

# Optimal Control for Autonomous Formula Student Car and Lap Time Simulation

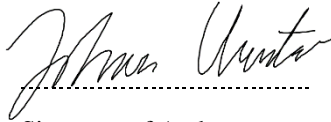
Johan Svendsen (18000450)

Programme: MSc Motorsports Engineering  
Module: P04791 – MSc Projects  
Submission: **Project progress**  
Year: 2018/19  
Word count: 10791

School of Engineering, Computing and Mathematics

## Statement of originality

Except for those parts in which it is explicitly stated to the contrary, this project is my own work. It has not been submitted for any degree at this or any other academic or professional institution.



Signature of Author

27/09/2019

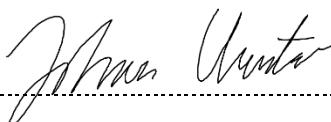
Date

**Regulations Governing the Deposit and Use of Master of Science Dissertations in the Department of  
Mechanical Engineering and Mathematical Sciences,  
Oxford Brookes University.**

1. The 'top' copies of projects submitted in fulfilment of Master of Science course requirements shall normally be kept by the Department.
2. The author shall sign a declaration agreeing that, at the supervisor's discretion, the dissertation may be submitted in electronic form to any plagiarism checking service or tool.
3. The author shall sign a declaration agreeing that the dissertation be available for reading and copying in any form at the discretion of either the project supervisor or in their absence the Head of Postgraduate Programmes, in accordance with 5 below.
4. The project supervisor shall safeguard the interests of the author by requiring persons who consult the dissertation to sign a declaration acknowledging the author's copyright.
5. Permission for anyone other than the author to reproduce in any form or photocopy any part of the dissertation must be obtained from the project supervisor, or in their absence the Head of Postgraduate Programmes, who will give his/her permission for such reproduction only to the extent which he/she considers to be fair and reasonable.

I agree that this dissertation may be submitted in electronic form to any plagiarism checking service or tool at the discretion of my project supervisor in accordance with regulation 2 above.

I agree that this dissertation may be available for reading and photocopying at the discretion of my project supervisor or the Head of Postgraduate Programmes in accordance with regulation 5 above.



Signature of Author

27/09/2019

Date

## Highlights

1. Combination of optimal trajectory and optimal controls for steering inputs of a Formula student car.
2. Testing controller performance at different velocities with a constant internal plant model.
3. Testing of controller performance around a full lap, with varying velocities.
4. Results comparing three optimal controllers in the equal conditions.

## Acknowledgements

I would like to thank my supervisor Denise Morrey and her PhD student Brady Planden for providing guidance and insight into controls and MSc project management. A big thanks to the Oxford Brookes Racing team for providing vehicle information and opportunities to perform test and gather data during the dissertation.

## Abstract

This paper outlines the process of creating an optimal controller for an electric formula student car to be used in the 2020 Formula Student Germany Competition. The computational limitation of the ETAS 910 is considered and thus the chosen controllers for the project are an MPC and LQR with and without gain-scheduling.

The controller needs a reference trace to follow, where optimal mathematics are applied to track data. Here both minimum curvature and shortest path optimisation were found and analysed as to which gave minimal lap time around Bruntingthorpe. Here it was found that around Bruntingthorpe minimum curvature was 2.89 seconds faster than shortest path.

A linear dynamics model was made which captures the lateral dynamics for use in the controller's plant model and as an initial vehicle model. For the optimal controller plant model, the equation of motion was transformed into a state-space representation. This was then validated against a derivatives analysis to ensure the dynamics were still consistent. A non-linear dynamics model was then made, with a MF 6.2 and weight transfer added. This was used on the vehicle model, to test whether the linear internal plant model would provide enough prediction capabilities for precise controls. The linear and non-linear model were then validated against trackside data. The parameters for the plant were found using CAD, moment of inertia testing and data from the Tyre Test Consortium.

The tuning of the controllers was based on key performance indicators, found to be used in industry. These were then tested on a step and sinusoidal input, before being tested on the optimal trajectory data around a full lap of Bruntingthorpe.

