Design and implementation of controllers for a platooning system of vehicles following a path

Author: Isaac Ambit Brao

Advisor: Víctor Repecho del Corral

Polytechnic University of Catalonia

This thesis was written as a part of

Master in Automatic Control and Robotics

Acknowledgements

I would like to thank Arnau Dòria and Víctor Repecho for giving me the opportunity and the material necessary to participate in this project.

Abstract

The goal of this thesis is to implement three different controllers, based on sliding mode control algorithms for a platooning system, to test each one of the controllers and analyze its performance in order to decide if it is feasible.

The work is based on a scientific paper that aims to find an Adaptive Cruise Control algorithm for platooning systems and control the nonlinear system using sliding mode control techniques.

The working model is a differential steering robot with two motorized front wheels and a castor wheel behind, equipped with ultrasound sensors in order to measure the distance to the car ahead and an infrared array sensor used to track a line.

The control algorithm runs in real time on an embedded system along with all the sensors, it has been programmed in C, since it is a low level programming language and allow to have a more efficient code.

Contents

List of Figures

List of Tables

1 Introduction

During the last years the interest in Advanced Driver Assistance Systems (ADAS) has been growing, this has result into an increasing interest for many research fields related to control, data acquisition, data processing and communication systems.

This thesis explores each one of the research fields mention above, focusing on topics such as Adaptive Cruise Control (ACC) [29] and vehicle to vehicle (V2V) [13] communications. The main challenge is to implement a robust control technique able to deal with the non linearities of the system and reject model uncertainty due to parameters imprecision.

Platooning system is an application of an Advanced Driver Assistance System categorized as Cooperative Adaptive Cruise Control (CACC) [30] or ACC. The main idea behind a platooning system [12] is to couple, without any physical structure, two or more vehicles in a convoy manner, where the vehicle ahead or leader is manually driven and the vehicles behind follow it. This is achieved by using the communications system and automated driving techniques, as a result the vehicles maintain a close safety distance and the human interaction with the vehicle is reduced.

The proposed control technique is sliding mode control (SMC) [10], a control technique widely used in power electronics, known by its robustness and high performance with non linear and time-varying systems. This control method modifies the system dynamics by means of a set-valued control action in such a way that forces the system to slide along a surface. The sliding surface is chosen to be a function of the states of the system.

The strengths of this control technique are the robustness against bounded disturbances, bounded parameters variation and unmodeled dynamics.

Three different sliding mode control algorithms are introduced, implemented, tested and analyzed. The first algorithm is the relay control which is, by nature, discontinuous, the second algorithm is a continuous approximation of the first algorithm using a sigmoid function, and the last one is the super-twisting algorithm (STA) [20], where the resulting control action is a combination of a discontinuous and continuous term.

1.1 Background

The idea of a self-driving cars is a concept that has been around since the beginning of 1900's, the first implemented approach of an autonomous vehicle was in 1925, when Houdina Radio Control Company [33] radio-controlled a car equipped with a transmitting antenna from another car.

However, the first platooning system approach arrived when Bel Geddes presented in the 1939 World's Fair, futurama [32], which was a model for automated roadways based on radio-controlled electric cars feed by an electromagnetic roadway.

The first tangible results of a platooning system were in 1972-73 with the ARAMIS [2] project developed in France, ARAMIS achieved to platoon twenty-five small transit vehicles running at 80 km/h, they would run in areas with high transit as a train without any mechanical coupling between vehicles, using ultrasonic and optical range sensor to estimate the distance to the vehicle in front, finally in 1987 the project was abandoned.

The Prometheus Project in Europe [31] (1980-1995) was another inflexion point where different automotive manufacturers, technology companies, universities and the government worked together to define the state of the art of autonomous vehicles and to build the first driver-less intelligent vehicles that would run on an advanced road system. The project concluded with notable results in areas such as telecommunications, vehicle control and artificial intelligence.

The research on the field of platooning systems still continues with great achievements during the last years. The most important project is Safe Road Trains for the Environment (SARTRE) [28].

SARTRE project (2009-2012), funded by the European Commission, was aimed to develop a prototype system to be integrated in a platooning without modifying the roadways, being able to interact with other vehicles not belonging to the platoon. The project explored and worked with radar, cameras, laser sensors and communication systems technologies.
The leading vehicle, driven by a professional driver, acts as a master controlling all the slave vehicles behind and setting a safety distance. Every vehicle is equipped with collision mitigation, adaptive cruise control, lane departure warn-

ing and brake control systems. It incorporated lateral position control for the first time in a platooning system. In January 2012 SARTRE carried out a demonstration in a public road near Barcelona, where a lead truck, controlled by an operator, followed by three cars driven entirely autonomously at speeds of up to 90 km/h with a gap between the vehicles of no more than 6 m.

Design and implementation of controllers for a platooning system of vehicles following a path, is inspired by the above mentioned project. The work, is a continuation of previous works, therefore some parts are based on other authors work. Part of this thesis is based on the paper "Sliding mode controllers for adaptive cruise control" [1] and a continuation of the thesis "Control of a Two-Wheeled Line Tracking Robot Using a Com-plete Mechanical Mode" [3]

1.2 Objectives

The main objective of this thesis is to design three different controllers based on SMC to control linear and angular velocity for a set of cars running inside a platooning system following a black line with a white background. More precisely, this can be divided in the following objectives.

- Designing of sliding mode surface for linear velocity and line following control.
- Implementing three different controllers based on sliding mode control.
- Analyzing the performance of each controller.

The objectives include the implementation for sensor readings, from the ultrasonic device and the infrared array sensor, a communication system for data analysis, and a motor modelling to understand its behaviour and limitations.

The tasks also require of working with different developing platforms such as Eclipse, Spyder and Matlab and programming languages such as C and Python.

1.3 Materials

1.3.1 Microcontroller

The microcontroller used for this project is the STM32F4-Discovery (figure 1.1) which its CPU is an ARM Cortex-M4.

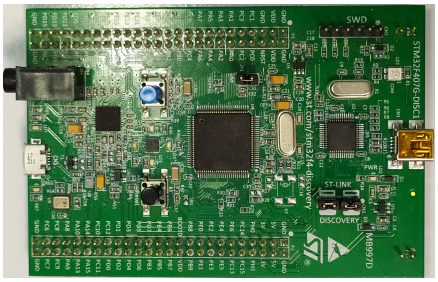

Figure 1.1: STM32F407VG

Its main characteristics are, the maximum frequency 168MHz, which allows fast computations very suitable to perform real time control, and its RAM memory, that results in handy to store data for a latter analysis.

1.3.2 Wi-Fi Module

Programmable Wi-Fi module ESP8266 (figure 1.2) with UART serial connection(baudrate 115200). Implements UDP and TCP/IP protocols as main feautres.

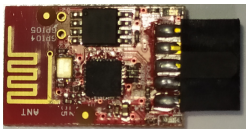

Figure 1.2: ESP8266 Wi-Fi module

1.3.3 Line Sensor

QTR-8A (figure 1.3) is made by a series of photoresistors, depending on the light detected by each sensor the voltatge will vary, thus if the sensor is placed on a dark surface the resistance of photoresistors will be low. However, a bright area would make the resistance increase and therefore the voltage fall. Depending on

the voltage of each photoresistor. It is possible to know where the sensor is placed.

Figure 1.3: QTR-8A photosensor

1.3.4 Ultrasonic Sensor

The HC-SR04 Ultrasonic sensor (figure 1.4), is a device that outputs a 40 kHz ultrasound burst of ultrasound. Once the eighth ultrasonic cycle has been sent, the HC-SR04 sensor will set to high "echo" pin. When the sound waves get reflected by an object these will come back and will be detected by the same sensor, then the "echo" pin will become low, therefore the time the "echo" pin remained high is the time it takes the ultrasonic wave to get reflected by an object and come back, knowing the speed of this ultrasonic wave, multiplying it by the time "echo" pin has been high and dividing by two, since the sound travel two times the distance between the sensor and the obstacle, we can know the distance from the sensor to the object. This will be used to estimate the distance between the actual vehicle and the one in front. It has to be taken into account that the measuring range is 2cm - 80cm/400cm (pracitcal/theorical) and it has an angular range of 15º, with an accuracy of 3mm.

Figure 1.4: HC-SR04 Ultrasonic sensor

1.3.5 Motor Driver

The electronic device L625PD is a dual full-bridge or H-bridge driver (figure 1.5) to control the direction and speed of each motor separately.

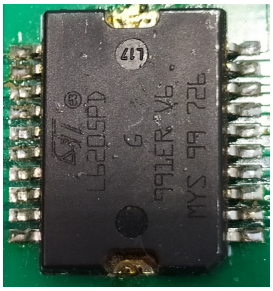

Figure 1.5: L625PD full bridge driver

1.3.6 DC Motors

The DC motors (figure 1.6) are used to generate a torque and an angular velocity on the wheels, the motor comes with a two channels hall-effect encoder that is going to be used to capture each wheel angular velocity.

Figure 1.6: DC motors

1.3.7 UPC PCB

The electronic board (figure 1.7) has been designed in the UPC, specifically to plug in the STM32F4-Discovery board, from STMicroelectronics, it has also sockets for the Wi-Fi module ESP8266, as well as wires to the line sensor, the board has incorporated a L298N dual full-bridge driver in it, to control the two DC motors. Connected to the board there are also three ultrasonic sensors, used to retrieve the distance between the vehicle and the closest obstacle.

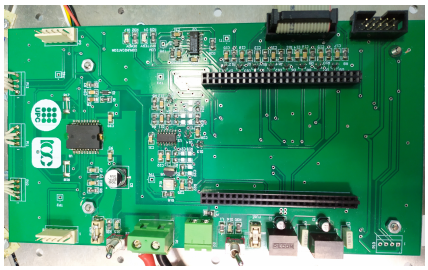

Figure 1.7: UPC electronic board

2 State of the Art

2.1 Platooning System

As mentioned in the previous chapter, platooning systems reproduce the behaviour of a convoy, where there is a leading vehicle at the front of the platoon which sets the limit velocity for all the other vehicles behind, but instead of a mechanical coupling between vehicles, control techniques and V2V communication are used to emulate the coupling behaviour.

According to the Society of Automotive Engineers (SAE) [18] depending on the degree of automation, vehicles can be classified in six different levels, where level 0 is the lowest automation rank and level 5 means that the vehicle is fully automated.

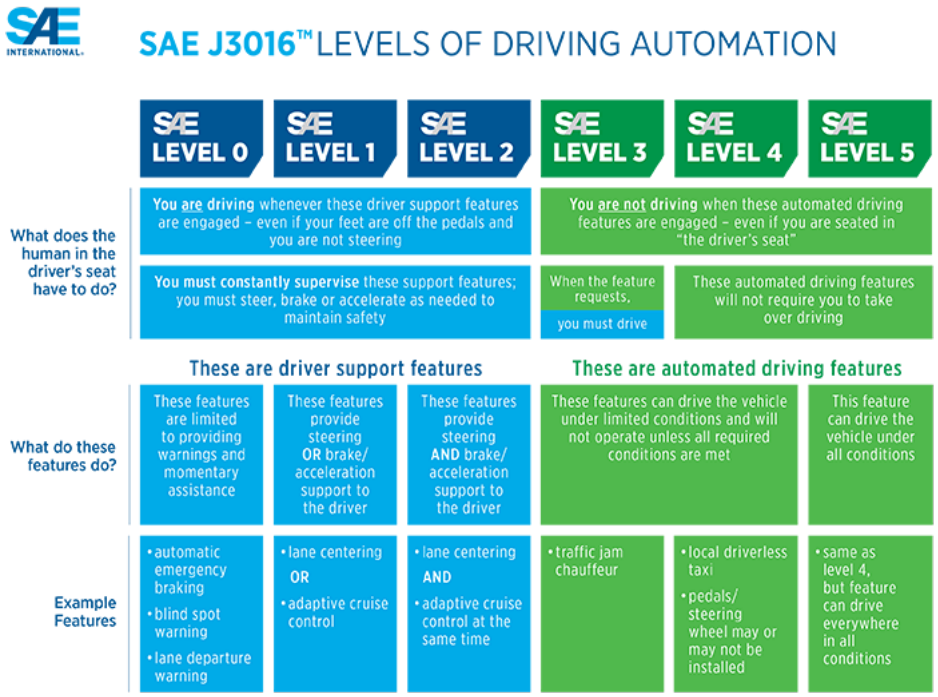

Figure 2.1: SAE level of automation chart [18]

Following the SAE criteria, platooning systems fit in the level one or two of automation, since human interaction is required, at least to drive the leader vehicle.

A platooning system would be considered of level one if only steering or acceleration is automated, however, if both features are automated at the same time it would belong to level two of driving automation.

The technology used to control platooning systems might vary depending on the given approach, some examples of platooning systems require the infrastructure to be modified in order to share data with the vehicles, also called, vehicle to infrastructure communication (V2I), another option which has a lower cost is to equip the vehicles with a set of sensors to have a good knowledge of the environment around.

Most of platooning systems implement the Adaptive Cruise Control (ACC) technology, which is a velocity controller and the key point, is that it adapts vehicle speed, given by a policy, to maintain a safety distance with the vehicle in front, the most commonly used sensors for this method are radar, lidar or cameras, although very often it may combine different kind of sensors at the same time, and extract a more reliable source of data using sensor fusion techniques such as Kalman filters or Bayesian networks.

The main issue with ACC is that it does not present string stability, which means that platoon is prompt to oscillations, due to abrupt braking and accelerations, when the leader vehicle accelerates/decelerates a delay is generated, and accumulated with every vehicle in the platooning system, causing the platoon to stretch and squeeze like an accordion.

An improvement of ACC is Cooperative Adaptive Cruise Control (CACC), which ensures string stability, this is achieved by using communications between vehicles (V2V), where the different vehicles of the platoon share information about the actual acceleration, velocity and steering, giving the possibility to the vehicles in the platoon to predict and anticipate actions, reducing the reaction time of the entire platoon and avoiding the accordion effect.

Platooning systems present many benefits and some drawbacks, giving room to research and future improvements.

The benefits of platooning systems are exposed in the following list:

- By decreasing the distance between vehicles at a minimum safety distance, the road capacity is increased.
- Due to instant and automatic breaking, the human reaction time disappear resulting in a gain on safety.

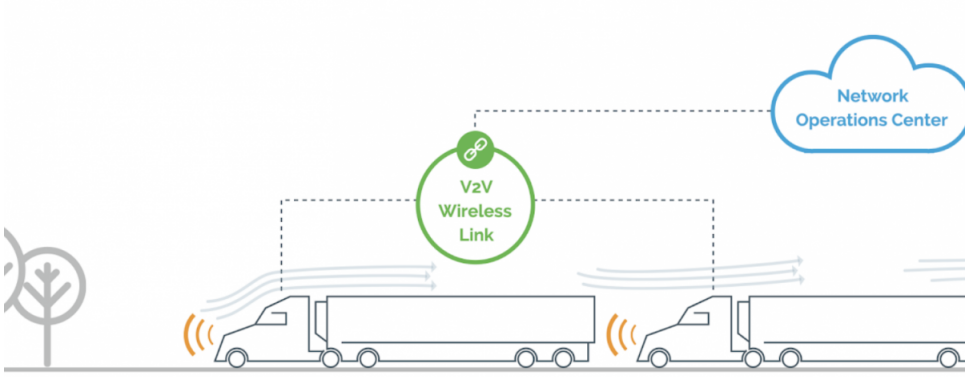

Figure 2.2: Vehicle to vehicle communications in a platooning system [27]

- When vehicles run at a constant speed there are less acceleration and deceleration, therefore the there is less amount of fuel consumed.
- As vehicles drive closer the aerodynamic drag is reduced, which improves the aerodynamic effectiveness, and the resistance to air of the vehicles behind decreases, reducing the CO_2 emissions by 10% according to a study conducted by the department of Vehicle Dynamics Laboratory in Auburn University.
- The fact that driving is automated, allows drivers to do other tasks.

