COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

GRADO EN INGENIERÍA EN TECNLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

DEVELOPMENT OF AN MPC ALGORITHM TO OPTIMALLY COACH HEAVY-DUTY TRUCK OPERATORS

Autor: José Mosquera Abellás

Director: Hosam K. Fathy

College Park, Maryland

June, 2020

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

“*DEVELOPMENT OF AN MPC ALGORITHM TO OPTIMALLY COACH HEAVY-DUTY TRUCK OPERATORS*”

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2019/20. es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.

PROYECTO RELIZADO POR EL ALUMNO

José Mosquera Abellás

Fdo.: Fecha: 03/ 06 / 2020

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Hosam K. Fathy

Fdo.: Fecha: 03/ 06/ 2020

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

GRADO EN INGENIERÍA EN TECNLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

DEVELOPMENT OF AN MPC ALGORITHM TO OPTIMALLY COACH HEAVY-DUTY TRUCK OPERATORS

Autor: José Mosquera Abellás

Director: Hosam K. Fathy

College Park, Maryland

June, 2020

ACKNOWLEDGEMENTS:

To my parents and family, for all the support they have given me during these 4 years of university; for encouraging me to keep going during hard times.

To all my friends, both from ICAI and the ones I made during my stay in the University of Maryland, for all the good moments we lived together and the help they gave me.

To my director Hosam, for giving an excellent mentorship and showing me how engineering research actually works.

Thank you.

DESARROLLO DE UN ALGORITMO DE CONTROL PREDICTIVO POR MODELO PARA ENTRENAR ÓPTIMAMENTE A LOS CONDUCTORES DE CAMIONES PESADOS.

Autor: Mosquera Abellás, José.
Director: Fathy, Hosam K.
Entidad Colaboradora: University of Maryland, College Park.

RESUMEN DEL PROYECTO

El consumo de combustible es uno de los aspectos más investigados en la industria automovilística, pues tiene especial impacto en las emisiones de gases contaminantes, así como en el coste de operación del vehículo, especialmente en vehículos pesados. Una manera de optimizar el consumo es usar un asistente de conducción que indique al usuario la velocidad a la cual el consumo es mínimo. De esta manera se logra reducir en gran medida los acelerones/frenazos ineficientes que realizan muchos conductores, los cuales aumentan significativamente el consumo.

Una forma de crear dicho asistente es mediante el uso de un algoritmo de control predictivo por modelo (MPC) que incorpore un modelo del conductor del vehículo para realizar sus predicciones. Desarrollaremos un modelo de conductor para realizar predicciones sobre la velocidad futura del vehículo y lo incorporaremos en un algoritmo de control predictivo por modelo. Dicho algoritmo servirá de base para la creación de un asistente de conducción que ayudará a reducir el consumo de combustible.

Finalmente evaluaremos el comportamiento del algoritmo ante diferentes situaciones de conducción en las cuales el camión debe seguir una velocidad de consumo óptima determinada, teniendo en cuenta las limitaciones físicas del vehículo así como los límites de velocidad de la carretera.

Palabras clave: Control predictivo, Asistente de conducción, Modelo de conductor, Consumo óptimo.

1. **Introducción:**

   Téngase en cuenta que el combustible supone un 29% del coste total de un camión pesado a lo largo de su vida útil (SCHI03), lo que supone el mayor coste de todos los que soporta un vehículo pesado (ATRI16). Dicho consumo de combustible también es el causante del 7% de las emisiones de gases de efecto invernadero en EE. UU. (EPA_19).

   A la hora de reducir el consumo de combustible se suele pensar en la automatización de la conducción como herramienta para lograr consumos óptimos, sin embargo existe un paso intermedio entre la automatización y el control completamente manual. Nos referimos a la conducción asistida, que puede lograr reducir el consumo de un vehículo sin llegar a automatizarlo completamente.

   El principal objetivo del proyecto es crear un algoritmo de control predictivo que incorpore un modelo del conductor que sirva de base para crear un hipotético asistente de conducción. Para ello utilizaremos la técnica de control predictivo por modelo

(MPC). El vehículo que se toma como referencia para desarrollar el algoritmo es un camión de mercancía pesado. Además, contaremos con el perfil de velocidad que optimiza el consumo de combustible en el camión.[1]

2. Definición del Proyecto:

Este proyecto forma parte de uno mucho mayor, desarrollado en diversas instituciones a lo largo de EE. UU. cuyo objetivo es reducir el consumo de combustible en vehículos pesados mediante la optimización de diversos problemas: trayectoria-velocidad, cambio de marchas, mezcla de combustible…El presente trabajo se centra únicamente en el desarrollo de un algoritmo para el asistente de conducción utilizando control predictivo basado en modelo.

El control predictivo basado en modelo es un proceso iterativo en el que se realizan sucesivas optimizaciones en un horizonte de tiempo finito (T). Dicho horizonte está dividido en *k* intervalos, cada uno de ellos de longitud *h*. Tomando como punto de partida el tiempo *t*, el algoritmo busca la acción de control óptima para todo el período [*t*, *t*+T] pero solamente se aplica la acción correspondiente al primer intervalo (*k*=1) y se calcula el estado del sistema en ese instante. Con la información del estado del sistema en dicho momento se repite el proceso anteriormente mencionado.

3. Descripción del algoritmo:

Para la creación del algoritmo, comenzaremos por formular un modelo de conductor en el que:

1. Las correcciones a la velocidad del vehículo se hagan a un ritmo proporcional a la diferencia entre la velocidad real del vehículo y la velocidad recomendada por el asistente de conducción.
2. Cuando se alcance el régimen permanente, el conductor seguirá perfectamente a la velocidad del asistente de conducción, lo que significa que la velocidad real del vehículo y la del asistente son iguales.

Dicho modelo se incorpora en el algoritmo de control para predecir la velocidad real del vehículo. El Sistema resultante se puede modelar de la siguiente forma:

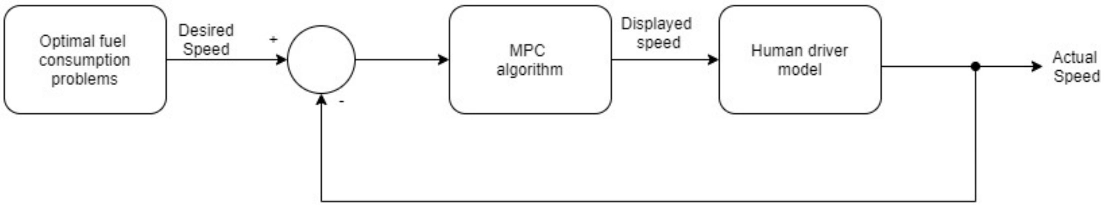

Figura 1:Modelo del sistema

El algoritmo de optimización MPC recibe como entrada el perfil de velocidades que minimiza el consumo de combustible así como la velocidad real del vehículo en el instante anterior, y devuelve como salidas la velocidad real del vehículo, junto con la

[1] Usaremos datos sintéticos por motivos de confidencialidad en el proyecto, VOLVO no nos permite utilizar los perfiles de velocidad reales. Esto no altera el desarrollo del algoritmo de control ni la funcionalidad de este, simplemente se mostrará su funcionamiento para otro perfil de velocidad.

velocidad que deberá mostrar el asistente de conducción y la acción de control para alcanzar dicha velocidad. El algoritmo se ha programado en MATLAB.

4. Resultados:

Tras ejecutar el algoritmo para un determinado perfil de velocidad podemos comprobar como su funcionamiento es el esperado. Se cumplen las dos premisas que habíamos puesto a nuestro modelo de conductor, obteniendo un perfecto seguimiento de la velocidad deseada. Esto se puede observar en las siguiente figuras.

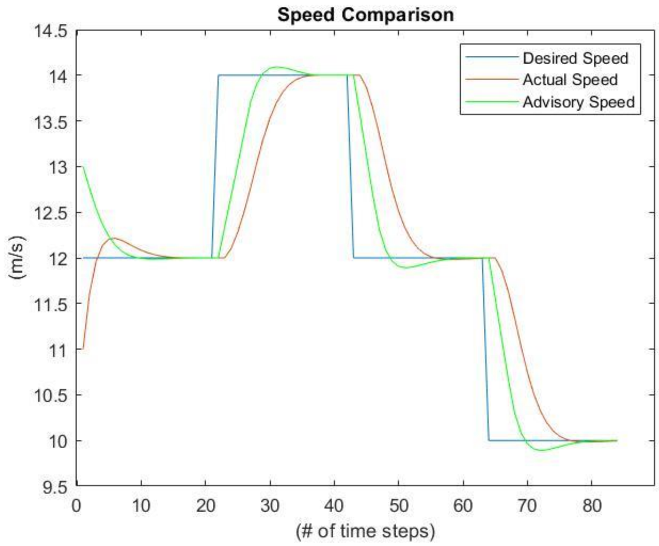

Figura 2: Resultado de la optimización (1).[2]

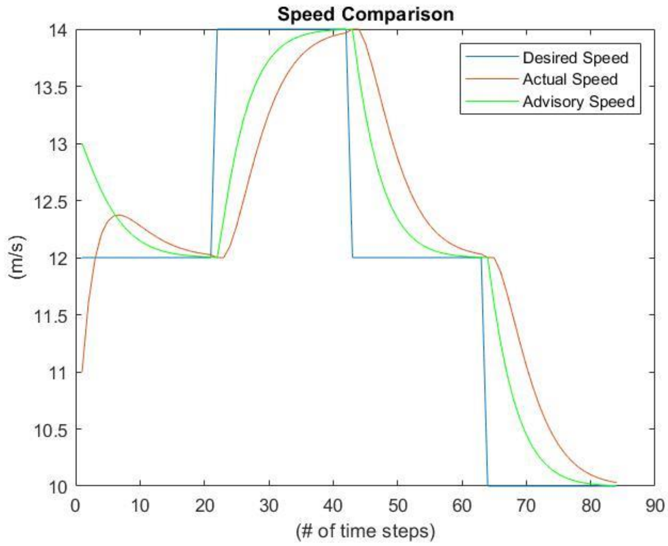

Figura 3: Resultado de la optimización (2).[2]

La Figura 2 corresponde a una optimización en la que se ha primado el seguimiento a la velocidad deseada frente a las acciones de control. Sin embargo, la Figura 3

[2] Desired speed= Velocidad deseada; Actual speed= Velocidad actual; Advisory speed= Velocidad del asistente de conducción; (# of time steps)=(nº de intervalos de tiempo)

corresponde a una optimización donde priman las acciones de control suaves frente al seguimiento de la velocidad deseada.

5. Conclusiones:

Tras realizar el Proyecto concluimos que el control predictivo por modelo es una técnica útil para la creación de un algoritmo que servirá de base para un asistente de conducción. Permite además modificar determinados parámetros para adaptar el algoritmo a diferentes estilos de conducción, en función de la persona que esté al volante del vehículo.

Una persona con menos experiencia necesitará una acción de control más suave para facilitar la conducción, sin embargo una persona con más experiencia al volante puede responder mejor frente a acciones de control más bruscas.

El algoritmo deberá ser probado en un entorno real, mediante su incorporación en un asistente de conducción en un vehículo para evaluar su rendimiento en situaciones no simuladas. Se deja una puerta abierta al refinamiento del modelo de conductor mediante el uso de técnicas más avanzadas.

6. Referencias:

[ATRI16] American Transportation Research Institute (ATRI), "An Analysis of the Operational Costs of Trucking: 2016 Update", Tech. Rep., 2016.

[EPA_19] U.S. Environmental Protection Agency, "U.S. Transportation Sector Greenhouse Gas Emissions'', 2019, https://nepis.epa.gov/Exe/ZyPDF.cgi?Dockey=P100WUHR.pdf.

[SCHI03] Schittler M., "State-of-the-art and emerging truck engine technologies for optimized performance, emissions and life-cycle costing", 9th Proc. Diesel Engine Emission Reduction Conference, Newport, 2003.