It was found that with the linear vehicle model, all controllers were showing excellent tracking, but with a non-linear vehicle model the non-linear LQR with gain-scheduling showed superior track to the others with a root mean square error of 0.0656 compared to the second best at 0.1936, while using less effort to achieve these values.

## Contents

Statement of originality .....	2
Highlights.....	3
Acknowledgements .....	3
Abstract .....	4
List of Figures .....	6
List of Tables .....	6
List of Abbreviations .....	7
List of Symbols .....	7
1. Introduction.....	8
1.1 Aim and Objectives.....	8
1.2 Background .....	8
2. Literature Review.....	9
3. Approach.....	13
4. Methodology .....	14
4.1 Research.....	14
4.2 Optimal trajectory Software and Mathematical approach .....	14
4.3 Plant model and Vehicle model Validation Strategy .....	16
4.4 Choice of controllers.....	19
4.5 Computational demand analysis .....	21
4.6 Testing Strategy .....	21
5. Results and Discussion .....	21
6. Conclusion .....	31
7. Future improvements .....	32
References.....	33
Appendix A – Supplementary Results .....	35
Appendix B – Vehicle parameters .....	39
Appendix C – Non-linear Simulink Code.....	39

## List of Figures

Figure 1 - Scope of Project .....	8
Figure 2 - Representation of Bicycle Model (Wu et al., 2018).....	10
Figure 3 - Plant and Vehicle Model Illustration.....	11
Figure 4 - LQR Full State Feedback Loop.....	11
Figure 5 - MPC prediction (MathWorks, 2019) .....	12
Figure 6 - Vehicle Coordinate System (SAE, 2019).....	16
Figure 7 - OBR18 Moment of Inertia Testing .....	18
Figure 8 - Optimum tire $F_x$ Fitting for LC0 FS Tire .....	19
Figure 9 - Bruntingthorpe Velocity Trace Data from Optimisation .....	20
Figure 10 - Filtering input data ( $U_f$ = unfiltered, $f$ = filtered).....	21
Figure 11 – Lat. and Long g vs. Velocity .....	22
Figure 12 - Track Map of Bruntingthorpe from Data and Google Earth .....	22
Figure 13 - Minimum curvature vs shortest path (a) Minimum curvature vs Opt (b) .....	23
Figure 14 - Derivative vs State space yaw rate from step.....	24
Figure 15 - System Characteristics Analysis .....	24
Figure 16 - Step Response Characteristics.....	25
Figure 17 - Validation of non-linear model against OBR18 data .....	26
Figure 19 - Controller Comparison Step Reference (Step val. = 1 step time = 0.5 s).....	26
Figure 19 - Controller Comparison Sinus Reference (Sine val. = 2 Freq. = 2 rad/s).....	27
Figure 20 - Linear Vehicle Model Results, All Controllers.....	28
Figure 21 - Non-Linear Vehicle Model Results, All Controllers.....	30
Figure 22 - All Gain-Schedule LQR Results (Step val. = 1 step time = 0.5) – Start of Appendix .....	35
Figure 23 - All LQR Results (Step val. = 1 step time = 0.5) .....	35
Figure 24 - All MPC Results (Step val. = 1 step time = 0.5) .....	35
Figure 25 - LQR Sine Results .....	36
Figure 26 - MPC Sine Results .....	36
Figure 27 - LQRGS Sine Results.....	36
Figure 28 - Gain-Scheduled LQR Track Result Linear Vehicle model.....	37
Figure 29 - LQR Track Result Linear Vehicle model.....	37
Figure 30 - MPC Track Result Linear Vehicle model .....	37
Figure 31 - LQRGS Results Non-Linear Vehicle Model.....	38
Figure 32 - MPC Results Non-Linear Vehicle Model .....	38
Figure 33 - LQR Results Non-Linear Vehicle Model.....	38
Figure 34 - Non-Linear Vehicle Model w. LQRGS Simulink Model .....	39

## List of Tables

Table 1 - Coding Language Comparison .....	14
Table 2 - Advantages and Disadvantage of Model based development .....	14
Table 3 - Optimal Trajectory Results.....	23
Table 4 - Results Step Response .....	25
Table 5 - Linear Vehicle Model Results and computation demand.....	29
Table 6 - Non-Linear Vehicle Model Results .....	30
Table 7 - Vehicle Parameters .....	39