The platooning system drawbacks are exposed in the following list:

- The rely on the communication system might be a target for hackers, creating hazardous situations.
- The fact that vehicles are controlled by a computer can give the feeling of lack of control to the driver.
- Drivers would be less attentive if a quick reaction is needed due to a vehicle failure.
- Platooning systems might not work well in towns due to the dense traffic.

2.2 Lyapunov Stability

Before explaining the sliding mode control method, the Lyapunov stability [7] results have to be introduced, since they set the basis of sliding modes, among other nonlinear control methods.

Lyapunov theory is used to extract conclusions from differential equations trajectories without having to find the exact solution and it is widely used for nonlinear systems analysis due to its toughness when they are up to be solved.

Based on the slides of control theory [4], the stability in the sense of Lyapunov is defined as follows:

Considering the dynamical system

$$\dot{x} = f(x)$$

where $x \in R^n$

- $x_e \in R^n$ is said to be an equilibrium point of the system if $f(x_e) = 0$
- The equilibrium point, x_e is stable, if and only if, for every $R > 0$ there exists $r > 0$ that implies

$$\|x(0) - x_e\| \leq r \implies \|x(t) - x_e\| \leq R, \forall t > 0$$

If not, the equilibrium point is unstable.

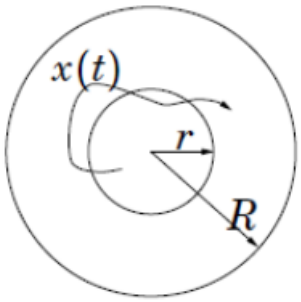

Figure 2.3: Representation of Lyapunov stability [4]

The Lyapunov theorem of asymptotic stability, is a stronger condition of the stability theorem that ensures the attractiveness of the equilibrium point.

The equilibrium point, x_e, is asymptotically stable if and only if is stable and there exists $r_0 > 0$ such that

$$\|x(0) - x_e\| \leq r_0 \implies \lim_{t \to +\infty} \|x(t) - x_e\| = x_e$$

The equilibrium point, x_e, is globally asymptotically stable, if and only if, it is asymptotically stable and for all $x(0)$

$$\lim_{t \to +\infty} x(t) = x_e$$

The Lyapunov theorem of exponentially stability, states that, the equilibrium point, x_e, is exponentially stable, if and only if, there exist $\alpha, \lambda, r_0 > 0$ such that,

$$\|x(0) - x_e\| \leq r_0 \implies \|x(t) - x_e\| \leq \alpha \|x(0) - x_e\| e^{-\lambda t}, \forall t > 0$$

Exponential stability implies asymptotic stability, but not the other way around.

The equilibrium point, x_e, is said to be marginally stable, if and only if, it is stable but not asymptotically stable.

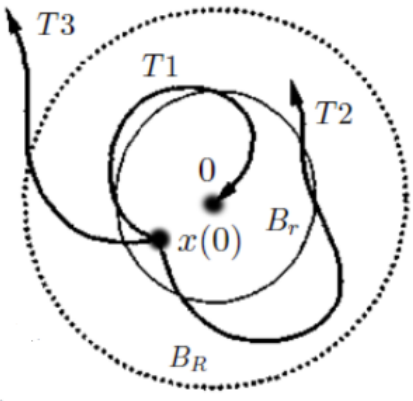

Figure 2.4: Representation of the three types of Lyapunov stability [4]

The figure 2.4 show the three possible types of stability in the sense of Lyapunov. Where T1 presents asymptotically and exponentially stability, T2 marginal stability and T3 is unstable.

Lyapunov's direct method, uses a Lyapunov function $V(x)$, that depends on the states of the system, to proof stability. If in a ball B_R the function $V(x)$ is positive

definite, has continuous partial derivatives, and its time derivative with respect to time for any state trajectory of $\dot{x} = f(x)$ is negative semi-definite

$$\dot{V} \leq 0, x \neq 0$$

then, $V(X)$ is said to be a Lyapunov function for $\dot{x} = f(x)$. Furthermore, if its time derivative with respect to time is negative definite, $\dot{V}(X) \leq 0, x \neq 0$ $V(x)$ is a stirct Lyapunov function.

A scalar continuous function $V(x)$ such that $V(0) = 0$ in a region defined by a ball B_R is said to be

- locally positive definite if $x \neq 0 \implies V(x) > 0$
- locally positive semi-definite if $x \neq 0 \implies V(x) \geq 0$

If the above properties and $V(0) = 0$ holds for the entire state space, the $V(x)$ is said to be globally positive definite and globally positive semi-definite respectively. If $-V(x)$ is positive definite or semi-definite, then $V(x)$ is said to be negative definite or semi-definite.

Local stability theorem states that, if in a region defined by a ball B_R, there exists a scalar function $V(x)$ with continuous first partial derivatives such that

- $V(x)$ is positive definite in B_R
- $\dot{V}(x)$ is negative semi-definite in B_R

then the origin is stable. Moreover, if $\dot{V}(x)$ is negative definite in B_R the origin is asymptotically stable.

According to the global stability theorem, a system is said to be globally asymptotically stable if there exists a scalar function $V(x)$ with continuous first partial derivatives such that

- $V(x)$ is positive definite
- $\dot{V}(x)$ is negative definite
- $V(x) \to \inf$ as $\|x(t)\| \to \inf$

The theorem of exponential stability assume that there exists a scalar function $V(x)$ with continuous first partial derivatives such that

- $V(x)$ is positive definite
- $\dot{V}(x)$ is negative definite and it satisfies $\dot{V}(x) \leq -\alpha V(x)$ for all x
- The sublevel sets $\{x|V(x) \leq c\}$ are bounded for all $c \geq 0$

Then the origin is exponentially stable.

If a real function $W(t)$ satisfies the inequality

$$\dot{W}(t) \leq -\alpha W(t)$$

where α is a real number, integrating both parts of the inequality,

$$W(t) \leq W(0)e^{-\alpha t}$$

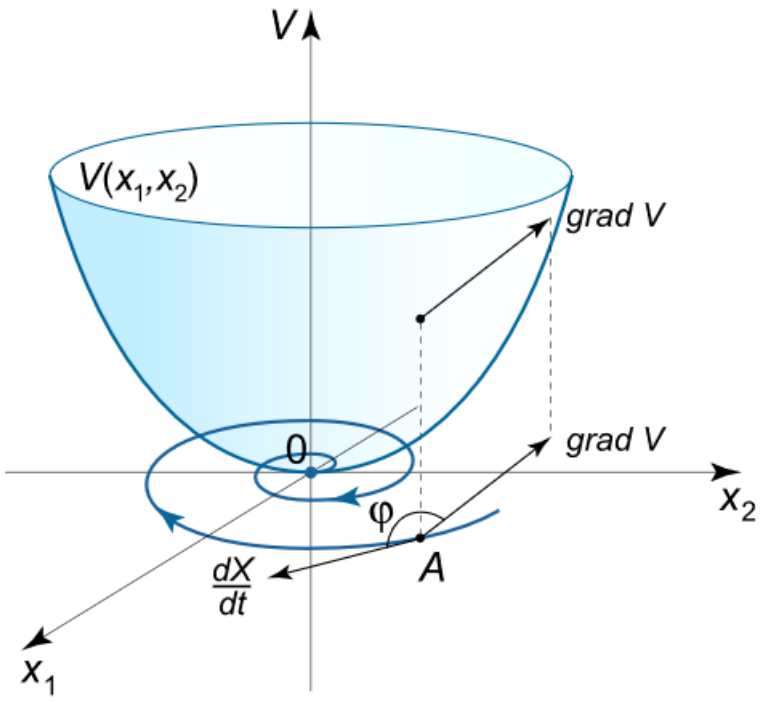

Figure 2.5: Lyapunov function representation [26]

2.3 Sliding Mode Control

Sliding mode control (SMC) is a variable structure control method, which means that the dynamics of the system are modified according to a stable hyperplane or sliding surface with simpler dynamics defined beforehand [6]. This makes this technique ideal for the control of nonlinear systems. With a discontinuous and high frequency control action, the state trajectories are forced to move and stay on the manifold $\sigma = 0$, once the system is in the manifold, it evolves following the dynamics described by this one.

Lyapunov theory sets the basis for sliding mode control, since the control action selected is one that makes the system stable in the sense of Lyapunov.

The idea of this method is to build an hyperplane, denoted by $\sigma(x(t))$, which is a function of the states of the system, and by means of a specific control action, force the system states trajectories to move towards the manifold $\sigma(x(t)) = 0$, ideally, once the trajectory reaches the manifold, it remains on it thereafter sliding over, then it is said that the system has entered in sliding mode. On the sliding surface the system evolves according to the dynamics defined by the manifold, therefore the manifold is, typically chosen to be a Linear Time Invariant (LTI) system, with simple dynamics, although it is not mandatory, usually the state errors are chosen to compound the sliding surface. A desired convergence rate to the equilibrium point, can be selected by choosing the adequate parameters for the manifold.

The hyperplane has the same dimension as number of inputs has the system, and the manifold dimension is the difference between the number of system states and the number of inputs, resulting into a order reduction with respect to the original system.

Due to the switching nature of the control action the system becomes insensitive to uncertain parameters correlated with the control action, robust against unmodelled dynamics and disturbances.

The design of sliding mode control is broken down into two separate parts:

The first part is the design of the sliding manifold $\sigma(x(t)) = 0$, in such a manner that the dynamics of the reduced order system evolve as desired.

The second part is the design of the control action, the control action is chosen in order to enforce the trajectories to move towards the sliding surfaces.

The control action is usually defined by the combination of two terms, one that is continuous and another one that is discontinuous. The discontinuous part is the one that forces the trajectory to move towards the surface, whereas the continuous part ensures the trajectory remains in the sliding surface once on it.

For the existence of sliding mode, the vector tangent to a state trajectory near the manifold has to be directed towards it. If all trajectories in the state space move towards the manifold, it is said to be a globally reachable sliding mode.

In order to ensure sliding mode, the hyperplane has to be stable in the sense of Lyapunov.
Choosing (2.1) as a Lyapunov function candidate it ensures it is positive definite and zero at the origin.

$$V(x) = \frac{1}{2}\sigma(x(t))^2 \tag{2.1}$$

To assure asymptotic stability to the manifold, the differentiation with respect to time of $V(x)$ must be strictly negative.

$$\dot{V}(x) = \sigma\dot{\sigma} \leq 0 \tag{2.2}$$

A stronger criteria is exponential stability which can be achieved by,

$$\dot{V}(x) = \sigma\dot{\sigma} \leq \gamma|\sigma| \tag{2.3}$$

and (2.3) ensures convergence to the surface in finite time, integrating both sides of the inequality, the time it takes an from an initial state to reach the hyperplane is upper bounded, which is given by the following inequality

$$t_r \leq \frac{|\sigma(x(t))|}{\gamma} \tag{2.4}$$

In the ideal sliding mode control the switching control action has an infinite gain and frequency. Nevertheless, in the practical sliding mode it is not feasible, the actuator might have some mechanical delay, the microcontroller need some time

to compute the control action, and an oscillatory behaviour called "chattering" around the manifold arises. due to this situation the system does not converge to the equilibrium but to a close region of it.

The chattering effect can be reduced by applying some variations of the before

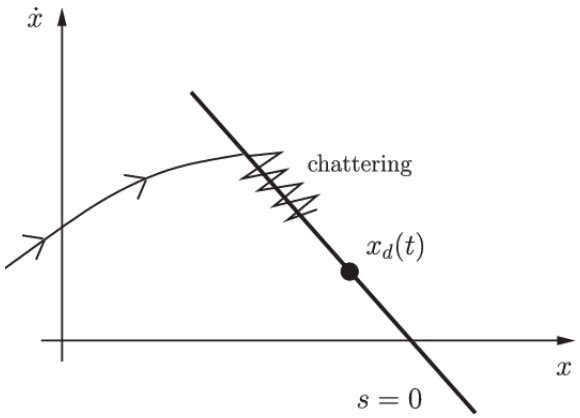

Figure 2.6: Representation of chattering effect [5]

mentioned discontinuous signal. The most used variations are the Sigmoid approximation, which consist in applying a sigmoid function to obtain the control action, instead of the on/off action, this method reduces the chattering but, on the other hand it relaxes the reachability condition,

$$Sigmoid = \frac{\sigma}{|\sigma + \epsilon|} \tag{2.5}$$

where $\epsilon > 0$ modifies the shape of the Sigmoid.

Another approach is the super-twisting algorithm where, the resulting control action is the sum of a discontinuous part and the integration of another discontinuous signal which results into a continuous term.

3 Modelling and Control Design

3.1 Model of the Vehicle

The prototype vehicle used for this project has a differential steering morphology, which means that there are two independent motorized wheels that allow the vehicle to move forward, backward and spin around, in this specific case the model is composed by two motorized wheels in front, and a castor wheel behind, to give stability at the back of the vehicle.

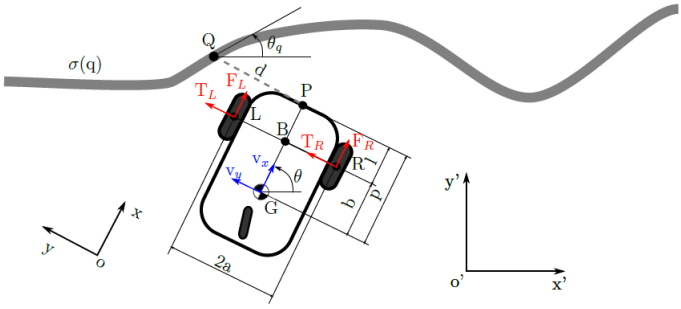

Figure 3.1: Differential steering mathematical model [3]

From the free body diagram represented by the figure 3.1 the resulting dynamic model, applying the momentum conservation is as follows:

$$\begin{aligned} F_R + F_L - f_r mg\cos\alpha - mg\sin\alpha - \frac{1}{2}\rho C_d A_x v_x^2 &= ma_x \\ T_R + T_L - f_r mg\cos\beta - mg\sin\beta - \frac{1}{2}\rho C_d A_y v_y^2 &= ma_y \end{aligned} \tag{3.1}$$

where the forces acting on the x axis are, F_L and F_R, that are the resulting forces produced by the left and the right wheel, $f_r mg\cos\alpha$ is the friction force opposing the motion, with f_r being the friction coefficient between the ground and the wheel, m the vehicle mass, g the gravity acceleration along the z axis and α is the slope of the ground between the x and z axis, $mg\sin\alpha$ is the force produced by the weight of the vehicle and $\frac{1}{2}\rho C_d A_x v_x^2$ is the force caused by the air opposing the direction of motion, with ρ being the air density, C_d the drag coefficient, A_x the frontal vehicle section and v_X, a_X the linear velocity and accelerations respectively.
The forces on the y axis are analogous to the ones on the x axis.

In order to simplify calculations the following assumptions have been made:

- The vehicle is considered as a rigid body modelled as a lumped point where all the mass is concentrated on the center of mass G.
- Along the x axis the friction coefficient is high enough to make the wheels do not slip, which results into a pure rotation of the front wheels with no friction force, it is also considered that the friction force acting on the castor wheel is negligible.
- Along the y axis the friction coefficient is high enough to prevent the wheels to move in this direction.
- The vehicle is considered to move on a flat terrain with no slope.
- The drag force caused by the air resistance it is considered to be negligible at the speed the vehicle is moving.