DEVELOPMENT OF AN MPC ALGORITHM TO OPTIMALLY COACH HEAVY-DUTY TRUCK OPERATORS

Author: Mosquera Abellás, José.
Supervisor: Fathy, Hosam K.
Collaborating Entity: University of Maryland, College Park.

ABSTRACT

Fuel consumption is one of the most researched aspects in the automotive industry, since it has a huge impact in greenhouse gas emissions, as well as in the cost of operation of a vehicle, especially in heavy-duty vehicles. One way of optimizing fuel consumption is to use a driver advisory that indicates the speed at which fuel consumption is minimum to the driver. By doing this, we can reduce the number of times the driver accelerates/brakes in a non-efficient manner that are really common among drivers. This kind of practices significantly increase fuel consumption.

One way of creating a driver advisory is by using a model-based predictive control (MPC) that use a model of a human driver to make its predictions. We will develop a human driver model to make predictions about the future speed of the vehicle and we will incorporate it into a model predictive control algorithm. That algorithm will be the foundation of a driver advisory to help reduce fuel consumption.

Finally, we will evaluate the behavior of the algorithm in different driving situations in which the truck must follow a given optimal fuel consumption speed, taking into account the speed limit, as well as the limits of the truck.

Keywords: Predictive control, Driver advisory, Driver model, Optimal consumption.

1. Introduction:

Keep in mind that fuel is the largest share of total costs (ATRI16) in heavy-duty trucks, with a share of more than 29% of its total cost through all its lifetime (SCHI03). That fuel consumption is also responsible for 7% of GHGs emissions in the US.

When researching about reducing fuel consumption, one of the first solutions that comes to mind is vehicle automation as a tool to reach optimal consumes. However, there is an intermediate step between automation and manual control. This solution is assisted driving systems, which are capable of reducing fuel consumption of a vehicle without fully automating it.

The main goal of the project is to create an algorithm using predictive control that incorporates a human driver model, which will serve as the foundation for a driver advisory system for a heavy-duty truck. To do that we will use Model Predictive

Control (MPC). Furthermore, we will have the speed profile that minimizes fuel consumption of the truck as an input. [3]

2. Project definition:

This project is a small piece of a bigger one, which is being conducted in several institutions across the US, whose objective is to reduce fuel consumption in heavy-duty vehicles by solving several optimization problems: speed trajectory optimization, gear shift optimization, powertrain optimization… This work has its focus on developing an algorithm for the driver advisory using model predictive control.

MPC is an iterative process in which successive optimizations are performed in a finite time horizon (T), divided into *k* time steps, each one of them with a length of *h*. Taking time *t* as the starting point, the algorithm will find the optimal control action for the time horizon [t, t+T]. The control action corresponding to the first time step (*k=1*) is implemented and feedback from the state of the plant at time instant *t+h* is received. Based on this new state, predicted using the plant model, a new optimal control action is calculated following the same process as before, moving the time horizon one step forward. This process is repeated continuously to obtain the optimal control action at each instant, accounting for the real behavior of the system using a model.

3. Algorithm description:

To develop the algorithm we start by formulating a human driver model in which:

1. Corrections to the vehicle speed are made at a rate that is proportional to the difference between the actual speed of the vehicle and the speed displayed on the driver advisory.
2. When steady state is achieved, the driver will track the advisory perfectly, meaning that the advisory speed should be equal to the actual speed of the vehicle.

That model is incorporated into the algorithm to predict the real speed of the vehicle. The resulting system is represented in the following figure:

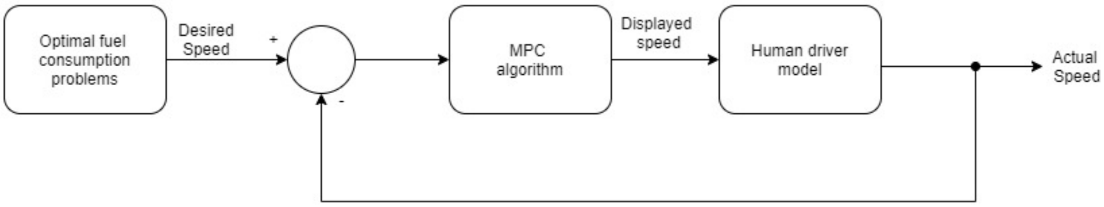

Figure 1:Model of the system

The MPC optimization algorithm gets the speed profile that minimizes fuel consumption and the actual speed of the vehicle in the previous instant as its inputs.,

[3] We used synthetic data instead of the real data obtained by other group members due to privacy issues. This does not modify the development of the control algorithm nor its functionality, the only consequence is that we will see its performance for a different speed profile.

and returns the real speed of the vehicle, the driver advisory speed and the control actions to reach that speed as its outputs. The algorithm has been coded in MATLAB.

4. Results:

After running the algorithm for a specific speed profile we can observe how its performance is what we first expected. Both requirements of the human driver model are satisfied, obtaining a perfect tracking of the desired speed (minimum consumption speed profile). We can see that in the following figures:

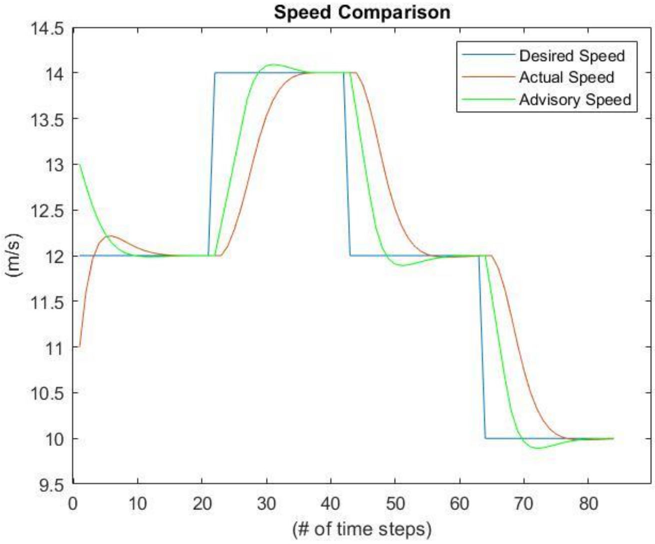

Figure 2:Optimization result (1)

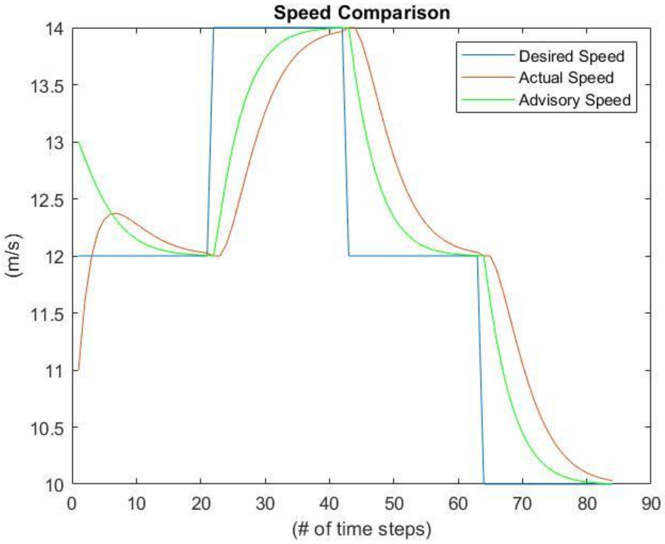

Figure 3: Optimization result (2)

Figure 2 corresponds to an optimization in which more importance was given to the desired speed tracking than to the smoothness of the control actions. On the other hand, Figure 3 corresponds to an optimization in which the smoothness of the control action was the priority, rather than desired speed tracking.

5. Conclusions:

After finishing the Project we conclude that model predictive control is a useful technique to create an algorithm to serve as the foundation for a driver advisory system. Furthermore, it gives enough freedom to modify certain parameters depending on the driving style of whoever is on the wheel.

A driver with less experience behind the wheel will require a smoother control action in order to easy the driving. However, a more experienced driver will we able to follow more abrupt control actions.

The algorithm should be tested in a real environment, through its incorporation in a driver advisory system in a heavy-duty truck to evaluate its performance in non-simulated scenarios. There is still room for refinement in the human driver model through more advanced techniques.

6. References:

[ATRI16] American Transportation Research Institute (ATRI), "An Analysis of the Operational Costs of Trucking: 2016 Update", Tech. Rep., 2016.

[EPA_19] U.S. Environmental Protection Agency, "U.S. Transportation Sector Greenhouse Gas Emissions'', 2019, https://nepis.epa.gov/Exe/ZyPDF.cgi?Dockey=P100WUHR.pdf.

[SCHI03] Schittler M., "State-of-the-art and emerging truck engine technologies for optimized performance, emissions and life-cycle costing", 9th Proc. Diesel Engine Emission Reduction Conference, Newport, 2003.

TABLE OF CONTENTS

Chapter 1. Introduction 5
1.1. Motivation: 5
1.2. Objectives: 6
1.2.1. Project goals: 6
1.2.2. Sustainable development goals: 7
1.3. Work Methodology 9
1.4. Tools and resources: 10
Chapter 2. State of the art 11
2.1. Driving technologies: 11
2.1.1. Advanced driver-assistance systems (ADAS): 12
2.1.2. Automated driving: 13
2.2. Model predictive control (MPC) systems: 13
Chapter 3. Model Predictive Control (MPC) 15
3.1. Fundamentals of MPC: 15
3.2. MPC vs PID: 16
Chapter 4. Human driver model 19
4.1. Existing models 19
4.2. Model formulation: 19
Chapter 5. Problem Formulation 23
5.1. Conventional/explicit formulation 23
5.1.1. Variable definition 23
5.1.2. Objective function specification 24
5.1.3. Constraint specification 25
5.2. MATLAB translation 26
Chapter 6. MATLAB MPC code 31
6.1. Step 1: MPC function 31
6.2. Step 2: Main script 32
Chapter 7. Parameter definition and simulation 33
7.1. Simulation 1: Acceleration to constant speed: 36

7.2. Simulation 2: Deceleration to constant speed: 38
7.3. Simulation 3: Series of steps: 39
7.3.1. High value of *r*: 40
7.3.2. Low value of *r:* 41
Chapter 8. Result analysis 43
Chapter 9. Conclusions and future works 47
Chapter 10. Bibliography 49
Chapter 11. Annex 51
11.1. Annex 1 51

List of figures:

Figure 1:Model of the system 14

Figure 2:Optimization result (1) 15

Figure 3: Optimization result (2) 15

Figure 4: Model Predictive Control strategy (BEHR20) 16

Figure 5: Control action for a high value of r 35

Figure 6: Control action for a small value of r 35

Figure 7: Speed comparison 37

Figure 8: Optimization results 37

Figure 9: Speed comparison 38

Figure 10: Optimization results 39

Figure 11: Speed comparison (high r) 40

Figure 12: Optimization results (high r) 40

Figure 13: Speed comparison (low r) 41

Figure 14: Optimization results (low r) 41

Figure 15: Rate of change of the advisory speed (small r) 44

Figure 16: Rate of change of the advisory speed (high r) 44

List of tables:

Table 1:Main SDGs of the project.. 8

Table 2:Computational time comparison 34

Chapter 1. Introduction

Transportation is evolving towards automation and the removal of fossil fuels as a source of power, by using alternative power sources like electricity instead. We find an intermediate step in this process in the concept of assisted/connected driving. Most of the current assisted driving technologies focus on reducing the chances of having a traffic accident: automatic emergency breaking, lane keeping systems, traffic jam assistants among others; or in facilitating some particular tasks such as parking assistants, adaptative lighting or rear-view video systems. As we can see none of these technologies have minimizing fuel consumption as a goal. In addition, the majority of these technologies are mainly developed to be implemented in conventional cars, leaving heavy-duty trucks as a secondary target. Furthermore, the majority of these technologies do not take into account the fact that humans are not a perfectly timed machine, they have delays and make mistakes when performing tasks.