## List of Abbreviations

CoG	=	Centre of Gravity
FS	=	Formula Student
OBR	=	Oxford Brookes Racing
DoF	=	Degrees of Freedom
(L)MPC	=	(Linear) Model Predictive Controller
LQR(GS)	=	Linear Quadratic Regulator (Gain Scheduled)
PID	=	Proportional Integral Derivative
EV	=	Electric Vehicle
RMSE	=	Root Mean Square Error
MAE	=	Mean Absolute Error
IACA	=	Integral of the Absolute Value of the Control Action
MC (Opt)	=	Minimum Curvature (Optimised)
SP	=	Shortest path
QP	=	Quadratic Programming
CAD	=	Computer Aided Design

## List of Symbols

$\delta$	=	Steered Angle at the Wheel
$v_x$	=	Longitudinal Velocity
$v_{max}$	=	Maximum Longitudinal Velocity
$v_y$	=	Lateral Velocity
$v_{y\dot{}}$	=	Pure Lateral Acceleration
$a_y$	=	Lateral Acceleration (Angular Acceleration)
$r$	=	Yaw Rate
$k_{(s)}$	=	Curvature (Length Along Curvature)
$x$	=	System States
$m$	=	Mass
$l_f/l_r$	=	CoG to Front/Rear Axle Length
$I_{zz}$	=	Moment of Inertia Z-Axis
$\alpha_r/\alpha_f$	=	Slip Angle Front / Rear
$F_{yf}/F_{yr}$	=	Lateral Force Front / Rear
$X/Y$	=	Distance in X/Y direction coordinate system (m)
$\alpha_i$	=	Track Boundaries

# 1. Introduction

## 1.1 Aim and Objectives

The aim of this project is to find the optimal controls to steer an autonomous electric formula student car with an optimal path.

1. Research and find at least 10 relevant journals in the field of lap time simulation and optimal path control for an autonomous EV by mid-April.
2. Create a 2DoF bicycle Plant model, to represent the dynamics of the EV formula student cars for accurate prediction in the controllers using MATLAB/Simulink by July.
3. Test the found algorithms ability calculate the optimal path around test circuit, with a predefined initial path derived from track data using Python by mid-August
4. Optimise controller parameters for minimal lap time around Bruntingthorpe, using MATLAB post processing by the start of September.
5. Evaluated the selected vehicle dynamics models and controllers, with regards to accuracy of prediction to achieve the desired performance by end of august.

## 1.2 Background

The purpose of this research comes from two new requirements in the motorsports industry. The first requirements come from major motorsports teams that want to evolve the driver models in their lap time simulators, in order to get better predictions in their simulations. Furthermore, it's clear that the industry moving towards autonomous driving in both motorsport (Mitchell, 2018) and the automotive sector (Basenese, 2017). A need has therefore emerged for better controllers, which can insure the better control of the car. With this ever-increasing demand, there has been an increase in the use of optimal controllers. This can be found both in the automotive sector, but also in other sectors such as in fighter jets. (Andersson *et al.*, 2018) In this paper there will be a focus on developing an optimal control system, for the Oxford Brookes Racing Formula Student car, to be used in the Formula Student Germany competition, which requires autonomous driving in skid pad and acceleration from 2020 onwards.

The project will span from the state input to the controller, to the output of the controller, with reference trajectory optimisation. Controlling the car will also require a motor at the steering rack, which will receive the output of the controller. The full control system will also include a state-estimator which will process the sensor signals, to give a better input for the controller. These two factors are large undertakings in itself and therefore this project is limited to the controller itself.