Applying the law of the momentum conservation along with the assumptions mentioned above, the simplified dynamic model is described by the equation (3.2)

$$\begin{aligned} F_R + F_L &= ma_x \\ T_R + T_L &= ma_y \end{aligned} \tag{3.2}$$

The vehicle lies in a rotating frame, therefore its acceleration is described by the following equality:

$$\begin{bmatrix} a_x \\ a_y \\ 0 \end{bmatrix} = \begin{bmatrix} \dot{v}_x \\ \dot{v}_y \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \omega \end{bmatrix} \times \begin{bmatrix} v_x \\ v_y \\ 0 \end{bmatrix} = \begin{bmatrix} \dot{v}_x - \omega v_y \\ \dot{v}_y + \omega v_x \\ 0 \end{bmatrix} \tag{3.3}$$

where $\dot{\theta} = \omega$ is the angular velocity of the vehicle around the z axis.

Substituting the acceleration given by the equation (3.3) in (3.2) we obtain:

$$\begin{aligned} F_R + F_L &= m(\dot{v}_x - \omega v_y) \\ T_R + T_L &= m(\dot{v}_y + \omega v_x) \end{aligned} \tag{3.4}$$

Applying the, simplified, angular momentum conservation in the center of mass:

$$(T_R + T_L)b + (F_R - F_L)a = I_z\dot{\omega} \tag{3.5}$$

where a and b are the distance between the center of the wheel and the center of mass, along the x and y axis respectively.
The inertial momentum of the vehicle is denoted by I_z and the angular acceleration by $\dot{\omega}$.

Following the assumptions and considering no slip between the wheels and the ground the conservation of angular momentum of each one of the wheels is defined as:

$$\begin{aligned} I\dot{\omega}_R + F_R r &= \tau_R - B_f\omega_R \\ I\dot{\omega}_L + F_L r &= \tau_L - B_f\omega_L \end{aligned} \tag{3.6}$$

where I is the inertial momentum of the wheel, ω_R, $\dot{\omega}_R$ and ω_L, $\dot{\omega}_L$ are the right and left wheel angular velocities and accelerations. The radius of the wheel is r, τ is the torque applied to the motor, and B_f the viscous coefficient that opposes the torque.

Further simplifications are made by considering the inertial momentum negligible compared to the other terms of the equality, resulting in the following equation:

$$\begin{aligned} F_R &= \frac{\tau_R - B_f\omega_R}{r} \\ F_L &= \frac{\tau_L - B_f\omega_L}{r} \end{aligned} \tag{3.7}$$

The relationship of the angular and linear velocity of the wheels is given by the wheel radius:

$$\begin{aligned} \omega_R &= \frac{v_R, x}{r} \\ \omega_L &= \frac{v_L, x}{r} \end{aligned} \tag{3.8}$$

Since the robot is considered as a rigid body the velocities of the wheels affect directly the velocity of the center of mass with the following relationship:

$$
\begin{aligned}
v_R = \begin{bmatrix} v_R, x \\ v_R, y \\ 0 \end{bmatrix} &= \begin{bmatrix} v_x + \omega a \\ v_y + \omega b \\ 0 \end{bmatrix} \\
v_L = \begin{bmatrix} v_L, x \\ v_L, y \\ 0 \end{bmatrix} &= \begin{bmatrix} v_x - \omega a \\ v_y + \omega b \\ 0 \end{bmatrix}
\end{aligned} \tag{3.9}
$$

Using the equations (3.9) and (3.8) the equation (3.7) becomes:

$$
\begin{aligned}
F_R &= \frac{\tau_R}{r} - \frac{B_f}{r^2}(v_x + \omega a) \\
F_L &= \frac{\tau_L}{r} - \frac{B_f}{r^2}(v_x - \omega a)
\end{aligned} \tag{3.10}
$$

Where the torque generated on each wheel is proportional to the current applied,

$$\tau = Ki \tag{3.11}$$

However, our control action is applied in terms of voltage. Applying Ohm's law $I = \frac{V}{R}$ we can obtain the relation between the torque generated according the voltage applied.

$$\tau = \frac{K}{R}V \tag{3.12}$$

Since we considered that the wheels do not slip, there is no velocity along y, therefore:

$$v_R, y = v_L, y = v_y + \omega b = 0 \tag{3.13}$$

and by differentiating equation (3.13) against time,

$$\dot{v}_y = -\dot{\omega} b \tag{3.14}$$

substituting equations (3.10) and (3.13) to equations (3.4) and (3.5) we obtain the

following expressions:

$$\dot{v}_x = \frac{1}{mr}(\tau_R + \tau_L) - \frac{2B_f}{mr^2}v_x - b\omega^2 \tag{3.15}$$

$$\dot{\omega} = \frac{1}{I_z + mb^2}\left(\frac{a}{r}(\tau_R - \tau_L) - \frac{B_f a^2}{r^2}\omega + mb\omega v_x\right) \tag{3.16}$$

As depicted in the free body diagram of the figure 3.1, in the world reference frame $X' - Y'$ the point on the line being tracked by the vehicle (q) can be expressed as:

$$\begin{pmatrix} x'_Q \\ y'_Q \end{pmatrix} = \begin{pmatrix} \sigma'_x(q) \\ \sigma'_y(q) \end{pmatrix} \tag{3.17}$$

The point being tracked can be also expressed as the distance between the center of gravity of the vehicle and the point tracked on the line from the vehicle reference frame $X - Y$, applying the rotation matrix that relates the frame of the vehicle and the frame of the world, plus the distance from the world reference frame to the center of mass of the vehicle.

$$\begin{pmatrix} x'_Q \\ y'_Q \end{pmatrix} = \begin{pmatrix} x'_G \\ y'_G \end{pmatrix} + R(\theta)\begin{pmatrix} p \\ d \end{pmatrix} \tag{3.18}$$

where R(θ) is the rotation matrix that relates the reference frame of the vehicle with the world, and θ is the angle between the world frame and the robot frame, p is the distance from the center of masses to the line tracker sensor along the X axis of vehicle's frame and d is the distance measured from the sensor to point of the line being tracked, along the Y axis.

$$R(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \tag{3.19}$$

Differentiating the equation (3.18) with respect to time, the variation of the tracked point results into the following equality:

$$\begin{pmatrix} \dot{x}'_G \\ \dot{y}'_G \end{pmatrix} + \omega R'(\theta)\begin{pmatrix} p \\ d \end{pmatrix} + R(\theta)\begin{pmatrix} 0 \\ \dot{d} \end{pmatrix} = \dot{q}\begin{pmatrix} \frac{\partial}{\partial q}\sigma'_x(q) \\ \frac{\partial}{\partial q}\sigma'_y(q) \end{pmatrix} \tag{3.20}$$

where the variation of the distance along Y is denoted by $\dot{d}$, the variation of the point being tracked by $\dot{q}$ and $R'(\theta)$ is the partial derivative of $\frac{\partial}{\partial\theta}R(\theta)$,

$$R'(\theta) = \begin{pmatrix} -\sin\theta & -\cos\theta \\ \cos\theta & -\sin\theta \end{pmatrix} \tag{3.21}$$

θ_q is the angle between the line and the world reference frame.

Considering the line to have a constant slope, the equation (3.20) can be rewritten as,

$$\begin{pmatrix} v'_x \\ v'_y \end{pmatrix} + \omega R'(\theta) \begin{pmatrix} p \\ d \end{pmatrix} + R(\theta) \begin{pmatrix} 0 \\ \dot{d} \end{pmatrix} = \dot{q} \begin{pmatrix} \cos\theta_q \\ \sin\theta_q \end{pmatrix} \tag{3.22}$$

Rearranging the equation (3.22) to isolate the variable $\dot{d}$ we get,

$$\begin{pmatrix} 0 \\ \dot{d} \end{pmatrix} = -R^{-1}(\theta) \begin{pmatrix} v'_x \\ v'_y \end{pmatrix} - \omega R^{-1}(\theta) R'(\theta) \begin{pmatrix} p \\ d \end{pmatrix} + R^{-1}(\theta)\dot{q} \begin{pmatrix} \cos\theta_q \\ \sin\theta_q \end{pmatrix} \tag{3.23}$$

where $R^{1-}(\theta)$ is the inverse rotation matrix of $R(\theta)$ and equals to:

$$R^{-1}(\theta) = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \tag{3.24}$$

which results into:

$$\begin{aligned} \dot{q} &= \frac{v'_x\cos\theta + v'_y\sin\theta - \omega d}{\cos\theta\cos\theta_q + \sin\theta\sin\theta_q} \\ \dot{d} &= v'_x\sin\theta - v'_y\cos\theta - \omega p + \dot{q}\left(-\sin\theta\cos\theta_q + \cos\theta\sin\theta_q\right) \end{aligned} \tag{3.25}$$

This can be simplified using the trigonometric equality of the sine and cosine of sum two angles $\sin(a \pm b) = \sin a\cos b \pm \cos a\sin b$ and $\cos(a \pm b) = \cos a\cos b \mp \sin a\sin b$.

Changing the variables according the following equality,

$$\theta_e = \theta - \theta_q \tag{3.26}$$

We obtain the following expression:

$$\dot{q} = \frac{v'_x \cos\theta + v'_y \sin\theta - \omega d}{\cos\theta_e} \tag{3.27}$$

$$\dot{d} = v'_x \sin\theta - v'_y \cos\theta - \omega p - \dot{q} \sin\theta_e \tag{3.28}$$

Equations (3.27) and (3.28) are expressed in terms of world reference frame, changing the coordinate system from world frame into vehicle frame is done by using the inverse of the rotation matrix used before to pass from vehicle frame to world frame.

$$\begin{pmatrix} v_x \\ v_y \end{pmatrix} = R^{-1}(\theta) \begin{pmatrix} v'_x \\ v'_y \end{pmatrix} \tag{3.29}$$

substituting equation (3.27) into (3.28) and applying the relation (3.29) to both the equations, the same equation expressed in vehicle's reference frame is obtained:

$$\begin{aligned} \dot{d} &= -v_y - \omega p - \tan\theta_e (v_x - \omega d) \\ \dot{q} &= \frac{v_x - \omega d}{\cos\theta_e} \end{aligned} \tag{3.30}$$

it can be further simplified if we consider that $v_y = -\omega b$, therefore $-v_y - \omega p = \omega(b - p)$ and since $l = p - b$ it results that $-v_y - \omega p = -\omega l$

$$\dot{d} = -\omega l - \tan\theta_e (v_x - \omega d) \tag{3.31}$$

The angle variation against time of the robot respect to the line is defined by differentiating equation (3.26):

$$\dot{\theta}_e = \omega - \omega_q \tag{3.32}$$

where $\dot{\theta}_q$ is defined as:

$$\dot{\theta}_q = c(q)\dot{q} \tag{3.33}$$

substituting equation (3.33) into equation (3.32) is described by the following expression:

$$\dot{\theta}_e = \omega - c(q)\frac{v_x - \omega d}{\cos\theta_e} \tag{3.34}$$

Therefore the equations that define the dynamic model are:

$$\begin{aligned}
\dot{v}_x &= \frac{1}{mr}(\tau_R + \tau_L) - \frac{2B_f}{mr^2}v_x - b\omega^2 \\
\dot{\omega} &= \frac{1}{I_z + mb^2}\left(\frac{a}{r}(\tau_R - \tau_L) - \frac{B_f a^2}{r^2}\omega + mb\omega v_x\right) \\
\dot{d} &= -\omega l - \tan\theta_e(v_x - \omega d) \\
\dot{\theta}_e &= \omega - c(q)\frac{v_x - \omega d}{\cos\theta_e}
\end{aligned} \tag{3.35}$$

3.2 Sliding Mode Control

The objective is to perform two different controls, a velocity control which reference is given by the Adaptive Cruise Control algorithm and a line following control. Both controls are done with sliding mode control method.

Considering two vehicles a leader and a follower, the distance between the bottom of the leader and the front of the follower is given by the following equation:

$$h(t) = x_L(t) - x_F(t) - l_L \tag{3.36}$$

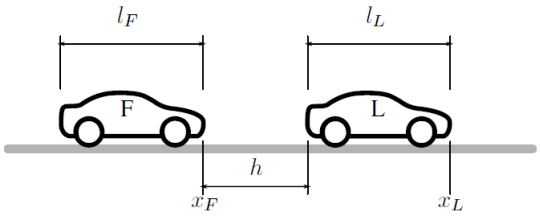

Figure 3.2: Representation of two vehicles [1]

Where x_L and x_F are the positions of the leader and follower vehicles respectively and l_L is the length of the leader vehicle.

Taking the derivative of (3.36) to obtain an equation that describes the distance variation between two vehicles,

$$\dot{h}(t) = v_L - v \tag{3.37}$$

where $v_L = \dot{x}_L$ and $v = \dot{x}_F$.

The Adaptive Cruise Control algorithm regulates the vehicle reference velocity guaranteeing a safety distance with the precedent vehicle, the safety distance depends on the velocity the vehicle is moving, the faster it moves the larger the safety distance has to be. The policy chosen that regulates the distance according to the velocity is defined by the following equation:

$$h_s(t) = h_0 + Tv \tag{3.38}$$

defining h_0 as the minimum safety distance between vehicles, and T is the time constant equivalent to the time it takes the vehicle to reach the position of its predecessor.

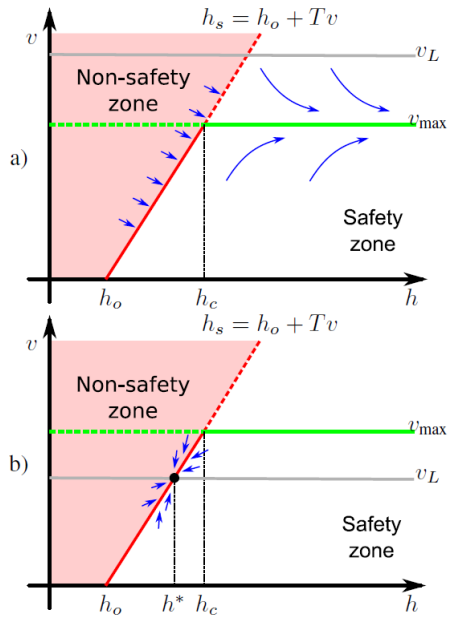

Figure 3.3: Distance/velocity policy [1]

Given this policy two cases arises.

The first case is when there is no vehicle ahead or the velocity of the ahead vehicle is greater than the actual vehicle maximum velocity, $v_{max} < v_L$, in which case the setpoint velocity is the maximum v_{max} shown in figure 3.3 a). In this case the control objective becomes regulating the velocity to v_{max}, assuming $v \to v_{max}$ and substituting it into equation (3.37) it becomes clear that, since the leader velocity is higher than the follower, the distance between both vehicles increases, therefore the safety distance is guaranteed.

The second case is when the vehicle in front drives at a lower velocity than the actual vehicle maximum speed, $v_{max} > v_L$, in which case the target is set to regulate a safety distance determined by the policy equation (3.38), only when the follower velocity becomes the v_L the equation (3.37) is at equilibrium.

The target safety distance in equilibrium, given by equation (3.38) becomes,

$$h^* = h_0 + Tv_L \tag{3.39}$$

The torque of each one of the motors τ_L and τ_R affect directly v_x in the equation (3.35), which implies that v_x must be regulated in such a manner that (3.37) rest in equilibrium.

The problem is reformulated as a velocity control depending on the distance to the leader vehicle, given by the following piecewise function,

$$v_d(h) = \begin{cases} \frac{h-h_0}{T} & h < h_c \\ v_{max} & h \geq h_c \end{cases} \tag{3.40}$$

where $h_c = h_0 + Tv_{max}$ is the target safety distance in case the vehicle is driving at its maximum speed.