In this project we will develop an algorithm that will serve as the foundation of a driver advisory system to effectively coach heavy-duty truck operators in an attempt to significantly reduce fuel consumption, incorporating a human driver model. These savings will have an impact not only on the truck operator expenses (fuel money) but also in the environment as a whole as a result of the reduction in CO2 emissions and other greenhouse gasses. According to the American Transportation Research Institute, fuel is the largest share of total costs (ATRI16) in heavy-duty trucks, with a share of more than 29% of its total cost through all its lifetime (SCHI03).

1.1.Motivation:

This Project is just a small piece of a larger one, the ARPA-E NEXT-Generation Energy Technologies for Connected and Automated on-Road Vehicles (NEXTCAR) Program, that has minimizing fuel/energy consumption has its goal, by increasing vehicle connectivity while using vehicle-to-vehicle (V2V), vehicle-to-infrastructure (V2I) and vehicle-to-everything (V2X) communication, that started in 2016. Many researchers from different institutions across the US (General Motors, Volvo Trucks, Pennsylvania State University,

University of Michigan…) have been working in this issue for years. Each one of them had a different task in the project: integrated vehicle power and thermal management, speed-trajectory optimization, vehicle platooning, engine cylinder optimization etc.

In particular, my branch of the project includes heavy-duty truck study and optimization to reduce fuel/energy consumption. More precisely the development of an MPC algorithm to coach the operator of the truck through a driver advisory that incorporates a human driver model to make its predictions.

This approach has never been taken before and it has a huge potential to increase fuel savings in heavy-duty truck since humans are responsible for a large amount of wasted energy/fuel when driving, as a result of unnecessary acceleration/deceleration, high speed, unnecessary lane switching or sub-optimal gear shifting. By coaching the driver through an advisor we could avoid most of this energy/fuel waste since they are making optimal decisions and obtaining close-to-optimal results. We should not forget that humans are not perfect-timed machines that react instantly to what they are asked to do, so our advisor should have a human driver model that accounts for that uncertainty.

The purpose of this project is to develop an effective advisor using MPC that could be incorporated to the main body of the NEXTCAR project. If we obtain a satisfactory result, the intention is to file a patent and commercialize the product combined with the rest of the findings made by the team. Keep in mind that this project is just a first approach to the problem in a limited time span. A door will remain open to further development and refinement of the results obtained with more complex techniques and models.

1.2.Objectives:

1.2.1. Project goals:

The main objective of the project is to develop a functional algorithm that serves as the foundation for a driver advisory system to coach heavy-duty truck operators. We will try to prove that MPC techniques are useful to create this type of algorithms, implementing a human driver model.

The second objective is to develop a human driver model to help the algorithm make predictions about the future speed of the vehicle. The model should account for the time delay in the response of the driver on its predictions.

The third goal is to prove that the implementation of the human driver model into the algorithm produces a satisfactory output when tracking specific speed profiles.

1.2.2. Sustainable development goals:

We will now analyze some of the Sustainable Development Goals (SGDs) defined by the United Nations (UN) (UN 15) that are pursued within the scope of this project. In the following table we can see a summary of these goals.

SDG DIMENSION	SDG identified	Role	Goal
Biosphere	SDG 13: Take urgent action to combat climate change and its effects (Acknowledging that the United Nations Framework Convention on Climate Change is the primary international, intergovernmental forum for negotiating the global response to climate change).	Secondary	Improve education, awareness-raising and human and institutional capacity on climate change mitigation, adaptation, impact reduction and early warning. (13.3)
Society	SDG 11: Making cities and human settlements inclusive, safe, resilient, and sustainable.	Secondary	By 2030, provide access to safe, affordable, accessible, and sustainable transport systems for all, improving road safety, notably by expanding public transport, with special attention to the needs of those in vulnerable situations, women, children, persons with disabilities and older persons. (11.2)

SDG DIMENSION	SDG identified	Role	Goal
Economy	SDG 8: Promote sustained, inclusive, and sustainable economic growth, full and productive employment, and decent work for all.	Primary	Improve progressively, through 2030, global resource efficiency in consumption and production and endeavor to decouple economic growth from environmental degradation, in accordance with the 10-year framework of programs on sustainable consumption and production, with developed countries taking the lead. (8.4)

Table 1:Main SDGs of the project

The main SDG that this project will pursue will be improving resource consumption efficiency in an attempt to detach economic growth from environmental degradation. Heavy-duty trucks are a key link in the supply and distribution chain for food and cargo as well as in other economic sectors like construction works. Without them it would be impossible to have food delivered to grocery stores, fuel in the gas stations or material in the construction sites. Given that, we can confirm that without them, economic activity in the country will be considerably low.

On the other hand, according to Energy Information Association's Annual Energy Outlook published in 2020, medium- and heavy-duty trucks fuel consumption accounts for 25.9% of the US total fuel consumption. That amount is expected to grow by a 4% through 2050 as a result of the increase in economic activity (EIA_20). They are also responsible for more than 7% of Greenhouse Gasses (GHG) emissions in the US (EPA_19).

The secondary SDGs for the project are improving human and institutional capacity on climate change mitigation as well as providing sustainable transportation systems for all. This project is an example of sustainable transportation technology that could be used by governments and implemented in public transportation systems to have a massive impact in climate change by significantly reducing GHGs emissions (one of the main causes of climate change).

If the output of the project is satisfactory, public institutions as well as individuals could benefit from it, providing a way to reduce pollution and fight climate change while they obtain a superior driving experience. If we apply this technology to public transportation vehicles, such as public city buses, we could achieve a significant reduction in GHGs emitted by the city public transportation system, resulting in a more sustainable city model. Everyone in the city would benefit from this measure since a reduction of GHGs to the atmosphere will significantly improve overall air quality.

1.3.Work Methodology

To solve the problem, we started by conducting thorough research in the subject of the project, given that no previous knowledge in MPC is possessed by the student. This research had familiarizing the student with MPC and other related optimizing processes as its main goal. At the same time, a detailed reading of all the previous work related to the topic of the project took place (NEXTCAR papers, automated driving papers, human driver model papers, etc.).

After the required knowledge was obtained, the project itself started. The first step was to create a functional human driver model to further implement it into the algorithm. To formulate the driver model, two objectives were set at first. By knowing what the model should do we were able to successfully formulate it.

After the model was created, an explicit formulation of the optimization took place. In this formulation we relied on a classic optimization problem structure, to later adapt it to our particular project and techniques. The human driver model was incorporated into the formulation and the MPC strategy was applied.

Once the problem was formulated, it was time to code the algorithm using MATLAB. In this step the algorithm itself was created. It required the adaptation of the explicit formulation into MATLAB language which was quite challenging.

Finally, several tests were conducted using the algorithm to evaluate its performance and check its functioning. After this testing phase, some corrections were incorporated into the algorithm to obtain its final version.

1.4. Tools and resources:

The primary tool that will be used through the course of the project will be MATLAB. MATLAB is a programming platform specially designed for engineers and scientists that allows the user to analyze data, develop algorithms and create models. We will rely on MATLAB to write the code needed for our algorithm, as well as to solve the iterative optimization problems needed for MPC. MATLAB has a specific toolbox to create and use MPC algorithms, but we will manually create the algorithm code, so we are not limited to the options given by that toolbox and have more design freedom in case we need it. MATLAB is available to us thanks to the University of Maryland (College Park, U.S.A.) student license, as well as to the license provided by the Universidad Pontificia de Comillas (Madrid, Spain).

We will also use the web page www.draw.io to create the diagrams and charts to be used in the project. Draw.io is a really powerful tool with a wide variety of templates that are useful to create diagram and other visual representations and models. This service is available for free to general public.

To edit and write all the annexes and the final thesis we will be using Microsoft Word, one of the most common word processor.

Another resource to be used in the project is an optimal speed (as a function of time) data set previously calculated by other members of the team using specific optimization techniques. This data is the result of solving a speed-trajectory optimization problem for heavy-duty trucks. We will use this data as an input to our MPC algorithm so it can be used to calculate the optimal speed to be displayed in the driver advisor.

Chapter 2. State of the art

2.1.Driving technologies:

First of all, we will define the different levels of driving automation, based on the standards J3016 elaborated by the Society of Automotive Engineers (SAE) (SHUT19). According to the SAE there are six different levels of automation, being 'Zero' the lowest and 'Five' the highest. The first three levels refer to driver support features, meaning that there has to be someone driving at all time, while levels 3 to 5 refer to automated driving features (there may not be a driver operating the vehicle).

- Level Zero (No automation): the vehicle is fully operated by the driver, with just a few momentary assistances or warnings.
- Level One (Driver assistance): there are some aid features such as steering or pedal support.
- Level Two (Partial Automation): aid is provided by having steering and pedal support.
- Level Three (Conditional Automation): the vehicle can drive itself when certain conditions are met, but there needs to be a driver when required by the support feature.
- Level Four (High Automation): In this level there is no requirement for the driver to take over the control of the vehicle, the vehicle can drive by itself when certain conditions are met.
- Level Five (Full Automation): This is the highest level of automation. There is no need for a driver to control the vehicle like in level 4, but the vehicle can drive by itself under all conditions.

The algorithm that will be developed in this project belongs to the first subcategory of automation levels: Level Zero to Two, more precisely it can be classified as Level Zero automation, where the vehicle is fully operated by the driver but with some assistance.

2.1.1. Advanced driver-assistance systems (ADAS):

This term refers to all those systems that have assisting the driver in an attempt to save fuel or prevent traffic accidents as its main goal, making the driving experience easier and safer for the driver. The development of this type of technologies began in the late 70s of the 20th century with the introduction of Anti-lock Braking System (ABS) into a serial production (ZIEB17). We will know highlight the main types of ADAS that are being used nowadays.

2.1.1.1. Cruise Control:

One of the main technological solutions to decrease fuel consumption on heavy-duty vehicles is Cruise Control System. Conventional Cruise Control (CCC) is the electronic system by which the speed of a given vehicle is held constant at a desired level stipulated by the driver. This technology was invented by Ralph Teetor in 1950s and was first implemented on a production vehicle in 1958, as an extra feature in a Chrysler car (SEAR18), even though it was not put into mass production until mid-80s (TATU03). Several improvements have been made to this technology ever since, being Adaptative Cruise Control (ACC) the most remarkable. ACC is considered one type of ADAS, while CCC is not.

ACC is a term used to define a wide variety of improvements made to the regular CCC, such as keeping a particular distance with the vehicle ahead or automatically limiting the speed as a result of the speed limit of a given road. ACC is classified as a level 1 automation according to the SAE standards.

2.1.1.2. Collision Avoidance System (CAS)

This system uses laser/radars and video cameras facing the upcoming road to predict imminent accidents and alert the driver in a timely manner to avoid the collision. They are also known as precrash systems. They are capable of decelerating the vehicle and apply automatic braking as well as pretensioning the seatbelt in order to avoid a collision or reduce the intensity of it. They also incorporate alarms to alert the driver of an imminent accident so he can act and prevent it. All these features vary depending on the manufacturer of the particular vehicle (SHAO11).

Blind spot information systems detect other vehicles that are approaching the main vehicle on their side or rear and gives a visual alert to the driver, so it is aware of the situation. Parking assistants help the driver perform the optimal movements to fit the car in a narrow parking spot. They can either do this through guidance or even by taking control of the vehicle.

2.1.2. Automated driving:

Another immediate solution to the problem we formulate is fully automated driving technologies. This means reaching Level Five, according to the SAE standards. When full automation is reached there is no room for human errors or bad driving-related behaviors (i.e. unnecessary breaking or accelerating, unnecessary lane switching…). An optimal fuel consumption can be reached by implementing automated driving since the vehicle will respond perfectly to the commands given by the controller, the speed, trajectory, gear shifting among other driving decisions will be as efficient as possible in order to reduce fuel consumption.