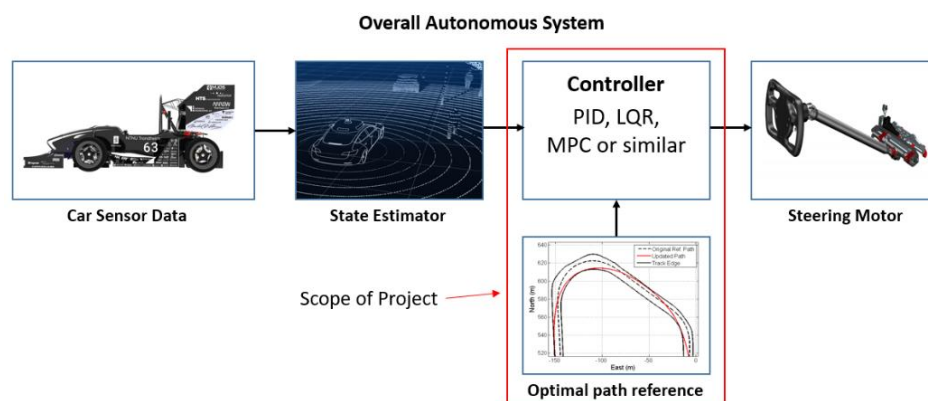


Figure 1 - Scope of Project



## 2. Literature Review

Below is the current state of the literature surrounding the autonomous control of a vehicle, along with literature about optimal path and simulation in general. The problem of optimal control has been divided into smaller section to improve the understanding of the individual parts. The first is the plant model which is used to describe the dynamics of the system in controllers. Secondly is the problem of giving the controller an optimal path as reference, which will require track optimisation. Third is the controls problem itself, in which different types of controllers are being used in industry.

### Optimal Path

Lap time simulators often use a pre-determined path, but as technology advances, the industry is pushing for transient simulation that can give more accurate results. The advantage and disadvantage of the different type of the different types of simulations have been investigated and compared in Siegler et al. (2000). The issue with having a given path, is that this might not be the optimal for every setup. This is also described in Casanova (2001), where he describes the theoretical best lap. There is therefore a need to develop a method to create a different trajectory dependent based on vehicle dynamics.

These methods have been found to be used to calculate the optimal path:

- Minimum curvature-based optimization (Henrique, 2016).
- Shortest path optimization (Marques, 2016).

Minimum curvature is based on the principle of less curvature will provide a higher velocity, defined by the following equation.

$$v_{max} = \sqrt{\frac{a_y}{k}} \quad (1)$$

The acceleration is limited by the level of grip between the track surface and the tire. As it can be seen from the equation, at smaller curvatures a larger velocity can be maintained. Given that lap time is calculated from.

$$laptime = \frac{distance}{avg.velocity} \quad (2)$$

The shortest path optimisation works with principle of minimising the distance travelled and sacrificing velocity. It takes the lap time simulation problem but tries to reach the minimum lap time by decreasing the numerator instead.

Which optimisation gives the best lap time is complex as the vehicles capabilities and track layout play a large role in which method give the optimal solution. Therefore, each optimisation will be attempted in the papers, for the OBR20 parameters and be compared to get the optimal lap around Bruntingthorpe.

### Plant and Vehicle Modelling

In lap time simulators the vehicle model often consists of equations of motion that describe the total behaviour of the car. (Criens et al. 2006) (Halkwill 2000). The reason for this is often that there is a need to model the exact behaviour of the vehicle, but this makes for complicated and computational heavy code. This level of complexity is also not needed to predict the overall behaviour of the plant.

In most control theory, the consensus is that transforming some of these equations into state-space is the optimal solution, as it makes the control problem easier (Carvalho et al. 2013). The complexity of the model is also reduced, which is described in Wang L. (2009). How to transform the classic bicycle

model, which is often used when modelling a car (Anderson et al. 2018), is described in Davin (2011). The equations for describing a state-space model are as follows:

$$\dot{x} = Ax + Bu \quad (3)$$

$$y = Cx + Du \quad (4)$$

$x$  is the states of the system, which is multiplied by the  $A$  matrix of the system, while the  $B$  matrix is multiplied by  $u$ , which is the input to the plant from the controller. Both the  $A$  and  $B$  are function of the plant dynamics.

The choice of plant model for the controller will be a balancing act between complexity and computational demand. In Borelli et al (2015), where one can see a preference towards using a bicycle model, with other papers adding a second bicycle model to simulate weight transfer. (Carvalho et al. 2013). Extensive research into this issue and the errors and downsides of the assumptions of the models when correlating data has been investigated in Patil (2017). Most notable is the assumption of linear tire characteristics, which is a large understatement. A tire only has a linear behaviour at very small slip angles as shown in figure 9.