Notice (3.40) is not smooth and as a consequence it is not differentiable in the entire domain this could lead to abrupt behaviour.

Differentiating equation (3.40) against time:

$$\frac{\partial v_d(h)}{\partial h} = \begin{cases} \frac{1}{T} & h < h_c \\ 0 & h \geq h_c \end{cases} \tag{3.41}$$

If the limit as h approaches the region defined by the next equation in the piecewise function (3.41) is the same, means that the function is first class C^1 and therefore differentiable. However we got that as the first region approaches the second $h \to h_c$ the limits differ.

$$\lim_{h \to -h_c} \frac{1}{T} = \frac{1}{T} \neq \lim_{h \to +h_c} 0 = 0$$

To solve this issue the following smooth piecewise function is considered,

$$v_{d\epsilon(h)} = \begin{cases} v_{max} + \frac{h-h_c}{T} & h - h_c \leq -\epsilon \\ v_{max} - \frac{1}{4\epsilon T}(h - h_c - \epsilon)^2 & |h - h_c| < \epsilon \\ v_{max} & h - h_c \geq \epsilon \end{cases} \tag{3.42}$$

where ϵ is greater than 0.

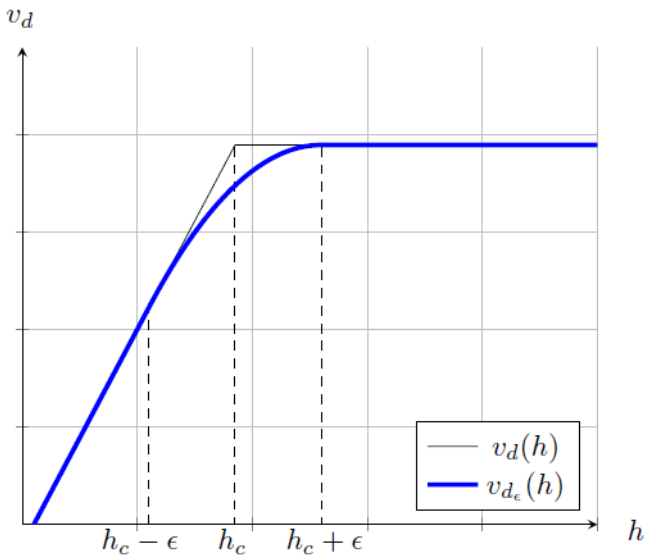

Figure 3.4: Distance/velocity smooth policy [16]

Differentiating the first class equation (3.40) against distance h the following equation is obtained,

$$\frac{\partial v_{d\epsilon(h)}}{\partial h} = \begin{cases} \frac{1}{T} & h - h_c \leq -\epsilon \\ -\frac{1}{2\epsilon T}(h - h_c - \epsilon) & |h - h_c| < \epsilon \\ 0 & h - h_c \geq \epsilon \end{cases} \tag{3.43}$$

When h approaches the boundary between first and second region, defined by $h - h_c \leq -\epsilon$ both regions approach to the same value.

$$\lim_{h \to -(h_c - \epsilon)} \frac{1}{T} = \frac{1}{T} = \lim_{h \to +(h_c - \epsilon)} -\frac{1}{2\epsilon T}(h - h_c - \epsilon)$$

When h approaches the boundary between second and third region, defined by $h - h_c < \epsilon$ both regions approach to the same value.

$$\lim_{h \to -(h_c + \epsilon)} -\frac{1}{2\epsilon T}(h - h_c - \epsilon) = 0 = \lim_{h \to +(h_c + \epsilon)} 0$$

To control velocity according the Adaptive Cruise Control algorithm and track the line using sliding mode control method, two hyperplanes are proposed.

The sliding surface that aims to control linear velocity following the reference given

by the ACC algorithm is defined as follows:

$$\sigma_1 = K(v_{d\epsilon(h)} - v) \tag{3.44}$$

Which is the velocity error multiplied by a constant gain K.
When sliding mode occurs, the resulting manifold dictates the new dynamics of the system. The manifold is chosen to be a stable Linear Time Invariant system for simplicity, and the gain K denotes the rate of decay.

The sliding surface designed to control the distance of the vehicle to the line is:

$$\sigma_2 = K_1 d + K_2 \dot{d} \tag{3.45}$$

It has been chosen to be the sum of the distance to the line error d and its time derivative $\dot{d}$, since it is with the second time derivative of the error distance where the control action appears. The gains K_1 and K_2 are chosen to be the rate of decay desired.

To ensure that the controller pushes the state trajectories towards the sliding manifold, it has to be demonstrated that the system is stable in the sense of Lyapunov.
To do so, the following Lyapunov function candidates are chosen.

$$V_1(\sigma_1) = \frac{1}{2}\sigma_1^2 \tag{3.46}$$

$$V_2(\sigma_2) = \frac{1}{2}\sigma_2^2 \tag{3.47}$$

These Lyapunov function, matches the criteria of being a function of σ_1 and σ_2, positive definite, as σ_1 and σ_2 increases the Lyapunov function increase as well, and it is zero only when both σ_1 and σ_2 are zero.

The last criteria it needs to accomplish is that its time derivative is negative definite for any value different than $\dot{V}(0,0)$.

$$\dot{V_1}(\sigma_1) = \sigma_1 \dot{\sigma_1} < 0 \tag{3.48}$$

$$\dot{V_2}(\sigma_2) = \sigma_2 \dot{\sigma_2} < 0 \tag{3.49}$$

To accomplish what equations (3.48) and (3.49) state, a control action must be chosen according.

The designing of the control action is composed as the sum of two terms, the first term is the control action that needs to be applied once the state is on the manifold, in which case $\dot{\sigma}(h, v) = 0$, called equivalent control, and the second term is a pushing action towards the manifold.

Using the model equations of the vehicle (3.35) In the term $\dot{\sigma}_1$ appears the torque applied to the motors $\tau_R + \tau_L$, which means it can be controlled, by computing the control action to make $\dot{\sigma}_1 = 0$ the equivalent control U_{eq} is obtained. Defining $\tau_R + \tau_L$ as U_1.

$$\dot{\sigma}_1 = \frac{\partial v_{d\epsilon(h)}}{\partial h}(v_L - v) - \dot{v}_x = \frac{\partial v_{d\epsilon(h)}}{\partial h}(v_L - v) - \frac{U_1}{mr} + \frac{2B_f}{mr^2}v_x + \omega^2 b = 0 \quad (3.50)$$

The resulting U_{1eq} control action is therefore,

$$U_{1eq} = \left(\frac{\partial v_{d\epsilon(h)}}{\partial h}(v_L - v) + \frac{2B_f}{mr^2}v_x + \omega^2 b\right) mr \quad (3.51)$$

Assigning $\frac{2B_f}{mr^2}v_x + \omega^2 b = f_1$, results into the following simplified equation.

$$U_{1eq} = \left(\frac{\partial v_{d\epsilon(h)}}{\partial h}(v_L - v) + f_1\right) mr \quad (3.52)$$

So, according to the Adaptive Cruise Control algorithm, the U_{1eq} has to be equal to,

$$U_{1eq} = f_1 mr + \begin{cases} \frac{mr}{T}(v_L - v) & h - h_c \leq -\epsilon \\ \frac{mr}{2\epsilon T}(\epsilon - h_c - h)(v_L - v) & |h - h_c| \leq \epsilon \\ 0 & h - h_c \geq \epsilon \end{cases} \quad (3.53)$$

The other part of the control action is the pushing term, this is chosen to be a discontinuous control action that depends on the sign of the hyperplane. This

transforms the system into a variable structure system.

$$U_{1c} = \begin{cases} K & \sigma < 0 \\ -K & \sigma > 0 \end{cases} \tag{3.54}$$

Where K is a positive constant greater than 0, the control action can be synthesized as $-Ksign(\sigma)$, where $sign(\sigma)$ is the sign of the sliding surface for a given state. Using a control action with the form $U_1 = U_{1c} + U_{1eq}$ we obtain the following control structure:

$$U_1 = Ksign\sigma_1 + \left(\frac{\partial v_{de(h)}}{\partial h}(v_L - v) + f_1 \right) mr \tag{3.55}$$

Substituting the control action given by equation (3.55) into (3.50) and according to the time derivative Lyapunov candidate function (3.48), ideally, we have that:

$$\dot{V}_1(\sigma_1) = \sigma_1 \dot{\sigma}_1 = -\frac{K}{mr} sign(\sigma_1)\sigma_1 = -K|\sigma_1| < 0 \tag{3.56}$$

Guaranteeing at least the asymptotic stability. To be more rigorous and guarantee the exponential stability and as a consequence a finite time convergence, the gain has to be design in such a manner that the following inequality holds.

$$\dot{V}_1(\sigma_1) = \sigma_1 \dot{\sigma}_1 = -\frac{K}{mr} sign(\sigma_1)\sigma_1 = -\frac{K}{mr}|\sigma_1| \leq -\gamma|\sigma_1| \tag{3.57}$$

With the expression (3.57) we can find an upper bound, for the time it would take the system to reach the manifold. by integrating $\dot{V}_1(\sigma_1) \leq -\gamma|\sigma_1|$

$$t_r \leq \frac{|\sigma_1(0)|}{\gamma} \tag{3.58}$$

The same procedure used to obtain U_1 is applied to obtain U_2, where $U_2 = \tau_R - \tau_L$. The first step is to calculate the equivalent control U_{2eq}, using the model equations of the vehicle (3.35) and $\dot{\sigma}_2$.

$$\dot{\sigma}_2 = \dot{d} - l\dot{\omega} - (1 + \tan^2\theta_e)\dot{\theta}_e(v_x - \omega d) - \tan\theta_e(\dot{v}_x - \dot{\omega}d - \omega\dot{d}) \tag{3.59}$$

Expression (3.59) can be further develop as:

$$\dot{\sigma}_2 = (1+\omega\tan\theta_e)\dot{d} + \alpha(d\tan\theta_e - l)(\frac{aU_2}{r} - \beta_2\omega + mb\omega v_x) - (1+\tan^2\theta_e)\dot{\theta}_e(v_x - \omega d) \\ - \tan\theta_e(\frac{U_1}{mr} - \beta_1 v_x - b\omega^2) \tag{3.60}$$

Where the following renaming of variables has been done in order to simplify the calculation $\alpha = \frac{1}{I_z+mb}$, $\beta_1 = \frac{2B_f}{mr^2}$, $\beta_2 = \frac{2B_f a^2}{mr^2}$.

Following the same procedure as above and equaling σ_2 to 0 we have the following expression:

$$U_{2eq} = \frac{r}{\alpha a(l - d\tan\theta_e)}\Big[(1+\omega\tan\theta_e)\dot{d} - \alpha(d\tan\theta_e - l)(\beta_2\omega - mb\omega v_x) \\ -(1+\tan^2\theta_e)\dot{\theta}_e(v_x - \omega d) - \tan\theta_e(\frac{U_1}{mr} - \beta_1 v_x - b\omega^2)\Big] \tag{3.61}$$

The same procedure applied to obtain the control action term that pushes the state trajectories to the hyperplane is done for control action U_2

$$U_{2c} = \begin{cases} K\frac{r}{\alpha a(l-d\tan\theta_e)} & \sigma < 0 \\ -K\frac{r}{\alpha a(l-d\tan\theta_e)} & \sigma > 0 \end{cases} \tag{3.62}$$

Where K is a positive constant greater than 0. The resulting control action has the form

$$U_2 = \frac{r}{\alpha a(l - d\tan\theta_e)}\Big[K + (1+\omega\tan\theta_e)\dot{d} - \alpha(d\tan\theta_e - l)(\beta_2\omega - mb\omega v_x) \\ -(1+\tan^2\theta_e)\dot{\theta}_e(v_x - \omega d) - \tan\theta_e(\frac{U_1}{mr} - \beta_1 v_x - b\omega^2)\Big] \tag{3.63}$$

Substituting the control action given by equation (3.63) into (3.59) and according to the time derivative Lyapunov candidate function (3.49), ideally, we have that:

$$\dot{V}_2(\sigma_2) = \sigma_2\dot{\sigma}_2 = -K\,sign(\sigma_2)\sigma_2 = -K|\sigma_2| < 0 \tag{3.64}$$

Guaranteeing at least the asymptotic stability. To be more rigorous and guarantee

the exponential stability and as a consequence a finite time convergence, the gain has to be design in such a manner that the following inequality holds.

$$\dot{V}_2(\sigma_2) = \sigma_2\dot{\sigma}_2 = -Ksign(\sigma_2)\sigma_2 = -K|\sigma_2| \leq -\gamma|\sigma_2| \tag{3.65}$$

With the expression (3.57) we can find an upper bound, for the time it would take the system to reach the manifold. by integrating $\dot{V}_1(\sigma_1) \leq -\gamma|\sigma_1|$

$$t_r \leq \frac{|\sigma_2(0)|}{\gamma} \tag{3.66}$$

So far it has been assumed an ideal case where the model parameters are exactly known, however this is far from reality, we might have a good estimation of the parameters but not an exact knowledge in addition, some parameters might change through time depending on the conditions they are exposed, therefore the equivalent control becomes an estimation $\hat{U}_{eq}$, when substituting the the equivalent control estimation into the derivative Lyapunov function candidate $\dot{V}$ the following system is obtained.

$$\dot{V} = -Ksign(\sigma) + |\hat{U}_{eq} - U_{eq}| < 0 \tag{3.67}$$

To ensure stability K gain has to be large enough to repel the term $|\hat{U}_{eq} - U_{eq}|$ which can be bounded by its maximum uncertainty value.

$$|\hat{U}_{eq} - U_{eq}| \leq \phi_{max} \tag{3.68}$$

By making $K \geq \phi_{max}$ Lyapunov asymptotic stability is guaranteed.
Another consideration to take into account are the disturbdances, if they are large enough to forbid the control action from pushing the states into the manifold, sliding mode control will not occur, that is why by having an estimation of the bounded disturbances it is possible to find a K able to reject them.

Finally, the control actions obtained U_1 and U_2 have to be related to the torque that has to be given to each separate wheel. From the definition of $U_1 = \tau_R + \tau_L$

and $U_2 = \tau_R - \tau_L$, we can obtain the following relationship.

$$\begin{aligned} \tau_R &= \frac{U_1 + U_2}{2} \\ \tau_L &= \frac{U_1 - U_2}{2} \end{aligned} \tag{3.69}$$

3.3 Kinematic Model

From the angular velocity of the wheels calculated from the encoders attached to each wheel, the linear and angular velocities of any point of the vehicle can be calculated, for simplicity the center of mass.

In the ideal case, it is assumed that the wheels do not slip, do not get deformed by the weight of the vehicle, and there are no bumps therefore, the linear velocity on the 'Y' axis is considered to be zero, and the linear velocity along X axis as well as the angular velocity can be computed by. For a differential steering vehicle, such as the one modelled, the velocity of the center of mass is given by the following formula.

$$\begin{bmatrix} V_x \\ V_y \\ \omega \end{bmatrix} = 2R \begin{bmatrix} 1 & 1 \\ 0 & 0 \\ \frac{2}{W_d} & -\frac{2}{W_d} \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} \tag{3.70}$$

Where V_x is the lineal velocity on the X axis, V_y is the lineal velocity on the Y axis and ω is the angular velocity around the Z axis. R is the wheel radius which is considered to be the same on the two front wheels W_d is the distance between the front wheels and ω_r, ω_l are the right and left wheels angular velocities.

