predictive control algorithm. For this strategy, the unicycle kinematic model is used as a vehicle model. Simulations are then presented to verify and examine the controller developed. A comparison to the conventional or quadratic controller is also provided.

Chapter 6 paraphrases the thesis' contents and summarizes the main results. Conclusions are then made, followed by the presentation of guideline about the possible future work.

1.6 Notation

This section will describe the basic notation established throughout this thesis. The following list is organized below:

- Scalars and vectors can be represented by lowercase letters.
- The notation x_i can refer to the ith element of a set of scalars or vectors x_1,x_2,...,x_n or the state variables in the control and the notation $\dot{x}_i(\cdot)$ is for a time-varying variables of x. Additionally, u refers to control inputs, y refers to outputs, $\hat{y}$ refers to the future or predicted outputs, and r refers

to reference or set point.The context or text where the notation is used makes it more clearer to understand.

- $\mathbb{R}$ is used to denote the set of real numbers. The set of n-real vectors is denoted as $\mathbb{R}^n$, the set of real p-vectors is denoted as $\mathbb{R}^p$, the set of real p-vectors is denoted as $\mathbb{R}^q$, and the set of real m x n matrices is denoted as $\mathbf{R}^{mxn}$.
- $\forall x \in \mathbb{R}$ means x belongs to the $\mathbb{R}$ Euclidean real vector space.
- A (system matrix),and B (input matrix) are constant matrices in the state equation of a plant model with dimensions n x n and n x p, respectively. C (output matrix), and D (direct transmission term) are constant matrices of the output equations in a plant model with dimensions q x n and q x p, respectively.
- Q (output or state weighting matrix) and R (input weighting matrix) are weight factors in the cost function which are real symmetric positive definite (n x n) and real symmetric positive definite, respectively.
- $J(\cdot)$ represents the quadratic cost function for the optimization problem.
- $(\cdot)^T$ denotes the matrix transpose, $(\cdot)^*$ denotes an optimal solution, and $(\cdot)_{min}$ and $(\cdot)_{max}$ denote the lower and upper limits of variables, respectively.
- $\|\cdot\|$ denotes the vector norm and $\|\mathbf{x}\|^2$ denotes the squared Euclidean norm or ℓ_2-norm. In general, ℓ_p-norms are denoted as $\|x\|^p = (\sum_{i=1}^{n} |x_i|^p)^{1/p}$ for $0 \leq p < \infty$. Note that this is done instead of the common representation $\|x\|_p$ to not get confused with cases involving weight factors Q and ρ.
- Time t represents the continuous time, while the time k represents the discrete time.
- z^{-1} denotes the backward shift operator satisfying $z^{-1}f(t) = f(t-1)$ for any time function $f(t)$.

2

Model Predictive Control

Contents

MPC is a form of control scheme based on iterative, finite-horizon optimization problem to obtain for the current control action using the current state of the plant as the initial state [25]. This current state is used to make predictions of future values of the states or outputs. Then the optimization yields a sequence of optimal control moves (that is, manipulated input changes) so that the predicted response moves to the set point in an optimal manner [26]. In other words, it is used to predict a sequence of input control changes so that the output tries to follow the target or minimize the difference between them. Shown in Fig. (2.1) is the basic principle of MPC. According to a receding horizon strategy of this sequence of control values, only the first is actually applied to the plant and the process is repeated again obtaining a different optimal control sequence.

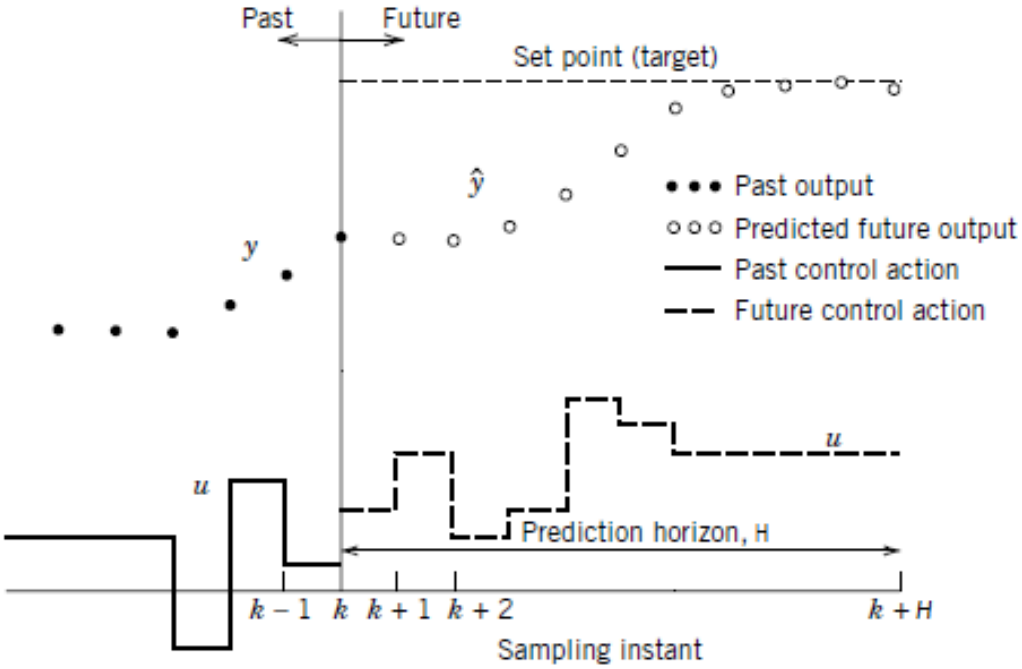

Figure 2.1: Model Predictive Control Basics (adapted from [26])

This chapter describes the basic requirements that are common to all model predictive controllers. First, it begins by presenting the requirements for optimal control systems which give the idea for MPC formulation. Then, it follows the presentation of additional elements for MPC.

2.1 System Requirements for Optimality

MPC has its roots in optimal control [18]. Classical control system design is generally a trial-and-error process in which various methods of analysis are used iteratively to determine the design parameters of

an acceptable system. The objective of optimal control theory is to determine the control signals that will cause a process to satisfy the physical constraints, and at the same time minimize or maximize some performance criterion [27]. The requirements for the formulation of optimal control system include but not limited to the following:

1. Mathematical model of the plant
2. Cost function
3. Constraints

2.1.1 Mathematical Model of the Plant

In the book of Naidu [28], a physical plant is described as a system with a set of linear or nonlinear differential or difference equations. The objective to model the process is to obtain a representation of the physical, biological, or information systems that adequately makes prediction about the response of the system. This section is mainly devoted to the systems or plants described by ordinary differential equations (ODE) in state-space form.

A state variables of the process at discrete current time k is defined as

$$x_1(k), x_2(k), \ldots, x_n(k) ,$$

and control inputs to the process at time k as

$$u_1(k), u_2(k), \ldots, u_n(k) ,$$

then the system may be described by n first-order differential equations in discrete time

$$\begin{aligned} \dot{x_1}(k) &= a_1(x_1(k), x_2(k), ..., x_n(k), u_1(k), u_2(k), ..., u_m(k), k) \\ \dot{x_2}(k) &= a_2(x_1(k), x_2(k), ..., x_n(k), u_1(k), u_2(k), ..., u_m(k), k) \\ \vdots & \qquad\qquad \vdots \\ \dot{x_n}(k) &= a_n(x_1(k), x_2(k), ..., x_n(k), u_1(k), u_2(k), ..., u_m(k), k) \end{aligned} \tag{2.1}$$

We then define the state vector, $x \in \mathbb{R}^n$ of the system as

$$x(k) = \begin{bmatrix} x_1(k) \\ x_2(k) \\ \vdots \\ x_n(k) \end{bmatrix} ,$$

and the control vector, $u \in \mathbb{R}^p$ as

$$u(k) = \begin{bmatrix} u_1(k) \\ u_2(k) \\ \vdots \\ u_n(k) \end{bmatrix}$$

The discrete-time state-space representation can then be written as

$$\dot{x}(k) = a(x(k), u(k), k) \tag{2.2}$$

where a is defined by looking in (2.1).

There are different classifications of a system. Systems are described according to being linear, nonlinear, time-invariant, and time-varying. If a system is *nonlinear* and *time-varying*, the discrete-time state equations are written as in (2.2). For a *linear*, *time-invariant* systems, it can be represented in the form

$$\dot{x}(k) = Ax(k) + Bu(k) \tag{2.3}$$

In a plant, a physical quantities are being observed or measured. These quantities are herein referred to as *outputs*, $y \in \mathbb{R}^q$ and are denoted in discrete time by

$$y_1(k),\, y_2(k),\, \ldots\, ,\, y_n(k).$$

If the outputs are *nonlinear*, *time-varying* functions of the states and controls, the output equations can thus be written by

$$y(k) = c(x(k), u(k), k) \tag{2.4}$$

If the output is related to the states and controls by a *linear*, *time-invariant* relationship, then

$$y(k) = Cx(k) + Du(k) \tag{2.5}$$

2.1.2 Cost Function

A cost function is a performance measure that needs to be minimized (or maximized) in an optimal control system. The designer selects a performance measure in order to evaluate the performance of a system quantitatively. In this section, several problems are discussed to provide insights for the selection of a performance measure. In selecting which performance measure to use, the designer has to define a mathematical expression which when minimized will give the most desirable performance of the system.

A possible performance measure to minimize the error of the output $x(k)$ from its reference $r(k)$ can have a form of

$$J = \sum_{i=1}^{N} [x_i(k) - r_i(k)]^2. \tag{2.6}$$

Using matrix notation, Eq. (2.6) will have a general form

$$J = [x(k) - r(k)]^T [x(k) - r(k)], \tag{2.7}$$

where $x(k) = [x_1(k), x_2(k), ..., x_N(k)]^T$ is the set of outputs and $r(k) = [r_1(k), r_2(k), ..., r_N(k)]^T$ is the

set of reference points. Equivalently, (2.7) can also be written as

$$J = \|x(k) - r(k)\|^2. \tag{2.8}$$

In general form, a weight factor Q can be added in Eq. (2.7) to get

$$J = [x(k) - r(k)]^T Q[x(k) - r(k)] = \|x(k) - r(k)\|_Q^2. \tag{2.9}$$

In the optimization of MPC, a conventional performance measure or cost function can be written as

$$J = \sum_{i=1}^{H} (\hat{y}(k+i|k) - r(k+i|k))^T Q(\hat{y}(k+i|k) - r(k+i|k)) + (u(k+i-1|k))^T R(u(k+i-1|k) \tag{2.10}$$

where $u(k+i-1|k)$, $i = 1, 2, ..., H$ is the sequence of manipulated variables to be optimized; $\hat{y}(k+i|k)$ is the state prediction generated by the model in (2.3) on the basis of information at discrete time k under the action of control sequence u; Q and R are strictly positive definite symmetric weighing matrices. This cost function is further modified, resulting to different variants of MPC.

2.1.3 Constraints

In a typical control system problem, constraints are always present. In the context of this thesis, constraints are referred to as the limitation or restriction imposed on the states, or control inputs depending upon the physical situation. We can have either equality or inequality constraints. A simple constraint can be written as

$$X_{min} \leq x(k) \leq X_{max}, \tag{2.11}$$

or

$$U_{min} \leq u(k) \leq U_{max} \tag{2.12}$$

In a bounded problem, a control history that satisfies the control constraints during the entire time interval [k_0, k_f] is called an *admissible control*. A state trajectory which satisfies the state variable constraints during the entire time interval [k_0, $k + f$] is called an *admissible trajectory*.

2.2 MPC Requirements

All MPC algorithms have the same basic elements or requirements for its implementation. They will only vary on the chosen options of the designers to modify one of these elements giving rise to different algorithms. The requirements for MPC have all of the requirements discussed in Sec. (2.1) with the

addition of some elements. These elements are

1. Prediction Model
2. Obtaining the control law

2.2.1 Prediction Model

The prediction model is the foundation of all MPC algorithms. Necessary components must be included in the design for obtaining the best possible model. The design must be complete enough to fully capture the process dynamics and must also have the capability to make the necessary calculation for predictions. Additionally, it must also allow theoretic analysis.

The prediction model can have the process model and the disturbance model. The process model is used to calculate the predicted output at future instants $\hat{y}(k+n|k)$. The designer can use different models to represent the outputs and measurable inputs. A disturbance model can also be included in the prediction model. This is used in order to describe the behavior which is not reflected by the process model such as the effect of non-measurable inputs, noise and model errors.

There are several possible form for the process model in a given MPC formulation. One of the process models used in this thesis, particularly in the implementation of a single integrator dynamics of Chapter 4, is the **State Space Model**. It has the following representation:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) \end{aligned} \tag{2.13}$$

The prediction for this model in a defined prediction horizon H, is given by

$$\hat{y}(k+i|k) = C\hat{x}(k+i|k) = C[A^i x(k) + \sum_{j=1}^{i} A^{j-1} Bu(k+i-j|k)] \tag{2.14}$$

where $i = 1, 2, ..., H$.

Another form of process model used in this thesis, particularly in the implementation of a unicycle dynamics of Chapter 5, is the **nonlinear models**. It has the following representation:

$$x(k+1) = f(x(k), u(k)) \tag{2.15}$$

where $f(\cdot)$ is implicitly defined by the originating differential equation that has an equilibrium point at the

origin (i.e. f (0,0)=0). Expanding this model to the future with prediction horizon H,

$$\begin{aligned} x(k+1|k) &= f(x(k|k), u(k|k)) \\ x(k+2|k) &= f(x(k+1|k), u(k+1|k)) \\ \vdots & \qquad\qquad \vdots \\ x(k+H|k) &= f(x(k+H-1|k), u(k+H-1|k)) \end{aligned} \tag{2.16}$$

the general equation of the prediction model can be written as

$$x(k+i+1|k) = f(x(k+i|k), \mathbf{u}(k+i|k)), \quad i \in [0, H-1] \tag{2.17}$$

2.2.2 Disturbance Model

A disturbance model [29] may also be included, together with the process model, in the prediction model. Though the disturbance model is irrelevant in the context of this thesis, it is still a good option to include this topic here. The choice to model the disturbances is as significant as the choice of the process model. The most common model used in MPC is the Controlled Auto-Regressive and Integrated Moving Average (CARIMA). In this model, the disturbances, which is the differences between the measured output and the one calculated by the model, are given by

$$n(k) = \frac{C(z^{-1})e(k)}{D(z^{-1})} \tag{2.18}$$

where $D(z^{-1})$ explicitly includes the integrator $\Delta = 1 - z^{-1}$, e(k) is a zero mean white noise and $C(z^{-1})$ is normally considered to be equal to one. This will later be incorporated with the transfer function process model as a demonstration.

Transfer function process model uses the concept of transfer function in its simplest form

$$A(z^{-1})y(k) = B(z^{-1})u(k) \tag{2.19}$$

with

$$\begin{aligned} A(z^{-1}) &= 1 + a_1 z^{-1} + a_2 z^{-2} + \cdots + a_{na} z^{-na} \\ B(z^{-1}) &= b_1 z^{-1} + b_2 z^{-2} + \cdots + b_{nb} z^{-nb} \end{aligned}$$

In this model, the prediction is given by

$$\hat{y}(k+n|k) = \frac{B(z^{-1})}{A(z^{-1})} u(k+n|k) \tag{2.20}$$

Now, given the Diophantine equation

$$1 = E(z^{-1})D(z^{-1}) + z^{-n}F(z^{-1}) \tag{2.21}$$

and substituting (2.18) to Eq. (2.21) yields

$$n(k) = E(z^{-1})e(k) + z^{-n}F(z^{-1})n(k) \tag{2.22}$$

Since the e(k) is a zero mean variable, the first term in Eq. (2.22) can be neglected. Therefore, the prediction will be

$$\hat{n}(k+n|k) = F(z^{-1})n(k) \tag{2.23}$$

If (2.23) will be integrated with the prediction model of transfer function in Eq. (2.20), the combined prediction can be obtained by

$$\hat{y}(k+n|k) = \frac{B(z^{-1})}{A(z^{-1})}u(k+n|k) + F(z^{-1})n(k) \tag{2.24}$$

taking note that n(k) is the disturbances and can also be expressed as $n(k) = y(k) - \frac{B(z^{-1})}{A(z^{-1})}u(k)$. So, Eq. (2.24) can be rewritten as

$$\hat{y}(k+n|k) = \frac{B(z^{-1})}{A(z^{-1})}u(k+n|k) + F(z^{-1})(y(k) - \frac{B(z^{-1})}{A(z^{-1})}u(k)) \tag{2.25}$$

Rearranging, yields

$$\hat{y}(k+n|k) = F(z^{-1})(y(k) + \frac{B(z^{-1})}{A(z^{-1})}(1 - z^{-n}F(z^{-1}))u(k+n|k) \tag{2.26}$$

Using the Diophantine equation in Eq. (2.21) and making $D(z^{-1}) = A(z^{-1})(1 - z^{-1})$, the following expression is obtained for the n-step ahead predictor

$$\hat{y}(k+n|k) = F(z^{-1})(y(k) + \frac{B(z^{-1})}{A(z^{-1})}(1 - z^{-n}F(z^{-1}))u(k+n|k) \tag{2.27}$$

In the particular case of Auto-Regressive and Integrated Moving Average (ARIMA), the disturbances can be represented as

$$n(k) = \frac{e(k)}{1 - z^{-1}} \tag{2.28}$$

and will have a prediction of

$$\hat{n}(k+n|k) = n(k) \tag{2.29}$$

Other polynomial models of higher order in the denominator of Eq. (2.28) can also be used.

2.2.3 Obtaining the Control Law

In order to obtain the values $u(k+n|k)$ in a fixed horizon, it is necessary to minimize the cost function in MPC. To do this, a model of predicted outputs $\hat{y}(k+n|k)$ is determined as a function of the states and future control signals. Then, this will be substituted in the cost function, obtaining an expression whose minimization leads to the optimal values. If the system is linear and the optimization with quadratic criterion has no constraints, this can be solved directly using analytical solution (See Appendix A). Otherwise, a numerical or iterative method of optimization will be used. After obtaining the set of optimal values for control, apply only the first step in the resulting optimal sequence. In the following sample, the state is measured again. Then, the optimization process is repeated again over the future. This principle is known as the receding horizon.

2.3 Linear Model Predictive Control

This section describes and summarizes the linear MPC which is the foundation of Chapter 4. By now, linear MPC theory is quite a mature control scheme. Recall that the computation and implementation of MPC depend on the model representation. Therefore, linear MPC refers to the use of linear system models in the prediction. Most of the time, it is always associated with the use of state space model. For the context of this thesis, the system is assumed to be described in state space by a linear discrete-time model as described in (2.13). For this model, the prediction model is described in (2.14).

The control objective is usually to steer the state to an equilibrium state. Assume that a full measurement of the state $x(k)$ is available at the current time k. Then for event (x, k) (i.e. for state x at time k), a receding horizon-based strategy is implemented by introducing the following optimization problem written on the following form

$$\underset{x_k^{k+H}, u_k^{k+H-1}}{\text{minimize}} \quad J(x(k), u(k)) \tag{2.30}$$

subject to

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ u(k+i) &\in \mathbb{U}, \quad i \in [0, H-1] \\ x(k+i+1) &\in \mathbb{X}, \quad i \in [0, H-1] \end{aligned} \tag{2.31}$$

where H is the prediction horizon, $x_k^{k+H} = [x(k+1), x(k+2), .., u(k+H)]^T$, $u_k^{k+H-1} = [u(k), u(k+1), .., u(k+H-1)]^T$, and $J(\cdot)$ is the objective function or the cost function. $\mathbb{U}$ and $\mathbb{X}$ are the sets of admissible values for the control and for the state, respectively. In (2.30), it tries to minimize the cost function subjected to different constraints found in (2.31).

Various MPC algorithms propose different cost functions for obtaining the control law. The general aim is that the future output ($\hat{y}$) on the considered prediction horizon must follow a determined reference or setpoint r, and at the same time, the control effort u necessary for doing so should be penalized. The conventional cost function that is used for linear MPC implementation is found in (2.10).

After solving the optimization problem (2.30) subject to the system dynamics in (2.13) and constraints in (2.31) using one of the solvers available, an optimizing value function J^* is obtained, together with an optimal control sequence $u^*(i|k)$, $i = k, k+1, ..., k+H-1$. In a receding horizon policy, only the first control $u^*(k|k)$ is implemented to obtain $x(k+1) = Ax(k) + Bu^*(k|k)$. The rest of the control found in the sequence are discarded, and $x(k+1)$ is now used to update the optimization problem in (2.30). This process is repeated each time using only the first control action to obtain a new initial condition, then shifting the horizon ahead one time step. This is the reason why MPC is also sometimes referred to as Receding Horizon Control (RHC) or Moving Horizon Control (MHC). The purpose of taking new measurements at each time step is to compensate for unmeasured disturbances and model inaccuracy, both of which cause the system output to be different from the one predicted by the model [25].