The downside about this option is that there is currently no production vehicle that incorporates this level of automation . There are some concept vehicles (BMW_20) that are able to exploit this kind of technology, but they are not available to general public yet. There are several reasons to explain this fact, such as governments laws/policies that do not allow these vehicles on the streets, there are no clear insurance terms regulating them, the current infrastructure is not valid to support them and also the confidence level that people have in this kind of vehicles may be a fact to keep in mind.

2.2.Model predictive control (MPC) systems:

There are many applications nowadays where we can see MPC strategies in action. MPC is a control strategy that is widely spread on the industry, with an intensive use in petrochemical plants. These plants use MPC to compute, in real time, the appropriate mix of substances that should be used in a specific process. It is also widely used to control multivariable processes, such as distillation columns or HVAC (heating, ventilation, and air conditioning). We can also see some examples of MPC in optimal trajectory problems, such as optimal trajectories for drones or autonomous vehicles. (CAIR18)

In Chapter 3 we explain the MPC control strategy to give the required knowledge to the reader.

Chapter 3. Model Predictive Control (MPC)

This chapter introduces the reader to Model Predictive Control (MPC) to provide a general idea of this technique and a knowledge base to understand the following work. MPC is used in a wide variety of multivariable control applications such as chemical plants control, spacecraft control or powertrains control for automobiles, as explained in the previous section.

3.1.Fundamentals of MPC:

MPC is a control technique that chooses the control action by repeatedly solving an optimal control problem, to minimize a particular cost function (*J*), over a moving time horizon. This optimization problem is subject to constraints in its inputs and outputs (ZHAN10). These constraints take into account the internal dynamics of the plant and its physical limitations. It is a form of online optimization; this means that the control actions are based on inputs that are not know in the beginning of the process but are given as the optimization process runs. Only the initial state of the system is given, and the subsequent states are predicted using a model of the plant. On the other hand we find offline optimization techniques , in which all the inputs are given before the optimization process starts, so there is no need for prediction of future state.

MPC is an iterative process in which successive optimizations are performed in a finite time horizon (T), divided into *k* time steps, each one of them with a length of *h*. Taking time *t* as the starting point, the algorithm will find the optimal control action for the time horizon [t, t+T]. The control action corresponding to the first time step (*k=1*) is implemented and feedback from the state of the plant at time instant *t+h* is received. Based on this new state, predicted using the plant model, a new optimal control action is calculated following the same process as before, moving the time horizon one step forward. This process is repeated continuously to obtain the optimal control action at each instant, accounting for the real behavior of the system using a model.

A graphic explanation of the MPC process can be found in Figure 4, where the control strategy is illustrated:

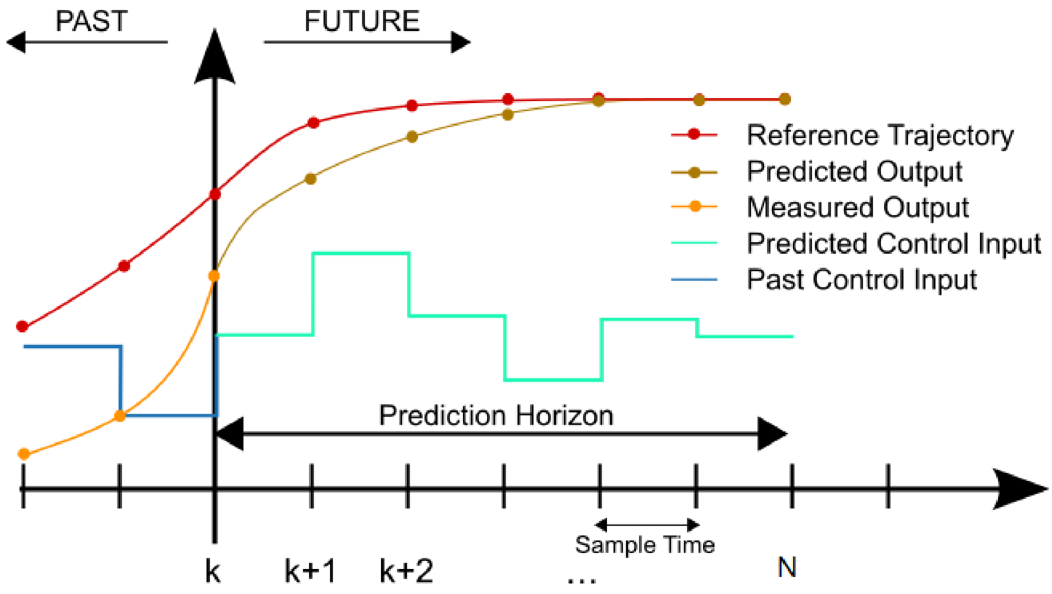

Figure 4: Model Predictive Control strategy (BEHR20)

At instant t=*k*, the control is computed for all the prediction horizon, until t=*T*. This means that the control action is calculated for N time steps. Once the optimal control is calculated, only the first step of the optimal control action is applied, and the time window is moved one step forward, causing the point t=*k+1* to be the new starting point. This process is repeated continuously.

3.2.MPC vs PID:

When selecting a control strategy we encounter different options to tackle the problem we are facing. The immediate alternative that comes to mind when facing this problem is a PID control, in which a proportional, an integral and a derivative control are combined together to achieve the desired objective. This control strategy is one of the most popular among control problems, but it does not fit the requirements of this particular problem.

A PID is an offline control technique in which the optimal control action is calculated at the current time for the remaining time, so it does not account for the possible deviation in the real behavior of the system. Furthermore, it requires a fine tuning of the control parameters to obtain accurate results. On the other hand, using MPC allows the problem to consider the deviations in the real output of the plant by constantly calculating the optimal control action for each time step (online control). In addition, it does not require as much parameter tuning as a PID control.

Chapter 4. Human driver model

In this chapter we conducted an analysis of the existing driver models that are frequently used in the industry for similar applications, and then we formulated our own human driver model. The model we formulated is specifically designed to satisfy a series of requirements we established.

4.1.Existing models

Human driver models are usually divided into longitudinal and lateral control models and formulated as separate units. In our case we focused only on longitudinal control models. By reviewing the literature (DELI07, PROK01) we found that most longitudinal control models are based on car following model. These models are based on the principle of maintaining a certain distance between the controlled car and the leading vehicle, taking into account the velocity of the car. Modeling the human behavior is a highly difficult task due to the dynamics involved in the process. Humans often act in a random, non-predictable way that requires multiparametric models with great complexity. One example of this can be found on (GUO_13), which combines both longitudinal al lateral control incorporating desired path perception and the internal dynamics of the vehicle.

4.2.Model formulation:

The models explained above will be a good to implement in our problem, but for the sake of simplicity and to verify whether taking an MPC approach to the problem is feasible, we will use a simpler one. To create the model, we used a state space model representation. A state space model reflects the existing relationship between the current state of the system and the predicted future state and output. The general formulation for a state space model with feedback, in continuous time, is as follows:

$$\dot{x}(t) = A \cdot x(t) + B \cdot u(t) \qquad \textit{Eq. 1}$$

$$\dot{y}(t) = C \cdot x(t) + D \cdot u(t)$$

In order to incorporate a state space model into the MPC algorithm we need to discretize Eq.1 as follows:

$$\mathrm{x}(k+1) = A_d \cdot \boldsymbol{x}(k) + B_d \cdot u(k) \qquad \textit{Eq. 2}$$

$$\mathrm{y}(k+1) = C_d \cdot x(k) + D_d \cdot u(k)$$

Where:

x (*k* + 1) is the state vector of the system at instant *k+1*.

x (*k*) is the state vector of the system at instant k.

u (*k*) is the control/input vector of the system at instant *k*.

y (*k* + 1) is the output vector of the system at instant *k+1*.

Ad is the matrix that models the relationship between current and previous state.

Bd is the matrix that models the relationship between the current state and the previous control/input.

Cd is the matrix that models the relationship between the current output and the previous state.

Dd is the matrix that models the relationship between the current output and previous control/input.

To create our model, we should first consider what features we want our design to have. We established that the desired features for the model are:

- Corrections to the vehicle speed are made at a rate that is proportional to the difference between the actual speed of the vehicle and the speed displayed on the driver advisory.

- When steady state is achieved, the driver will track the advisory perfectly, meaning that the advisory speed should be equal to the actual speed of the vehicle.

Taking these required features into account we formulated the following human driver model:

$$\dot{x}_{ACT}(t) = \lambda \cdot \left(x_{ADV}(t) - x_{ACT}(t)\right) \qquad \textit{Eq. 3}$$
$$\dot{x}_{ADV}(t) = u(t)$$

Where:

x_{ACT} (t) is the actual speed of the vehicle at time *t*.

x_{ADV} (t) is the speed of the advisory at time *t*.

u (*t*) is the control/input vector of the system at time *t*.

λ is the change proportionality constant.

$\dot{x}_{ACT}(t)$ *and* $\dot{x}_{ADV}(t)$ correspond to the first order derivative with respect to time of x_{ACT} (t) and x_{ADV} (t) respectively.

As we can see, both requirements are satisfied by this model. When steady state is achieved, $\dot{x}_{ACT}(t) = 0$ which implies that x_{ADV} (t) $= x_{ACT}$ (t) necessarily. At the same time, the rate of change of the actual speed of the vehicle is proportional to the difference between advisory and actual speed, with a proportionality constant of λ .

To use this model on the optimization MPC algorithm we first need to discretize it. To do this we used the finite difference method, which yields accurate approximations for small values of Δt . For a generic variable *a*:

$$\dot{a}(t) = \frac{da}{dt} \approx \frac{\Delta a}{\Delta t}; \frac{a_{t+1} - a_t}{\Delta t} \qquad \textit{Eq. 4}$$

By taking Eq. 4 concept and adapting it into Eq. 3, we obtained:

$$x_{ACT}(k+1) = x_{ACT}(k) + \lambda \cdot h \cdot \left(x_{ADV}(k) - x_{ACT}(k)\right) \qquad \text{Eq. 5}$$
$$x_{ADV}(k+1) = x_{ADV}(k) + u(k) \cdot h$$

Where h is the sampling time (previously Δt *on Eq*.4).

By taking a look at the model, we immediately obtain *Ad* and *Bd:*

$$Ad = \begin{pmatrix} 1-\lambda \cdot h & \lambda \cdot h \\ 0 & 1 \end{pmatrix};\ Bd = \begin{pmatrix} 0 \\ h \end{pmatrix};$$

Cd and *Dd* are equal to 0 in this model formulation.

With all the model matrices defined we now have a full model that can be incorporated into the MPC algorithm to try its performance by running different simulations.

Chapter 5. Problem Formulation

In this chapter we explain how the problem was formulated in order to develop the algorithm. We have to remember that the ultimate goal of the formulation is to build a successful algorithm to track the fuel-consumption optimized speed accounting for the driver delay through a model.

5.1.Conventional/explicit formulation

A conventional formulation for an optimization problem consists of three major steps: Variable definition, objective function specification and constraint specification. We followed this standard procedure to formulate a dynamic linear quadratic programming problem (a quadratic objective function subject to linear constraints), making adaptations to our particular case as needed. A general dynamic linear quadratic program formulation has the following structure:

$$\min_{\mathbf{x},\mathbf{u}} \int_0^T \frac{1}{2}(\mathbf{x}\,Q\,\mathbf{x}^T + \mathbf{u}\,R\,\mathbf{u}^T)\mathrm{dt}$$

Subject to:

$$\mathbf{x}(0) = \mathbf{x}_0$$
$$\dot{\mathbf{x}} = A\,\mathbf{x} + B\,\mathbf{u}$$

$$\mathrm{lbu} \leq u(t) \leq \mathrm{ubu}$$
$$\mathrm{lbx} \leq x(t) \leq \mathrm{ubx}$$

5.1.1. Variable definition

First, we have to analyze what variables we are going to optimize in the problem, as well as consider those parameters that are not subject to optimization but are going to be used in the problem. After conducting a thorough analysis we concluded that our optimization variables would be:

- x_{ACT} (t) is the actual speed of the vehicle at time t.
- x_{ADV} (t) is the speed of the advisory at time t.
- u (t) is the control/input vector of the system at time$t$.