When controlling the steering input, then a plant model that represents the lateral characteristics is needed, which in Patil (2017) the linear bicycle model proves to do. Turning the plant model non-linear will make the controls problem more advanced and in Borelli (2015) it can also be seen that the nature of the optimal controllers tends to make up for the errors in the non-linearity.

The gap in many of the papers is that the plant model is updated when checking the controller at different velocities. This gives the optimal condition for the controller, but for use in racing then it's not possible to change the plant model with a linear controller. This paper aims to test the viability of having a linear controller, for a non-linear plant model.

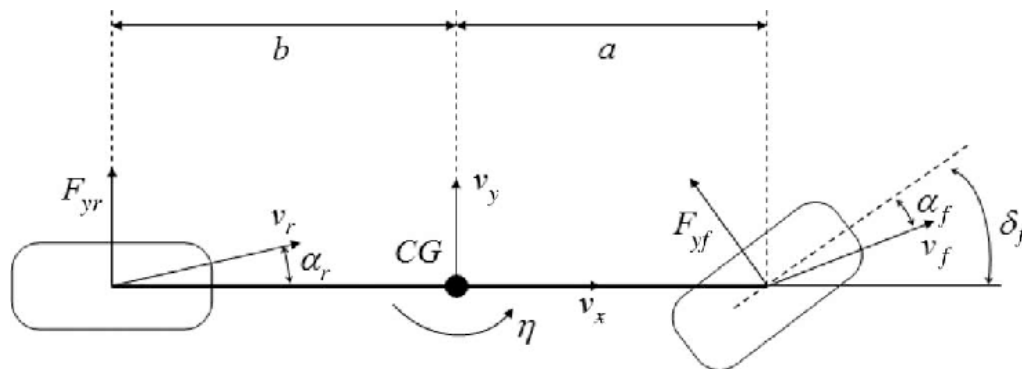


Figure 2 - Representation of Bicycle Model (Wu et al., 2018)

It's important to also mention the difference between plant model and vehicle model. A plant model is the dynamics that are used for the optimisation for the controller and can be both linear and non-linear. Changing the plant model from linear to non-linear makes the controls more complicated and often require non quadratic solution. (Wang L., 2009) The vehicle model can on the other hand be as complex as is desired in order to capture the dynamics, so the response comes as close to the actual car behaviour. Below is a figure in which this is visualised.

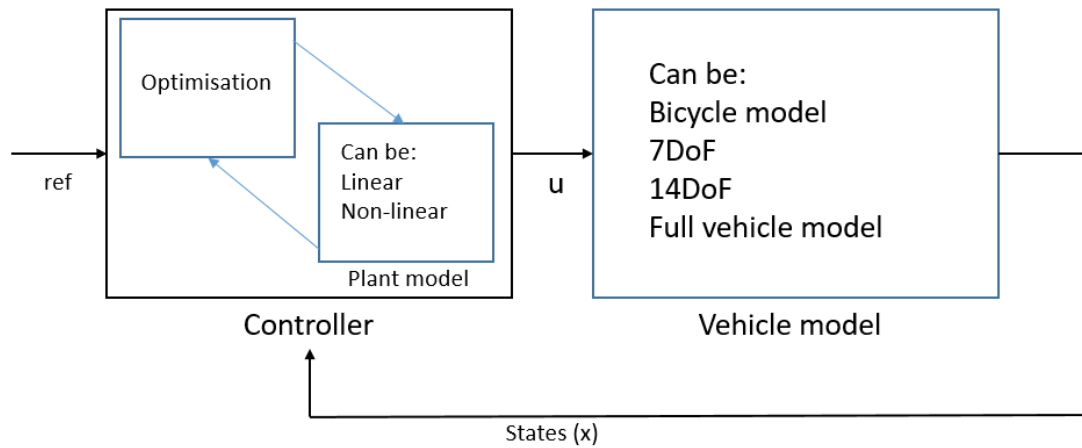


Figure 3 - Plant and Vehicle Model Illustration

## Controller Theory

In lap time simulation PID's are often used (Halkwill, 2000) because of their simplicity and relatively good results if correct tuning has been applied, but the capabilities of this controller are not considered optimal. There is therefore a gap to improve the general laptime simulators by using optimal controllers.