2.4 Nonlinear Model Predictive Control

The practical interest for Nonlinear Model Predictive Control (NMPC) is driven by the fact that many systems are in general inherently nonlinear, and today's processes need to be operated under tighter performance specifications. At the same time, more and more constraints(i.e. from environmental and safety considerations) need to be satisfied. In these cases, linear models are often inadequate to describe the process dynamics and nonlinear models have to be used. This motivates the use of NMPC.

In NMPC, the system is described or approximated by a discrete-time model as shown in (2.15). The optimization problem is still performed with the optimization problem detailed in Sec. (2.3). The same scheme with the linear MPC still holds to obtain the control law. A popular choice of the cost function is already given in (2.10).

One of the modifications of (2.10) is the addition of terminal cost in the objective function. This will be considered in the implementation. The cost function is now given by [30]

$$\begin{aligned} J = \Omega(\hat{y}(k+H|k)) + \sum_{i=1}^{H} (\hat{y}(k+i|k) - r(k+i|k))^T Q(\hat{y}(k+i|k) - r(k+i|k)) \\ + \sum_{i=1}^{H} (u(k+i-1|k))^T R(u(k+i-1|k) \end{aligned} \tag{2.32}$$

where $\Omega(\hat{y}(k+H|k)) = \hat{y}^T(k+H|k)\ P\ \hat{y}(k+H|k)$ and P is the terminal weight factor. Further modification of this cost function is used in [31] with time-invariant weight matrices, and is evaluated in [32–34] with

time-varying weights to minimize or eliminate the steady-state error. This cost function is used and evaluated as well in Chapter 5.

3

Sparsity

Contents

"Sparse" or "Sparsity" of a vector denotes that some elements of the vector are zero. In other words, it refers to the fact that only very few entries in a vector are non-zero. This appears to be widely used to many research fields including neuroscience, image processing, computer vision, signal processing, and even statistics in the beginning of the 20^{th} century [19]. This chapter discusses the background of regularization and the use of ℓ_1-norm to sparsify a solution.

3.1 The Use of ℓ_p-norm

Consider a matrix $A_1 \in \mathbb{R}^{mxn}$ and a vector $\bar{y} \in \mathbb{R}^m$. The vector $\mathbf{x} \in \mathbb{R}^n$ can be inferred either from noiseless observations of $\bar{y}$

$$\bar{b} = A_1 x \tag{3.1}$$

or from noisy observations

$$\bar{y} = A_1 x + \xi. \tag{3.2}$$

In the low dimensional case, when the number of observations is larger than the number of variables such that $m > n$, as in classical statistical setups, system identification and adaptive control, the desired x is the unique solution to the system of linear equation. In the high dimensional case when $m < n$, the system is under-determined (more unknowns than equations), which implies that the solution is not unique, provided that there is at least one. Assuming a matrix A_1 with full-rank (its columns span the entire $\mathbb{R}^n$) guarantees that a solution can be sought to.

Commonly, a single solution is only opted and the fact that the problem can have many or infinite number of solutions is a major concern. To achieve a more well-defined solution, additional criteria are required . A familiar technique for this approach is through *regularization*. This technique is preferable as it favors a smaller values from the possible solution it has evaluated.

A general optimization problem can be defined by rewriting eq. (3.1) as

$$\begin{aligned} \min_{x} \quad & J(x) \\ \text{s.t.} \quad & \bar{y} = A_1 x \end{aligned} \tag{3.3}$$

A well-known choice of $J(x)$ is the squared Euclidean norm. Using Lagrange multipliers and introducing the concept of *Lagrangian*, the optimization problem in (3.3), can be written in *Lagrangian* form as

$$\mathcal{L}(x) \quad = \quad \|x\|^2 \quad + \quad \lambda^T (A_1 x - \bar{y}), \tag{3.4}$$

where λ is the Lagrange multipliers for the constraint set. To find the unique solution x^*, take the

derivative of $\mathcal{L}(x)$ with respect to x and obtain

$$\frac{\partial \mathcal{L}(x)}{\partial x} = 2x + A_1^T \lambda \tag{3.5}$$

By equating Eq. (3.5) to zero, the solution is then

$$x^* = -\frac{1}{2} A_1^T \lambda \tag{3.6}$$

Substituting x in Eq. (3.1) by the solution above yields

$$\bar{y} = A_1 x^* = -\frac{1}{2} A_1 A_1^T \lambda \quad \Rightarrow \quad \lambda = -2(A_1 A_1^T)^{-1} \bar{y} \tag{3.7}$$

Plugging the value of λ from Eq. (3.7) to Eq. (3.6) gives the familiar closed-form pseudo-inverse solution

$$x^* = -\frac{1}{2} A_1^T \lambda = A_1^T (A_1 A_1^T)^{-1} \bar{y} = A_1^+ \bar{y} \tag{3.8}$$

Recall that matrix A_1 was assumed to be full-rank, thus the matrix $A_1 A_1^T$ is positive-definite and invertible. The use of ℓ_2 is common in many areas of engineering due to its simplicity as obviously seen by the above closed-form and unique solution. This, however, does not imply that ℓ_2 is the best choice for regularization [35, 36]

Before exploring for other options, recall first the definition of convexity for sets and for functions [37].

Definition 3.1.1. (Convex sets) A set $\mathbb{C}$ is *convex* if $\forall \mathbf{x_1}, \mathbf{x_2} \in \mathbb{C}$ and $\forall \alpha \in [0, 1]$, the convex combination or line segment $\mathbf{x} = \alpha \mathbf{x_1} + (1 - \alpha)\mathbf{x_2}$ lies also in the set $\mathbb{C}$.

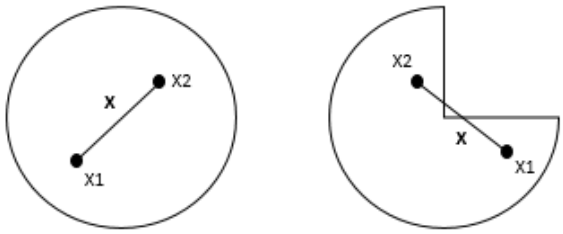

Figure 3.1: Simple Convex and Nonconvex Sets. Left. The circle set is convex. Right. The pie shaped set is not convex, since the line connecting the two points is not contained in the set.

In order for the optimization problem to be confirmed as a convex, another convexity requirement will be added. This requirement is given by another definition.

Definition 3.1.2. (Convex function) A function $f(x)$: $\mathbb{C} \rightarrow \mathbb{R}$ is *convex* if $\forall x_1, x_2 \in \mathbb{C}$, and $\forall \alpha \in [0, 1]$, thus

$$f(\alpha x_1 + (1 - \alpha)x_2) \leq \alpha f(x_1) + (1 - \alpha)f(x_2). \tag{3.9}$$

Geometrically, the inequality in (3.9) means that the line connecting between two points on the graph of a function lies above or on the function's plot (also called the *epigraph* of a function) as shown in Fig. (3.2). To examine it further, the left side of the inequality is the result from evaluating the convex combination of x_1 and x_2 to the function f. The reflection of the obtained point to the line connecting the two points is the right side of the inequality.

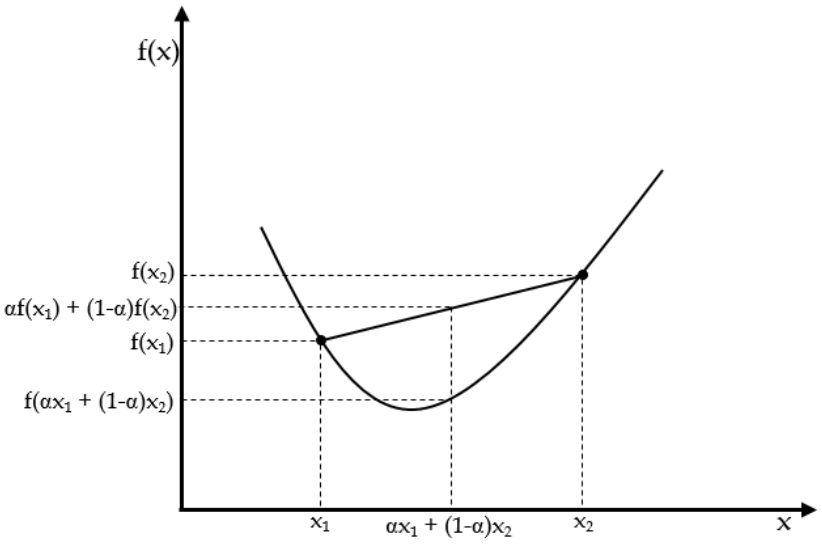

Figure 3.2: Convex Function Graph. The line connecting any two points on the graph of a function lies above or on the graph.

The convexity of the squared Euclidean norm is another reason why it is a well-known regularization. It is strictly convex and it satisfies all of the above definitions. With this fact, there are many other choices for the functions which are convex. Some of which may not lead to a closed-form solution but it is still considerable as long as it guarantees the convergence to the global minimum of the problem. Interesting choices for convex functions are the so-called ℓ_p-norms defined as

$$\|\mathbf{x}\|^p = (\sum_{i=1}^{n} |x_i|^p)^{1/p}. \tag{3.10}$$

The succeeding discussion will only give a strong emphasis on ℓ_1-norm due to its ability to sparsify the solution. This norm has the property of producing many elements with zero values. Other ℓ_p will be

covered in Sec. (4.3) as obstacle constraints.

The ℓ_1-norm, sometimes known as the Least Absolute Deviations (LAD) or Least Absolute Errors (LAE), is basically defined as the summation of absolute values of a vector. That is,

$$\|\mathbf{x}\|^1 = \sum_{i=1}^{n} |x_i| \,. \tag{3.11}$$

It has been typically used as a regularization function beyond the least-square context, leading to the problems of the form

$$\min_{x \in \mathbb{R}^p} \quad f(x) + \rho \|x\|_1. \tag{3.12}$$

where $f\colon \mathbb{R}^p \to \mathbb{R}$ is a cost function. By studying the geometry of the ℓ_1-ball, the sparsity-inducing effect of the ℓ_1-norm can be shown. If the ℓ_1-norm function is drawn in the form $\|x\|^1 \leq \mu$ where μ is a constant, the graph looks like as shown in Fig. (3.3).

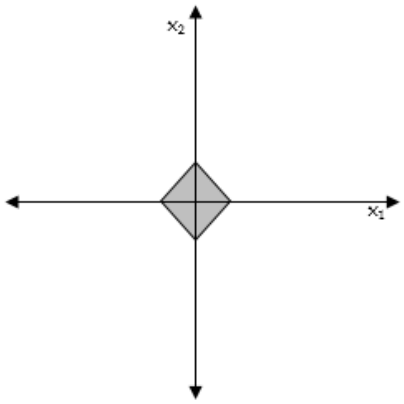

Figure 3.3: Geometric Interpretation of ℓ_1-norm ball

The shape of the graph is like a tilted square or a square with a 45^0 rotation. In a three-dimensional space, this can be interpreted as an octahedron. It is noticeable in the graph that only the tips can be considered as sparse. Meaning to say, either the x_1 or x_2 component of a point is equal to zero. Assuming that a line is drawn in the form of $x_2 = mx_1 + b$, where m is the slope and b is the y-intercept, the sparse solution of the line can be found by growing the ℓ_1-norm ball to touch this line as shown in Fig. (3.4).

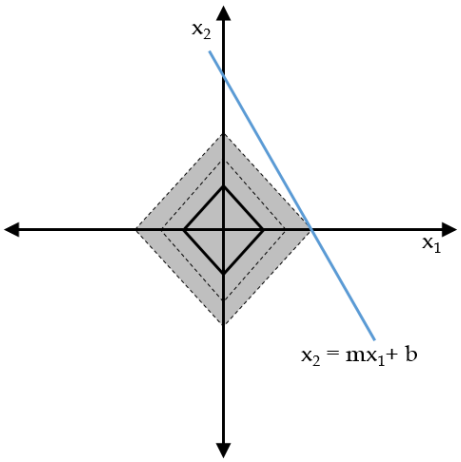

Figure 3.4: Sparse solution of the intercept of a line with the ℓ_1-norm ball.

By growing the ball connotes that the constant μ is increased. It is also expected that the touch point is most likely the tip of the ℓ_1-norm ball. Since the tip of the ball is a sparse point, the solution defined by the touch point is also a sparse solution. In the touch point, the constant μ is said to be the smallest ℓ_1-norm within all the possible solutions. So by putting the ℓ_1-norm in the cost function will most likely keep looking for a solution with a smaller μ (that is, at the ”sparse” tip of the ℓ_1-norm). A real cost function case is typically shrinking the ball to find a touch point, not making it bigger from the origin.

4

Linear Sparse Predictive Control

Contents

This chapter presents the development process and the results of the controller for linear model formulation. It begins with describing the vehicle model (for this chapter, the model used is a single integrator), and progressively develops until the design process is reached. After which, the control strategy developed will be called sparse predictive control or sparse controller, because of the type of the cost considered.

After the design is fully covered, testing and evaluations of this controller will be provided. A simulation setup is provided with the final aim to evaluate the performance of the controller and show the advantages of implementing sparsity in the cost function.

4.1 Single Integrator

The plant model is one of the main requirements of any MPC implementation. The controller optimizer uses this model to make predictions of the plant future behavior, which is defined as the prediction model. In doing so, the optimal control input can be determined and applied accordingly. There are many different models to describe the dynamics of the vehicles. This section describes the simplest vehicle model in a plane, called a single integrator, which is used as a vehicle model in this chapter.

In a single integrator model, the motion of the vehicle is given by

$$\dot{x} = u \tag{4.1}$$

where $\dot{x} = [x_1, x_2]^T$ are the states and $u = [u_1, u_2]^T$ are the control vectors. The discretized equivalent for this can be defined as

$$\begin{aligned} x_1(k+1) &= x_1(k) + T_s u_1(k) \\ x_2(k+1) &= x_2(k) + T_s u_2(k) \end{aligned} \tag{4.2}$$

where T_s is the sampling time. Thus, the state-space representation of this model can be written as

$$x(k+1) = \begin{bmatrix} x_1(k+1) \\ x_2(k+2) \end{bmatrix}, \quad A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} T_s & 0 \\ 0 & T_s \end{bmatrix}, \quad C = A$$

4.2 Control Design

This section will incorporate the MPC described in Chap. 2 to sparsity. In most cases the actual model is represented as an approximation to a linear model but nonlinear models (which is considered in the following chapter) can also be used in an MPC formulation. This chapter considers a linear plant

model in its discrete-time state-space representation

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) \end{aligned} \tag{4.3}$$

The prediction for this model in a defined prediction horizon H, is given by

$$\hat{Y} = WU + \Gamma X \tag{4.4}$$

where the matrices W, Γ, X, $\hat{Y}$, and U are defined accordingly in Appendix A.

In the MPC algorithm, there are various ways to write the cost function for obtaining the control law (refer to Appendix A for the analytical solution of control law). The general purpose of having the function is that the error between the future output and the given set point or reference is minimized on the considered prediction horizon, H. At the same time, the control input must also be penalized. Considering only the system dynamics, a general optimization problem will have the form

$$\begin{array}{lll} \min\limits_{u} & \sum\limits_{i=1}^{H} \|y(k+i|k) - r(k+i|k)\|_Q^2 \; + \; \|u(k+i-1|k)\|_R^p & \\ \text{s.t.} & x(k+1) = Ax(k) + Bu(k) & \\ & y(k) = Cx(k) + Du(k) & \end{array} \tag{4.5}$$

where p is either the ℓ_1 or ℓ_2 norms. For a quadratic or ℓ_2-norm formulation, the cost function in (4.5) can be expressed as in (2.10). The controller utilizing a cost function in (4.5) is henceforth called a quadratic controller.

For the case of this thesis, constraints are always involved. It is either a constraint in the system or a constraint for obstacle avoidance problem or both. The constraints are expressed as linear and nonlinear inequalities. The nonlinear constraints are detailed in Sec. (4.3) which are intended for obstacle concern. Introducing the vector $Z_k = [\hat{Y}\, U]^T$ that contains the set of future outputs and inputs, the mathematical optimization problem in (4.5) can be modified as

$$\begin{array}{lll} \min\limits_{Z_k} & \|M_y Z_k - \bar{Y}\|_Q^2 \; + \; \|M_u Z_k\|_R^1 & \\ \text{s.t.} & [I \quad -W] Z_k = \Gamma X & \\ & \begin{bmatrix} I \\ -I \end{bmatrix} Z_k \leq \begin{bmatrix} Z_{max} \\ -Z_{min} \end{bmatrix} & \\ & h(M_y Z_k) \leq 0 & \end{array} \tag{4.6}$$

where $\bar{Y}$ is a set of reference points defined in Appendix A, and M_y and M_u are selection matrices which extract from the vector Z_k the inputs and outputs, respectively. The last two constraints represent the bound for control input and output constraints, and the nonlinear function for obstacle constraints, respectively.

4.3 Obstacle Constraints

This section will present the equations used to form the obstacles which are also used as constraint parameters in the controller. The encoding of obstacle constraints assumes that the shape of every obstacle follows that of the shape of ℓ_p-norms.

4.3.1 The ℓ_p-norm balls

Given the background of some of the norms in Chap. 3, its equations are explored further and some information is added which will guide through the implementation of the obstacles constraints. A vector norm $\|x\|$ can be thought of as the length or magnitude of a vector $x \in \mathbb{R}^n$ [38]. It is any mapping from $\mathbb{R}^n$ to $\mathbb{R}$ that follows these three properties:

$$\|x\| > 0 \quad if x \neq 0$$

$$\|ax\| = |\alpha| \, \|x\|$$

$$\|x + y\| = \|x\| + \|y\| \qquad \text{(triangle inequality)}$$

where vectors x, $y \in \mathbb{R}^n$ and scalar $a \in \mathbb{R}$. Some of the vector norms for the vector $\mathbf{x} = (x_1, x_2, ..., x_n)^T \in \mathbb{R}^n$ are

$$\|x\|^1 = \sum_{i=1}^{n} |x_i| \tag{4.7}$$

$$\|x\|^2 = (\sum_{i=1}^{n} x_i^2)^{1/2} \tag{4.8}$$

$$\|x\|^\infty = \max_{1 \leq j \leq n} |x_i| \tag{4.9}$$

The Eq. (4.7) is for the ℓ_1-norm. As mentioned in Chap. 3, it is the sum of absolute values of the elements in vector, x. It can be expanded as $\|x\|^1 = |x_1| + |x_2| + ... + |x_n|$. The second equation, Eq. (4.8) is the most popular example of norm and it is called the Euclidean or ℓ_2-norm. It is defined in another way as $\|x\|^2 = (x_1^2 + x_2^2 + ... + x_n^2)^{1/2}$. The last equation, Eq. (4.9) is known as the infinity norm or ℓ_∞-norm. It measures the maximal component of a vector and takes the form $\|x\|^\infty = max\{|x_1|, |x_2|, ..., |x_n|\}$. The geometrical interpretations in 2-D space of these different norms can be shown in Fig. (4.1).

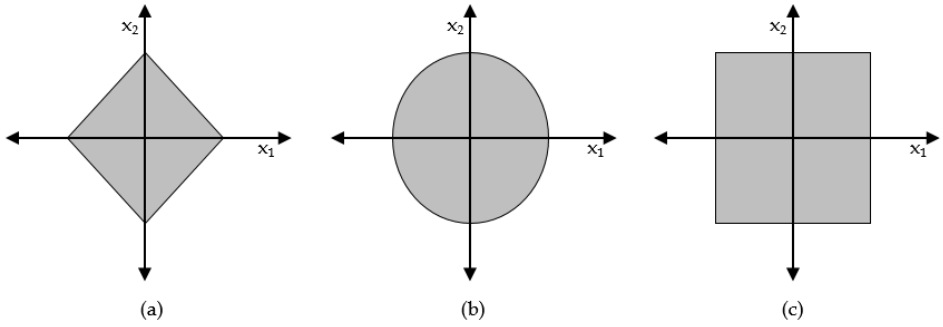

Figure 4.1: Different Norms Geometric Interpretations in 2D Space. The figures in (a), (b), and (c) correspond to the graphs of ℓ_1, ℓ_2, and ℓ_∞-norms, respectively.