To calculate the angular velocity needed for each wheel, given a reference linear velocity V_x and angular velocity ω we can make use of equation (3.70), and calculate its inverse to obtain the inverse kinematics.

$$\begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} = \frac{1}{4R} \begin{bmatrix} 1 & \frac{W_d}{2} \\ 1 & -\frac{W_d}{2} \end{bmatrix} \begin{bmatrix} V_x \\ \omega \end{bmatrix} \tag{3.71}$$

From the odometry it is also possible to obtain an estimate of the trajectory the robot, this is done by integrating the linear and angular velocity through time.

$$\begin{aligned} x(t) &= \int_0^t V_x \cos\theta(t) dt \\ y(t) &= \int_0^t V_x \sin\theta(t) dt \\ \theta(t) &= \int_0^t \omega dt \end{aligned} \tag{3.72}$$

Since the data is obtained at discrete amount of time, the equation (3.72) is approximated by:

$$\begin{aligned} x_k &= x_{k-1} + V_{x_{k-1}} \cos\theta_{k-1}\Delta t \\ y_k &= y_{k-1} + V_{x_{k-1}} \sin\theta_{k-1}\Delta t \\ \theta_k &= \theta_{k-1} + \omega_{k-1}\Delta t \end{aligned} \tag{3.73}$$

So far, we have always assumed that the vehicle wheels do not slip, there are no bumps and there is no deformation of the wheels, however, in the real world this does not apply, and the problems mentioned above occur quite often, this rise to problems such us bad localization in autonomous vehicles, that is why these kind of systems must have an extra localization strategy that does not accumulate error such as a lidar or a landmark system.

4 Implementation

4.1 Controller Implementation

The main goal of this thesis is to implement a series of controllers based on sliding mode control to ensure the follower vehicle follows the leader, keeping a safety distance given by the Adaptive Cruise Control algorithm, described by equation (3.38).

Three different control algorithms have been implemented, the first one, is a first order sliding mode control (FOSMC) based on an on/off action, the second algorithm is a variation of the first one, also belonging to the family of (FOSMC) with the difference that the control action is defined by a Sigmoid, which results into different properties and the last control algorithm is a second order sliding mode controller (SOSMC), called super-twisting algorithm (STA).

These controllers have been chosen because they present different properties that are interesting to compare and analyze.

The control action is computed at a fixed time rate, in order to have a stable frequency when computing the sliding surfaces state values and the control actions. This is done with the implementation of a callback function that happens every time an interruption of the timer occurs. The frequency execution is, somehow, determined by the hardware, mainly by the resolution of the encoders and the time it takes to convert the line sensor output. The data obtained from the sensor, is the one used to to compute the control actions.

Ideally the frequency of the control action in sliding mode control is infinite, however this is not realistic due to the delays caused by mechanical elements such as the backlash of the reduction gear and software delays caused. In addition, a high frequency signal can generate undesired effects in the control such as “chattering” and it might damage the hardware elements.

Therefore, the control action frequency seeks for a compromise between robustness and avoiding undesired effects. After some iterations and taking always into account the time limitations caused by the lecture of the encoders, the final control action frequency is 100Hz.

To know how the DC motors will behave, they are modelled.

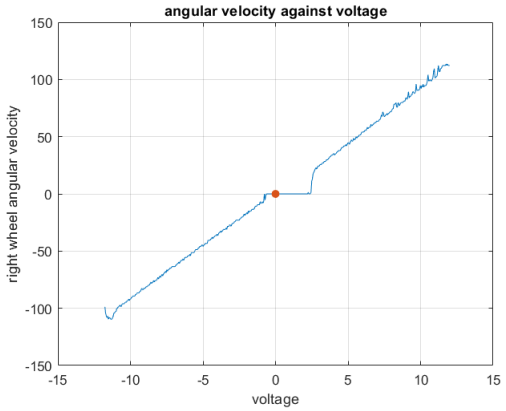

Figure 4.1: Plot of right wheel angular velocity against voltage for motor modelling

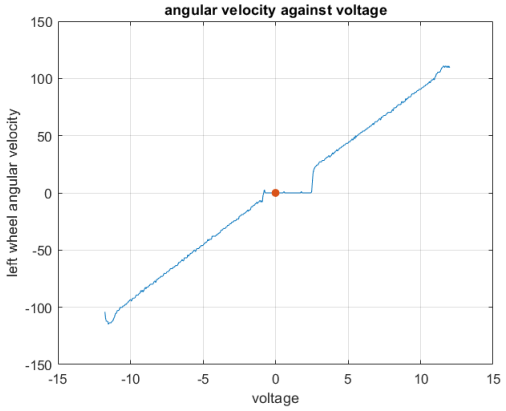

Figure 4.2: Plot of left wheel angular velocity against voltage for motor modelling

Using "polyfit" function, with degree three, in MATLAB to find a polynomial function that parameterizes the resulting angular velocity given a voltage, we obtain that:

$$\omega_L = 0.0090V^3 + 0.0079V^2 + 8.4396V - 2.1376$$

$$\omega_R = 0.0087V^3 + 0.0382V^2 + 8.4266V - 2.7130$$

The second and third order coefficients are small enough to be neglected, therefore the motor behaviour is approximately linear and the constants for each motor are, $K_L = 8.4396 - 2.1376$ and $K_R = 8.4266V - 2.713$ for the left and right motor respectively.

4.1.1 First Order Sliding Mode Control (on/off)

The first implementation is a first order sliding mode controller (FOSMC) based on an on/off or rely control. This method is applied to control the distance to the line and velocity. Using equations (3.55) and (3.63) and taking into account the parameter uncertainties. The gain K for each one of the control actions can be computed.

Considering that the maximum parameters uncertainty is given by ϕ_{max},

$$|\hat{U}_{eq} - U_{eq}| < \phi_{max}$$

By choosing a gain $K > \phi_{max}$ we can assure Lyapunov stability, however it should be tested because in real world application there are disturbances perturbing the system.

After testing different parameters and seeing that the dynamics of the motor moving backward are quite different than the moving forward the control actions chosen for velocity control and distance to the line control respectively are,

$$U_{velocity} = U_{1eq} + 1.8sign(\sigma) + 4$$

$$U_{distance} = U_{2eq} + 3 * sign(\sigma)$$

Where U_{1eq} and U_{2eq} are defined by equations (3.53) and (3.61) and the function sign is described by,

$$U = \begin{cases} K & \sigma < 0 \\ -K & \sigma > 0 \end{cases} \tag{4.1}$$

The main drawback of this control method, is that it appears the "chattering" effect due to the high frequency switching.

4.1.2 First Order Sliding Mode Control (sigmoid)

A softer approximation to the on/off control is the implementation of a control signal based on a sigmoid, which is continuous and differentiable. The resulting control action is the following,

$$u = -K\frac{\sigma}{\sigma + \epsilon} \tag{4.2}$$

where K is a positive constant, $\frac{\sigma}{\sigma+\epsilon}$ defines the sigmoid function and ϵ is the parameter that allows tho modify the slope of the sigmoid.

Figure 4.3 shows a plot for a constant $K = 1$ and for different values of ϵ equal to

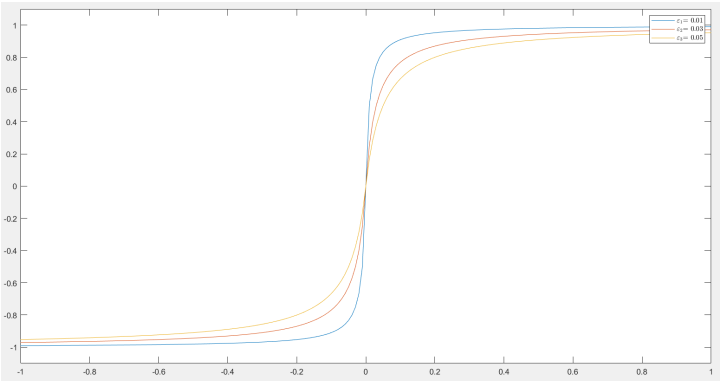

Figure 4.3: Plot of a sigmoid with different ϵ

0.01, 0.03 and 0.05. Notice that the larger ϵ the softer the slope is, and it becomes further to an ideal on/off signal, the chosen ϵ value must be a compromise between a control action with high enough frequency to force the state to move towards the manifold but at the same time not too high that it could damage the actuators, on the other hand if ϵ is too large we risk the system no to be enough reactive. This approach attenuates the chattering effect but as a trade-off of losing robustness.

After some iterations the final equations that define the control action are,

$$U_{velocity} = U_{1eq} + 1.8\sigma/|(\sigma + 0.025)| + 4$$

$$U_{distance} = U_{2eq} + 3 * \sigma/|(\sigma + 0.001)|$$

Where the parameters defining the shape of the sigmoid are $\epsilon_1 = 0.025$ and $\epsilon_1 = 0.001$

4.1.3 Second Order Sliding Mode Control (Super-twisting algorithm)

The last approach implemented is the super-twisting algorithm (STA) which is a second order sliding mode control method (SOSMC), this algorithm is composed by two control actions, one which is discontinuous and depends on the sign of sigma, and another one which is the integration of a discontinuous signal. This method allows to avoid the undesired chattering effect. The control action is formulated as follows.

$$U = \hat{U}_{eq} + K(U_1 + U_2)$$

Where,

$$U_1 = -k_1|\sigma|^{\frac{1}{2}} sign(\sigma)$$

$$\dot{U}_2 = -k_2 sign(\sigma)$$

In order to calculate $\dot{u}_2$ the integration has to be done,

$$U_2 = -K_2 \int_{t_0}^{t_f} sign(\sigma) dt$$

The integration operation is done in discrete chunks of time since the last integration,

$$U_2 = U_2 + -k_2 sign(\sigma)\Delta t$$

An anti-windup has also been implemented in order to prevent limit the integral value. The resulting control action is given by,

$$\hat{U}_{eq} + 1.175(8\sqrt{|\sigma|} sign(\sigma) + U_{2c})$$

Where $U_{2c} = U_{2c} - 2sign(\sigma)\Delta t$, and Δt is a constant sample time of 10 milliseconds.

4.2 Timer Configuration

During this project the timers have been widely used to perform many tasks, such as executing functions at a constant rate of time, generate PWM signals, retrieve counter value once an interrupt occurs to read, for instance, the distance with the vehicle ahead. The procedure to configure timers with an STM32 device is as follows.

Timers in STM32 are configured using the following parameters: (F_{clock}, $ClockDivision$, $Prescale$, $Period$, $AutoReload$). Where F_{timer} is the resulting frequency obtained at which the timer is going to be working, F_{clock} is the internal clock frequency of the timer, $ClockDivision$ is, as the name indicates, the division it is performed on the F_{clock}, which can be 1, 2 or 4 and the $Prescale$ is another division that can be applied to the F_{clock} but it gives more flexibility since it can go from 0 to $2^{PrescaleBits} - 1$, $PrescaleBits$ is the prescale resolution, it may vary depending on the timer. Finally, a $Period$ must be defined, the value of the period indicates the number until the timer will count (up/down), once this value is reached the timer will overflow, the $Period$ resolution is given by $PeriodBits$, resetting the counter number and applying the $AutoReload$ value to the counter.

The equation that gives the resulting timer frequency is shown below.

$$F_{timer} = \frac{F_{clock}}{ClockDivision \times (Prescale + 1)} \tag{4.3}$$

The timer cycle, when configured in counter up mode is determined by the following equation

$$T_{timer} = \frac{1 + Period - AutoReload}{F_{timer}} \tag{4.4}$$

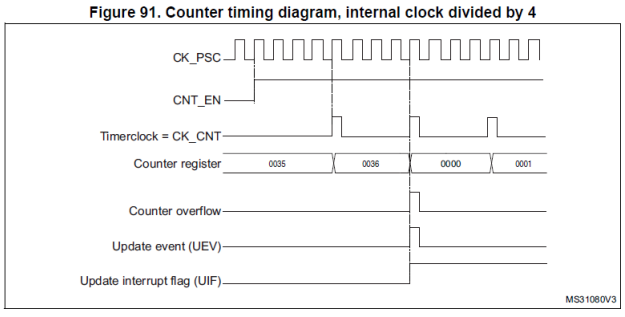

Figure 4.4: Example of timer configuration parameters [8]

4.3 DC Motor Control

The motor is fed with a with pulse wide modulation (PWM) signal.

To obtain the actual speed we use the encoder (2channel Hall-effect) attached to an interruption that counts every time there is a change, another timer is programmed to have an interruption every few milliseconds, then the amount of pulses of the encoder are counted and divided by the total amount of time between timer interruption, and the angular speed of each wheel is obtained.

Given that the encoder has two channels it is possible to know the direction at which it is spinning.

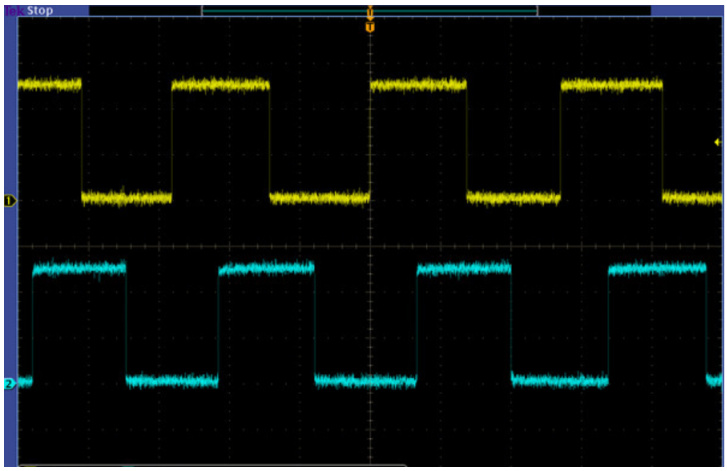

Figure 4.5: Encoder pulses captured by an oscilloscope channel A (yellow), channel B (blue) [21]

When the wheel is moving counterclockwise, the yellow signal has to be seen from left to right, therefore, when there is a falling edge in channel A the channel B will be high. On the other hand, if the wheel is moving clockwise, when there is a falling edge in channel A the channel B will be low, and the direction is always determined.

The two motor supply wires are connected to a dual full bridge driver (L6205PD) outputs (output 1, output 2). The truth table of the driver is shown in the figure 4.1

If both outputs are either high or low the motor will not move, however, if output 1 is high and output 2 is low the motor will turn clockwise and if output 1 is low and output 2 is high the motor will turn counterclockwise.

INPUTS			OUTPUTS	
EN	IN1	IN2	OUT1	OUT2
L	X	X	High Z	High Z
H	L	L	GND	GND
H	H	L	Vs	GND
H	L	H	GND	Vs
H	H	H	Vs	Vs

Table 4.1: dual full bridge driver truth table [25]

The inputs of the driver come from the PWM signals generated by the timer 1. Timer 1 is configured to generate two PWM signals, one for each motor, with its negated signal, each one of the PWM signal together with its negated are wired to input 1 and input 2 of the full bridge, respectively.

The configuration of timer 1 is as follows: $F_{clock} = 168MHz$, since timer 1 uses the APB1 bus, $ClockDivision = 1$, $Prescale = 0$, $Period = 8399$, $AutoReload = 0$, knowing that $PrescaleBits = 16$ and $PeriodBits = 16$ which give a $Prescale$ and $AutoReload$ resolution of $= 65536$. The resulting timer frequency and its period is given by:

$$F_{t1} = \frac{168 \times 10^6}{1 \times (0+1)} = 168 \times 10^6 \text{Hz}$$

$$T_{t1} = \frac{8400}{F_{t1}} = 50\mu s$$

The Pulse value that determines the PWM analog value once it matches the counter is modified during the execution of the code according to the control law and it affects the angular velocity of the wheels.