Note that these variables are consistent with the ones used to formulate our human driver model.

On the other hand, we can also define a parameter that would be used in the problem formulation. It is not an optimization variable since its value will not change/be optimized, it is given. This parameter is the desired speed to be followed by the truck at each time t. This parameter is the result of previously calculated (by other team members) optimization problems that had minimizing fuel consumption has its objective. We will name this parameter $\mathrm{V}_{\mathrm{DES}}$(t).

5.1.2. Objective function specification

The objective function (also known as cost function in the literature) is a key element in the problem formulation. It is responsible for defining the optimization that will take place on the problem. A proper definition of the cost function is needed in order to obtain the pursued results. It is also in charge of determining the importance of each optimization variable with respect to each other through the use of weighting coefficients. We can penalize one variable in favor of other one.

Our objectives are to minimize the tracking error between the desired speed and the actual speed of the vehicle, and not to have sharp changes in the advisory speed to allow the driver to follow the advisory commands properly. Taking these considerations into account we formulated the following objective function:

$$\min_{y,\mathbf{u}} J = \int_{t_0}^{t_0+T} \frac{1}{2}\left[(y)^2 + u(t)^2\right]$$

$$\min_{\mathbf{x,u}} J = \int_{t_0}^{t_0+T} \frac{1}{2}\left[q \cdot (V_{\mathrm{DES}}(t) - X_{\mathrm{ACTUAL}}(t))^2 + r \cdot u(t)^2\right]\mathrm{dt}$$

In order to incorporate the cost function into the algorithm we discretized it as follows:

$$\min_{\mathbf{x,u}} J = \sum_{k=0}^{N} \frac{1}{2}\left[q * (V_{\mathrm{DES}}(k) - X_{\mathrm{ACTUAL}}(k))^2 + ru(k)^2\right]$$

We develop the first squared difference and we remove the constant terms since they do not affect the optimization problem. This term was $\mathrm{V}_{\mathrm{DES}}{}^{2}$ which was the first term of the squared difference development. We obtained the following cost function:

$$\min_{\mathbf{x,u}} J = \sum_{k=0}^{N} \frac{1}{2}\left[q \cdot (X_{\mathrm{ACTUAL}}(k))^2 + ru(k)^2 - 2 \cdot q \cdot V_{\mathrm{DES}}(k) \cdot X_{\mathrm{ACTUAL}}(k)\right]$$

where: $h = \textit{time step length; } T = \textit{time horizon; } N = T \,/\, h = \textit{number of time steps}$

We observe that the cost function varies quadratically with $x_{ACT}(k)$ and $u\,(k)$, and linearly with $x_{ACT}(k)$.

5.1.3. Constraint specification

The constraints are those equations and inequations that impose a restriction on the possible values of the states and/or control of the system. We used constraints to ensure a physically feasible result of the problem. The value of the states has to be kept within some limits to avoid feasible but no physically possible results. For example, we may obtain that the optimal value of the control to minimize the cost function on a given situation is greater than the maximum value that the truck operator can satisfy on a real scenario. We have to differentiate between two types of constraints: equality constraints and inequality constraints.

5.1.3.1. Equality constraints:

Equality constraints are those that have the form: A·x = B . The human driver model that was developed in previous sections falls into this constraint category. The model is incorporated into the problem has an equality constraint. As a result, we used the following constraints:

$$\mathrm{x}(k+1) = A_d \cdot \boldsymbol{x}(k) + B_d \cdot u(k)$$

$$where:$$

$$Ad = \begin{pmatrix} 1-\lambda \cdot h & \lambda \cdot h \\ 0 & 1 \end{pmatrix};\ Bd = \begin{pmatrix} 0 \\ h \end{pmatrix};$$

5.1.3.2. Inequality constraints:

This type of constraint is generally used to impose bounds to the optimization variables. This means, to establish a maximum and a minimum value for a given variable. In this case we used the following constraints:

$$lbx \le x(k) \le ubx$$

$$lbu \le u(k) \le ubu$$

Where:

lbx is the lower bound vector for the states vector *x(k)*

ubx is the upper bound vector for the states vector *x(k)*

lbu is the lower bound for the input/control *u(k)*

ubu is the upper bound for the input/control *u(k).*

5.2.MATLAB translation

Now that we formulated the problem in an explicit form, we have to adapt that formulation into a specific format that is accepted by MATLAB. In order to do this, we first have to determine what format is required by the specific solver to be used on the resolution of the optimization problem.

We will use a function called *quadprog* to performance each optimization in the algorithm. This function is a quadratic solver that finds the minimum for a problem with a quadratic objective function and linear constraints, like the ones we have in this problem. The inputs to *quadprog* are:

- *H:* a symmetric real matrix that contains the quadratic objective term.
- *f:* a real vector that contains the linear objective term.
- *A* and *Aeq* : real vectors that contain the coefficient matrices for the inequality and equality constraints, respectively.
- *b* and *beq :* real vectors that contain the independent term for the inequality and equality constraints, respectively.
- *lb* and *ub*: real vector that contain the lower bound and upper bound respectively for all the optimization variables.
- *x0:* a real vector that specifies the initial point of the problem.

The standard *quadprog* problem hast the following format:

$$\min_x \frac{1}{2}x^T Hx + f^T x \text{ such that } \begin{cases} A \cdot x \leq b, \\ Aeq \cdot x = beq, \\ lb \leq x \leq ub. \end{cases}$$

This causes an immediate rename in our notation, where we should incorporate into one single vector both the *n* state variables (*x*) and the *p* inputs (*u*). We named this vector *v*.

$$v = \binom{x}{u}; v \in R^{N*(n+p)\ x\ 1}$$

Following this new notation, our objective function can be re-written in order to adapt it to the standard *quadprog* format:

$$min\ \frac{1}{2}v^T Hv + F^T v$$

To create the H matrix we have to incorporate the quadratic term corresponding to the states to the one corresponding with the control. We do this by creating an augmented matrix formed by two other matrices diagonally concatenated.

$$H = \begin{pmatrix} \begin{pmatrix} Q & 0 \\ 0 & R \end{pmatrix} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \begin{pmatrix} Q & 0 \\ 0 & R \end{pmatrix} \end{pmatrix}; \; Q = \begin{pmatrix} q & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & q \end{pmatrix}; \; R = \begin{pmatrix} r & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & r \end{pmatrix};$$

$$H \in R^{N*(n+p)\ x\ N*(n+p)}; \qquad Q \in R^{n\ x\ n}; \qquad R \in R^{p\ x\ p};$$

The vector *f* is defined as follows:

$$F = \begin{pmatrix} f \\ \vdots \\ f \end{pmatrix}; \; f = \begin{pmatrix} -V_{DES} \\ 0 \\ 0 \end{pmatrix};$$

$$F \in R^{N*(n+p)\text{-}1\ x\ 1}; \qquad f \in R^{(n+p)\ x\ 1};$$

Once that the objective function was correctly formulated in the appropriate format, it was time to formulate the constraints. As the standard *quadprog* form says, we should differentiate between equality and inequality constraints in the algorithm. In our case the equality constraints correspond to the human driver model, and there are no inequality constraints besides the bounds for the variables.

The equality constraints are formulated as follows:

$$A_{eq} \cdot v = b_{eq}$$

$$\begin{bmatrix} \begin{bmatrix} 1_{\mathrm{nxn}} & 0_{\mathrm{nxp}} \end{bmatrix} & \begin{bmatrix} 0_{\mathrm{nxn}} & 0_{\mathrm{nxp}} \end{bmatrix} & \cdots & \begin{bmatrix} 0_{\mathrm{nxn}} & 0_{\mathrm{nxp}} \end{bmatrix} \\ -\begin{bmatrix} A_d & B_d \end{bmatrix} & \begin{bmatrix} 1_{\mathrm{nxn}} & 0_{\mathrm{nxp}} \end{bmatrix} & \cdots & \vdots \\ \vdots & \ddots & \ddots & \begin{bmatrix} 0_{\mathrm{nxn}} & 0_{\mathrm{nxp}} \end{bmatrix} \\ \begin{bmatrix} 0_{\mathrm{nxn}} & 0_{\mathrm{nxp}} \end{bmatrix} & \cdots & -\begin{bmatrix} A_d & B_d \end{bmatrix} & \begin{bmatrix} 1_{\mathrm{nxn}} & 0_{\mathrm{nxp}} \end{bmatrix} \\ \begin{bmatrix} 0_{\mathrm{nxn}} & 0_{\mathrm{nxp}} \end{bmatrix} & \cdots & \begin{bmatrix} 0_{\mathrm{nxn}} & 0_{\mathrm{nxp}} \end{bmatrix} & -\begin{bmatrix} A_d & B_d \end{bmatrix} \end{bmatrix} \mathbf{v} = \begin{bmatrix} \mathbf{x}_0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

The initial state of the system is already known and specified through x_0, while the remaining *N* states have to be predicted using the driver model. The only thing left to define are the bound for the variables, but the definition is still the same as the one given in 5.1.3.2, no additional modification is needed to implement it into the algorithm.

Chapter 6. MATLAB MPC code

Now that all the required elements are formulated in a way that can be understood by *quadprog* we are going to code the algorithm. To do this we divided the process in two steps. In the first step, we created a function to run the MPC process given some inputs and return the calculated outputs. On the second step, we created a script in which we generated synthetic data[4] to test the algorithm, and a call to the function created on the first step is made.

At first, we did not take this two-step approach and we created all the code in the same script. This first approach led us to some issues with the results. We observed how the time horizon was not being updated correctly and thus we were obtaining incorrect results, in which the algorithm was not updating the time instant correctly. By applying this two-step process we solved the issue.

6.1.Step 1: MPC function

In this step, the main objective was creating a function that ran the MPC process in a separate script from the main body of the code. It is in this step where the majority of the algorithm is coded.

The function code can be divided in 3 major parts. On the first part, we initialized and defined the optimization parameters: *n, p, h, λ, T, N, Ad, Bd*, *q* and *r*. The value for the optimization parameters is discussed in Chapter 7 since they vary between different simulation cases.

On the second part, we created the augmented matrices that were defined on 5.2 using specific MATLAB commands such as *blkdiag* to create the diagonal block matrix *H*, or

[4] We used synthetic data instead of the real data obtained by other group members due to privacy issues. If the real data was used and shown in this thesis, we would have had to sign a confidentiality agreement with Volvo. We considered that it was more meaningful for educational purposes to use synthetic data and allow free access to the thesis.

vertcat to create matrix *F*. Furthermore, the bounds on the variables are also specified in this part.

Finally, on the third part the call to the optimization function *quadprog* is made. This call is made inside a loop for the *N* time steps of the process. They main tasks that take place inside the loop are updating the constant term of the equality constraint (*beq*), the prediction of the future state of the system using the human driver model, and the collection of the optimization results for each iteration. Furthermore, we used the command *exitflag* when calling *quadprog* to check whether the optimization process returns an optimal result or otherwise the problem is unfeasible.

This function has the desired speed (*VDes*) and the initial state of the system (*x0*) as its inputs. The returned outputs to the main scripts are the optimal control actions *(u)* for the time horizon, and the state of the system (*x*) through the time horizon.

The actual code for the function can be found on the first annex of this project, Annex 1. The value for the parameters included in the annex is not representative. They were assigned random values as an example. The actual values for each simulation will be discussed on Chapter 7.