As a general case, all of these are defined to formulate the constraints: 1) the center point C(c_1,c_2), 2) the vector x that contains all the possible points in the figure boundary, and 3) a constant d_{safe} which is either the length from the tip to the center for ℓ_1-norm, the radius for ℓ_2-norm, or the length of from the side to the center for ℓ_∞-norm. The equation for each of the norms can then be formulated as

$$\|\mathbf{x} - C\|^1 = |x_1 - c_1| + |x_2 - c_2| = d_{safe} \tag{4.10}$$

$$\|\mathbf{x} - C\|^2 = (x_1 - c_1)^2 + (x_2 - c_2)^2 = d_{safe}^2 \tag{4.11}$$

$$\|\mathbf{x} - C\|^\infty = max\{|x_1 - c_1|, |x_2 - c_2|\} = d_{safe} \tag{4.12}$$

In MATLAB, the norm ℓ_p-norm can be calculated by the function *norm*, which takes the value of $0 \leq p < \infty$ as an optional argument. For the ℓ_∞-norm, it will take the Inf command for the p parameter, which is the optional argument in the function. If p is not defined, the default value of it is the Euclidean norm.

4.3.2 Safe Distance Constraint

The vehicle representation is taken into account in formulating the constraints for a multi-vehicle scenario or it can be interpreted as a moving obstacle. For collision avoidance, the vehicles are required to stay at least a distance apart from each other. In this thesis, the vehicle is represented by a circular shape with center at C(c_1,c_2) and can be imagined as a rigid body that moves in the plane. The constraint can be formulated as

$$\sqrt{(x - c_1)^2 + (y - c_2)^2} = d_{safe} \tag{4.13}$$

where d_{safe} is the allowable distance between vehicles.

4.4 $fmincon$ Solver

There are several available optimization methods, and solvers are normally designed in order to find the desired solution in an optimization problem using one or the combination of these methods. In this section, the $fmincon$ function solver, which is used to solve the optimization problem in (4.6), will be explored.

The $fmincon$, a function in MATLAB, is a powerful non-linear programming solver. It is used to find the minimum value of the constrained nonlinear function specified by [39]:

$$\begin{array}{lrcl} \min\limits_{x} & f(x) & & \\ \text{s.t.} & c(x) & \leq & 0 \\ & ceq(x) & = & 0 \\ & A(x) & \leq & b \\ & Aeq(x) & = & beq \\ & lb \leq & x \leq & ub \end{array} \tag{4.14}$$

where x is the controlled variables, which can be either scalars or vectors, and $f(x)$ is the cost or objective function to be optimized which returns a scalar. The lower and upper bounds of x are lb and ub, respectively. $A(x)$ and $Aeq(x)$ are the linear constraints being bounded by vectors b and beq, respectively. The $c(x)$ and $ceq(x)$ are nonlinear constrained functions which return vectors.

To demonstrate the use of this function, given a bound-constrained of control input u such that -1 $\leq$ $u \leq$ 1, the code using *fmincon* can be structured as

Listing 4.1: Sample $fmincon$ implementation

```
%% Optimization with constraint (using fmincon)
A = []; % in linear inequalities A*x ? b
b = []; % in linear inequalities A*x ? b
Aeq = []; % in linear equalities Aeq*x = beq
beq = [];% in linear equalities Aeq*x = beq
lb = -1*ones(1,H); % lower bound in lb ? x ? ub
ub = 1*ones(1,H); % upper bound in lb ? x ? ub
nonlcon = []; % contains  nonlinear inequalities c(x) or equalities ceq(x)
%fmincon function
u = fmincon(@(u) (norm(W*u(:) + F)^2 + rho* norm(u(:))^2),u0,A,b,Aeq,beq,lb,...
    ub,nonlcon,options);
```

where W and F are the parameters defined in Appendix A, $u(:)$ indicates a set of control inputs and u0 is the set of initial points of optimization with size depending on the prediction horizon, H.

As can be seen in the code, the syntax of $fmincon$ is

$$x = fmincon(fun, x0, A, b, Aeq, beq, lb, ub, @nonlcon, options)$$

where fun is the objective function in the optimization problem. The nonlinear inequalities $c(x)$ or equalities $ceq(x)$ is defined in $nonlcon$. The $nonlcon$ is a separate file or function handle that solves for the optimality of the system subjected to the defined nonlinear constraints which can be coded as follows

Listing 4.2: Sample $nonlcon$ function

```
function [c,ceq] = nonlcon(x)
    c = x^2 % Compute nonlinear inequalities at x.
    ceq = x^3 % Compute nonlinear equalities at x.
end
```

There are five different algorithms in the $fmincon$ options. These are

- 'interior-point' (default)
- 'trust-region-reflective'
- 'sqp'
- 'sqp-legacy'
- 'active-set'

The $interior-point$ and $trust-region-reflective$ are large-scale algorithms, whereas the other three algorithms are medium-scale algorithms. The large-scale algorithms use linear algebra that does not need full matrices to operate. This may be done internally by storing sparse matrices, and by using sparse linear algebra for computations whenever possible. Furthermore, the internal algorithms either preserve sparsity, such as a sparse or do not generate matrices. Also, the users do not need to specify any sparse matrices to use these methods. On the other hand, medium-scale algorithms internally create full matrices and use dense linear algebra. If a problem is sufficiently large, full matrices take up a significant amount of memory, and the dense linear algebra may require a long time to execute. Choosing these algorithms means the users want to access extra functionality, such as additional constraint types, or possibly for better performance [40].

4.5 Selection of Parameters

This section will present how the parameters can be chosen to guarantee a reasonable results and to make sure that the controller performs accordingly. As previously mentioned, the $fmincon$ function in MATLAB is used to solve the optimization problem. The vehicle starts from somewhere in the southwest corner to the reference point or goal where the vehicle is tasked to reach and is located at the northeast corner of the simulation area or environment as shown in Fig. (4.2). The output variables are represented as $[x, y]^T$, which is equivalent to the states $[x_1, x_2]^T$.

The tuning process is not always straightforward, and some assumptions can be deduced to favor one from the other such as the processing speed and navigation path. It will be discussed here how the relevant parameters will be chosen.

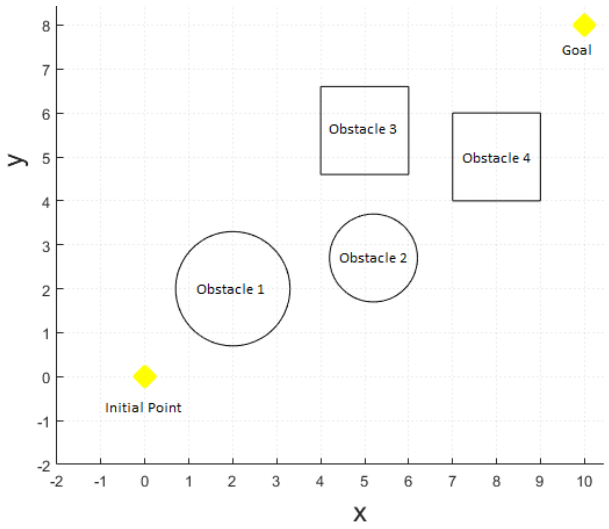

Figure 4.2: The Defined Environment

4.5.1 Selection I: Starting Point

The fixed parameters used throughout the trial process are shown in Table 4.1.

Parameter	Value
H	10
T_s	0.1s
Q	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
R	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
end point	$\begin{bmatrix} 10 \\ 8 \end{bmatrix}$

Table 4.1: Fixed Parameters for Validation I

Different initial positions of the vehicles are considered: $a)$ $(x, y) = (0, 0)$, $b)$ $(x, y) = (1, -1)$, $c)$ $(x, y) = (-1, 2)$, and $d)$ $(x, y) = (0.5, 1)$. The results are shown in Fig. (4.3). Note that the trajectory of the sparse controller is adjusted so it will look like settling in the goal perfectly. In real simulation, the trajectory produced by sparse controller does not converge in the goal point (with minimal gap only). This will be addressed in the following chapter. As seen in the figure, the green dotted line refers to the trajectory or path taken by the vehicle from the initial point until reaching the goal. These paths are discretized as the result of running the controller in a receding horizon strategy.

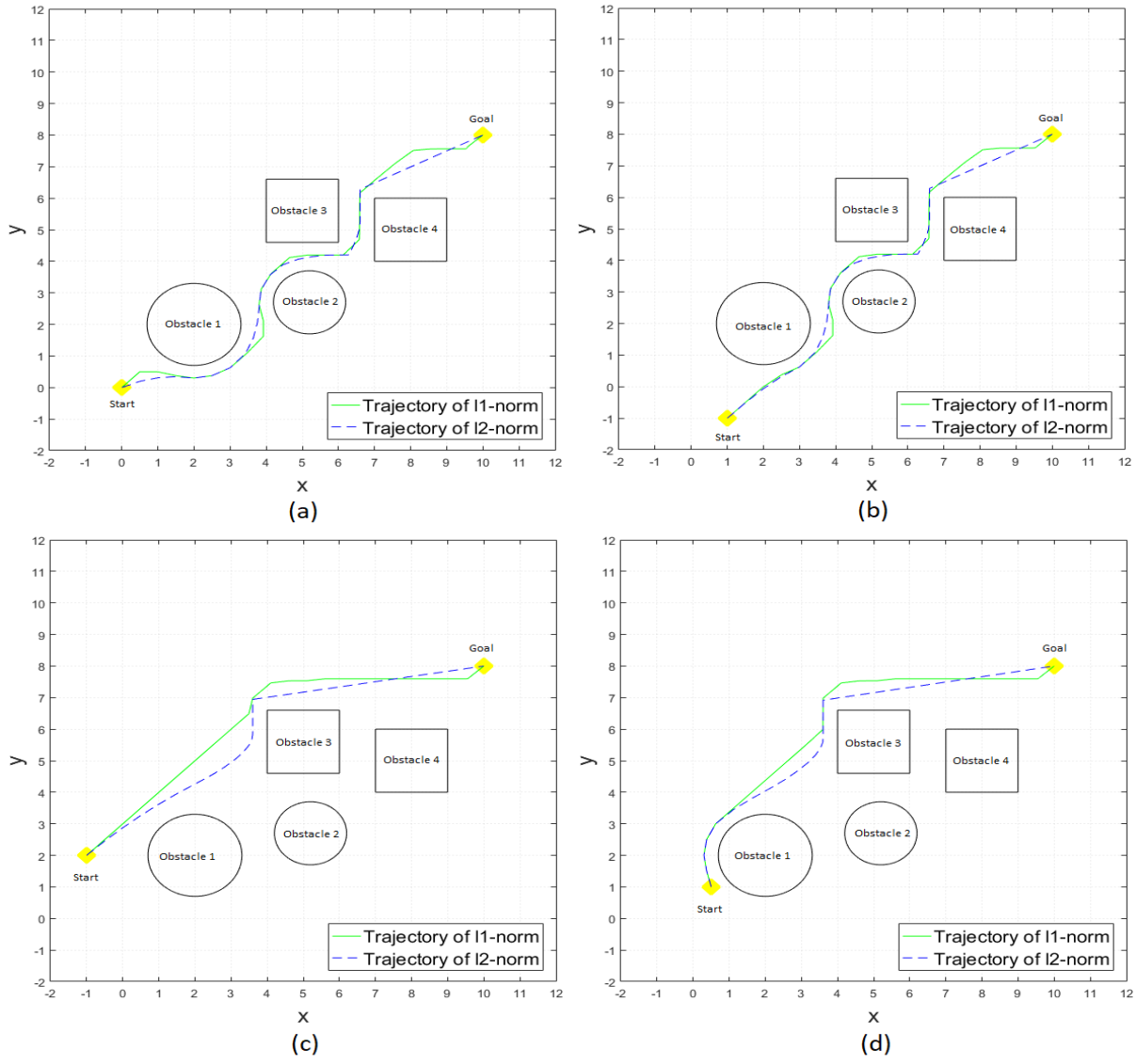

Figure 4.3: Validation 1 Results. a) result obtained with initial point at (0,0), b) result obtained with initial point at (1,-1), c) result obtained with initial point at (-1,2), and d) result obtained with initial point at (0.5,1).

This subsection will try to evaluate if the controller gives the desired path planning regardless of the initial position. It is conducted to see the limitations of the proximity of the initial point of the vehicle to

one of the obstacles, if there is any, given the defined conditions and parameters. Choices are made to cover all the possibilities for the initial point(directly away from the obstacle and near to the obstacle). As can be observed, generated path taken varies with the initial point. Furthermore, there are no limitations to the initial point since the vehicle dynamics used does not impose limits at the wheel's velocities, meaning, the vehicle can turn in any way possible.

4.5.2 Selection II: Weight Factors Q and R

The fixed parameters used throughout the experiment are shown in Table (4.2).

Parameter	Value
H	10
T_s	0.1s
initial point	$\begin{bmatrix}0\\0\end{bmatrix}$
end point	$\begin{bmatrix}10\\8\end{bmatrix}$

Table 4.2: Fixed Parameters for Validation II

To assess the importance of the weight factors in the overall process, different weight factors Q and R of the controller are considered: $a)$ Q = $\begin{bmatrix}5 & 0\\0 & 0.5\end{bmatrix}$ R = $\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$, $b)$ Q = $\begin{bmatrix}0.5 & 0\\0 & 5\end{bmatrix}$ R = $\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$, $c)$ Q = $\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$ R = $\begin{bmatrix}5 & 0\\0 & 0.5\end{bmatrix}$, and $d)$ Q= $\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}$ R = $\begin{bmatrix}0.5 & 0\\0 & 5\end{bmatrix}$.

The results are show in Fig. (4.4). Again, the same case as before, the generated paths are discretized as the result of running the controller in a receding horizon strategy. The demonstration is similar to Fig. (4.3a) but this time the weight factors are varied. As it can be observed, the results are not exactly the same anymore. This is because a penalty is imposed to one of the outputs or inputs that make the calculations prefer one direction over the other. For Fig. (4.4a) and Fig. (4.4b), the deviations of the outputs from the goal point are penalized while the inputs remain unchanged. On the other hand, the inputs are penalized while the deviations remain unchanged for Fig. (4.4c) and Fig. (4.4d).

In Fig. (4.4a), the weight factor Q is structured such as the deviation in x-position is subjected to higher value over the y-position while nothing is done with the weight factor R in the control input cost. As a result of this, it is noticeable that the path taken by the vehicle leans more toward the x-coordinate. In the other case which can be seen in Fig. (4.4b), the deviation in y-position is the one being imposed with a higher value. Thus, the resulting trajectory is different from the last case. This time, the y-coordinate is more preferred to follow. In Fig. (4.4c), the value of control input u_1 is penalized more than the control input u_2 while nothing is done to the output cost. As a result, the path taken by the vehicle leans toward the y-coordinate. In Fig. (4.4d), u_2 is penalized more than the u_1. This yields to a path which is more toward the x-coordinate and that the movement along y-coordinate is limited.

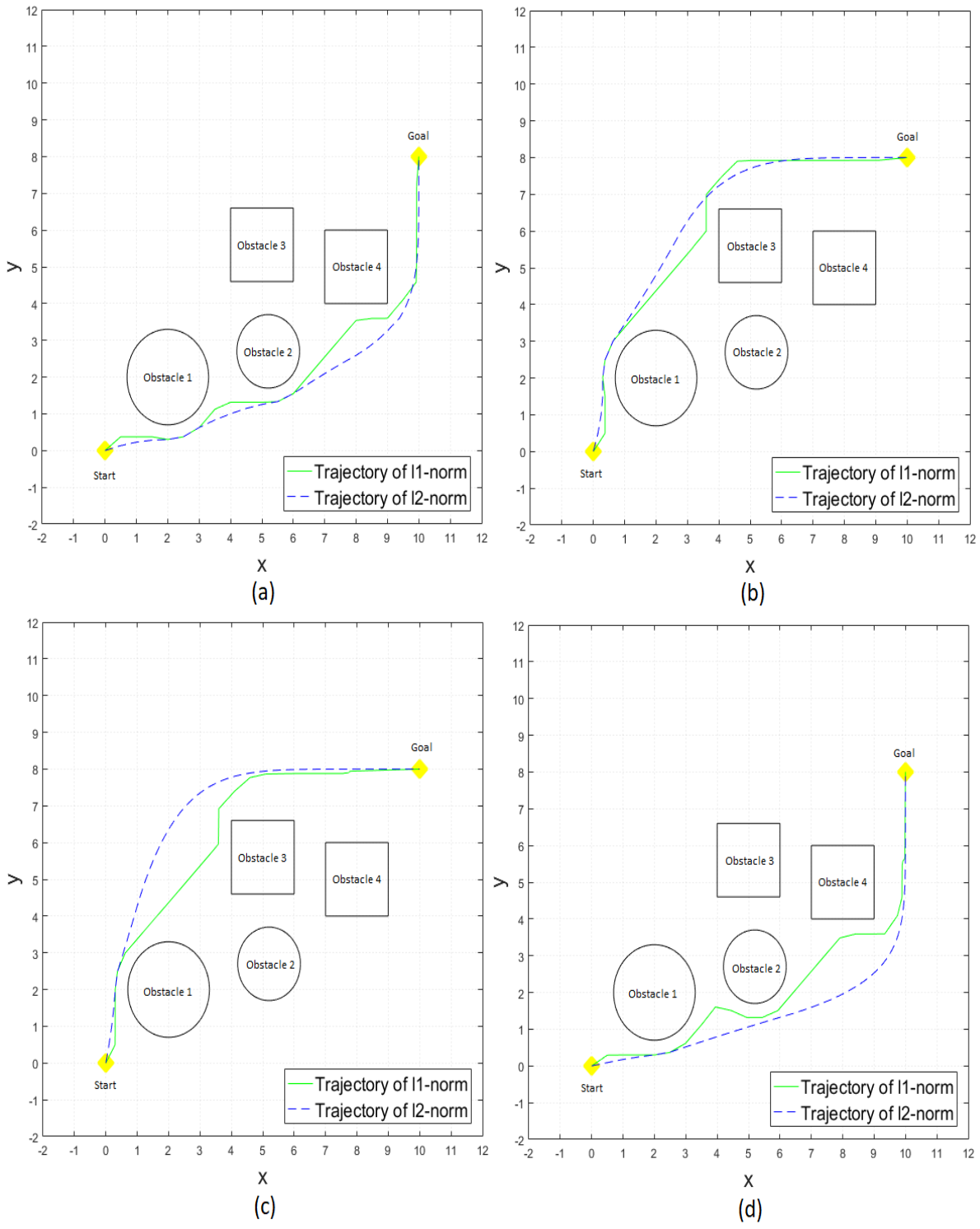

Figure 4.4: Validation 2 Results. Results obtained with weight factors $a)$ Q = $\begin{bmatrix} 5 & 0 \\ 0 & 0.5 \end{bmatrix}$ R = $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $b)$ Q = $\begin{bmatrix} 0.5 & 0 \\ 0 & 5 \end{bmatrix}$ R = $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $c)$ Q = $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ R = $\begin{bmatrix} 5 & 0 \\ 0 & 0.5 \end{bmatrix}$, and $d)$ Q= $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ R = $\begin{bmatrix} 0.5 & 0 \\ 0 & 5 \end{bmatrix}$.

It can be concluded that the tuning of weight factor Q means giving preference to the direction where the vehicle can conveniently take. On the other hand, tuning the weight factor R connotes that the movement of the vehicle will be restrained. The matrix R is sometimes called move suppression factor, since increasing it penalizing the value of control effort heavily. The tuning of these parameters will be based mostly on the designer, the corresponding control dynamics, and the defined environment.

4.5.3 Selection III: Prediction Horizon H

The fixed parameters used throughout the experiment are shown in Table (4.3).

Parameter	Value
T_s	0.1s
Q	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
R	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$
initial point	$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$
end point	$\begin{bmatrix} 10 \\ 8 \end{bmatrix}$

Table 4.3: Fixed Parameters for Validation III

To analyze the effect of the prediction horizon H, different horizon lengths are considered and its corresponding computation time is recorded. The computation time refers to the duration to solve the optimization problem starting from the initial point until reaching the target point. The stopwatch timer (tic and toc), a built-in function in MATLAB, is used to record the time of the execution of the program. The tic function is placed right before the start of the loop function that solves for the optimization problem in a receding horizon strategy. The toc function is placed just after the program exits the loop and after the the goal is reached. The time obtained also depends on the specifications of the computer used. After recording, the plot of H vs. total computational time is shown in Fig. (4.5).

Notice that as the horizon length increases, the total elapsed time for the entire process also increases. Tuning this parameter does not affect other parameters for the case of the model used. For wheeled vehicles (models involving orientation), horizon lengths can affect how the vehicles traverse. This will be discussed in the following chapter.