Attached to the motor there is a two-channel hall effect sensor, providing a 48 counts per revolution of the motor shaft when counting both edges of both encoder channels, coupled to the motor shaft there is a reduction gear of 9.68:1 gear ratio, when the motor shaft makes 9.68 revolutions the wheel has turn one revolution. The following formula relates the count of pulses to the wheel revolutions.

$$PulsesToRevolution = \frac{1}{PulsesPerRevolution \times ReductionGear}$$

Substituting the expression parameters by motor values we get:

$$PulsesToRevolution = \frac{1}{48 \times 9.68} = 2.152 \times 10^{-3} \frac{Rev}{Pulses}$$

When an event occurs timer 3 and 4 both with two channels connected to each channel of the hall sensor, enter to the interrupt routine where a counter is incremented. Using the timer 7, configured to have a period of 10ms.

$$F_{t7} = \frac{84 \times 10^6}{1 \times (999 + 1)} = 84 \times 10^3 \text{Hz}$$

$$T_{t7} = \frac{1 + 839}{F_{t7}} = 10ms$$

Once it overflows it enters to the interrupt routine where the angular velocity of the wheel is calculated.

$$\omega_{wheel} = \frac{2\pi \times EncoderPulses \times PulsesToRevolution}{LapsedTime}$$

where, $LapsedTime = 10ms$.

The control action obtained from the sliding mode control is used to establish the new PWM value.

Two different approaches have been implemented in order to control the speed of the wheel, the first one is a controller compound by a proportional gain a feed-forward and an integrator, taking as a reference the desired angular velocity of each wheel separately and the angular velocity read by the encoder.

The other approach has been to model each motor, by saving a tuple of the input given to the motor and the angular speed of the wheel, by increasing the input slowly until the maximum speed of the motor, and doing the analogous for the wheel going backwards, we get the data of the angular speed given a control action, by inverting the data we get the control action given an angular speed, the data can be approximated with a linear regression and the values found are the following, in this fashion we obtain a function that returns the control action needed to get a specific angular speed.

4.4 Distance Sensor

The sensor used to get the distance between the vehicle and the vehicle in front is the ultrasound sensor HC-SR04, the sensor has four pins which are Vcc, Trigger, Echo and Ground. The Vcc pin is the supply voltage the device needs in order to work, five volts according to the datasheet. The Trigger, when activated, generates an eight cycle burst of ultrasound at 40kHz and right after sets the Echo pin to five volts, the ultrasound waves move through the air until they collide with some object and they bounce back, until the HC-SR04 ultrasound receiver detects the high frequency sound and the Echo pin is set to zero volts.

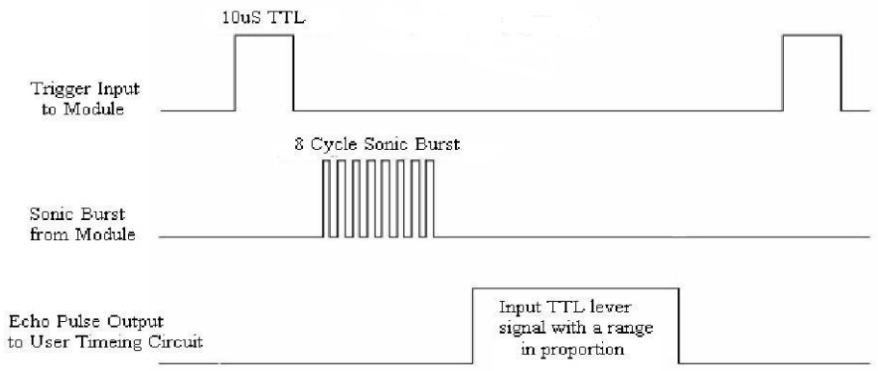

Figure 4.6: Timming diagram [17]

The time the Echo pin remained high indicates the time for the ultrasonic wave to go from the HC-SR04 device to collide with an object and come back (T_{high}). Since the speed of the ultrasound waves at 25 degrees Celsius is 340m/s (V_{sound}), taking into account that the ultrasound waved travelled twice the distance from the sensor to the object, this distance can be computed as $Distance = \frac{V_{sound}T_{high}}{2}$.

According to the data sheet of the product some considerations must be taken, the operational range of the device indicates that it can detect an object within 0.02m to 4m and an angle of 15 degrees from the receiver with an accuracy of 3mm. It is suggested that the measuring cycle be higher than 60ms in order to prevent overlap of measurements, the time the Trigger pin must be high to generate the ultrasound pulses must be of $10\mu S$.

Timer 10 of STM32 gives the option to generate a PWM signal, using the PB8 pin of the microcontroller, once the counter matches a predefined, called $Pulse$ value

set by the user, this is used to generate the Trigger signal for the HC-SR04, that creates an eight cycle burst of ultrasounds. To obtain the $60ms$ cycle mentioned on the device data sheet we set the timer parameters as follows: $F_{clock} = 168MHz$, since timer 10 uses the APB1 bus, $ClockDivision = 1$, $Prescale = 165$, $Period = 65535$, $AutoReload = 0$, knowing that $PrescaleBits = 16$ and $PeriodBits = 16$ which give a $Prescale$ and $AutoReload$ resolution of = 65536, using equations (4.3) and (4.4) we obtain the following:

$$F_{t10} = \frac{168 \times 10^6}{1 \times (165 + 1)} \approx 1.012 \times 10^6 \text{Hz}$$

$$T_{t10} = \frac{65536}{F_{t10}} \approx 64.76ms$$

The PB8 pin will remain low until the counter value matches $Pulse = 65525$ then it will become high until the counter reaches its maximum $Period$ value, set at 65535 at which an overflow will occur and the PB8 pin will become low again, therefore the time PB8 will remain high is going to be determined by:

$$T_{high} = \frac{65536 - 65525}{F_{t10}} \approx 10.87\mu S$$

As a result, an ultrasound signal is sent every $64.76ms$ with a Trigger pulse of $10\mu S$ as specified in the data sheet.

Timer 2 is used to capture the counter value every time there is a rising or falling edge on the Echo pin, since there are three HC-SR04 devices, three channels of timer 2 are attached to the Echo pin of each of the devices (channel 1 PA15, channel 2 PB3, channel 3 PB10). The HC-SR04 resolution of 3mm has been taken into account in order to calculate the timer frequency, $T_{3mm} = \frac{0.003}{340} \approx 8.82\mu S$ being $8.82\mu S$ the time needed for the ultrasound wave to travel 3mm, the timer parameters configured as follows, $F_{clock} = 84MHz$, since timer 2 uses the APB2 bus, $ClockDivision = 1$, $Prescale = 99$, $Period = 65535$, $AutoReload = 0$, knowing that $PrescaleBits = 16$ and $PeriodBits = 32$ which give a $Prescale$ resolution of 65536 and $AutoReload$ resolution of 2^{32}, the resulting time per count is given by:

$$F_{t2} = \frac{84 \times 10^6}{1 \times (99 + 1)} \approx 840 \times 10^3 \text{Hz}$$

$$T_{t2} = \frac{1}{F_{t2}} \approx 1.191\mu S$$

Since the sampling time $T_{t2} < T_{3mm}$ variations of the order of the HC-SR04 resolution will be detected.

Whenever a rising edge is detected, an interrupt will occur and a callback function will be called where the value of the timer counter will be saved in the variable $ICRisingValue$, once the return ultrasound wave is detected by the receiver the falling edge in the Echo pin will trigger a callback function, which will save the timer counter value in the variable $ICFallingValue$, if $ICFallingValue$ is greater than $ICRisingValue$ the difference is calculated and the result is multiplied by the factor that relates counts to meters, on the other hand if $ICFallingValue$ is smaller than $ICRisingValue$, means that an overflow of the counter has occurred and the total amount of counts is $(Period - ICRisingValue + ICFallingValue)$ then the result is multiplied by the factor that relates counts to meters.

4.5 Line Sensor

The line sensor is an array of eight infrared emitters and infrared sensors, depending on the infrared amount of light the sensor increases or decreases its voltage, the higher the amount of infrared light the higher the voltage and the vice versa. According the data sheet of the sensor the separation of between sensor is $0.375" = 0.9525cm$.

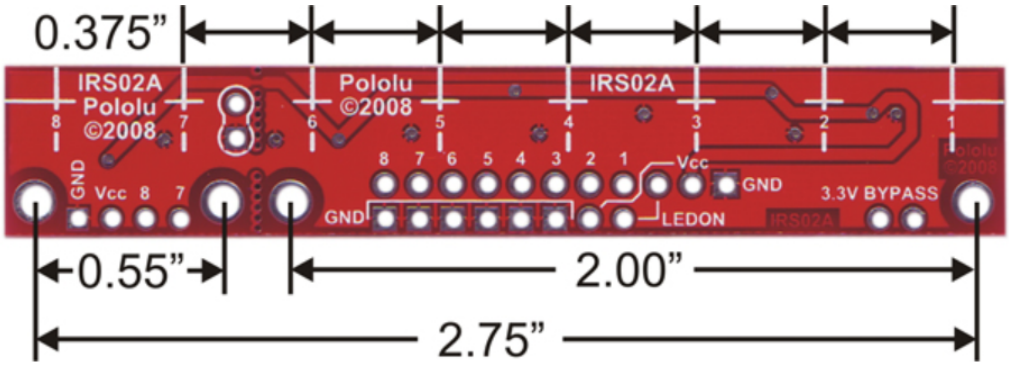

Figure 4.7: Infrared array sensor [22]

Each sensor output is wired to an analog to digital converter channel of the microcontroller (ADC1). The internal clock used by the ADC1 runs at a frequency of 84MHz, the clock is prescaled by 4 and since a resolution of 12bits is required the ADC will need to perform 15 cycles, therefore, the time it takes to convert the analog output signal from the sensor into a digital one is given by the following expression:

$$T_{conversion} = \frac{15 \times 4}{84 \times 10^6} \approx 0.714\mu s$$

The conversion is done every time there is an interruption on timer one, that is every $50\mu s$, the converted value of each one of the eight channels is saved in an array. Since the resolution is 12bits the value of the conversion range is 0 to 4095, considering the middle of the sensor as the zero distance point, the values of the sensor outputs from one to four are considered as positive distance, and the values of the outputs from five to eight as negative. The vehicle is following a black tape in a white background. When a sensor is close to a less reflecting surface its conversion from analog value will increase otherwise, on the other hand, when is facing a reflecting surface the value will decrease, knowing that the distance between receivers is $0.009525m$ the distance of each sensor to the middle is computed

and multiplied by the analog value obtained, this will give a value indicating how far from the middle point is the sensor.

Sensor number	Distance to middle [m]
1	0.03334
2	0.02381
3	0.01429
4	0.00477
5	-0.00477
6	-0.01429
7	-0.02381
8	-0.03334

Table 4.2: Distance to the middle IR sensor

The operation to obtain the distance of the center of the infrared sensor to the line is the following:

$$D = \frac{1}{4096}\left[0.03334(V_1 - V_8) + 0.02381(V_2 - V_7) + 0.01429(V_3 - V_6) + 0.00477(V_4 - V_5)\right]$$

where V_x is the analog to digital conversion value of the channel x and $\frac{1}{4096}$ is the scaling factor from timer counts to reflecting intensity normalized to one.

4.6 Communication System

The communication vehicle to server is done via wifi using the module Wifi ESP8266, the module can be configured via serial port, by default the Wifi ESP8266 is configured according to the paremeters of the following list

Serial communication configurations

- Baudrate 115200Bits/s
- Eight bits length word
- No parity
- One stop bit

The USART3 of the microcontroller, used to establish a serial communication with the WiFi, has to be configured to have the same configuration as the WiFi device.

The operation mode of the wifi can also be configured, in our case it is configured as a station device, since the goal is to connect to an access point. The configuration of the device is done via serial communication the order of commands are as follows:

First all all, the echo from the Wifi device is deactivated with the command "ATE0", this avoids the Wifi module to reply with the same message sent by the microcontroller.

Then with the command "AT+CWMODE=1" is set in station mode, since it is going to act as a device connecting to an access point (i.e. router).

The command AT+CIPSTA=ip,gateway,mask" defines the "ip" address, Gateway address, and a mask for the Wifi ESP8266. Where "ip = 192.168.173.X" must be in the same range as the "gateway = 192.168.173.X" and the server address (PC) "ip = 192.168.173.X", finally the mask restricts the range of ip that we are going to be working on "mask = 255.255.255.X" only if the ip of the server that has an address of the form of "192.168.173.X" will be reached.

The "AT+CWJAP=SSID,PASSWORD" connects to an access point if the network name and the password are correct, otherwise the Wifi module responds with an error code. In order to enable multiple vehicles connecting to the same access

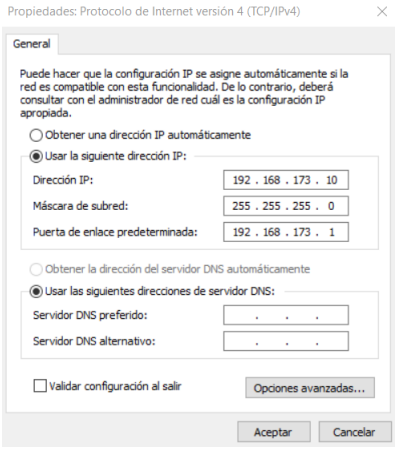

Figure 4.8: Ip configuration

point "AT+CIPMUX=1" command must be sent to the Wifi ESP8266, this is the case since several vehicle can be communicating to the server.

Finally a connection must be started, by sending the command "AT+CIPSTART=link ID,type,remote IP,remote port[,UDP localport,UDP mode]", where link ID is the connection ID given since multiple connection is allowed, the type field specifies the protocol used either TCP or UDP, the remote IP and remote port are the server address and the port where the messages will be directed and finally if UDP protocol is selected a local port has to be defined, the UDP mode specifies if the destination peer might or might not change.

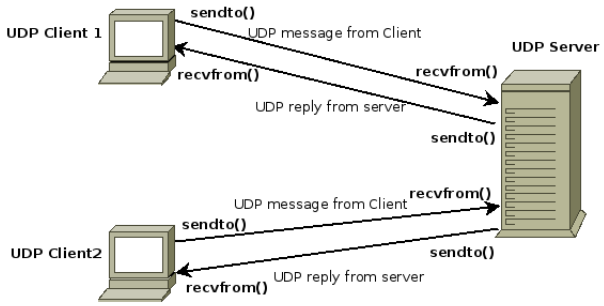

Figure 4.9: UDP connection client-server [24]

For the purposes of the project the UDP protocol has been chosen, the main differences between UDP and TCP protocol are:

- TCP is a connection-oriented protocol meaning that it requires to establish a

connection between the sender and the receiver before sending data, on the other hand, UDP does not establish a connection.

- TCP is also a reliable protocol, it verifies that the data sent is well delivered, checking also possible message corruption and re-sending the message if necessary. UDP does not check if the message is well delivered which can lead to data loss, therefore, it is not reliable.
- TCP provides flow control, ensuring that the sender does not send too many messages that might saturate the receiver, this is achieved by the implementation of two buffers, one for messages to send and another one to receive, the receiver tells the sender the room left in the receive buffer not to drop messages away if full, UDP does not implement a flow control, if a receiver buffer is full messages would be drop apart.
- TCP does ordering, guarantying the order of packets sent, UDP, however, send the packets without ordering. them

Although TCP has more features than UDP, due to its paradigm is slower and not a good option if the data requires to be delivered fast and efficiently. For messages such as odometry, velocity reference or starting/stopping the vehicle, the sender does not need to verify if the data has been well received, as long as the received data has the least delay possible.