### Optimal controllers

Defining optimal, can be difficult and normally depends on what the engineer's objective is. But for controllers there is a specific group that is categories as optimal. It is considered optimal when the system is designed to provide a minimum performance index. The performance of the controller also must be described in terms of the state variables, in a cost function as shown below (Dorf, 2010).

$$J = \int_0^{t_{final}} g(x, u, t) dt \quad (5)$$

In the field of vehicle controls, two optimal controllers have shown to have excellent tracking capabilities but with distinct advantages and disadvantages. These are Linear Quadratic Regulators and Model Predictive Controllers, which both come under the term of optimal controllers.

### Linear Quadratic Regulators

Are what is known as a full state feedback controller (Dorf, 2010) where the close-loop poles are used to create the gains. For the state feedback a linear transfer function or state space representation is used, to which the formulation can be seen in equation 3 and 4. Figure 5 shows the layout of an LQR controller.

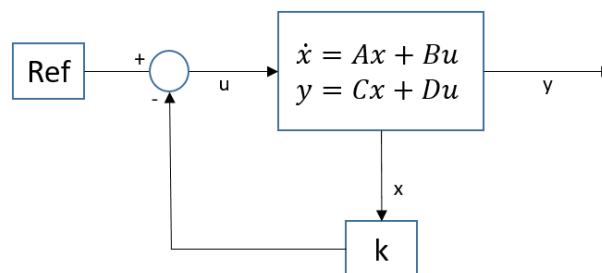


Figure 4 - LQR Full State Feedback Loop

In an LQR the gains  $k$  is found using the closed loop characteristic of the plant model, with  $Q$  and  $R$  values to represent the which state to prioritise more ( $Q$ ) and how big a penalty the actuation  $u$  needs ( $R$ ). (Mathworks, 2019)

Cost function for LQR

$$J = \int_0^{\infty} (x^t Q x + u^t R u) dt \quad (6)$$

In matrix formulation the equation becomes

$$J = \int_0^{\infty} \left( [x_1, x_2 \dots x_n] \begin{bmatrix} Q_1, 0, \dots, 0 \\ 0, Q_2, \dots, 0 \\ 0, 0, \dots, Q_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + [u_1, u_2 \dots u_n] \begin{bmatrix} R_1, 0, \dots, 0 \\ 0, R_2, \dots, 0 \\ 0, 0, \dots, R_n \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} \right) \quad (7)$$

The optimisations produce the gains  $k$ , for the state of the system to create the output of the controller  $u$ , as shown below and in figure 4.

$$u = -kx \quad (8)$$

The optimal equation for LQR is what is known as a quadratic function and means that the system has one definite minimum value, which makes for a simpler controls problem. LQR has also been seen in papers to have great tracking capabilities (Lenzoa, 2018), which makes it a good candidate for this project.

#### Model Predictive Controllers

This controller also uses full state feedback but uses an internal plant model in the controller which is used to predict the behaviour of the model for a given prediction horizon. The controller will then take a single or several steps known as control horizon (Wang L., 2009). This process has been depicted in figure 5. The representation of an MPC is represented in figure 4, as unlike the LQR the optimisation is happening online and as such the controller uses the plant to constantly predict the future behaviour of the vehicle.

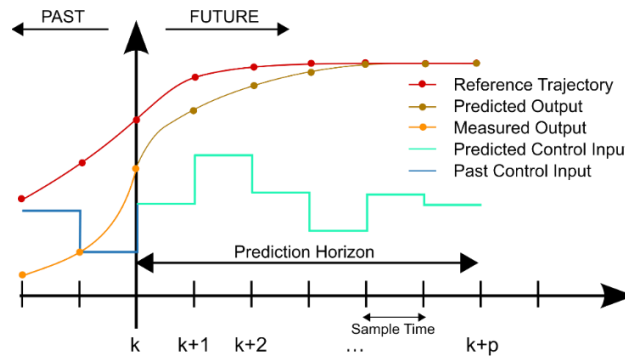


Figure 5 - MPC prediction (MathWorks, 2019)

As MPC is an optimal controller, it has a cost function which is stated below.

$$J = \sum_{i=1}^p w_e e_{k+i}^2 + \sum_{i=0}^{p-1} w_{\Delta u} \Delta u_{k+i}^2 \quad (9)$$

Looking at Bruschetta (2017), Liniger (2017) and other literature the excellent tracking capabilities of the MPC becomes clear. The downside of the MPC is the computational demand from having to run the prediction simulation constantly, which most literature do not evaluate.