It is recommended to keep H smaller for computational reasons, but it must be sufficiently large for the feasibility of the optimization problem.

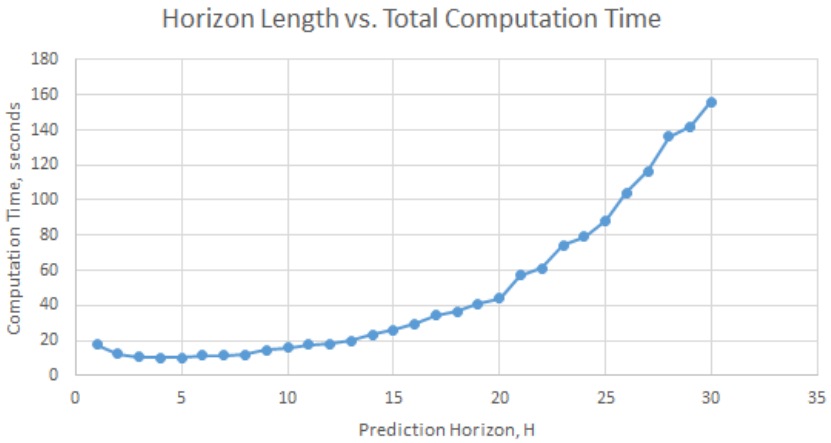

Figure 4.5: Validation 3 Result. Plot of horizon length vs. elapsed time in seconds

4.5.4 Selection IV: Sampling Time

Choosing this parameter is an imperative process. There are many to consider in simulations to make it more realistic. Depending on the system dynamics used, the sampling time T_s must be tuned for it to have a smooth trajectory to represent a vehicle path that looks like avoiding the obstacles. Choosing a wrong one will most likely have a result as shown in Fig. (4.6). This result is obtained by having a sampling time T_s = 0.6s and with parameters similar to Table (4.1).

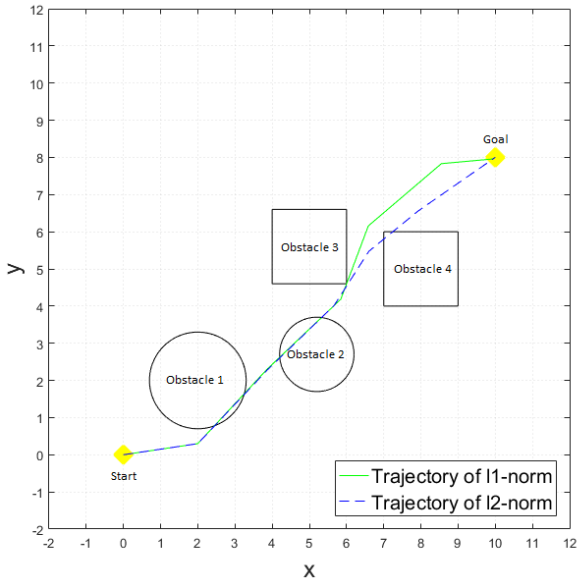

Figure 4.6: Sample trajectory with incorrect sampling time T_s

4.6 Simulation Results

This section will present the results from the simulation conducted. The results are taken after steering the vehicle from a defined starting point to a target or the goal point with the objective to avoid certain fixed obstacles in a defined environment as shown in fig. (4.2).

This will evaluate the suitability of the sparse predictive control, with the single integrator as the vehicle model, to an environment involving obstacles. Parameters are chosen with basis to the selection of parameters process as presented in the previous section.

4.6.1 Quadratic vs. Sparse Control Performance

This part of the section will compare the performance of the quadratic and sparse controllers. The fixed parameters used in both evaluations are shown in Table (4.4). These parameters are chosen to comply and provide a desirable results, enough to have a better comparison between the two controllers.

Parameter	Value
H	10
T_s	0.1s
Q	$\begin{bmatrix} 20 & 0 \\ 0 & 1 \end{bmatrix}$
R	$\begin{bmatrix} 10 & 0 \\ 0 & 1 \end{bmatrix}$
initial point	$\begin{bmatrix} 1 \\ -1 \end{bmatrix}$
end point	$\begin{bmatrix} 10 \\ 8 \end{bmatrix}$

Table 4.4: Fixed Parameters for the evaluation

The control inputs are bounded by $-5 \leq u_i \leq 5$ in every iteration. Furthermore, to make the movement of the vehicle more realistic, the position is constrained by

$$\begin{bmatrix} x_{k-1} - 2 \\ y_{k-1} - 2 \end{bmatrix} \leq \begin{bmatrix} x_k \\ y_k \end{bmatrix} \leq \begin{bmatrix} x_{k-1} + 2 \\ y_{k-1} + 2 \end{bmatrix}$$

Results for the control inputs of both controllers are presented in Fig. (4.7) while the results for the outputs are shown in Fig. (4.8).

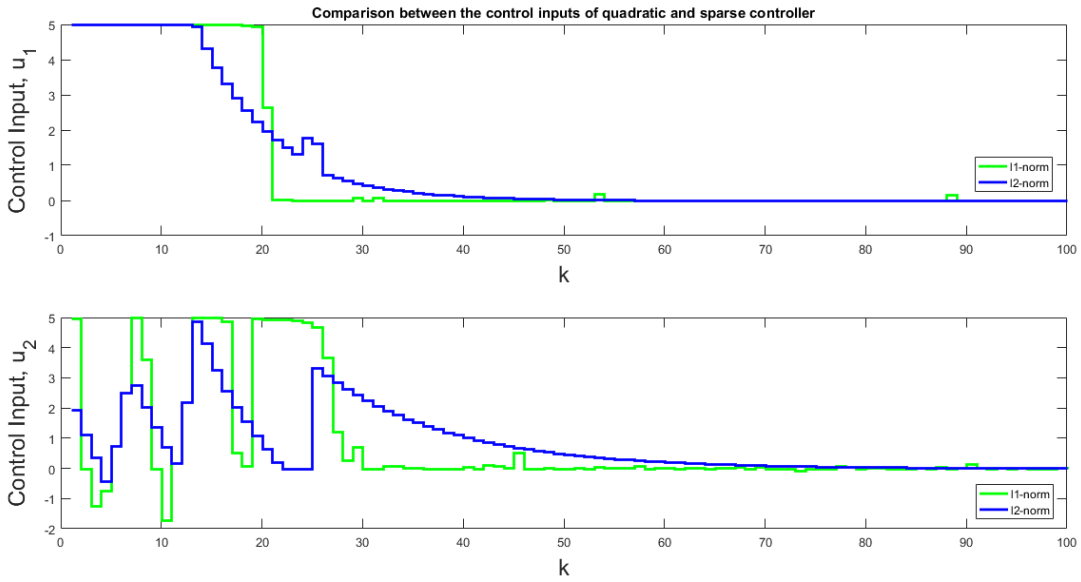

Figure 4.7: Control inputs plots for the quadratic and sparse controllers

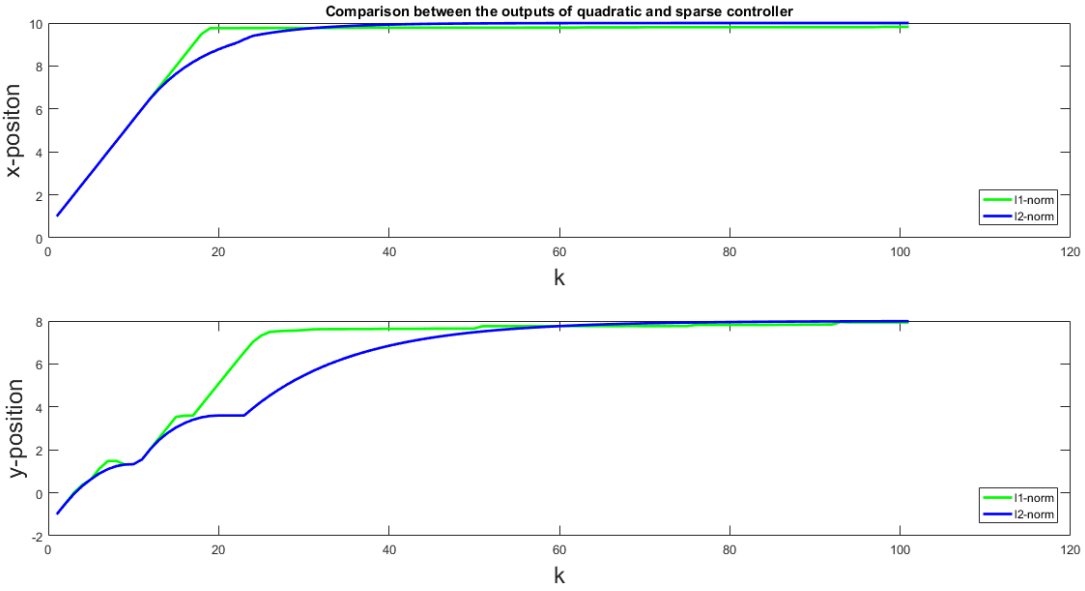

Figure 4.8: Position plots for the quadratic and sparse controllers

In both figures, the green line is the trajectory or the line resulted from using the sparse controller. The blue line is the trajectory obtained from using the quadratic controller, which is the typical implementation in many MPC cost function variants. The difference between the two controllers can be shown in different trajectories produced. In sparsity, it is expected that the control set will contain few non-zero elements. Even if it is applied in a receding horizon method, the characteristic of sparsity will still be visible as a whole. As shown in Fig. (4.7), the control inputs' trajectory of sparse controller tends to immediately

cut the control as opposed to a decaying trend of a quadratic controller at a later time, k. This aspect of sparse controller is more desirable than the quadratic controller since it minimizes more the control effort, thus yielding a better fuel performance if it relates to vehicles. However, this difference is not so evident if one looks into the position plot in Fig. (4.8). But by looking into the y-position, one can justify that at some point, the sparse-controlled vehicle will accelerate more than the quadratic-controlled one.

Solving for the total cost of the sparse controller in the presented example, the following result is provided:

$$J_{\ell_1} = \sum_{k=0}^{N=100} (\hat{y}_{k+i} - r_i)^T Q(\hat{y}_{k+i} - r_i) + IRu_{k+i-1} \quad = 9.54\mathbf{x}10^4 \tag{4.15}$$

where I is a $1\text{x}2H$ matrix consisting of ones-signed elements to obtain only the absolute values, and i = $1, ..., H$. Furthermore, the total cost for the quadratic controller is calculated as

$$J_{\ell_2} = \sum_{k=0}^{N=100} (\hat{y}_{k+i} - r_k)^T Q(\hat{y}_{k+i} - r_k) + (u_k)^T R(u_k) \quad = 1.10\mathbf{x}10^5 \tag{4.16}$$

Comparing the result, notice that the quadratic controller is more costly than the sparse controller by a substantial percentage or margin of

$$\%difference = \frac{J_{\ell_2} - J_{\ell_1}}{J_{\ell_1}} \times 100 = 15.7\%$$

4.6.2 Different Set of Obstacles

This subsection evaluates both controllers in an environment with the ℓ_1-norm ball as obstacle. For this demonstration, the obstacle 4 in (4.2) will be replaced with the ℓ_1-norm ball. The parameters found in Table (4.4), as well as the conditions used, are still applied.

The trajectory results from both controllers are shown in Fig. (4.10) while its corresponding control effort and position plots are shown in Fig. (4.9).

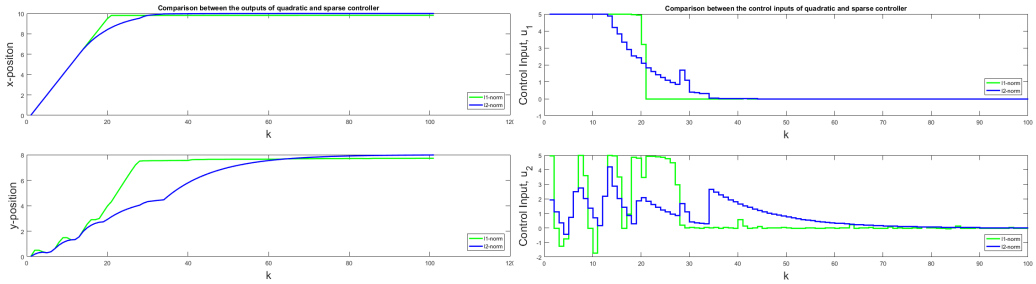

Figure 4.9: Position and control effort plots of sparse and quadratic controllers with ℓ_1-ball as obstacle

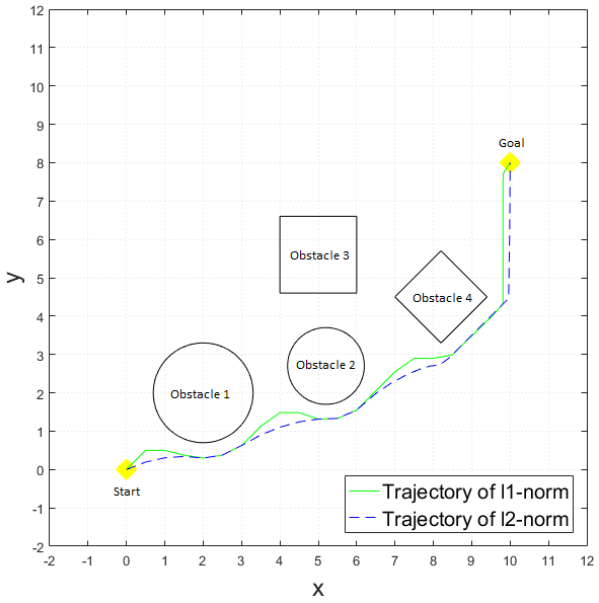

Figure 4.10: Trajectories of sparse and quadratic controllers with ℓ_1-ball as obstacle

As it can be seen in the position plot in (4.9), the vehicle with the sparse controller reaches the goal point first. This observation is similar to the observation in the previous part of the section. But notice that the gap on the y-position of both controllers is more noticeable. For this simulation, it can be stated that the sparse controller handles the hard constraints better. From the control effort, it is also evident that the control effort plot of sparse controller does not exhibit the exponential decay plot as seen from the quadratic control effort plot. The plot can be explained again by the nature of the ℓ_1-norm.

4.6.3 Sparse controller in a dynamic environment

This subsection demonstrates the application of the sparse controller in a dynamic environment. The parameters found in Table (4.4), as well as the conditions used, are still applied as the previous subsection. Dynamic environment is referred to the environment where the presence of moving obstacles is taken into account. For this demonstration, the moving obstacle is the vehicle itself. It is assumed that two vehicles are initiated in different starting points with the same goal point to reach. One vehicle size is bigger than the other (meaning, the radius of one vehicle is two times larger than the other one). With this scenario, it is expected that one will not reach the goal point due to safety distance constraint. The vehicles are simulated as if they are traversing in a one-way street. Shown in Fig. (4.11) is the trajectory and position plots of this simulation.

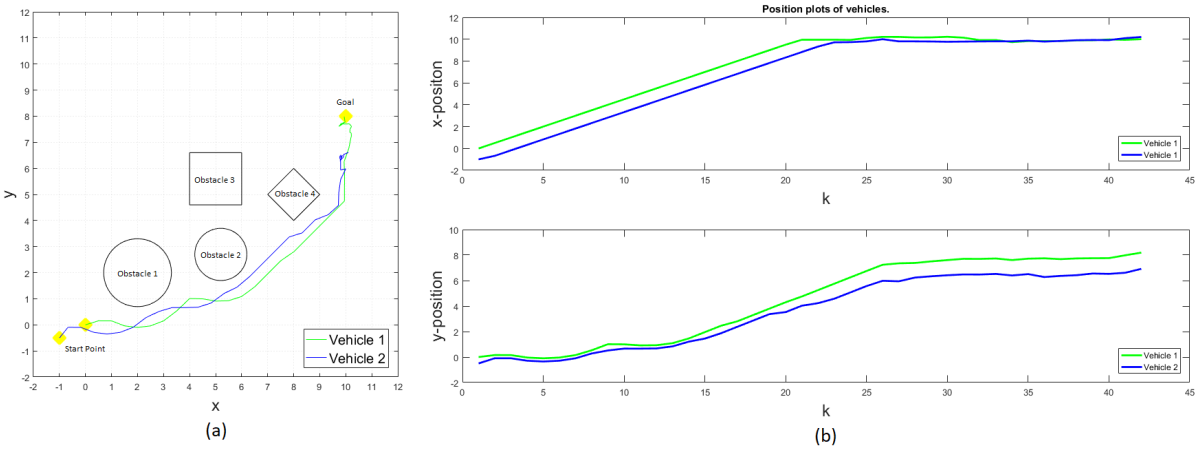

Figure 4.11: Trajectory and position plots for vehicles in a dynamic environment. Result in (a) is the trajectory plot and in (b) is the position plot.

As anticipated, one vehicle is seen not completely reached the goal point because of the other vehicle and the safe distance constraint. As can be seen in the position plot, one vehicle has to maintain the safe distance from the other vehicle, which looks like it is following the other vehicle.

5

Nonlinear Sparse Predictive Control

Contents

This chapter presents the development process and the results of another vehicle model called the unicycle model. This model involves now the orientation of the vehicle as an additional state together with the position. With the model used, a nonlinear sparse controller will then be developed. The optimization problem is still solved using the MATLAB routine $fmincon$ as described in (4.4).

5.1 Unicycle Model

Another vehicle model is the unicycle model. This model assumes that a vehicle has one steerable drive wheel as shown in Fig. (5.1). Moreover, this model ignores balancing concerns and it is assumed that the motion of the vehicle cannot slip laterally so that the translational velocity is in the direction of heading, i.e. a pure rolling contact between the wheels and the ground.

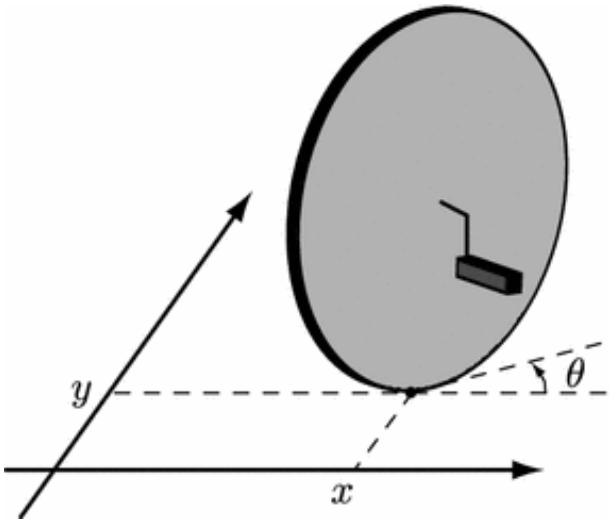

Figure 5.1: The Unicycle Model (adapted from [41])

In this model, the rider can set the pedaling speed and the orientation of the wheel with respect to the z-axis (perpendicular to the plane). Let r be the wheel radius and ω be the pedaling angular velocity of the unicycle with respect to the xy plane. Hence, the speed or linear velocity of the unicycle is defined as $s = r\omega$. The angular velocity can be defined as the rate of change of angular displacement with respect to time. Mathematically, $\dot{\theta} = \omega$. With this, the state equations can be derived as [42]

$$\begin{aligned} \dot{x} &= v cos\theta \\ \dot{y} &= v sin\theta \\ \dot{\theta} &= w \end{aligned} \tag{5.1}$$

where state $[x, y, \theta]^T$ describes the position and orientation of the center of the vehicle, and the control inputs denoted by $u = [v, \omega]^T$ are the linear and angular velocities, respectively.

Since the MPC, as described in this thesis, requires the model to be computed in discrete time, it is important to discretize the model in (5.1). Considering a sampling time T_s, and a sampling instant k, the discretized equivalent equations for this model can be expressed as

$$\begin{aligned} x(k+1) &= x(k) + T_s v(k) cos\theta(k) \\ y(k+1) &= y(k) + T_s v(k) sin\theta(k) \\ \theta(k+1) &= \theta(k) + T_s w(k) \end{aligned} \tag{5.2}$$

or, in a compact formulation,

$$\mathbf{x}(k+1) = f_d(\mathbf{x}(k), \mathbf{u}(k)) \tag{5.3}$$

5.2 Control Design

Same as presented in the previous chapter, this section will incorporate the MPC described in Chap. 2 to sparsity.