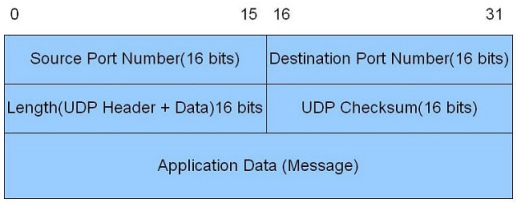

Figure 4.10: UDP protocol structure [14]

The server is running in a PC and must have the same IP address as the one specified on the client packet, the port at which the packets will be sent is also specified by the UDP protocol and it is the port at which the server will be listening, the server respond to client messages or petitions and can also send commands to the microcontroller.

5 Data Analysis

5.1 Data Collection

In order to analyze the controller performance, specific data is required. The data selected to study the behaviour of the system under the three different algorithms is the sliding hyperplanes σ_1 and σ_2, the distance between vehicles and the two control actions computed for the Adaptive Cruise Control algorithm and the line follower. All this variables might vary a lot from one iteration of the control loop to another, therefore it is crucial to retrieve all the data used.

The first approach, was to send the data via WiFi to the computer server, however this method was not feasible, because the data had to be sent at a frequency of 100Hz at least taking into account that each variables has a size of four bytes and has to be sent every 10ms, transmission speed should be at least,

$$\frac{(\text{number of bytes to send}) \times (\text{size of a byte})}{\text{sending rate}} = 19200\text{bits/s}$$

A priori seems possible, but actually many packets were lost.

The solution chosen, has been to store all the data in the ram memory, more precisely in the stack region, and once the recording has completed the data is sent via serial communication to the PC, where with a program developed using python, the data is read and store in a text file.
Finally, the data is loaded in MATLAB, analyzed and plotted.

The tests executed emulate the behaviour of a vehicle joining to a plattoning system, where the vehicle is moving at its maximum velocity until a vehicle ahead is found, then it adapts its velocity to maintain a safety distance given by the policy in (3.38). A similar test has been done with each one of the controllers.

5.2 Data Analysis

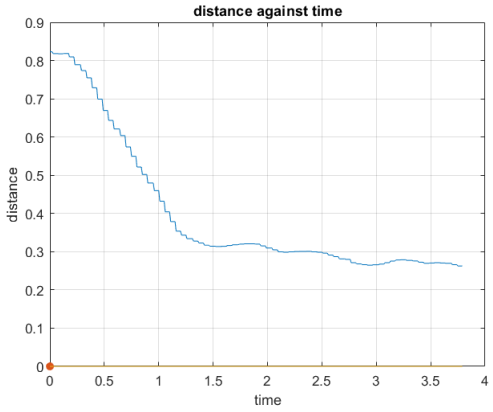

Figure 5.1: Distance to the obstacle ahead on/off control

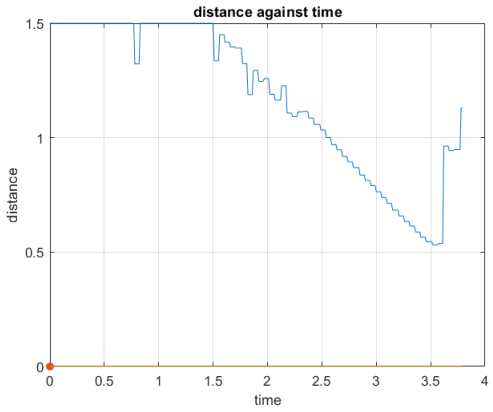

Figure 5.2: Distance to the obstacle ahead sigmoid function control

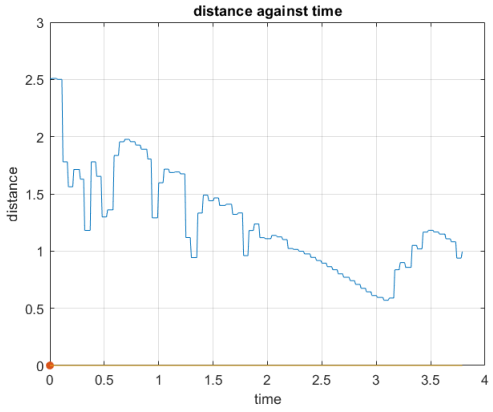

Figure 5.3: Distance to the obstacle ahead super-twisting algorithm

As we can appreciate in figures 5.1, 5.2 and 5.3 in the three cases the vehicle starts without any close obstacle in front and as the time passes the distance to the obstacle ahead decreases, meaning that it is approaching to a vehicle.

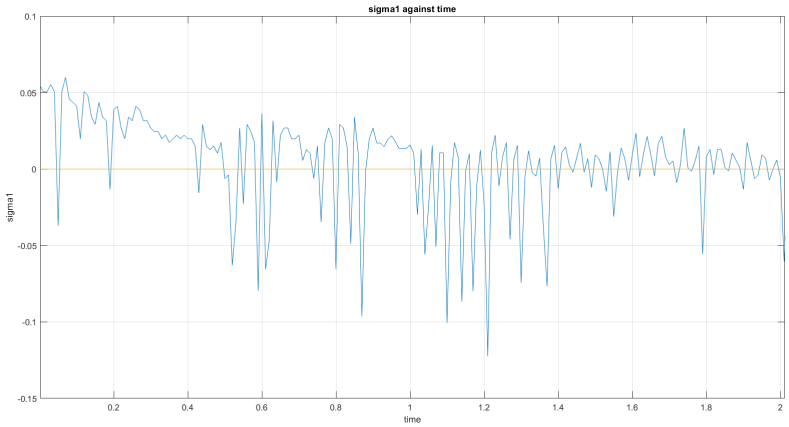

Figure 5.4: σ_1 ACC algorithm on/off

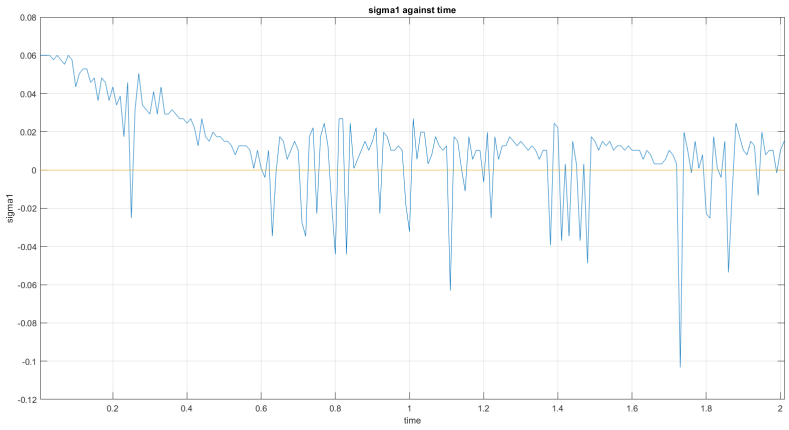

Figure 5.5: σ_1 ACC algorithm sigmoid function

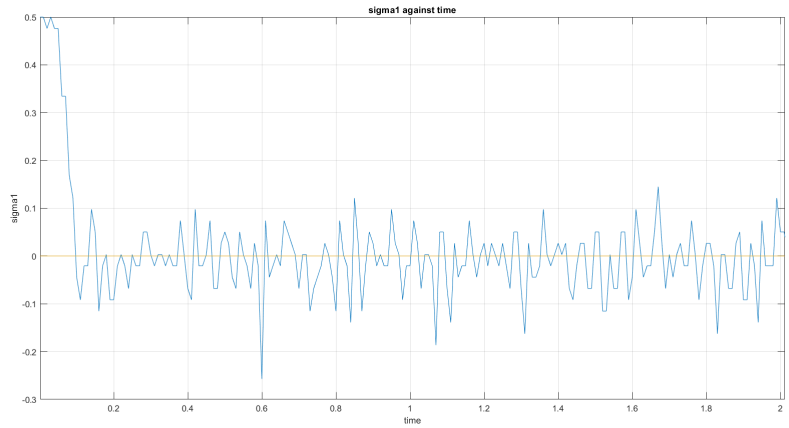

Figure 5.6: σ_1 ACC algorithm super-twisting algorithm

These three plots show the evolution of the sliding variable, in an ideal sliding mode σ would be pushed towards the x axis and remain on it thereafter, nevertheless this does not occur in real world because the signal does not have an infinite

switching frequency that forces the system to remain on the manifold, there are also computational and mechanical delays that generate those oscillations around the manifold.

To compare the performance of each one of the controllers the mean value and the standard deviation is calculated, with this values we can have an idea of how much differs from the manifold and how disperse is the data.

The mean values obtained for each one of the plots are, $\mu_1 = 0.0027$, $\mu_2 = 0.0099$ and $\mu_3 = -0.0023$, the standard deviations by controller are, $std_1 = 0.0257$, $std_2 = 0.0233$ and $std_3 = 0.0884$ we can appreciate that all mean values are quite close to each other, on the other hand, the standard deviation of the on/off controller and the sigmoid function are slightly lower than the super-twisting algorithm, this makes sense if we see the control action of each controller, shown below in figures 5.7, 5.8 and 5.9.
It is interesting to see how in figure 5.6 how σ reaches the manifold starting from an initial condition far from it.

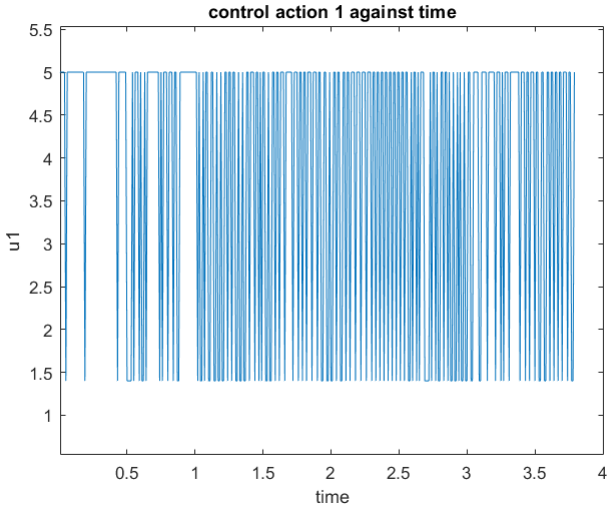

Figure 5.7: control action from ACC on/off control

Control action of on/off controller and sigmoid function are more aggressive than the super-twisting algorithm, this causes the system to react faster and as a consequence the σ values are closer giving a lower standard deviation, the problem of aggressive control actions is that they can damage the hardware, overheating of the driver, and they affect unmodelled dynamics which are presumably fast.

To analyze the line follower controllers, the same procedure as above is followed,

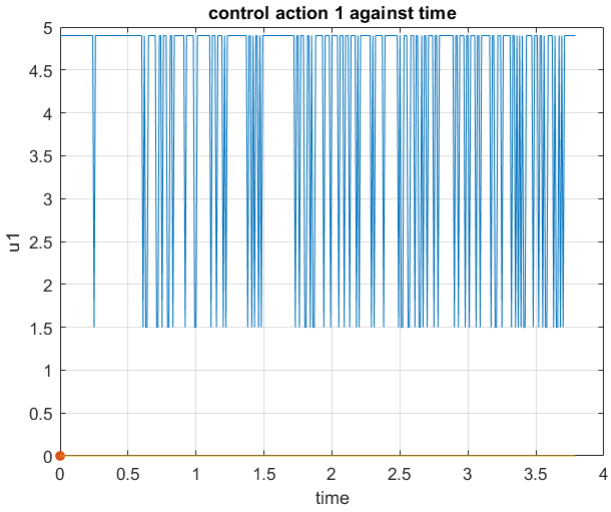

Figure 5.8: control action from ACC sigmoid fucntion control

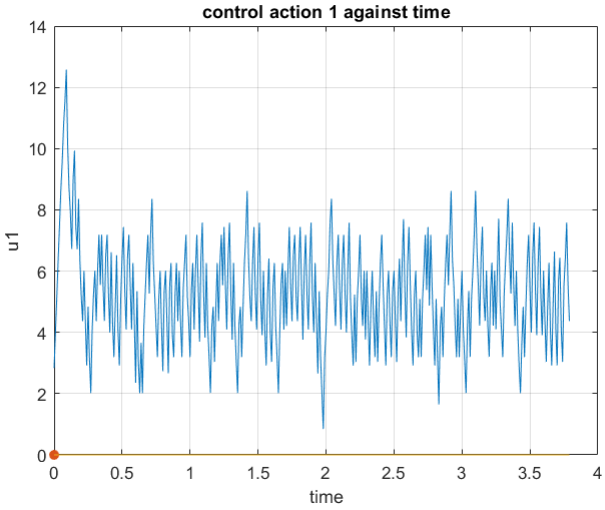

Figure 5.9: control action from ACC super-twisting algorithm control

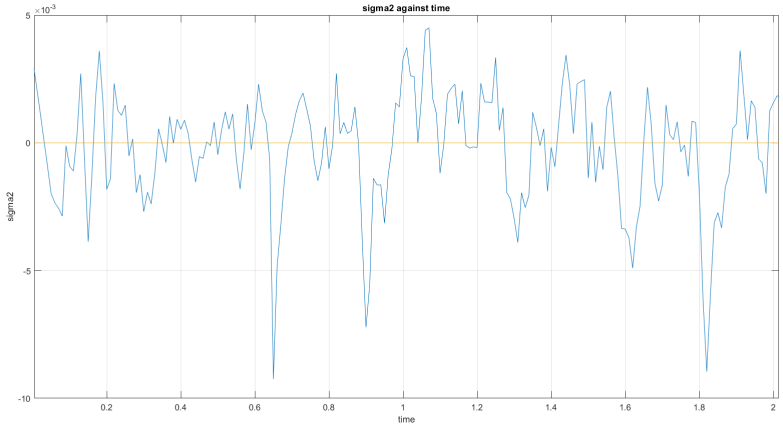

Figure 5.10: σ_2 line follower on/off controller

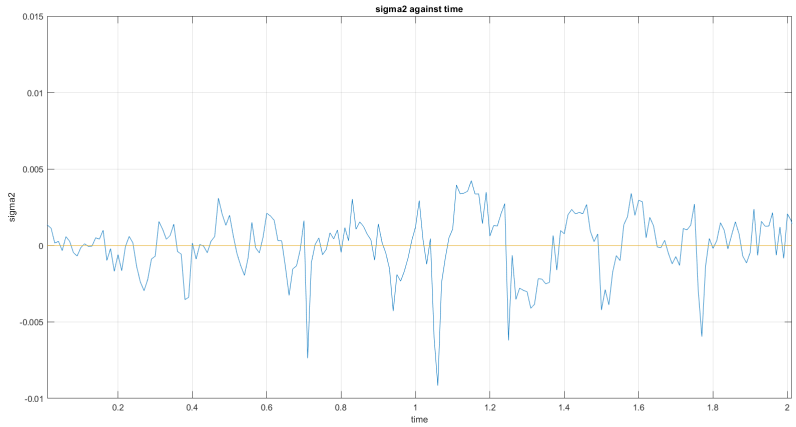

Figure 5.11: σ_2 line follower sigmoid function controller

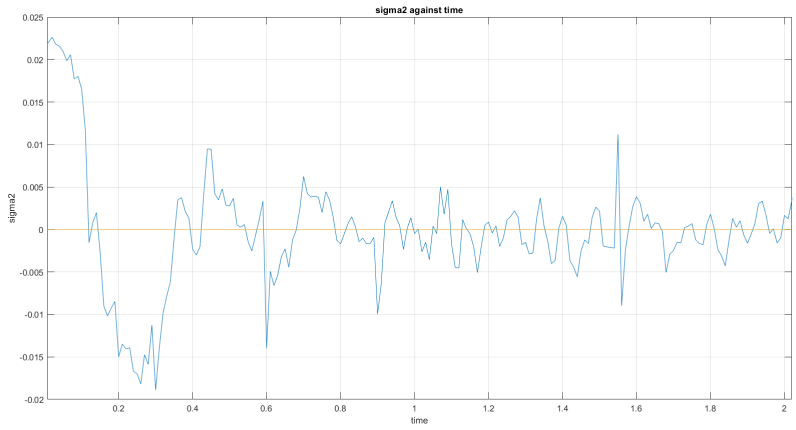

Figure 5.12: σ_2 line follower super-twisting algorithm controller

the mean value and the standard deviation are computed.