6.2.Step 2: Main script

We used the main script to generate synthetic data for the desired speed. We decided to create a series of steps each one of the with a different value, that represents the desired speed of the truck at each time instant. There are multiple ways to do this, but we chose to use several *for loops* to create the series of steps. Furthermore, we also used this script to make the call to the MPC function described on section 6.1 and to plot the results provided by that function. Even though it is the main script, the actual computation is performed on the MPC function script code.

Chapter 7. Parameter definition and simulation

In this chapter we will run several simulations using the code described on Chapter 6. We evaluated different driving situations in which the truck would be required to modify its speed to analyze the algorithm performance. Additionally, we observed how different values of the parameters affect the simulation results.

In order to run the simulations we first have to determine the parameters of the algorithm. For all the simulations, the number of states (*n*) is equal to 2, since we are using a two-state system. Those states are the actual speed (X_{ACT}) and the advisory speed (X_{ADV}). The number of inputs (*p*) is equal to one. This input is the rate of change of the advisory speed ($\dot{x}_{ADV}$). The proportionality constant, λ, is set to $e^{-0.5} \approx 60\%$.

When deciding the value for the time-related parameters we kept in mind that computational time is a key factor on this problem. A large computational time will not allow a real implementation on a truck, since the optimal actions will not be displayed to the truck operator on time for it to modify its speed. In this case, we have to assign a value for the time horizon, *T*, as well as to the time step for discretization, *h*, that do not increase computational time significantly. The smaller the time step is, the larger the computational time is. For the time horizon, the greater it is, the larger the computational time is. We conducted several tests to choose the optimal combination of time horizon and time step length, to reduce computational time.

T (s)	h (s)	N=T/h	COMPUTATIONAL TIME (s)	RESULT
2	0.1	20	3.5	Not optimal
2	0.2	10	2.6875	Not optimal
2	0.5	4	2.8125	Not optimal
5	0.1	50	10.875	Not optimal
5	0.2	25	4.26 56	Not optimal
5	0.5	10	2.7344	Not optimal

T (s)	h (s)	N=T/h	COMPUTATIONAL TIME (s)	RESULT
10	0.1	100	36	Optimal
10	0.2	50	9.0156	Optimal
10	0.5	20	3.2813	Optimal
15	0.1	150	97.6719	Optimal
15	0.2	75	17.6250	Optimal
15	0.5	30	5.1250	Optimal

Table 2:Computational time comparison[5]

Taking Table 2 as our reference, we concluded that computational time has a strong positive correlation with the number of time steps, *N*, on the problem. Do not forget that *N* determines the size of the matrices in our problem, so the bigger the matrices MATLAB has to operate with, the larger the computational time is. We chose *T*=10 and *h*=0.5, since it is the combination that gave a smaller computational time and an optimal solution at the same time.

Using the weighting parameters *q* and *r* we can penalize certain behaviors of the algorithm. We can choose between having a really precise following of the reference or instead having smooth control actions. A high value of *q* penalizes the deviations from the reference while a high value of *r* penalizes the control actions. If we choose a high value of *r* we will obtain smooth control actions and vice versa. We can see an example of this phenomenon in the following figures (Figure 5 and Figure 6):

[5] The classification between "optimal" and "not optimal" results is made according to the results observed in those simulations. If the simulation result does not converge to the desired speed it is classified as not optimal. On the other hand if the result is satisfactory it is classified as optimal.

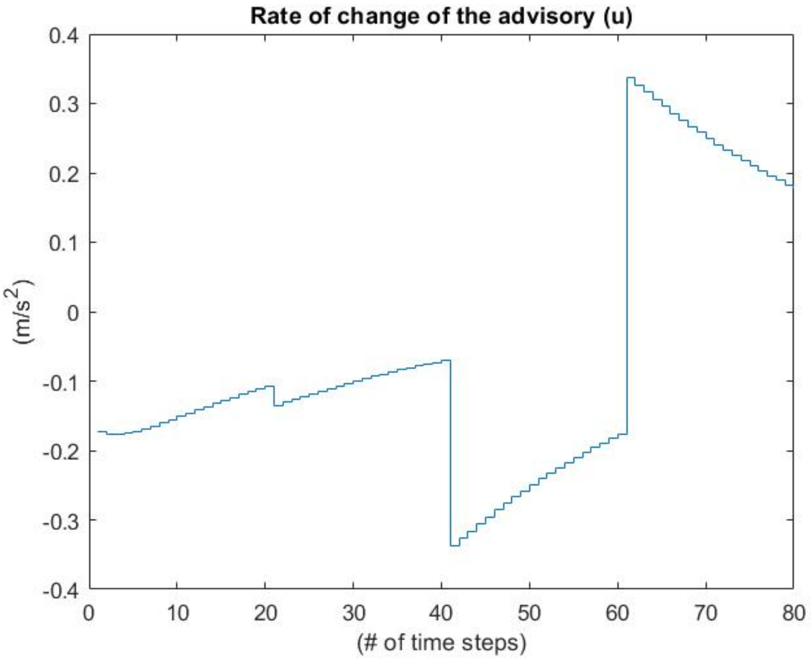

Figure 5: Control action for a high value of r

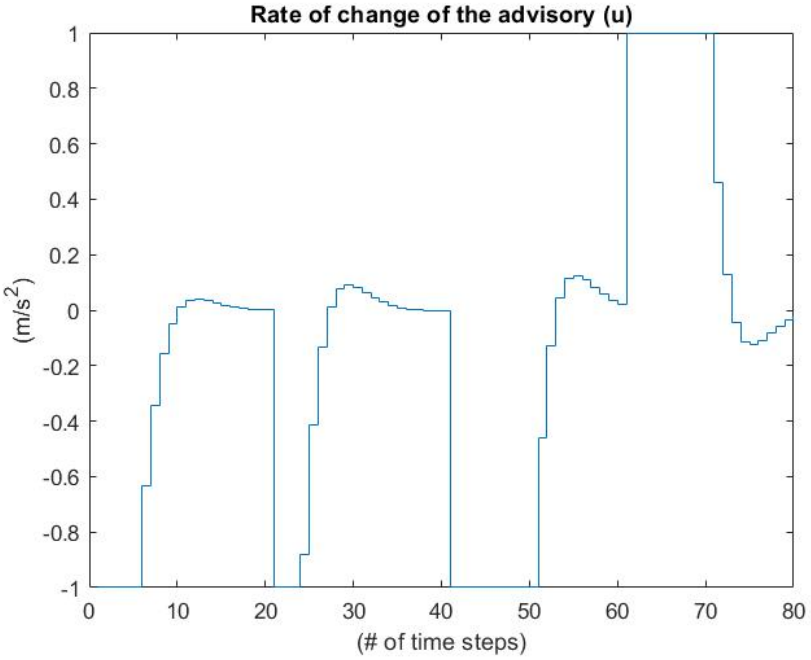

Figure 6: Control action for a small value of r

As we can observe on both figures, the control action is smoother for a high value of *r* than for a low value. The same phenomenon is observed when changing the value of q.

Furthermore, we established the boundaries on the variables of the problem to ensure that the obtained results were feasible in a real environment. We considered that the truck will operate on a semi urban environment, where the maximum allowed speed on the road is 50 km/h≈14 m/s. Knowing this we set the driver advisory so it is capable of advising higher speeds to the driver (18m/s at most) in order to achieve the steady state faster. The average acceleration rate for a heavy-duty truck is 0.68 m/s^2 (GUAN16). On the other hand, the maximum deceleration rate for a heavy-duty truck, in the speed range that we will consider is 0.876 m/s^2, (MARU12). Rewriting these values into matrix form:

$$lbx = \begin{pmatrix} 0 \\ 0 \end{pmatrix};\ ubx = \begin{pmatrix} 14 \\ 18 \end{pmatrix};\ lbu = (-0.876);\ \ ubu = (0.68);$$

The lower bound for the states is set to zero since we are assuming that the truck will only be moving forward, hence a negative speed is not possible.

7.1.Simulation 1: Acceleration to constant speed:

On this simulation, the truck will start with an initial actual speed of 0 m/s. The desired speed is equal to 2 m/s. The initial state of the advisory speed is set to 3 m/s.

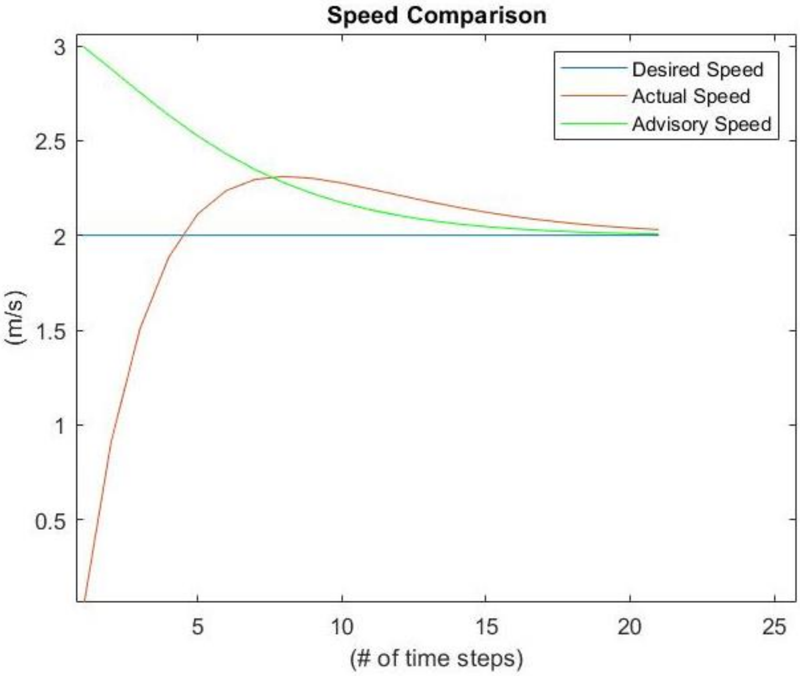

Figure 7: Speed comparison

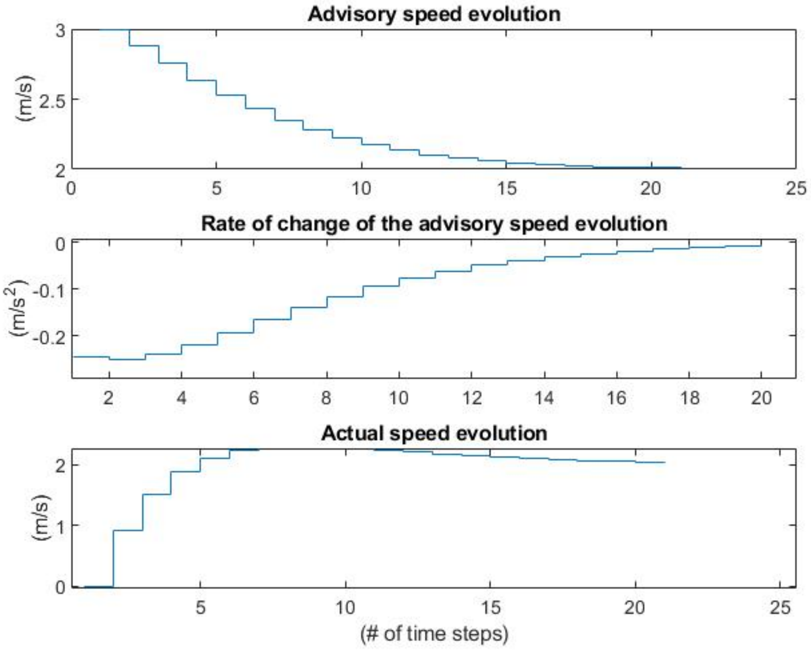

Figure 8: Optimization results

After running the algorithm with the previously mentioned initial conditions, we obtained the results shown in figures 7 and 8. We used a continuous plot on Figure 7 to have a more clear view of the speeds (actual speed, desired speed and driver advisory speed) and easy the comparison process. The actual results are discrete, as shown on Figure 8 through a stairs plot. We will follow this procedure in all the simulations.