As proposed in [32], the polar coordinate transformation yields a smoother trajectory and also eliminates the steady-state error effectively. By doing so, the following state transformation to polar coordinates from the Cartesian coordinates is presented, assuming that the reference or goal point is in the origin $(x_g, y_g, \theta_g) = (0, 0, 0)$ and point (x, y) is the center of gravity of the vehicle.

$$\begin{aligned} e &= \sqrt{x_e^2 - y_e^2} \\ \phi &= tan^{-1}(\frac{y_e}{x_e}) \\ \alpha &= \phi - \theta \end{aligned} \tag{5.4}$$

where x_e = x - x_g and y_e = y - y_g. This transformation is used in the cost function to be optimized by the MPC algorithm. Letting $\mathbf{X}$ = $[e\ \phi\ \alpha]^T$ and defining the prediction model as

$$\mathbf{X}(k+i+1|k) = f_d(\mathbf{X}(k+i|k), \mathbf{u}(k+i|k)), \quad i \in [0, H-1] \tag{5.5}$$

the cost function in (2.10) is modified for nonlinear sparse controller formulation and can be rewritten as

$$\begin{aligned} J(k) &= \sum_{i=1}^{H} \mathbf{X}^T(k+i|k)\, \mathbf{Q}\, \mathbf{X}(k+i|k) + \mathbf{R}\, |\mathbf{u}(k+i-1|k)| \\ J(k) &= \sum_{i=1}^{H} \|\mathbf{X}(k+i|k)\|_Q^2 \ + \ \|\mathbf{u}(k+i-1|k)\|_R^1 \end{aligned} \tag{5.6}$$

For the quadratic controller, it will take the form

$$\begin{aligned} J(k) &= \sum_{i=1}^{H} \mathbf{X}^T(k+i|k)\,\mathbf{Q}\,\mathbf{X}(k+i|k) + \mathbf{u}^T\,(k+i-1|k)\mathbf{R}\,\mathbf{u}(k+i-1|k) \\ J(k) &= \sum_{i=1}^{H} \|\mathbf{X}(k+i|k)\|_Q^2 \;+\; \|\mathbf{u}(k+i-1|k)\|_R^2 \end{aligned} \tag{5.7}$$

For simplicity, both controllers will have the combined form as

$$J(k) = \sum_{i=1}^{H} \|\mathbf{X}(k+i|k)\|_Q^2 \;+\; \|\mathbf{u}(k+i-1|k)\|_R^p \tag{5.8}$$

where p = 1 is for the sparse controller and p = 2 is for the quadratic controller. As mentioned in Sec. (2.4), another form of cost function in the form of (2.32) is also considered here, and will be mostly used for the simulations. This cost function is also modified and will have the following structure

$$J(k) = \Omega(\mathbf{X}(k+H|k)) + \sum_{i=1}^{H} \|\mathbf{X}(k+i|k)\|_Q^2 \;+\; \|\mathbf{u}(k+i-1|k)\|_R^p \tag{5.9}$$

where $\Omega((\mathbf{X}(k+i|k))(k+H|k))$ = $\mathbf{X}^T(k+H|k)\;P\;\mathbf{X}(k+H|k)$. The terminal weight factor P = N*Q, where N is an arbitrary number.

The mathematical optimization problem can therefore be stated as

$$\begin{aligned} \min_{\mathbf{x},\mathbf{u}} \quad & J(k) \\ \text{s.t.} \quad & \mathbf{x}(k+i+1|k) \quad = f_d(\mathbf{x}(k+i|k), \mathbf{u}(k+i|k)) \\ & \mathbf{u}_{min} \quad \leq \quad \mathbf{u} \quad \leq \mathbf{u}_{max} \\ & h(\mathbf{x}) \qquad\qquad \leq 0 \end{aligned} \tag{5.10}$$

where $h(\cdot)$ represents the nonlinear function for the obstacle constraints.

5.3 Obstacle Constraint

This section describes the formulation of obstacles for environment constraints. The obstacles for this implementation are best approximated as a circle. Suppose that the obstacle is given (the radius and the center location are defined), the obstacle avoidance constraint is formulated as

$$\left\| \begin{bmatrix} x(k) \\ y(k) \end{bmatrix} - \begin{bmatrix} x^j_{obstacle}(k) \\ y^j_{obstacle}(k) \end{bmatrix} \right\| \geq d_{safe} \tag{5.11}$$

where d_{safe} is a safety constant measured from the jth obstacle's center $[x^j_{obstacle}(k), y^j_{obstacle}(k)]^T$ to the vehicle's center.

5.4 Validations

Validation must be made to verify a working nonlinear sparse controller with a unicycle model. Instead of repeating all of the validation processes presented in the previous chapter, validation for this unicycle model begins with K$\ddot{u}$hne $et\ al.$, [32] simulation conditions. The following control variable constraints are chosen as follows:

$$\mathbf{u}_{min} = \begin{bmatrix} -0.47\,m/s \\ -3.77\,rad/s \end{bmatrix}, \quad \mathbf{u}_{max} = \begin{bmatrix} 0.47\,m/s \\ 3.77\,rad/s \end{bmatrix}$$

The sampling period used is T_s = 0.1s and the weighing matrices are

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$$

which means that equal penalties are given to the position states, half of the penalty is imposed in the orientation, and the control effort is penalized less. Throughout this section, these values are used as well as the cost function in the form of (5.8), unless stated otherwise.

5.4.1 Different Initial Positions

It will be appropriate to first verify if the controllers developed (both the quadratic and the sparse predictive controller, with a higher emphasis on the latter) is applicable to perform in a free environment, meaning that there are no obstacles present, regardless of its initial positions.

Given six different initial positions: (x_0, y_0, θ_0) = $a) (-2, 2, \frac{\pi}{2})$, $b)$ $(2, 2, \frac{\pi}{2})$, $c)$ $(2, 0, \frac{\pi}{2})$, $d)$ $(2, -2, \frac{\pi}{2})$, $e)$ $(-2, -2, \frac{\pi}{2})$, and $f)$ $(-2, 0, \frac{\pi}{2})$. Each of the positions is assumed to be occupied by a vehicle. Each vehicle is tasked to reach the set point or the goal point located in the origin $(x_g, y_g, \theta_g) = (0, 0, 0)$, with subscript g denotes the goal point. The trajectory results are shown in Fig. (5.2) with prediction horizon H = 5 (as used in the paper of K$\ddot{u}$hne $et\ al.$, [32]). The cost function used is in the form of (5.8).

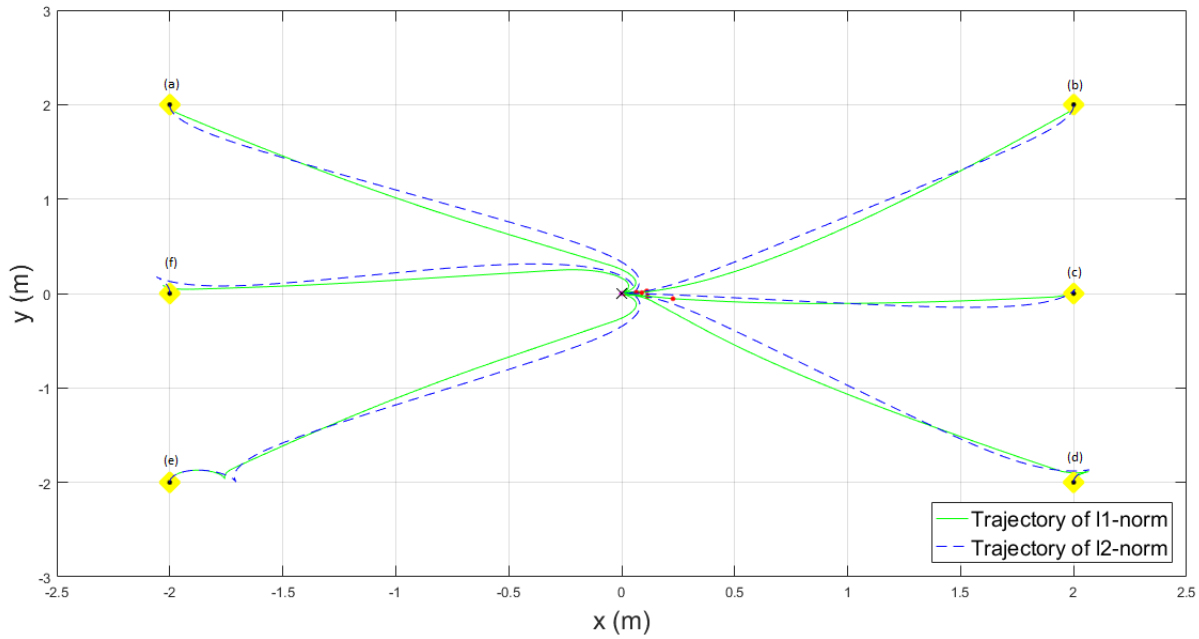

Figure 5.2: Trajectory plots for the quadratic and sparse controllers using the unicycle model

As seen in Fig. (5.2), the yellow-shaded block with a $dot(\cdot)$ is the set point where the vehicle initially starts while the (x) mark is where the vehicle must reach or the goal point. To be consistent with the trajectory labels as used for the results in the previous chapter, the green solid line is for the trajectory resulted from the sparse controller. The blue dashed line is then the trajectory obtained from using the quadratic controller. Moreover, the red-colored $(\cdot)$ is the final position reached using the sparse controller while the magenta-colored $(\cdot)$, which might not be visible since it overlaps the origin, is for the quadratic controller. It can be deduced from this figure that both controllers satisfy the objective to perform in an obstacle-free environment. It is noticeable, however, that the trajectories of both controllers slightly differ from each other but is understandable. Furthermore, it is visible that some final points of the sparse controller do not converge well to the goal point or the origin.

5.4.2 Different Initial Orientations

Since the model involves now an orientation of the vehicle, it is important to demonstrate if the sparse controller developed is capable of steering regardless of the initial orientations. With the same environment and conditions, namely the initial position of the vehicle and the goal point, different initial orientations θ_0 will be considered. The orientation for the goal point is still the same as the previous case, θ_g = 0 and initial orientation θ_0 = $\frac{\pi}{2}$ will be excluded from this validation. The initial orientations to be performed are the following: $a)$ $\theta_0 = 0$, $b)$ $\theta_0 = -\frac{\pi}{2}$, $c)$ $\theta_0 = \pi$, and $d)$ $\theta_0 = \frac{\pi}{4}$.

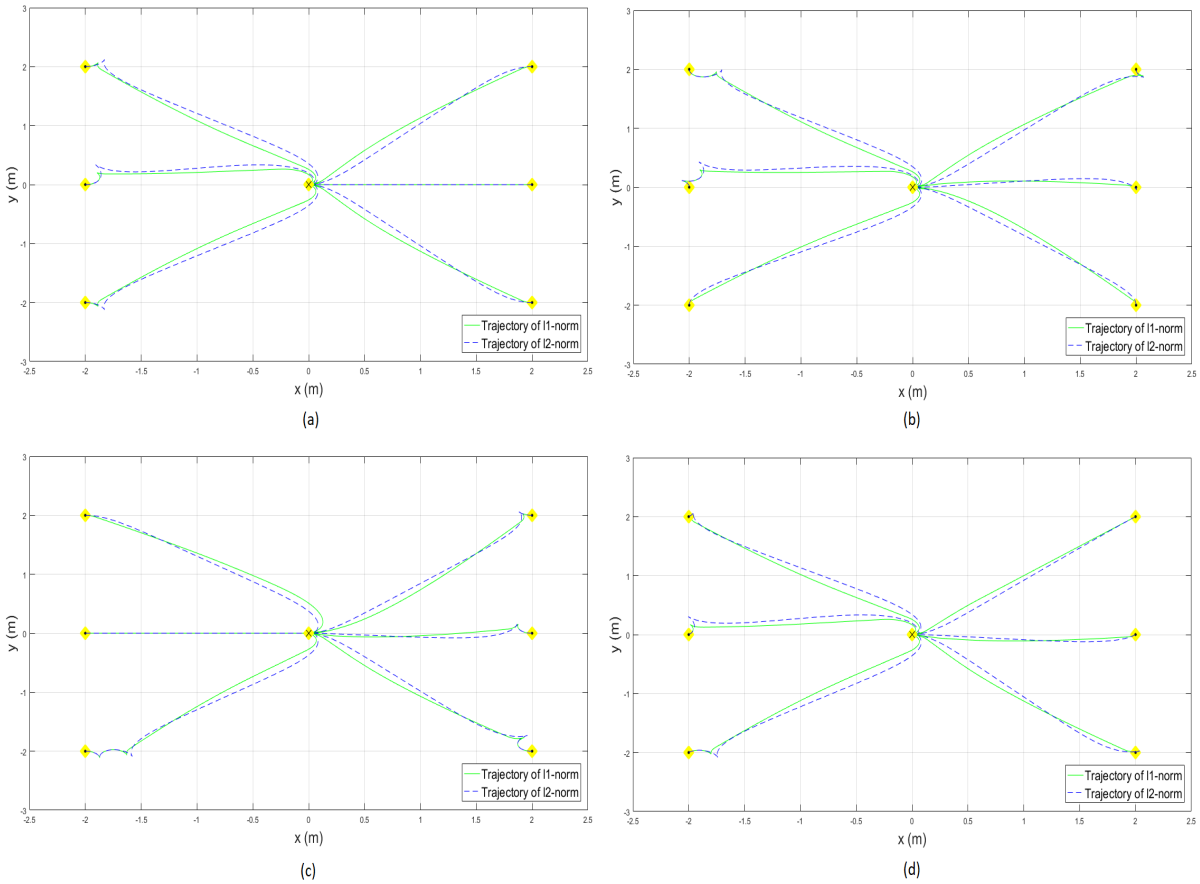

Figure 5.3: Trajectory plots for the quadratic and sparse controllers using the unicycle model with different initial orientations. Results obtained with the following initial orientations: $a)\ \theta_0 = 0$, $b)\ \theta_0 = -\frac{\pi}{2}$, $c)\ \theta_0 = \pi$, and $d)\ \theta_0 = \frac{\pi}{4}$.

All are still performed with the prediction horizon of $H = 5$. Furthermore, the aim is still to move the vehicle towards the goal point. It is assumed that both controllers converge to the origin. The results of this demonstration are shown in Fig. (5.3). It can clearly demonstrate that the controller developed can performed regardless of the initial positions and orientations.

5.4.3 Convergence Correction: Increasing the Prediction Horizon

To minimize the error of convergence of the sparse controller, as seen in Fig. (5.2), it is proposed to increase the prediction horizon. Shown in Fig. (5.4) and Fig. (5.5) the results of these changes on both controllers. Fig. (5.4) is taken with prediction horizon $H = 10$ while Fig. (5.5) with prediction horizon $H = 20$.

It can be observed that increasing the prediction horizon will minimize the convergence error of the sparse controller. Furthermore, as the prediction horizon increases, the trajectory of sparse controller tends to behave as the quadratic controller.

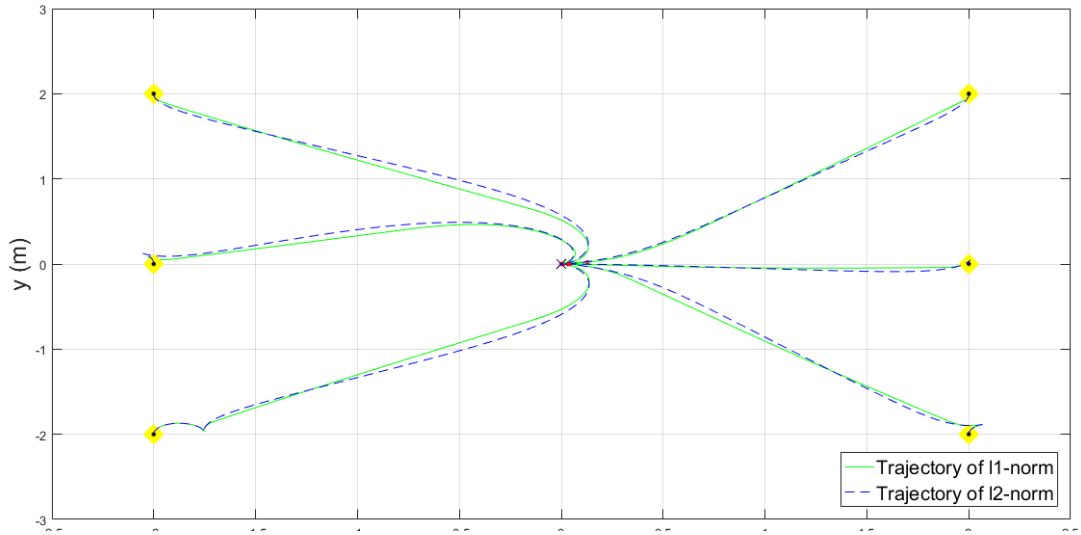

Figure 5.4: Trajectory plots with H = 10

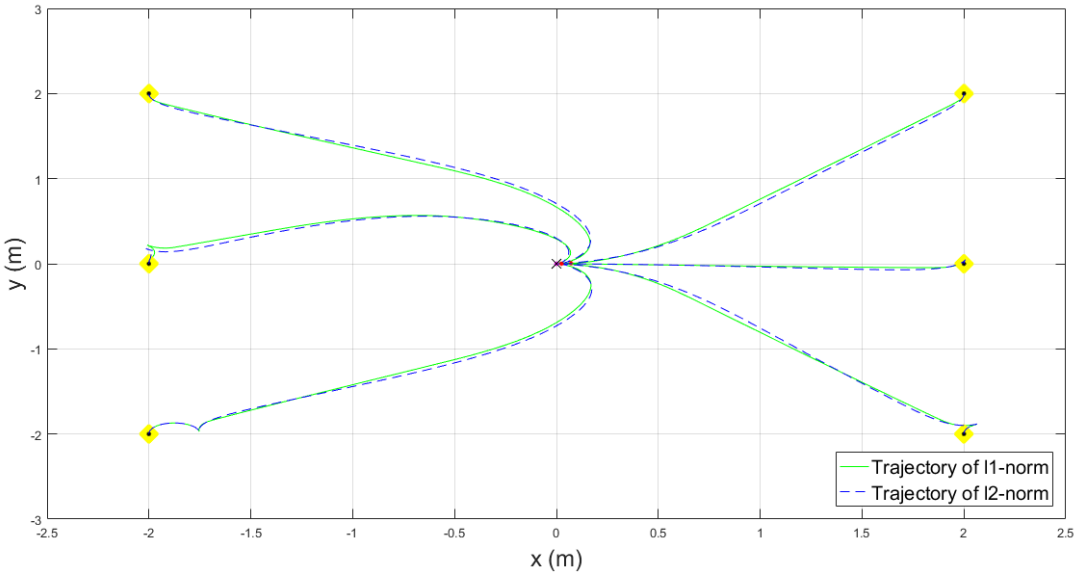

Figure 5.5: Trajectory plots with H = 20

5.4.4 Convergence Correction: Addition of Terminal Cost

In an attempt to reduce or eliminate the convergence error, modification on the cost function is also suggested. This modification is given in (5.9). For this modification, the weight factor Q remains constant, meaning, it is time-invariant. The terminal state is penalize with a weight factor P = N*Q. For this experiment, different numbers of N are chosen to be simulated. These numbers are: $a)$ $N = 1$, $b)$

$N = 5$, $c)$ $N = 10$, and $d)$ $N = 50$. These values are simulated with H = 10 and the results are all displayed in Fig. (5.6), considering only the sparse controller.

Notice how the terminal state converges in the origin as the terminal weight factor P is increased by tuning the value of N. It can be seen that at a certain value of N, the trajectory generated will start to change. In Fig. (5.6d), this change can be observed. Note also that the terminal state of some vehicles for this value of N tends to become unsteady, meaning a fluctuations can be seen as the vehicle tries to settle in the goal point.