## Real Time Simulation

One of the main issues with controls system is balancing computational requirement and accuracy of the prediction. This is described in Stankovic (1988) where the real time issue is further described. For this project the MPC will have the largest demand, as it must run the optimisation for each step, unlike the LQR which does the optimisation offline. Secondly it can be seen that the trajectory optimisation can also be done offline (Heilmeier *et al*, 2019). to decrease the computational demand of the online running.

## Knowledge Gap

Optimal path is used in motorsport and optimal controllers are used in many different industries. The unique aspect of this project is to combine these two areas, to create the optimal control for the specific purpose of running the OBR20 car in a formula student Germany. The ETAS 910 which will run the full controls system has a limited computational demand, which means that the not only tracking error will be investigated but if the code is running real time and the demand. This is another novelty as most of the papers that deal with optimal controller only focus on the performance of the controller and not the computational demand, which this paper will put an emphasis on.

Another gap which is being investigated in this paper, is the performance of linear internal plant model with a varying velocity vehicle model. Most papers evaluate the performance of the controllers with constant velocity and an updated plant model to match, but when dealing with race cars then the velocity will always be changing.

As shown in literature, most papers test the controller on a linear vehicle model, which as shown in figure 2 isn't the case. Therefore, this paper puts a focus on testing the controller on a non-linear plant which should give a much closer representation of the final performance around a track.

## 3. Approach

The approach has been to investigate as many methods as possible, to achieve the best possible result. The overall problem of optimal controls was studied to be able to divide the problem into smaller sections. This meant that each part of the problem could be optimised, and the optimal solution could be found.

As this project was the first coding experience, care was taken to learn proper coding structure and procedures. This was done through a seminar with ITK engineering, which briefed the controls and simulation group about naming convention, structure and knowledge about embedding Simulink code.

As this project is done from a motorsports engineering background, then the approach is changed to involve a larger research into the plant and vehicle modelling compared to other papers. In motorsports engineering one knows the importance of using correct plant modelling and how severe the differences in having a more complex vehicle model can make. Therefore, the approach in this paper is to take the vehicle model to new heights, in order to add to current literature.

As this is the first year of formula student autonomous from Oxford Brookes Racing, a bigger emphasis will be put on learning about different controllers and using software's which are freely available to all members of the team. This is done so this project can be carried on and evolved until the 2020 regulations. The team has already received the ETAS 910 and therefore all decision has to consider the available computational power from this unit.

## 4. Methodology

### 4.1 Research

An investigation was made into the coding languages and the available toolboxes, which could aid in controller creation, optimising trajectory and has open source code which can serve as basis for some of the code. The features of each of the investigated coding language are listed below.

*Table 1 - Coding Language Comparison*

Has It = X, Lesser Extend = (X), Does Not Have It = Blank			
Feature	C++	Python	MATLAB / Simulink
Embeddable code	X	X	X
High level language	X	(X)	
Controls toolbox			X
Optimal path optimisation		X	
Open source code	X	X	X

MATLAB and Python was chosen for their low-level coding languages, toolboxes and libraries. MATLAB and Simulink will be used for coding the controllers and Python will be used for the optimal trajectory optimisation, based on the toolboxes and open source code.

Oxford Brookes Racing decided to go with a model-based development, which is done through Simulink an additional software to the MATLAB license. Below is a comparison of model based and classic embedded coding, to provide an overview of the reasoning behind the choice.