The results obtained by the data extracted are, $\mu_1 = -1.9454e^{-4}$, $\mu_2 = 4.8298e^{-4}$ and $\mu_3 = 0.0014$ and the standard deviations by controller are, $std_1 = 0.0021$, $std_2 = 0.0026$ and $std_3 = 0.0073$.

Same as before, since the on/off control is more aggressive as it can be appreciated in the figure 5.13, the mean value and standard deviation are smaller.

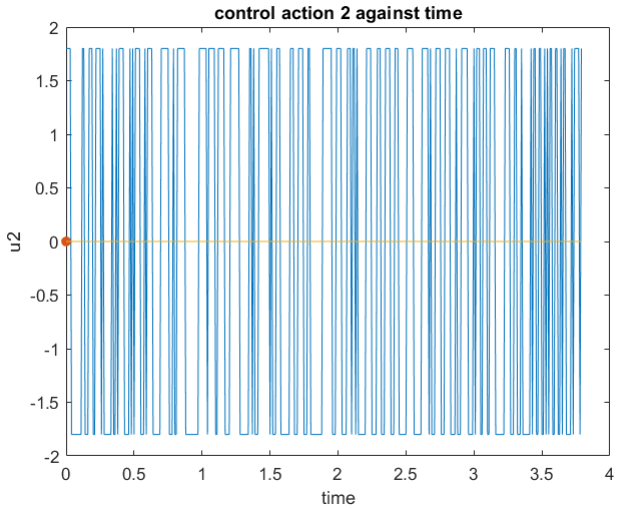

Figure 5.13: control action from line follower on/off control

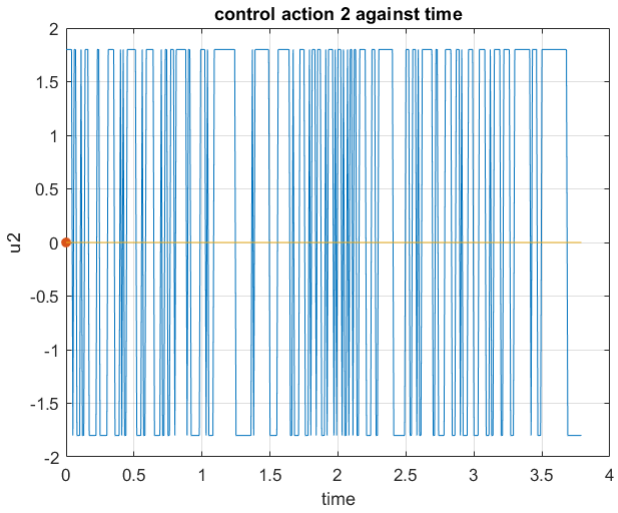

Figure 5.14: control action from line follower sigmoid function control

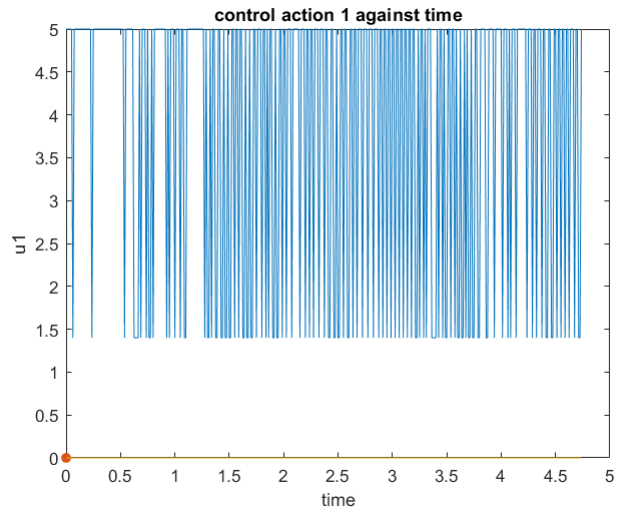

Figure 5.15: control action from line follower super-twisting algorithm control

6 Conclusions

The main goal of this thesis was to design a series of controllers based on the method sliding mode control, for a platooning system and test its performance.

The objectives have been achieved, two sliding surfaces, one to control the linear velocity and another one to control the distance to the line have been implemented and all three controllers have a correct performance. Notice, that when using the on/off control method the driver tends to overheat if the frequency and gain are too high, although this controller seems to be the most robust among the three, super-twisting algorithm is preferable, even if it is not as robust as on/off its control action is smoother, which generates less peaks of current that might damage the materials, it is an acceptable trade-off. It has been practically proved that sliding mode control could be a good solution to implement in a platooning system for adaptive cruise control.

When modelling the motor it appears to be an important deadlock, and the lower angular velocity is $23\frac{rad}{s}$ translated into linear velocity of a wheel of radius 0.035m is $V_{lin} = \omega r = 0.805\frac{m}{s}$ when moving forward, the vehicle however is limited to $0.5\frac{m}{s}$ because higher velocities cause problems on the line following control, therefore, since the desired speed is lower than the minimum speed the motor can provide, to achieve the reference velocity the control action is constantly switching, and sometimes generates peaks of velocity that push the states away to the manifold.

Although the routine where sliding surface and the control actions are calculated and applied is running at $100Hz$, it could have a higher frequency, but it is limited by the wheel velocity calculation, it would make no sense to set the routine at a higher frequency than the state variables used to compute the sliding surface.
Decreasing the encoder calculation time is not either an acceptable solution because the error in the lecture increases dramatically resulting in sudden changes of velocity that generate oscillations in the control.
A solution would be to change the actual reduction gear by another one with a higher value, increasing the resolution and at the same time decreasing the motor deadlock.

7 Environmental Impact

The hardware components used for development of the project have been inherited from previous works. The project is based on software development, therefore, the environmental impact during its development consists on the use of electricity consumed.

8 Budget

The cost of the entire project can be divided in labour and materials costs

8.1 Labour Costs

The student did not earned money for the development of this master thesis as it is considered to be under the academic environment. In case the working hours were paid, the labour costs budget would be:

Concept	PVP[€/h]	Units[h]	Total[€]
Research and documentation	20	50	1,000
Design and implementation	20	200	4,000
Data collection and analysis	20	30	600
Report writing	20	150	3000

Table 8.1: Labour Costs

8.2 Material Costs

The material used for this project is constituted by the electronic and mechanic components of the robot, development tools requiring of license and a personal computer. As mobile phone has been used as an access point to connect the robot and the server.

Concept	PVP[€]	Units[h]	Total[€]
Components of the Robot			
DC/DC Converter 12V/5V	5.49	1	5.49
DC/DC Converter 12V/3.5V	2.35	1	2.35
Motor Driver	6.49	1	6.49
Development Board	17.58	1	17.58
DC Motor	29.45	2	58.90
Gear Motor Bracket Pair	6.36	1	6.36
Scooter/Skate Wheel 12V/5V	2.16	2	4.32
Wheel Adapter 12V/3.5V	5.55	2	11.10
Support Wheel	6.45	1	6.45
Ultrasonic Sensor	16.74	3	50.22
Line Sensor	8.06	1	8.06
Wi-Fi Module	6.40	1	6.40
Battery	32.50	1	32.50
Connector	1.10	1	1.10
PCB1	21.66	1	21.66
Development Tools			
MATLAB license	0	1	0
Personal Computer	1200	1	1,200
Personal Computer	150	1	150

Table 8.2: Material Costs

Concept	Total Costs[€]
Labour Costs	8,600
Material Costs	1588.98
Subtotal	10,188.98
VAT(21%)	2,139.69
Total	12,328.67

Table 8.3: Total Costs

9 Apendix

Parameter	Value	[SI]
a	0.085	m
b	0.03	m
l	0.068	m
r	0.035	m
L	0.25	m
W	0.18	m
m	1.175	Kg
I_z	9.292×10^{-3}	Kgm^2
B_f	1.543×10^{-4}	$\frac{Kgm^2}{s}$

Table 9.1: Model parameters value

9.1 Technical Conepts

ADAS : Advanced Driver Assistance Systems, electronic systems that help the driver at some point.

ACC: Adaptive Cruise Control, a technique that adapts vehicle velocity according to a safety distance.

CACC: Cooperative Adaptive Cruise Control, it is an extension of ACC but the vehicles communicate between them.

V2V: Vehicle to vehicle, communications between vehicles.

V2I: Vehicle to infrastructure, communications between and an infrastructure such as a highway.

SMC: Sliding mode control, a control technique used on nonlinear systems.

STA: Super-twisting algorithm, is a control algorithm based on sliding mode control.

UART: Universal asynchronous receiver- transmitter it is a configurable electronic device to receive and transmit data.

LTI: In control theory a linear time invariant system does not depend on time.

Chattering: An oscillatory effect that occurs in sliding mode control due to delays

References

[1] Arnau Dòria-Cerezo et al. “Sliding mode controllers for adaptive cruise control”. Report UPC. July 2018.

[2] Olivier Salvi. “Aramis (personal rapid transit)”. Report INERIS. April 2014.

[3] Judit Ruiz Cabo. “Control of a Two-Wheeled Line Tracking Robot Using a Complete Mechanical Model”. Bachelor’s Thesis. UPC, April 2018.

[4] Domingo Biel, Arnau Dòria-Cerezo, and Josep M. Olm. “Dynamic behaviour”. Slides of a presentation. February 2020.

[5] Riccardo Furlan, Francesco A. Cuzzola, and Thomas Parisini. “Friction compensation in the interstand looper of hot strip mills: A sliding-mode control approach”. Report Elsevier. February 2007.

[6] Utkin Vadim I. and Hao-Chi Chang. “Sliding mode control on electro-mechanical systems”. Report The Ohio State University. September 2002.

[7] Jean-Jacques E. Slotine and Weiping Li. *Applied Nonlinear Control*. Prentice-Hall International Editions, 1991.

[8] *STM32F405/415, STM32F407/417, STM32F427/437 and STM32F429/439 advanced ARM®-based 32-bit MCUs*. 15th ed. STMicroelectronics. July 2017.

[9] Antonella Ferrara. “Sliding Mode Control Handout Advanced Automation and Control Course”. Slides of a presentation. Accessed: 15 of February 2020.

[10] Ben Gallup. “Sliding Mode Control: A Comparison of SlidingSurface Approach Dynamics”. Slides of a presentation. Accessed: 15 of February 2020.

[11] Wu-Chung Su. “Sliding Mode Control with Industrial Applications”. Slides of a presentation. Accessed: 15 of February 2020.

[12] ACEA. *what is a truck platooning?* URL: `https://www.acea.be/uploads/publications/Platooning_roadmap.pdf`. Accessed: 12 of April 2020.

[13] National highway traffic safety administration. *Vehicle-to-Vehicle Communication*. URL: `https://www.nhtsa.gov/technology-innovation/vehicle-vehicle-communication`. Accessed: 19 of April 2020.

[14] Stephen Cooper. *A guide to UDP (User Datagram Protocol)*. URL: https://www.comparitech.com/net-admin/guide-udp-user-datagram-protocol/. Accessed: 9 of July 2020.

[15] Mohammed Dahleh, Munther A. Dahleh, and George Verghese. *Lectures on Dynamic Systems and Control*. Accessed: 25 of March 2020.

[16] Arnau Dòria-Cerezo et al. "A first order sliding mode-based adaptive cruise controller". Report UPC.

[17] ElecFreaks. *Ultrasonic Ranging Module HC - SR04*. URL: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf. Accessed: 9 of July 2020.

[18] SAE International. *SAE International Releases Updated Visual Chart for Its "Levels of Driving Automation" Standard for Self-Driving Vehicles*. URL: https://www.sae.org/news/press-room/2018/12/sae-international-releases-updated-visual-chart-for-its-%5C%E2%5C%80%5C%9Clevels-of-driving-automation%5C%E2%5C%80%5C%9D-standard-for-self-driving-vehicles. Accessed: 7 of May 2020.

[19] Stanford lectures. "Basic Lyapunov theory". Slides of a presentation. Accessed: 25 of March 2020.

[20] Wen-Bin Lin and Huann-Keng Chiang. "Super-Twisting Algorithm Second-Order Sliding Mode Controlfor a Synchronous Reluctance Motor Speed Drive". Report Hindawi.

[21] Pololu. *34:1 Metal Gearmotor 25Dx64L mm MP 12V with 48 CPR Encoder (No End Cap)*. URL: https://www.pololu.com/product/3240. Accessed: 8 of July 2020.

[22] Pololu. *QTR-8A and QTR-8RC ReflectanceSensor Array User's Guide*. URL: https://www.pololu.com/docs/pdf/0J12/QTR-8x.pdf. Accessed: 9 of July 2020.

[23] Lisa Pradhan. *Truck Platooning: History, Benefits, Future*. URL: https://trucks.cardekho.com/en/news/detail/truck-platooning-history-benefits-future-945.html. Accessed: 27 of April 2020.

[24] Pythonic. *UDP - Client And Server Example Programs In Python*. URL: https://pythontic.com/modules/socket/udp-client-server-example.

[25] STMicroelectronics. *DMOS DUAL FULL BRIDGE DRIVER*. URL: https://pdf1.alldatasheet.com/datasheet-pdf/view/22534/STMICROELECTRONICS/L6205PD.html. Accessed: 7 of February 2020.

[26] Peloton Tech. *Method of Lyapunov Functions*. URL: https://www.math24.net/method-lyapunov-functions/. Accessed: 8 of July 2020.

[27] Peloton Tech. *V2V Communication in Platooning)*. URL: https://peloton-tech.com/v2v-communication-platooning/. Accessed: 8 of July 2020.

[28] Road traffic technology. *Project SARTRE (Safe Road Trains for the Environment)*. URL: https://www.roadtraffic-technology.com/projects/the-sartre-project/. Accessed: 27 of April 2020.

[29] Wikipedia. *Adaptive cruise control*. URL: https://en.wikipedia.org/wiki/Adaptive_cruise_control. Accessed: 29 of March 2020.

[30] Wikipedia. *Cooperative Adaptive Cruise Control*. URL: https://en.wikipedia.org/wiki/Cooperative_Adaptive_Cruise_Control. Accessed: 29 of March 2020.

[31] Wikipedia. *Eureka Prometheus Project*. URL: https://en.wikipedia.org/wiki/Eureka_Prometheus_Project. Accessed: 27 of April 2020.

[32] Wikipedia. *Futurama (New York World's Fair)*. URL: https://en.wikipedia.org/wiki/Futurama_(New_York_Worlds_Fair). Accessed: 27 of April 2020.

[33] Wikipedia. *Houdina Radio Control*. URL: https://en.wikipedia.org/wiki/Houdina_Radio_Control.

[34] Wikipedia. *Lyapunov stability*. URL: https://en.wikipedia.org/wiki/Lyapunov_stability. Accessed: 20 of March 2020.

[35] Wikipedia. *Rotating reference frame*. URL: https://en.wikipedia.org/wiki/Rotating_reference_frame. Accessed: 19 of January 2020.

[36] Wikipedia. *Sliding mode control*. URL: https://en.wikipedia.org/wiki/Sliding_mode_control. Accessed: 16 of February 2020.