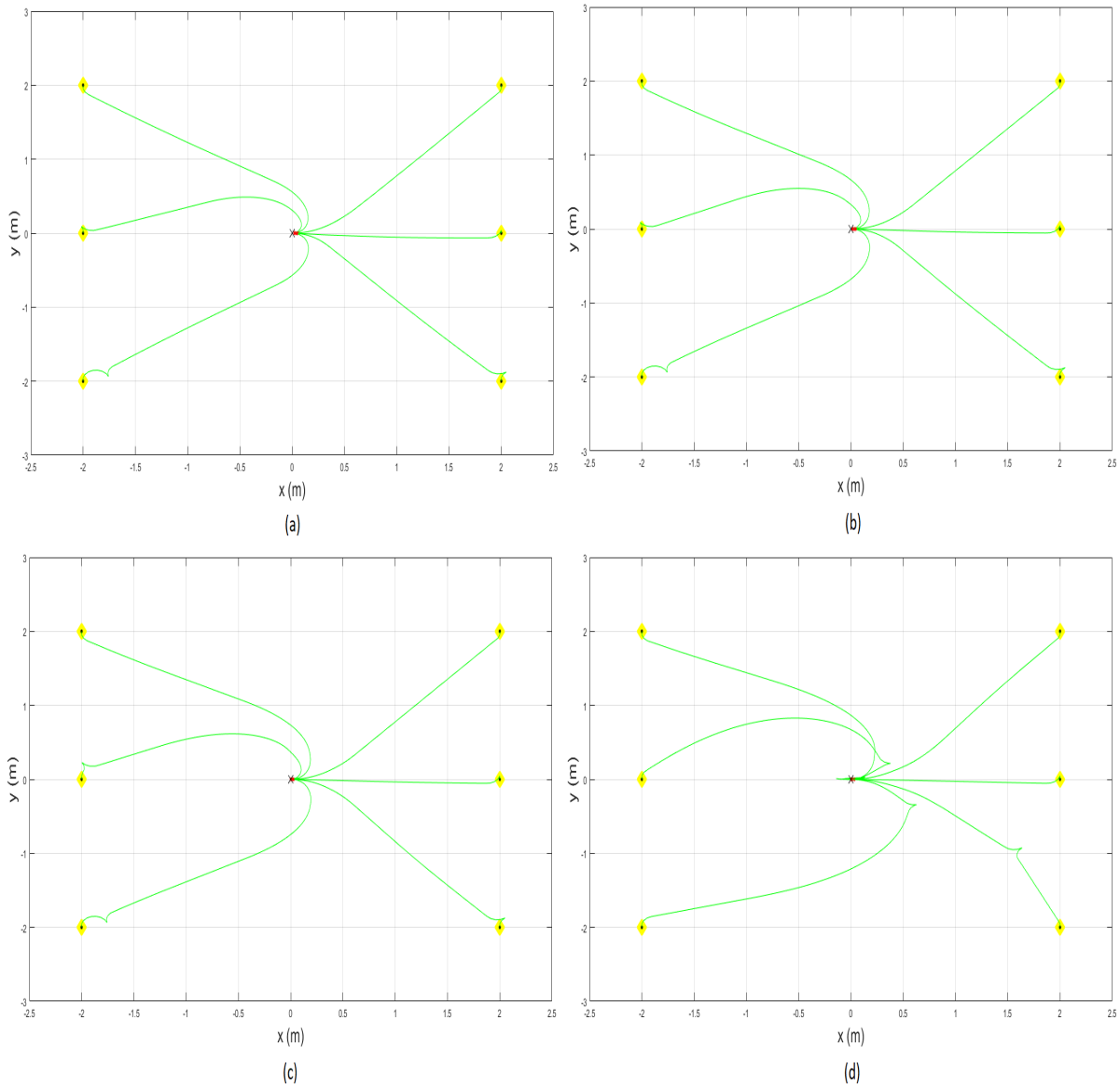

Figure 5.6: Trajectory plots for a different cost function of sparse controller Results obtained with the following N: $a)$ $N = 1$, $b)$ $N = 5$, $c)$ $N = 10$, and $d)$ $N = 50$.

Though it is verified in here that adding a terminal cost will indeed improve the convergence of the terminal state, recall that a careful choice of weight factors, not only the terminal weight factor, is still a concern. Nevertheless, the addition of terminal cost is recommended in the context of this thesis to solve the issue in convergence error for the nonlinear sparse controller, and hereby be carried out in the succeeding sections.

5.5 Obstacle Avoidance Problem

This section considers a defined environment consisting of a single obstacle. In a general case, most of the shapes of obstacles in a real-world environment have no definite form and are difficult to describe. In the simulations to follow, the irregular shapes of the obstacles can best be represented as a disc with a radius of r_b, i.e. for this simulation r_b = 0.5m. The vehicle is also assumed with a shape of a circle with radius r_v = 0.2m.

The vehicle is simulated to start at point (x_0, y_0, θ_0) = $(-2, 2, \frac{\pi}{2})$ and is tasked to steer at point (x_g, y_g, θ_g) = $(0, 0, 0)$. The obstacle is centered at point (x_b, y_b) = $(-1, 1)$. The parameters used are shown in Table (5.1).

Parameter	Value
T_s	0.1s
Q	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}$

Table 5.1: Fixed Parameters for Obstacle Avoidance Problem

Fig. (5.7) shows the trajectories for both quadratic and sparse controllers utilizing the cost function in (5.9) under the prediction horizon H = 10, and weight matrices R = $\begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$ and P = $20 * Q$. Still, the green solid line is for the trajectory generated using the sparse controller while the blue dashed line is for the quadratic controller. For the additional trajectories shown, the black dotted and dash-dotted lines are for the sparse and quadratic controllers without the obstacle, respectively.

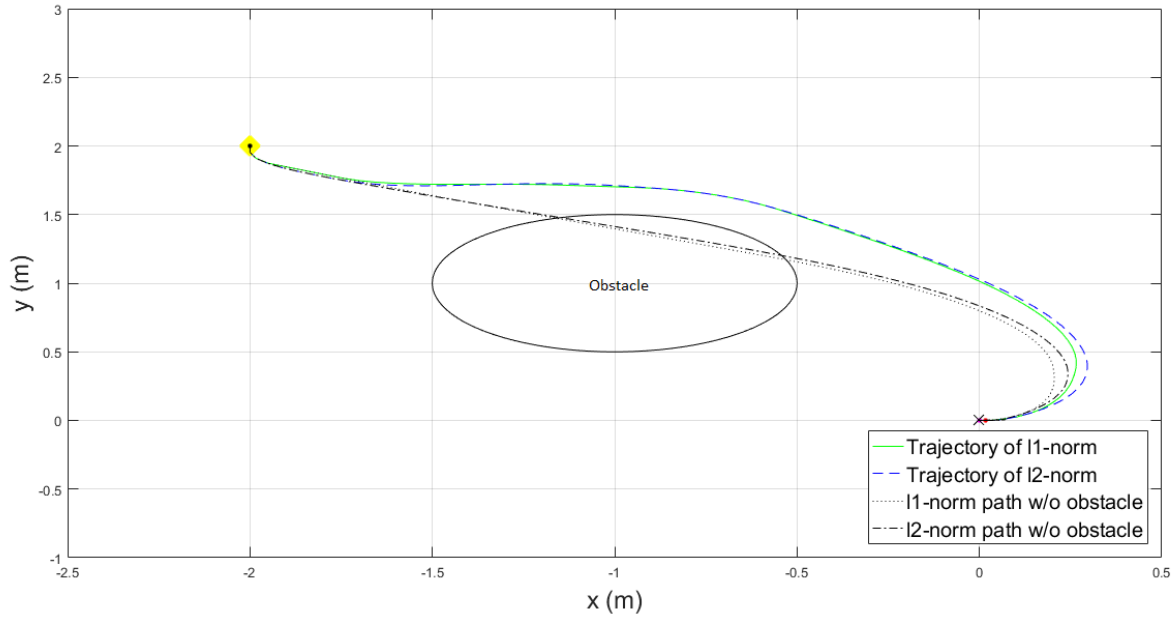

Figure 5.7: Trajectory for sparse and quadratic controllers with obstacle.

As shown in Fig. (5.7) and with parameters used as provided, the sparse controller is able to handle the obstacle constraint, the same with the quadratic controller. The trajectories with the obstacle can be described as the path taken by the vehicle when it traverses the environment. Upon detecting the obstacle in its way to the origin, it keeps a distance away from the it and changes its direction of heading.

The cost over time is shown in Fig. (5.8), while the control effort (linear and angular velocities) as well as the individual state plots are shown in Fig. (5.9).

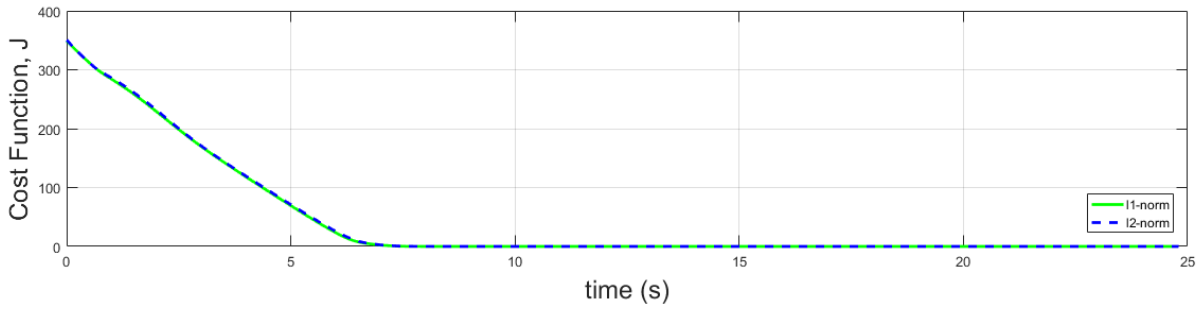

Figure 5.8: Cost over time for sparse and quadratic controllers with obstacle.

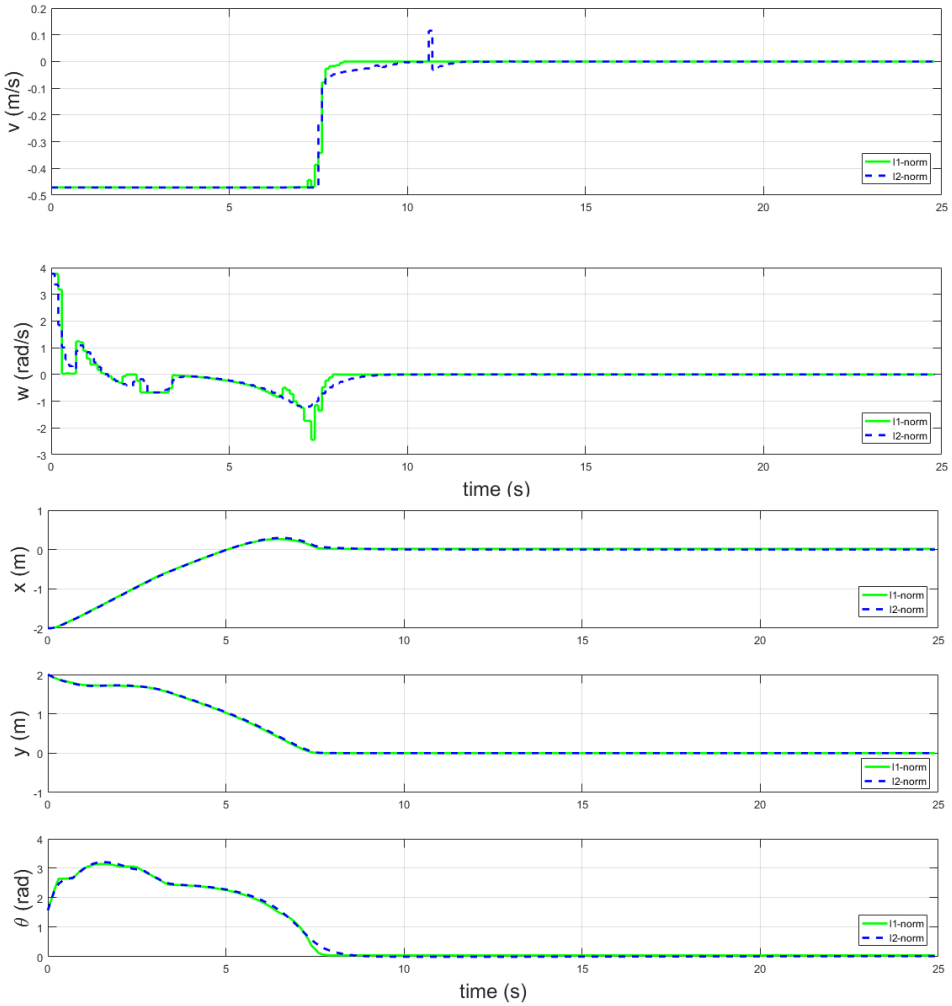

Figure 5.9: Control effort and state plots. The first two plots are for the control effort while the remaining three plots are for the state.

The total cost, considering only the situation with obstacle, is calculated as 10800 for the sparse controller while 10900 for the quadratic controller. Note that a small difference is due to the use of smaller penalties in the control effort (1/10th penalties are imposed in the form of matrix R).

Consider now a higher penalty than the previous case with $R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ but this time, $H = 20$ is used for both controllers to perform the obstacle avoidance (using $H = 10$ will make the quadratic controller not feasible for this demonstration). The cost over time is shown in Fig. (5.10), while the control effort and individual state plots are shown in Fig. (5.11).

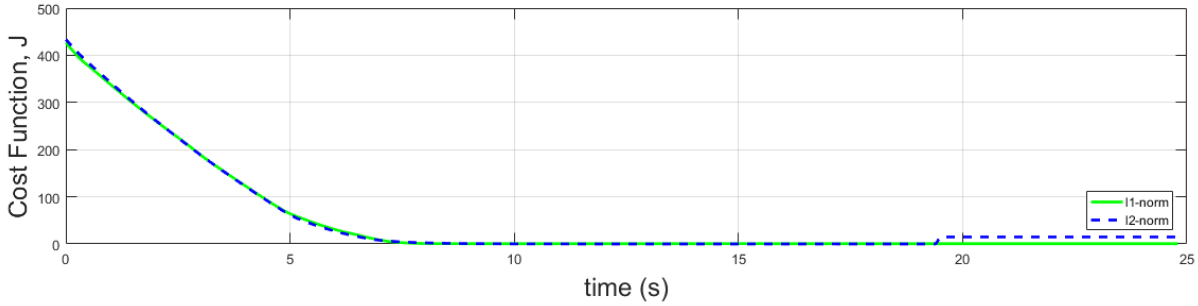

Figure 5.10: Cost over time for a higher penalty on the control cost.

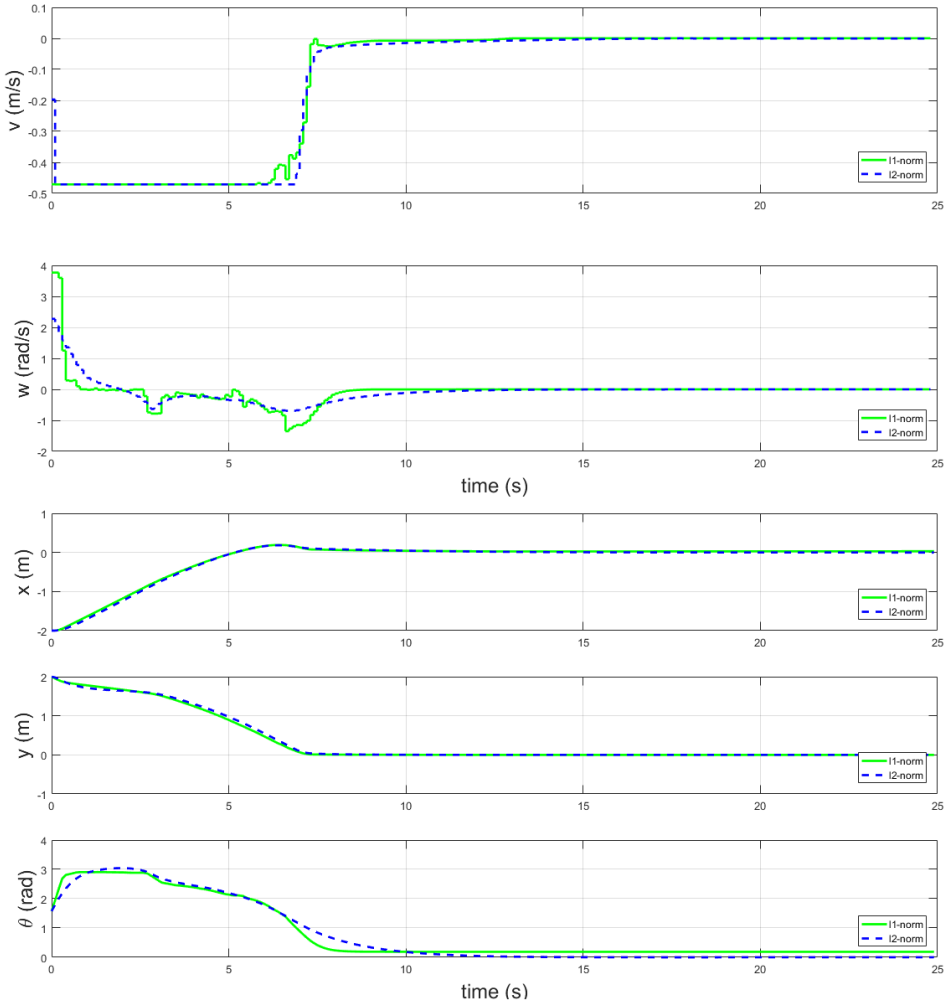

Figure 5.11: Control effort and state plots for a higher penalty on the control cost.

The total cost calculated from (5.10) for the sparse controller is 12500 while 13300 is calculated for the quadratic controller. With these figures, it can be said that the sparse controller minimizes more the cost incurred during the whole process with a satisfactory performance comparable to the quadratic

controller.

5.5.1 Prediction Horizon Choice

This subsection is created as an additional evaluation to the importance of tuning the prediction horizon. It will be demonstrated here the trajectories produced by using different horizons in an obstacle avoidance problem. For this demonstration, the parameters used is shown in Table (5.2).

Parameter	Value
T_s	0.1s
Q	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}$
R	$\begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$

Table 5.2: Fixed Parameters for Cost vs. H Evaluation

Fig. (5.12) depicts the result of performing different horizons with $P = 20 * Q$.

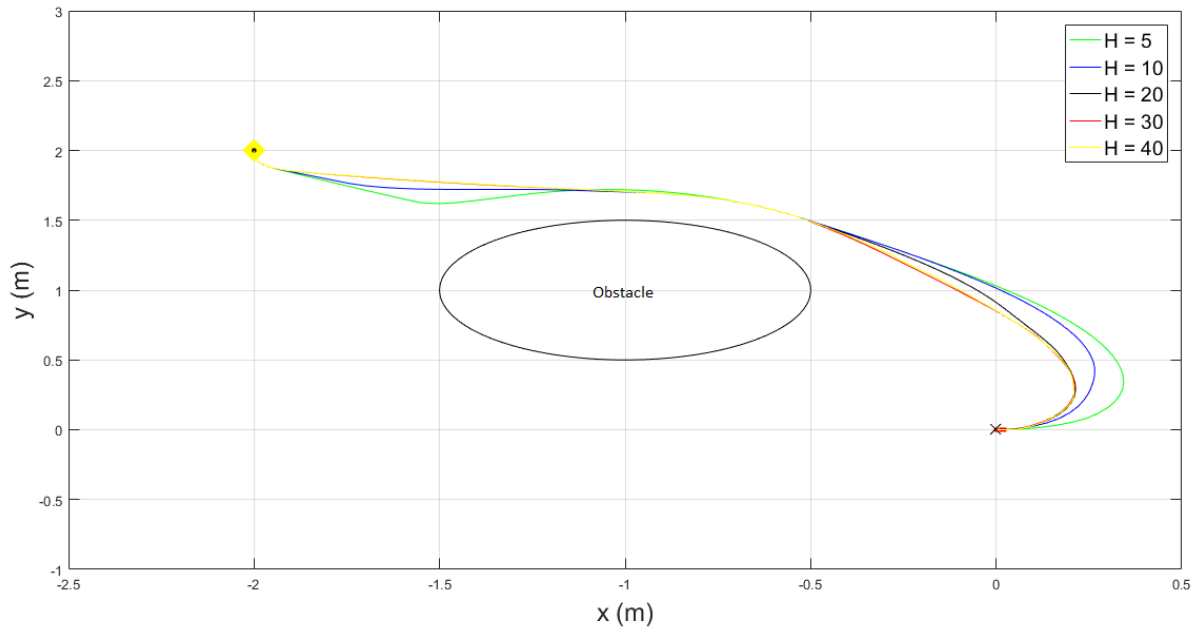

Figure 5.12: Trajectories of different horizons in obstacle avoidance.

As shown in Fig. (5.12), the solid line in green is for $H = 5$, in blue for $H = 10$, in black for $H = 20$, in red for $H = 30$, and in yellow for $H = 40$. The figure implies that larger prediction horizons are required to ensure a smooth maneuver by giving the vehicle more information on a look ahead basis, thus planning its path earlier upon detecting the obstacle.

Fig. (5.13) shows a different scenario with $P = 10 * Q$.