7.2.Simulation 2: Deceleration to constant speed:

In this occasion we will simulate a deceleration process from a constant speed to another constant speed. The initial speed is 13 m/s and we want to achieve a speed of 11 m/s. The initial state of the driver advisory is 12 m/s.

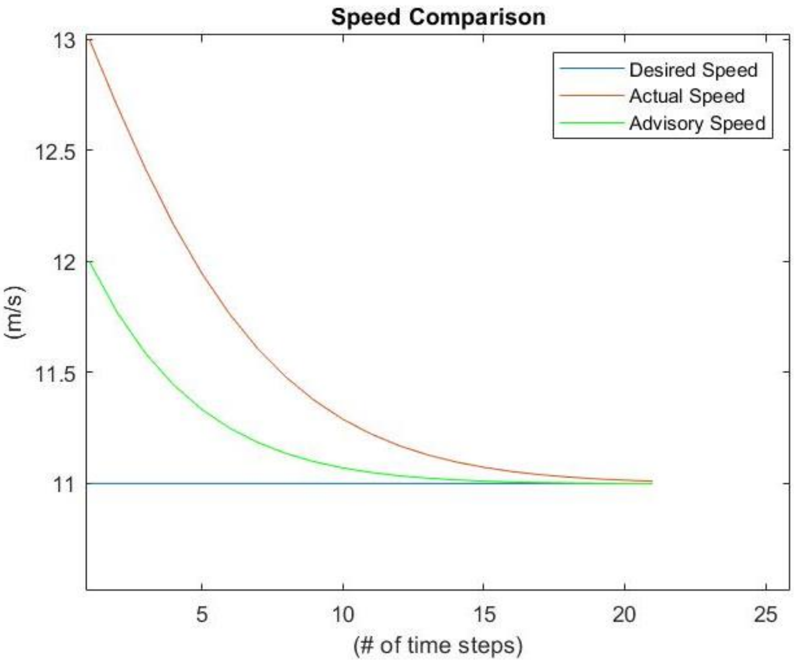

Figure 9: Speed comparison

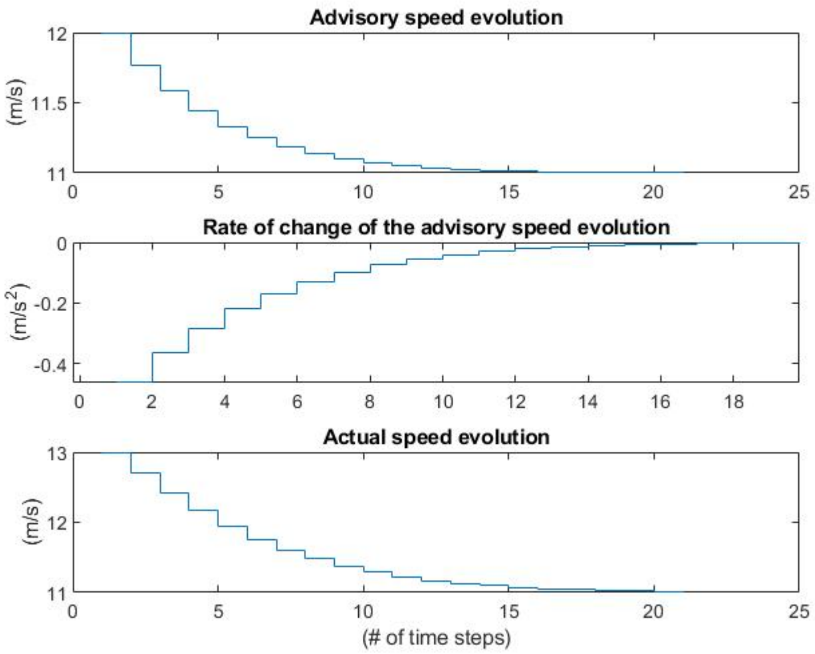

Figure 10: Optimization results

7.3.Simulation 3: Series of steps:

In this simulation we will observe the behavior of the algorithm when the desired speed is made by a series of different speeds instead of a single one. The desired speed is equal to 12 m/s at first, then equal to 14 m/s, then to 12 m/s and finally to 10 m/s. The initial speed of the truck is 11m/s, and the initial advisory speed is 13 m/s.

We will perform this simulation both for a large value of *r* and a small value of *r* to see the difference in the control action, as well as in the reference tracking performance.

7.3.1. High value of *r*:

The chosen value for *r* is 5. In the following figures we can see the simulation results.

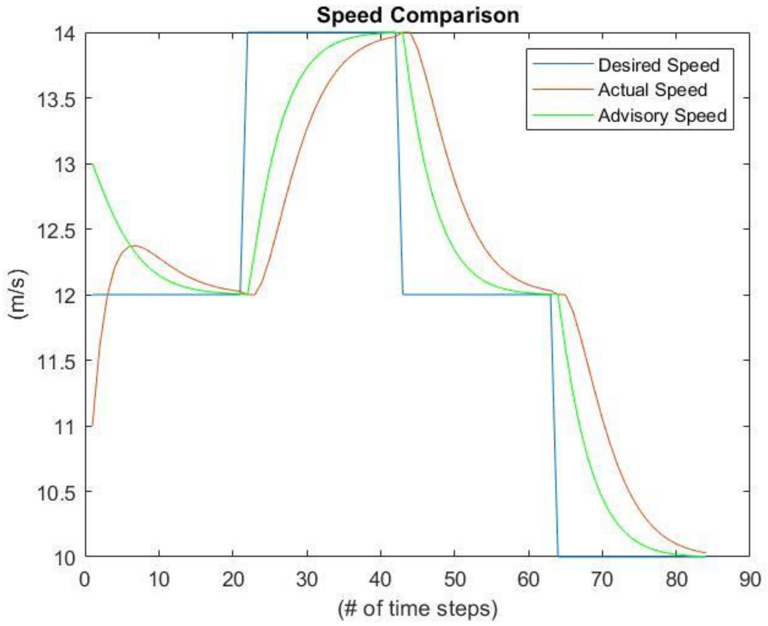

Figure 11: Speed comparison (high r)

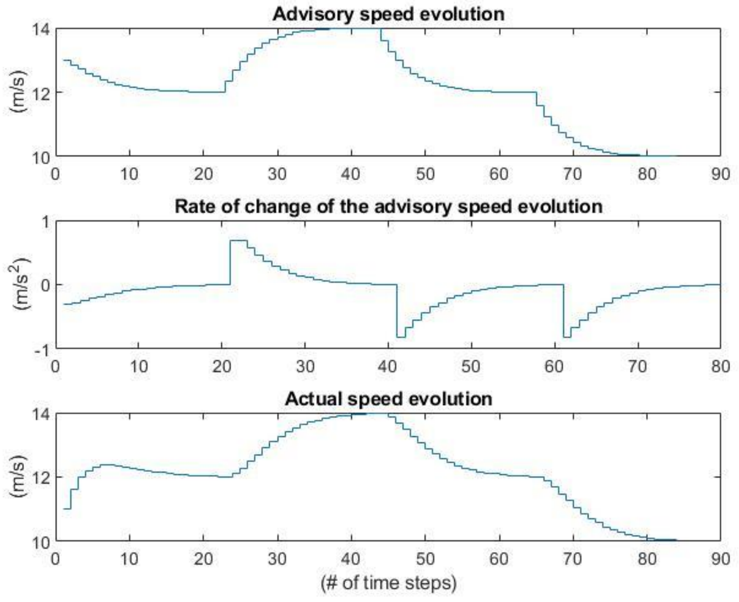

Figure 12: Optimization results (high r)

7.3.2. Low value of *r*:

The chosen value for *r* is 1. In the following figures we can see the simulation results.

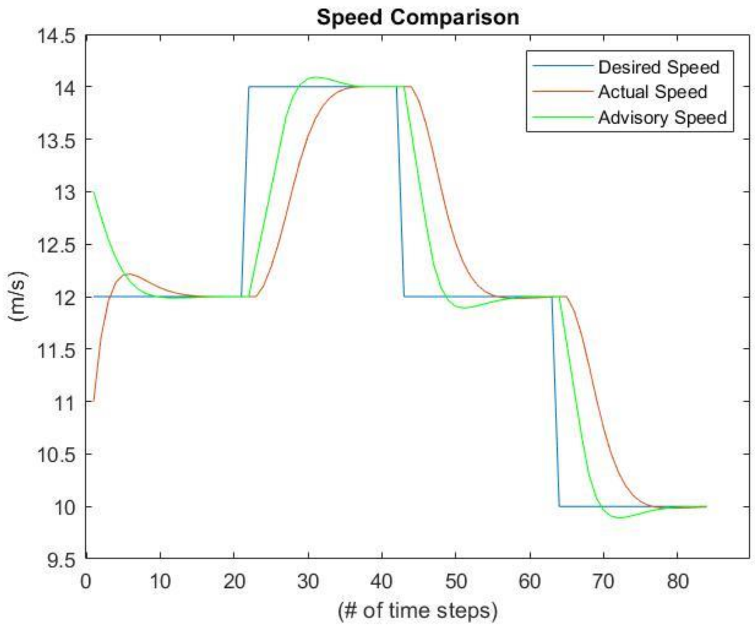

Figure 13: Speed comparison (low r)

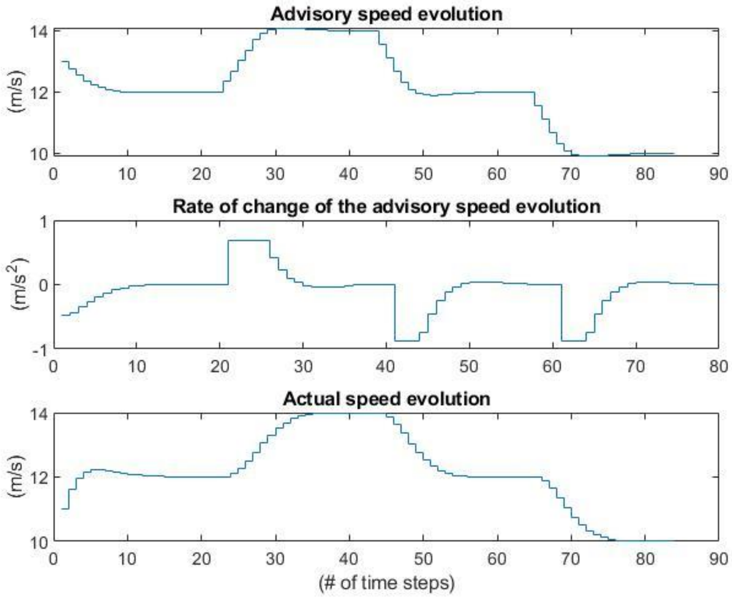

Figure 14: Optimization results (low r)

Chapter 8. Result analysis

In this chapter we conducted an analysis of the results obtained in the previous simulations. The differences between simulations are highlighted, and the algorithm performance is evaluated.

On the first two simulations (7.1,7.2) a relatively high value of *r* was used, which explains why the control action prediction is quite smooth. On the other hand, the smoothness of the control action compromises the reference tracking performance, causing the actual speed to achieve the desired speed in a large time. We can clearly see on Figure 7 how the actual speed does not reach the desired speed in the established time horizon, but the control action shown in Figure 8 is considerably smooth, allowing the operator to perform small changes of speed, achieving a more comfortable driving experience.