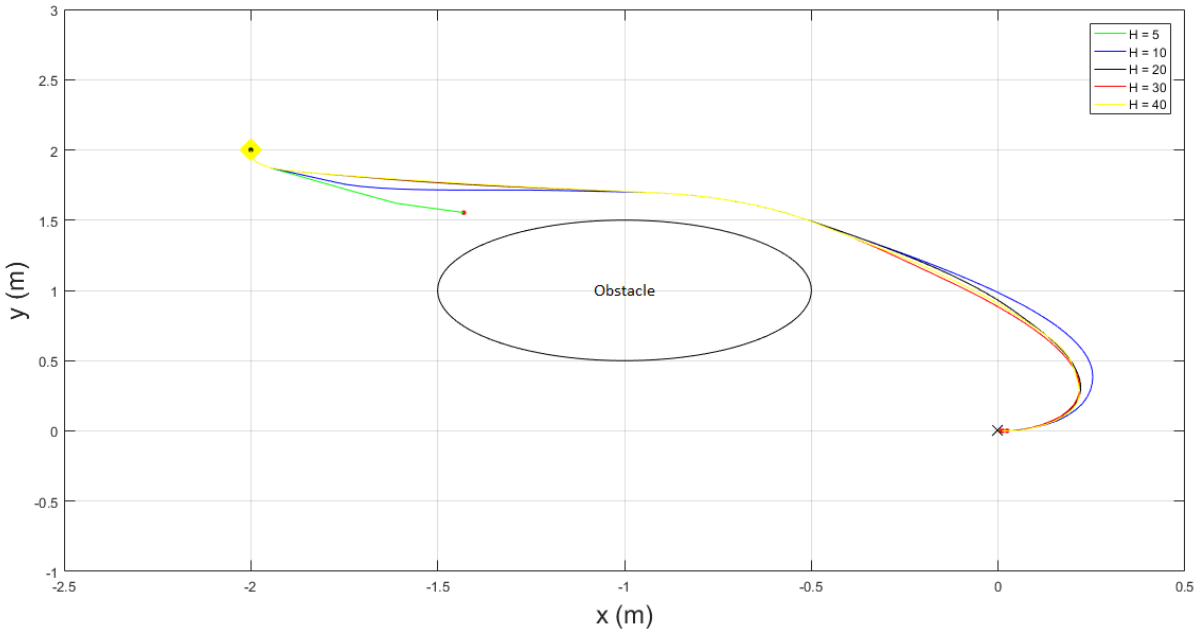

Figure 5.13: Trajectories of different horizons in obstacle avoidance with $P = 10Q$.

It can be seen that in this case, the vehicle is stuck and is unable to move due to having a smaller horizon length ($H = 5$).

5.5.2 Conventional Cost Function

This subsection determines whether the cost function in (5.8) can be considered in an obstacle avoidance problem. For this demonstration, the parameters used are found in Table (5.3).

Parameter	Value
T_s	0.1s
Q	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}$
R	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

Table 5.3: Fixed Parameters

Fig. (5.14) displays the result of the demonstration for the quadratic controller.

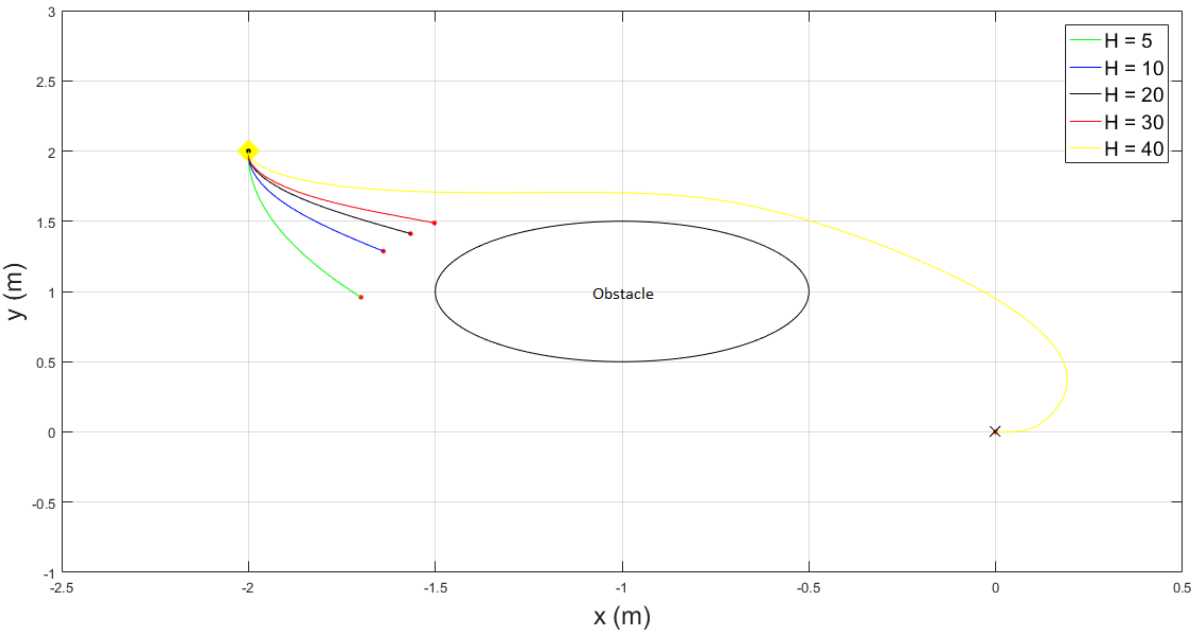

Figure 5.14: Trajectory for quadratic controller using a conventional cost with obstacle.

On the other hand, Fig. (5.15) reveals the result for the sparse controller.

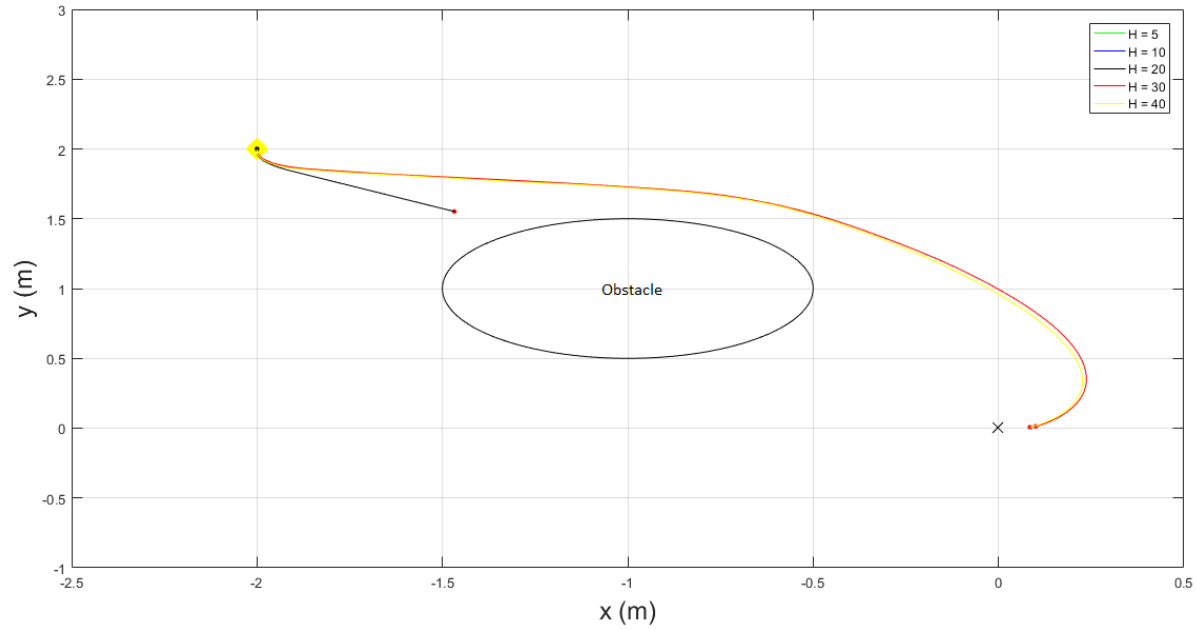

Figure 5.15: Trajectory for sparse controller using a conventional cost with obstacle.

It can be noticed that by using the cost in (5.8), both controllers can only perform the obstacle avoidance problem with higher prediction horizons. Prediction horizons higher than $H = 40$ (it is assumed) are required for the quadratic controller while values higher than $H = 30$ are necessary for the sparse con-

troller to perform in this problem. Aside from its capability to minimize the steady-state error, the addition of terminal cost is still preferred, especially in (wheeled) vehicles for nonlinear formulation, because of its flexibility, i.e. tuning necessary parameters that can be applicable even with smaller prediction horizons.

5.5.3 Multiple Obstacles

This subsection is provided to discern the performance of the developed controller in a multiple obstacles scenario. For this demonstration, the cost function to be used is still in the form of (5.9) with the parameters used found in Table (5.4).

Parameter	Value
H	30
T_s	0.1s
Q	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}$
R	$\begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$
P	20*Q

Table 5.4: Fixed Parameters

The environment this time is assumed to be occupied by 2 obstacles represented as circles. One obstacle is situated with center at $C_1(-1.5, 2)$ and the other is at $C_2(-3.5, 3)$. The radius of one is two times larger than the other (r_1 = $0.5m$, r_2 = $1m$). The vehicle is initiated at point (x_0, y_0, θ_0) = $(-6, 6, \frac{\pi}{2})$ with goal to reach at point (x_g, y_g, θ_g) = $(0, 0, 0)$. The resulting trajectory is shown in Fig. (5.16).

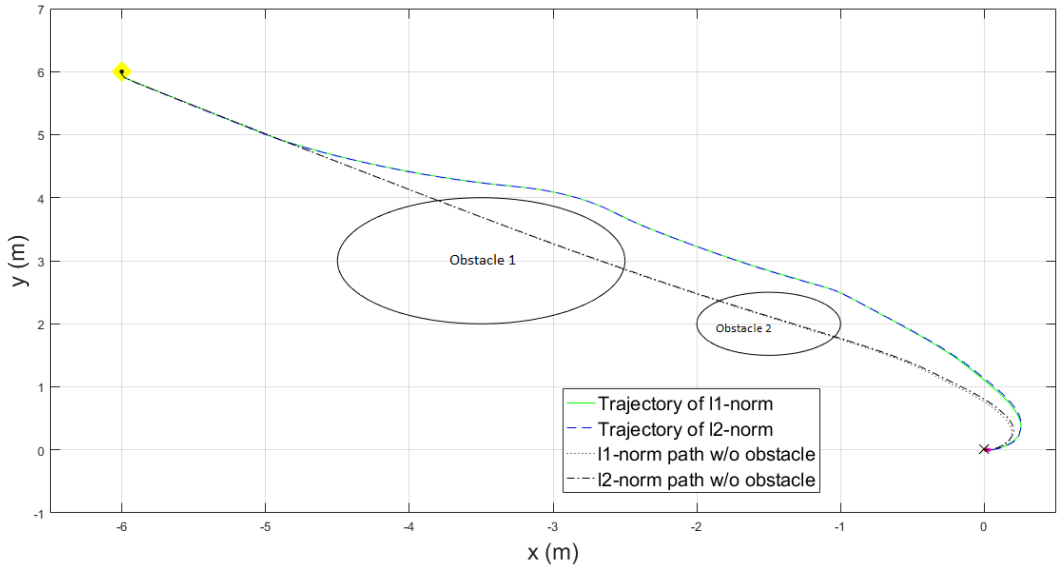

Figure 5.16: Trajectories of both controllers in obstacle-free and multiple obstacles problem.

The trajectories of both controllers in a free environment are also provided. As it can be seen, both controllers can also be applied in a multiple obstacles scenario where the obstacles are represented as a disc or circle. The control efforts of both controllers in multiple obstacles are revealed in Fig. (5.17).

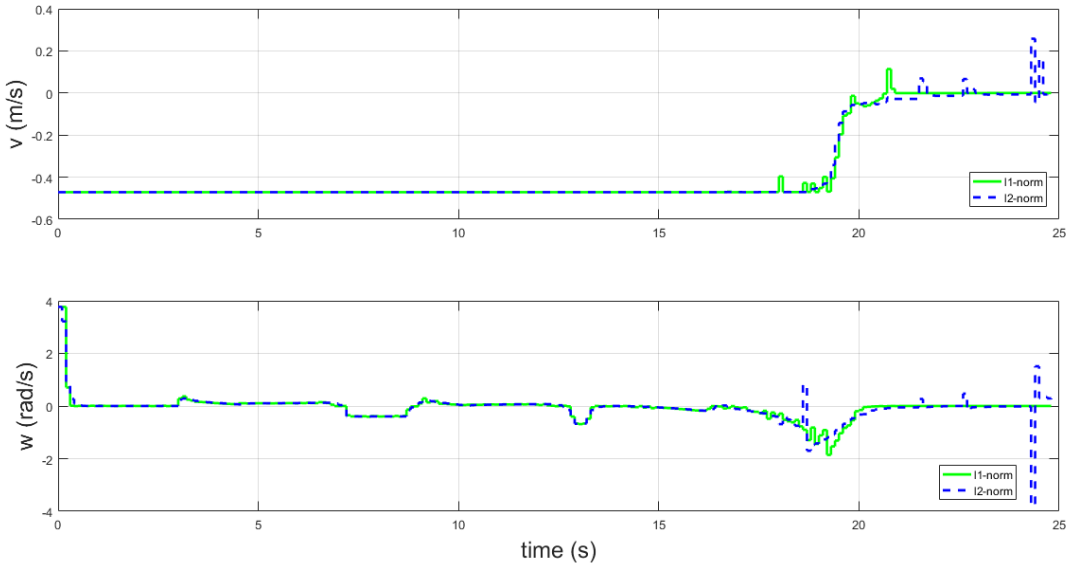

Figure 5.17: Control Effort of both controllers for multiple obstacles problem.

It is seen that the control effort for sparse controller has more stabilized region than the quadratic one. This stabilized region can be illustrated by the nature of the ℓ_1-norm penalty or sparsity, where it penalizes the number of nonzero elements and force many elements to be equal to zero. As a consequence, it can also be noticed that fluctuations of control values are minimal for sparse controller compare to the quadratic controller.

5.6 Evaluation of the Effect of Parameters on the Cost

The results presented in this section are done in an environment with one obstacle just like the environment as shown in Fig. (5.7). The purpose of this section is to present the results of fine tuning significant parameters such as the prediction horizon, and weight matrices.

5.6.1 Cost vs. Prediction Horizon

To check the effect of prediction horizon to the cost function of nonlinear sparse predictive control algorithm, the following fixed parameters are used as shown in Table (5.5).

Parameter	Value
T_s	0.1s
Q	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}$
R	$\begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$
P	20*Q

Table 5.5: Fixed Parameters for Cost vs. H Evaluation.

Different prediction horizons are considered to determine its corresponding total cost. These horizons are the following: $H = 10$, $H = 20$, $H = 30$, and $H = 40$. Fig. (5.18) depicts the cost over time of the horizon choices.

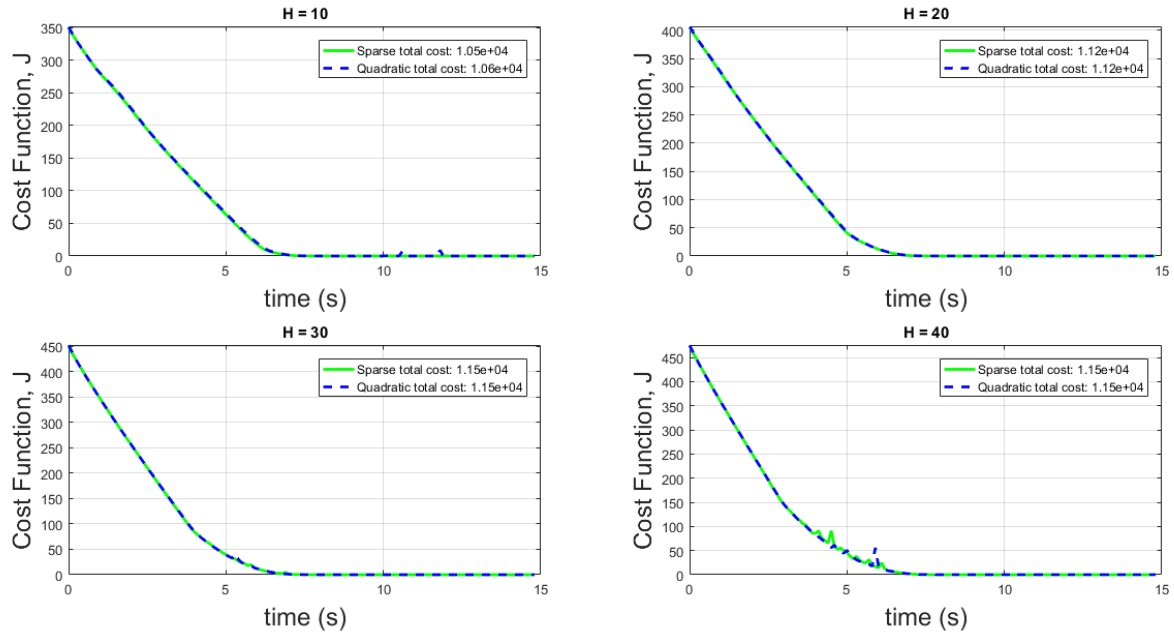

Figure 5.18: Cost over time of different horizons used.

For the purpose of this evaluation for the effect of horizons on the cost of both controllers, fixed parameters assigned remain as is. In the figure, the trajectories of the cost can be seen as exponentially decreasing. The corresponding total cost incurred by both controllers in each defined horizon is also shown. It can be inferred that increasing the prediction horizon will make the process more costly. This information is an additional consideration to what is pointed out in Sec. (5.4.4). However, for this demonstration at a certain point, the increase of the total cost is only minimal or is not clearly seen. This can be justified by the minimization problem defined. Since the aim here is to approach the origin, the values in higher horizons are smaller ones thus these values do not really contribute to the totality of cost.

5.6.2 Cost vs. Weight Matrices

To determine the effect of weight matrices Q, R, and P to the cost function of nonlinear sparse predictive control algorithm, the following fixed parameters are used. Only one case per weight matrix will be considered. It is assumed that a higher value on every weights is enough to learn the significant effect of these weights.

Parameter	Value
H	20
T_s	0.1s

Table 5.6: Fixed Parameters for Cost vs. Weights Evaluation

For the effect of weight factor Q, which penalizes the deviation of the state from the goal point, the following weight matrices are considered: $Q = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 10 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}$, $R = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$ and P = 20*Q. Fig. (5.19) shows the comparison of the cost over time between the result as seen in Sec. $(5.6.1)$ with $H = 20$ and with the new defined matrix Q.

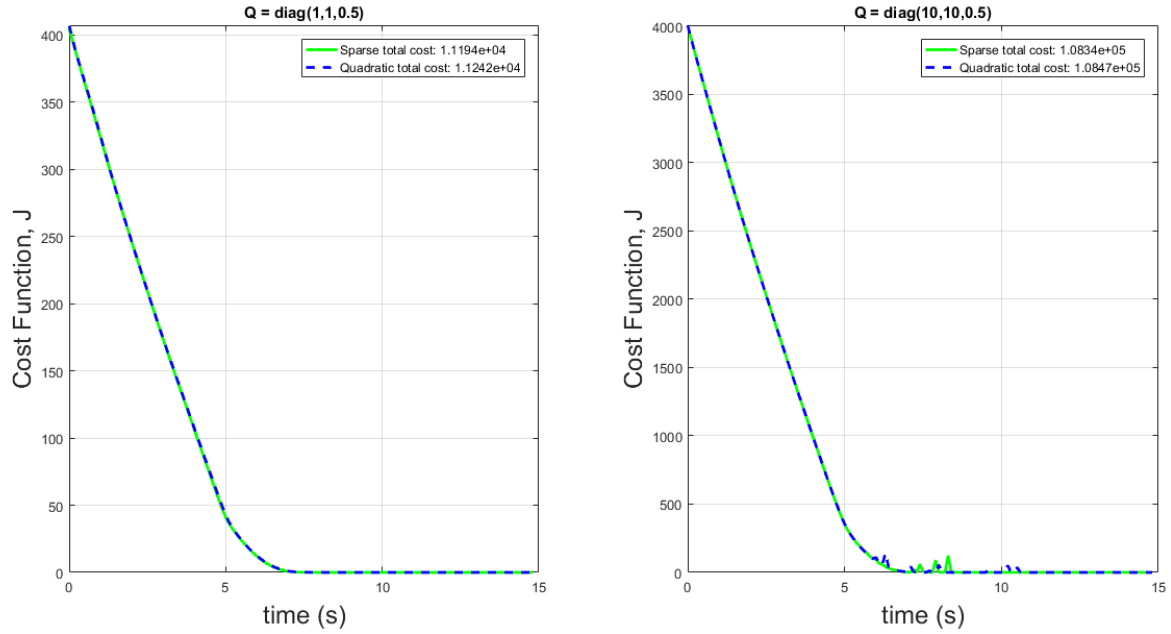

Figure 5.19: Cost over time of the change of matrix Q.

It can be seen that a huge difference has been calculated as a consequence of penalizing the position about 10 times higher. The difference can also be approximated as 10 times more of the former result.

Evaluating the effect of weight factor R, which penalizes the control effort, the following weight matrices are considered: $Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}$, $R = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$ and P = 20*Q. Again, the only change from Sec.

(5.6.1) is the weight factor R with 5 times the penalty now on the inputs. Comparison result is shown in Fig. (5.20).

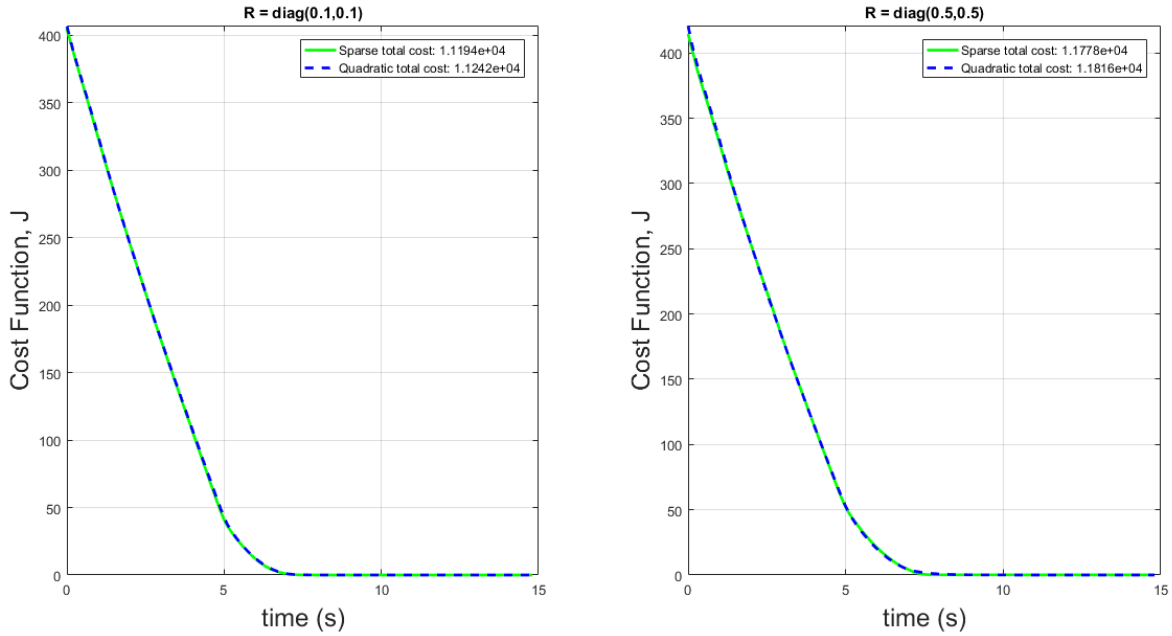

Figure 5.20: Cost over time of the change of matrix R.

Comparing the values shown in the figure, a slight change occurs. This can be analyzed by recalling the constraints on the control inputs, which are only small values. Furthermore, even though the change is 5 times of the matrix R in Sec. (5.6.1), it is still half the penalty(0.5) of a whole. For demonstration purposes, the only change is with matrix R and increasing this penalty will result to inability of the vehicle to steer.

Since the cost function used also has an additional terminal cost with weight factor P, the following weight matrices are used for evaluation: $Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}$, $R = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$ and P = 40*Q. Again, the only change from Sec. (5.6.1) is the weight factor P with two times the penalty now on the terminal cost. Result is shown in Fig. (5.21).

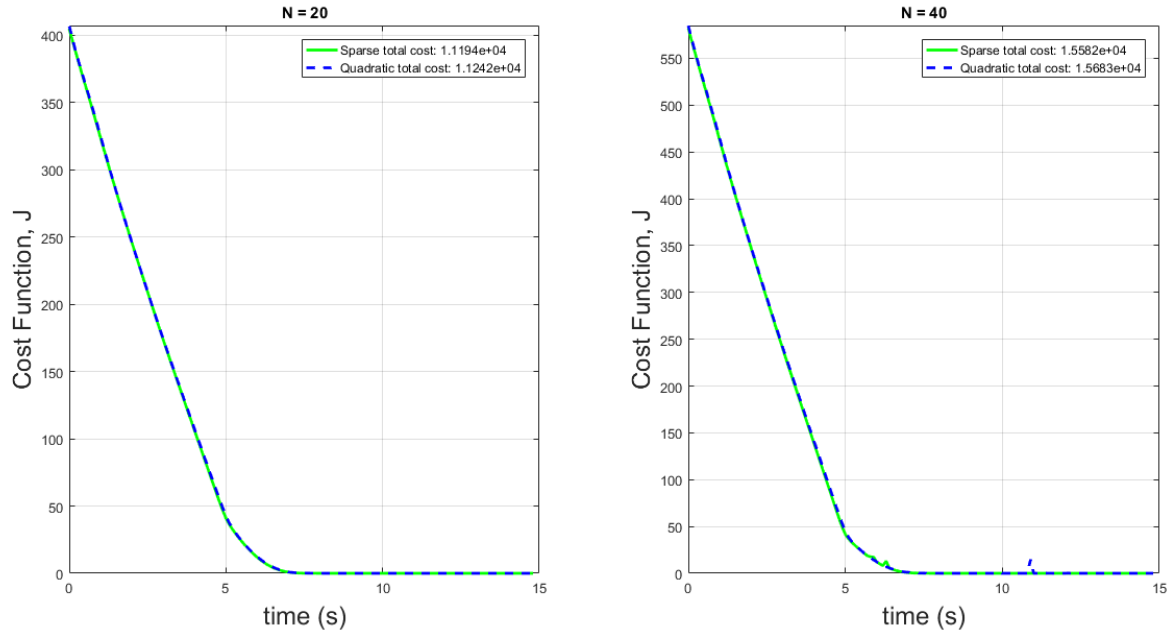

Figure 5.21: Cost over time of the change of matrix P.

Comparing the values indicated in the figure, a big difference is also observed in relation to the change in matrix Q previously. This increase can be explained with penalizing the quadratic nature of the terminal cost.

6

Conclusions and Future Work

The aim of this thesis is to develop a control strategy utilizing the Model Predictive Control (MPC) scheme and sparsity for vehicles modeled as linear, as well as nonlinear systems. MPC is a control strategy that generates the control action by solving an optimization problem in a finite receding horizon principle, with the measured state as the initial state, that can handle the state and input constraints directly. The main components of the MPC algorithm include the performance measure or the cost function. In this work, the conventional cost function of MPC is modified by applying the idea of sparsity, which penalizes the the number of nonzero elements in the control sequence. This modification resulted to what is known, in the context of this thesis, as the sparse predictive control strategy.

Chapter 2 gives the necessary knowledge and background on MPC. On the other hand, Chapter 3 provides an adequate idea on sparsity.

In Chapter 4, a sparse predictive control for linear system is instigated. For the demonstration, the single integrator is used as the plant model, which is also one of the main components of the MPC algorithm. Selection of parameters are performed to address the viability of the controller developed. It is then compared to the performance of the conventional MPC which, in the context of the thesis, is called the quadratic controller. Both controllers are evaluated to perform in an obstacle avoidance problem where the objective is to reach in a fixed goal point from a defined starting point. A multiple fixed obstacles are given in the environment with the shape of all ℓ_p-norm balls. Furthermore, a dynamic environment is also demonstrated by performing two vehicles traversing together without colliding to themselves and the given fixed obstacles.

In Chapter 5, a nonlinear model is presented. A unicycle model is used for the nonlinear representation. The parameters for both sparse and quadratic controllers are also validated for the demonstration. For the simulation, both controllers are applied in a free and fixed obstacle-inhabited environments.

The results suggest that the sparse predictive control developed is better, if not as good as the quadratic controller. Though a significant issue is found in sparse controller for its incapacity to reach the goal point as desired, but this drawback is addressed by adding a terminal cost in the objective function. Fine tuning of parameters will also help solve the problem without affecting the solution of the sparse controller, but the parameters chosen might not be suitable for the quadratic controller for comparison purposes. The result for the sparse controller is shown to stabilize more the movement of the vehicle after reaching the goal point. The control effort in the entire process or journey is able to demonstrate the nature of sparsity, thus believing to minimize the fuel consumption if it relates to a real world scenario.

Finally, in the future and since this work only considers the simple models to represent a vehicle, it would be impressive to consider a more complex model of vehicles, i.e. bicycle model with consideration to the road frame. Instead of having one or two vehicles, it will be proper if the pursuit-evasion games of multiple vehicles is considered in the simulations.

It would be interesting to assess also the performance of sparse controller in a real-world scenario with the use of mobile robots. With the evaluation, it will be appropriate to include the disturbance model and investigate the robustness of the controller developed.

Bibliography

[1] D. F. Golnaraghi and D. B. C. Kuo. (2017) Automatic Control Systems. [Online]. Available: https://www.accessengineeringlibrary.com/browse/automatic-control-systems-tenth-edition#preface05

[2] E. Feron. (2016) Advances in Control System Technology for Aerospace Applications. [Online]. Available: https://books.google.pt/books?id=aI6QCgAAQBAJ&printsec=frontcover#v=onepage&q&f=false

[3] (2007) Chemical Process Dynamics and Controls Book I. [Online]. Available: https://open.umich.edu/sites/default/files/downloads/chemical_process_dynamics_and_controls-book_1.pdf

[4] M. A. Stephens, C. Manzie, and M. C. Good, "Model Predictive Control for Reference Tracking on an Industrial Machine Tool Servo Drive," *IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS*, vol. 9, no. 2, pp. 808–816, 2013.

[5] Y. Yang, S.-C. Tan, and S. Y. R. Hui, "Adaptive Reference Model Predictive Control With Improved Performance for Voltage-Source Inverters," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 2, pp. 724–731, 2018.

[6] Navigant Research Leaderboard: Automated Driving Vehicles. [Online]. Available: https://www.navigantresearch.com/research/navigant-research-leaderboard-automated-driving-vehicles

[7] Waymo Formally Applies for Fully Driverless Car Tests in California. [Online]. Available: https://www.engadget.com/2018/04/13/waymo-applies-fully-driverless-car-tests-california/

[8] R. E. Kalman, "Contributions to the Theory of Optimal Control," *Bulletin de la Societe Mathematique de Mexicana*, no. 5, pp. 102–119, 1960.

[9] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of ASME, Journal of Basic Engineering*, no. 87, pp. 35–45, 1960.

[10] N. R. Ruchika, "Model Predictive Control: History and Development," *International Journal of Engineering Trends and Technology*, vol. 4, no. 6, pp. 2600–2602, 2013.

[11] S. J. Qin and T. A. Badgwell. (2002) A Survey of Industrial Model Predictive Control Technology. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0967066102001867

[12] J. Richalet, A. Rault, J. Testud, and J. Papon. (1978) Model Predictive Heuristic Control: Applications to Industrial Processes. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0005109878900018

[13] C. Cutler and B. Ramaker, "Dynamic Matrix Control - A Computer Control Algorithm," *Automatic Control Conference*, 1980.

[14] J. M. Maciejowski, *Predictive Control with Constraints*. Pearson Education Limited, 2002.

[15] D. Clarke, C. Mohtadi, and P. Tuffs, "Generalized Predictive Control-Part I. The Basic Algorithm," *Automatica*, vol. 23, no. 2, pp. 137–148, 1987.

[16] D.W. Clarke and C. Mohtadi, and P. Tuffs, "Generalized Predictive Control-Part II. Extension and Interpretations," *Automatica*, vol. 23, no. 2, pp. 149–160, 1987.

[17] W. Kwon and S. Han, *Receding Horizon Control Model Predictive Control for State Models*. Springer-Verlag London Limited, 2005.

[18] J. B. Rawlings and D. Q. Mayne, *Model Predictive Control: Theory and Design*. Nob Hill Publishing, LLC, 2009.

[19] J. Mairal, F. Bach, and J. Ponce, "Sparse Modeling for Image and Vision Processing," *Foundations and Trends in Computer Graphics and Vision*, vol. 8, no. 2-3, p. 85–283, 2014.

[20] H. Ohlsson, F. Gustafsson, L. Ljung, and S. Boyd, "Trajectory Generation Using Sum-of-Norms Regularization," *49th IEEE Conference on Decision and Control*, pp. 540–545, December 15-17, 2010.

[21] M. Nagahara and D. E. Quevedo, "Sparse Representations for Packetized Predictive Networked Control," *IFAC 18th World Congress*, pp. 84–89, August 28 - September 2, 2011.

[22] I. M. Ross, "How to Find Minimum-Fuel Controllers ," *Guidance, Navigation, and Control Conference and Exhibit*, pp. 1–10, August 16-19, 2004.

[23] M. Gallieri and J. M. Maciejowski, "Stabilising Terminal Cost and Terminal Controller for ℓasso-MPC: Enhanced Optimality and Region of Attraction ," *European Control Conference (ECC)*, pp. 524–529, July 17-19, 2013.

[24] S. K. Pakazad, H. Ohlsson, and L. Ljung, "Sparse Control Using Sum-of-Norms Regularized Model Predictive Control," *52nd IEEE Conference on Decision and Control*, p. 5758–5763, December 10-13, 2013.

[25] L. Dai, Y. Xia, M. Fu, and M. S. Mahmoud, "Discrete-Time Model Predictive Control, Advances in Discrete Time Systems Magdi Mahmoud, IntechOpen, DOI: 10.5772/51122," December 5, 2012. [Online]. Available: https://www.intechopen.com/books/advances-in-discrete-time-systems/discrete-time-model-predictive-control

[26] D. E. Seborg, T. F. Edgar, and D. A. Mellichamp, *Process Dynamics and Control*, 2nd ed. John Wiley & Sons, Inc., 2004.

[27] D. E. Kirk, *Optimal Control Theory*, ser. Dover. Dover Publications, Inc., 2004.

[28] D. S. Naidu, *Optimal Control Systems*. CRC Press LLC, 2003.

[29] E. F. Camacho and C. Bordons, *Model Predictive Control*. Springer-Verlag London Limited, 1999.

[30] L. Grune and J. Pannek, *Nonlinear Model Predictive Control Theory and Algorithms*. Springer-Verlag London Limited, 2011.

[31] C.-H. Hsieh and J.-S. Liu, "Nonlinear Model Predictive Control for Wheeled Mobile Robot in Dynamic Environment," *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pp. 363–368, July 11-14,2012.

[32] F. Kühne, W. F. Lages, and J. M. G. da Silva Jr., "Point Stabilization of Mobile Robots with Nonlinear Model Predictive Control," *IEEE International Conference on Mechatronics and Automation*, pp. 1163–1168, July 2005.

[33] F. Kühne, W.F. Lages and J.M.G. da Silva Jr., "Mobile Robot Trajectory Tracking Using Model Predictive Control," *VII SBAI/IEEE LARS*, pp. 1–7, September 2005.

[34] H. van Essen and H. Nijmeijer, "Nonlinear Model Predictive Control for Constrained Mobile Robots," *VII SBAI/IEEE LARS*, pp. 1157–1162, September 2001.

[35] *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing, author = Michael Elad, Edition = First, year = 2010, publisher = Springer Publishing Company*.

[36] R. M. D. Bras, "Adaptive Control with Sparse Models," Master's thesis, Instituto Superior Tecnico, Lisbon, Portugal, 9 2017.

[37] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[38] W. Cheney and D. Kincaid, *Numerical Mathematics and Computing*, 6th ed. Thomson Higher Education, 2008.

[39] fmincon, "fmincon." [Online]. Available: https://www.mathworks.com/help/optim/ug/fmincon.html

[40] fmincon Algorithms, “Choosing the Algorithm.” [Online]. Available: https://www.mathworks.com/help/optim/ug/choosing-the-algorithm.html#brppuoz

[41] B. A. Francis and M. Maggiore, *Flocking and Rendezvous in Distributed Robotics*. Springer International Publishing, 2016.

[42] S. M. LaValle, *Planning Algorithm*. Cambridge University Press, 2006.

A

Analytical Solution

Contents

A.1 Prediction Model

The representation from eq. (4.3) can be extended accordingly with a prediction horizon, *H*. The result will be called as the prediction model with a generalized form for the state variable, *x*:

$$\begin{aligned}
x(k+1|k) &= Ax(k|k) + Bu(k|k) \\
x(k+2|k) &= A^2x(k|k) + ABu(k|k) + Bu(k+1|k) \\
x(k+3|k) &= A^3x(k|k) + A^2Bu(k|k) + ABu(k+1|k) + Bu(k+2|k) \\
&\vdots \qquad\qquad\qquad \vdots \\
x(k+H|k) &= A^Hx(k|k) + A^{H-1}Bu(k|k) + A^{H-2}Bu(k+1|k) + \cdots + Bu(k+H-1|k)
\end{aligned} \tag{A.1}$$

By introducing the vector X which contains the set of n state variables to be considered, and the vectors $\hat{Y}$ and U which are the set of future outputs and inputs, respectively

$$\hat{Y} = \begin{bmatrix} y(k+1|k) \\ y(k+2|k) \\ y(k+3|k) \\ \vdots \\ y(k+H|k) \end{bmatrix}, \qquad U = \begin{bmatrix} u(k|k) \\ u(k+1|k) \\ u(k+2|k) \\ \vdots \\ u(k+H-1|k), \end{bmatrix}, X = \begin{bmatrix} x_1(k|k) \\ x_2(k|k) \\ x_3(k|k) \\ \vdots \\ x_n(k|k) \end{bmatrix}$$

Using the generalization of eq. (A.1), the prediction at time k is then expressed as

$$\hat{Y} = WU + \Gamma X \tag{A.2}$$

where the matrices W and Γ are defined as

$$W = \begin{bmatrix} CB & 0 & 0 & \dots & 0 \\ CAB & CB & 0 & \dots & 0 \\ CA^2B & CAB & CB & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ CA^{H-1}B & CA^{H-2}B & CA^{H-3}B & \dots & CB \end{bmatrix}, \qquad \Gamma = \begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ \vdots \\ CA^H \end{bmatrix}$$

A.2 Control Law

One way to solve the optimization problem in (4.5) is to do it manually. The objective function to be minimized, considering only the quadratic criterion, can also be written as

$$J = \sum_{i=1}^{H} \|y(k+i|k) - r(k+i|k)\|_Q^2 \;+\; \|u(k+i-1|k)\|_R^2 \tag{A.3}$$

Considering Q = R = I where I is an identity matrix, and defining $\bar{Y}$ as

$$\bar{Y} = \begin{bmatrix} r(k+1|k) \\ r(k+2|k) \\ r(k+3|k) \\ \vdots \\ r(k+H|k) \end{bmatrix},$$

eq. ($A.3$) can also be written as

$$J = (\hat{Y} - \bar{Y})^T(\hat{Y} - \bar{Y}) + U^T U \tag{A.4}$$

By substituting the vector-matrix notation of $\hat{Y}$ in eq. (A.2) to (A.4), the following expression is obtained:

$$J = (WU + \Gamma X - \bar{Y})^T(WU + \Gamma X - \bar{Y}) + U^T U \tag{A.5}$$

To simplify the equation, let F = Γ X - $\bar{Y}$ since these vector parameters are not dependent to U . Afterwards, expand the expression to get

$$\begin{aligned} J &= (WU + F)^T(WU + F) + U^T U \\ J &= (W^T U^T + F)(WU + F) + U^T U \\ J &= W^T U^T WU + W^T U^T F + WUF^T + F^T F + U^T U \end{aligned} \tag{A.6}$$

Simplifying the last expression in ($A.6$) to get

$$J = U^T(W^T W + I)U + 2UWF^T + F^T F \tag{A.7}$$

Let M = W^T W + I since again these vector expressions do not depend on U, the quadratic cost function of eq. (A.7) can be rewritten as

$$J = U^T MU + 2UWF^T + F^T F \tag{A.8}$$

In order to find the optimal value of U, compute the gradient of eq. (A.8) with respect to U and equate the result to zero

$$\nabla_U J = \frac{\partial J}{\partial U} = 2MU + 2W^T F = 0 \tag{A.9}$$

The control giving the minimum cost is therefore:

$$U^* = -M^{-1}W^T F \tag{A.10}$$

Only the first element of U^* will be applied for the current state, so from ($A.10$)

$$u(k|k) = -[1\ 0\ 0. \ldots . 0]M^{-1}W^T(\Gamma X - \bar{Y}) \tag{A.11}$$

Afterwards, the whole process is repeated at the next time interval in a receding horizon fashion obtaining a new set of control inputs.