To better illustrate this phenomenon, we performed the third simulation for a high value of *r* (7.3.1), and for a small value of *r* (7.3.2). When the value of *r* is high, the rate of change of the driver advisory is smooth, but the tracking of the desired speed is a bit poor. On the other hand, when the value or *r* is smaller, the tracking of the desired speed is significantly better, achieving the steady state in less time, but the control action is more abrupt. Furthermore, there is also some overshoot on the response of the system. We can observe this in more detail in the following figures:

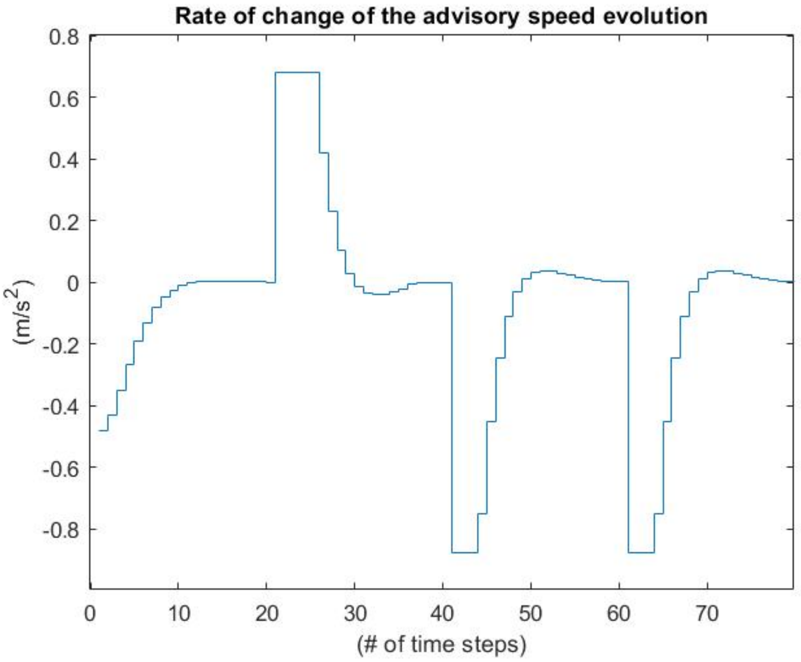

Figure 15: Rate of change of the advisory speed (small r)

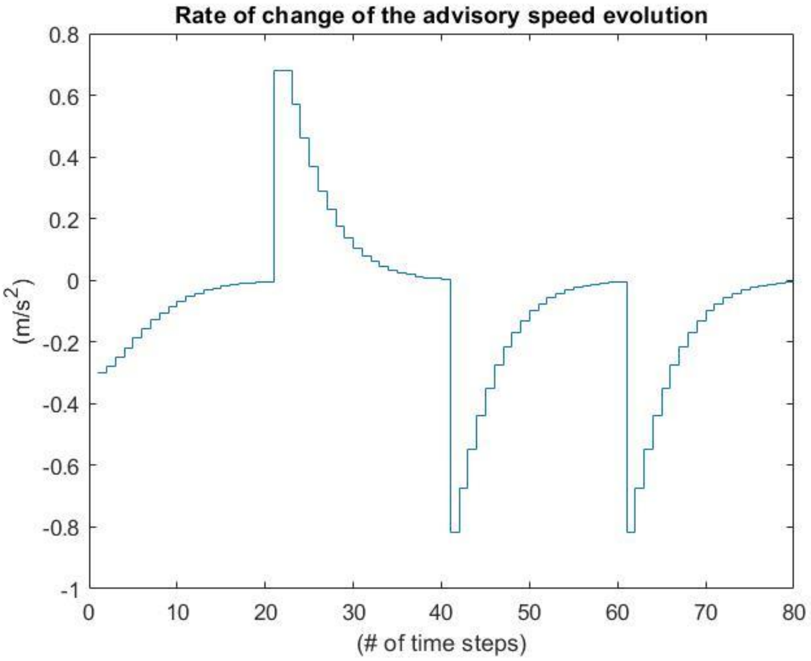

Figure 16: Rate of change of the advisory speed (high r)

The control action represented by Figure 15 would require a more experienced driver whose driving skills are more refined. The control actions of Figure 16 are smoother; hence an average truck operator is able to follow them with no trouble. This behavior allows the algorithm to be adjusted for different type of drivers, achieving a better tracking of the desired speed for those drivers who are more precise, while giving a more relaxed control for those with less precision but still considerably accurate.

As we observed on the figures of Chapter 7, the algorithm works how it is supposed to work. There are corrections being made to the actual speed at a constant rate, that is proportional to the difference between the actual speed of the vehicle and the advisory speed, accounting for the driver delay. Furthermore, when the steady state is achieved, the actual speed of the truck is equal to the advisory speed, so the rate of change of the speed advisory is equal to zero. Perfect tracking of the desired speed occurs when the appropriate value of r is used.

Chapter 9. Conclusions and future works

The most remarkable conclusion that we obtained from the project is that MPC is a valid and functional technique to be implemented in a driver advisory system, and consequently it will help reduce fuel consumption and GHGs emissions, at the same time that it helps trucking companies save money. The algorithm developed has shown a satisfactory output in the simulations conducted, which was the main goal of the project. It has shown accurate tracking capabilities of the desired speed, depending on the parameters chosen.

Furthermore, we developed a human driver model that has shown great performance when predicting future speeds of the truck. The use of a linearized model has allowed the algorithm to successfully predict the future speed of the truck to use it in the optimization process, obtaining optimal control actions.

Another remarkable conclusion of the project is that the algorithm developed has enough freedom to adapt it to different driving styles and skills in order to achieve a better performance. By tuning parameters like r we can obtain different types of advised speed profiles, which will require more/less experienced truck operators to follow them, depending on the precision demanded by that profile. For example, if the operator of a specific truck is someone with many years of experience behind the wheel, we can tune r so the control actions are more abrupt, obtaining faster optimal tracking of the desired speed. On the other hand, if the truck is being driven by a new driver, we can tune r to obtain smoother control actions, that will also lead to optimal results but in larger time span.

The objectives of the project have been achieved, but there is still room for improvement and refinement. The main future works will involve the refinement of the human driver model in order to achieve more precise predictions of the speed of the truck. Furthermore, the dynamics of the truck can be incorporated into the algorithm, so the advised speed also considers all the parts of the system formed by truck and driver.

This algorithm has to be tested in a real environment by building it into a truck to test its real performance. Keep in mind that the theoretical behaviors predicted using a model may differ from the actual behavior in a real environment, where more factors affect the final output.

Chapter 10. Bibliography

[ATRI16] American Transportation Research Institute (ATRI), "An Analysis of the Operational Costs of Trucking: 2016 Update", Tech. Rep., 2016.

[BEHR20] Behrendt, M. "Un esquema MPC discreto", 2020. https://es.wikipedia.org/wiki/Control_predictivo_por_modelo#/media/Archivo:MPC_scheme_basic.svg.

[BMW_20] BMW of San Antonio, "BMW to Unveil Level-5 Self-Driving Car by 2021", 2020, https://www.bmwofsanantonio.com/bmw-to-unveil-level-5-self-driving-car-by-2021.htm.

[CAIR18] Cairano, S., L. Kolmanovsky, "Real-time optimization and model predictive control for aerospace and automotive applications", Annual American Control Conference (ACC), 2018.

[DELI07] Delice, İ., S. Ertugrul, "Intelligent Modeling of Human Driver: A Survey", IEEE Intelligent Vehicles Symposium, 2007.

[EIA_20] U.S. Energy Information Administration, "Annual Energy Outlook 2020'', 2020, https://www.eia.gov/outlooks/aeo/.

[EPA_19] U.S. Environmental Protection Agency, "U.S. Transportation Sector Greenhouse Gas Emissions'', 2019, https://nepis.epa.gov/Exe/ZyPDF.cgi?Dockey=P100WUHR.pdf.

[GUAN16] Guangchuan Y., H. Xu, Z. Wang, Z. Tian, "Truck acceleration behavior study and acceleration lane length recommendations for metered on-ramps", International Journal of Transportation Science and Technology, vol. 5, p. 93-102, 2016.

[GUO_13] Guo H., Y. Ji, T. Qu, H. Chen, "Understanding and Modeling the Human Driver Behavior Based on MPC", IFAC Proceedings Volumes, vol. 46, p.133-138, 2013.

[MARU12] Maurya, A., P. Bokare, "Study of deceleration behaviour of different vehicle types", International Journal For Traffic And Transport Engineering, vol. 2 of 3 vol., p.253-270, 2012.

[PROK01] Prokop G., "Modeling human vehicle driving by model predictive online optimization", Vehicle System Dynamics, vol. 35, no. 1, pp. 19-35, 2001.

[SCHI03] Schittler M., "State-of-the-art and emerging truck engine technologies for optimized performance, emissions and life-cycle costing", 9th Proc. Diesel Engine Emission Reduction Conference, Newport, 2003.

[SEAR18] Sears, D., "The Sightless Visionary Who Invented Cruise Control.", Smithsonian Institution, 2018, www.smithsonianmag.com/innovation/sightless-visionary-who-invented-cruise-control 180968418/.

[SHAO11] Shaout, A., D. Colella, "Advanced Driver Assistance Systems - Past, present and future'', 7th Int. Computer Engineering Conf., 2011.

[SHUT19] Shuttleworth, J., "SAE J3016 automated-driving graphic'', 2019, https://www.sae.org/news/2019/01/sae-updates-j3016-automated-driving-graphic.

[TATU03] Tatum, M., "What is cruise control?'', 2003, http://www.wisegeek.com/what-is-cruise-control.htm.

[UN__15] UN General Assembly, "Sustainable Development Goals'', 2015, https://sustainabledevelopment.un.org/?menu=1300.

[ZHAN10] Zhang, P., "Advanced industrial control technology'', Amsterdam: William Andrew/Elsevier, 2010.

[ZIEB17] Ziębiński, A., R. Cupek, D. Grzechca, L. Chruszczyk, "Review of advanced driver assistance systems (ADAS)'', 2017.

Chapter 11. Annex

11.1. Annex 1

```
function [x,u] = MPCrun(VDes,x0)
%% Initialization parameters:
n = 2; % number of state variables (x)
p = 1; % number of input variables (u)
h = 0.25; % time step
lambda=exp(-0.25);
T = 10; % time horizon
N = T/h; % Number of time steps
Ad = [1-lambda*h lambda*h; 0 1];
Bd = [0; h];
C = [1 0];
Q = C'*C;
R = 0.8;
%% Augmented matrices:

% Bounds on the variables
lb =[];
ub = [];

% Augmented matrices loop:
for i=1:N+1

Qdiaglist{i} = blkdiag(Q,R);
Fdiaglist{i}=[-VDes(i); 0; 0];
```

```
Adiaglist{i} = [eye(n,n) zeros(n,p)];
Adiaglist2{i} = -[Ad Bd];

% We specify the limits (bounds) of the variables:
lb = [lb;[-20; -20; -1]];
ub = [ub;[20; 20; 1]];
end

Q_aug = blkdiag(Qdiaglist{:});
F_aug= vertcat(Fdiaglist{:});

Aeq_aug = [[zeros(n,N*(n+p)) ; blkdiag(Adiaglist2{1:end-
1})] zeros((N+1)*n,n+p)];
Aeq_aug = Aeq_aug + blkdiag(Adiaglist{:});

% b matrix definition:
beq_aug = [x0;zeros(N*n,1)];
%% MPC optimization run:
x(:,1) = x0;
opt = optimoptions('quadprog','Display','final');
for k = 1:1:10/h %Prediction horizon

    % We update beq matrix in each iteration
    beq_aug = [x(:,k);zeros(N*n,1)];

    beq_aug_list{k}=beq_aug; % Auxiliary list to check
the process
```

```
    % We run the optimization for the current iteration
    [v, minI,exitflag] =
quadprog(Q_aug,F_aug,[],[],Aeq_aug,beq_aug,lb,ub,[],opt)
;
    vlist{k}=v; % Auxiliary list to check the process

    %We get the value of the control in the current
iteration
    u(k)=v(3);

    % We predict future state:
    x(:,k+1) = Ad*x(:,k) + Bd*u(k);
end
end
```