*Table 2 - Advantages and Disadvantage of Model based development*

Model based development compared to Embedded coding	
Advantages	Disadvantages
<ul style="list-style-type: none"> <li>• Low level language easy to learn</li> <li>• Direct link between requirements and implementation</li> <li>• If done correctly then fault finding would be easier</li> </ul>	<ul style="list-style-type: none"> <li>• Not hardware-oriented</li> <li>• Can be difficult to optimize specific code</li> </ul>

### 4.2 Optimal trajectory Software and Mathematical approach

For the optimal trajectory calculation, it was chosen to go with Python as the coding language, as this had an optimal path library and a large community of open source code in the area. Open source code from the Technical University of Munich, will serve as the basis for this area of the project. (Heilmeyer, 2019). The optimization problem initially starts as a minimum curvature problem.

$$\min_{[\alpha_1 \dots \alpha_n]} \sum_{i=1}^N k_i^2(t) \quad (10)$$

The optimization problem is then reformulated in terms of a QP problem, that is subject to the maximum and minimum boundaries and steering angle that the car can achieve. The derivation and matrix manipulation are described in (Heilmeier *et al.*, 2019).

$$\min_{[\alpha_1 \dots \alpha_n]} \frac{1}{2} \alpha^T H \alpha + f^T \alpha \quad (11)$$

$$\text{Subject to: } \alpha_i \in [\alpha_{i \min}, \alpha_{i \max}] \quad \forall 1 \leq i \leq N$$

an optimisation will then be done by then iterating on the curvature until the solution converges.

The shortest path problem can be formulated in minimizing the distance between the discretization points as so. (Cardamone *et al.*, 2010)

$$S^2 = \sum_{i=1}^n d(P_{i+1} - P_i) \quad (12)$$

Which in its QP problem form, with the conditions being the boundaries (Cardamone *et al.*, 2010) can be written as.

$$\min_{[\alpha_1 \dots \alpha_n]} \alpha^T H_s \alpha + B_s \alpha \quad (13)$$

$$\text{Subject to: } \alpha_i \in [\alpha_{i \min}, \alpha_{i \max}] \quad \forall 1 \leq i \leq N$$

To calculate the velocity trace and capabilities of the car, a ggV representation is used. The ggV plot consist of the maximum long and lateral g at different velocities. Bruntingthorpe data will be interpolated, which will provide an accurate prediction as the car has been seen to do these accelerations around the track.

One could use a theoretically derived ggV plot, but biggest problem with doing this is the friction coefficient, which change massively dependent on the surface. Tools exist which can measure this, but the cost eliminates the possibility of a formula student team and thus track data is the best option.

The velocity traces are calculated first by using equation (1) while only using the minimum lateral g from the ggV. The maximum velocity of the vehicle is then used to cut the trace, to which a forward calculation for positive longitudinal g predict accelerations. Next the procedure is reversed to calculate the negative acceleration of the car. (Heilmeier *et al.*, 2019). The velocity trace is then produced using

$$a_{xi} = \frac{v_{xi+1}^2 - v_{xi}^2}{2l_i} \quad (14)$$

The optimisation will serve as the reference for the controller and it is therefore important to consider the limitations of these methods. When creating the ggV plot from data, then it's very important to apply filtering to not capture g-force values which comes from noise. In this project the signal processing toolbox in MATLAB will be used to process the track data. When looking at the created ggV plot it's also important to understand, that this is an optimal representation and not the real car. Therefore, downscaling should be used when initially running the car and then slowly bringing the velocity up, which is normal practice in autonomous racing (Roborace, 2019)

The output of the optimisation will be:

- $k_s$  (m)
- X and Y (m)

- $k$  (rad/m)
- $v_x$  (m/s) and  $a_x$  (m/s<sup>2</sup>)

As it can be seen the two model references  $a_y$  and yaw rate are missing, so these needs to be calculated.

$$a_y = v_x^2 * k \quad (15)$$

$$r = \frac{a_y}{v_x} \quad (16)$$

### 4.3 Plant model and Vehicle model Validation Strategy

#### Choosing plant model

The purpose of the controller is to steer the vehicle as close to the optimal line as possible. To do this the controller requires a plant that captures lateral dynamics of the vehicle, where the yaw rate and lateral acceleration are the desired values to trace as shown in (Borelli *et al.*, 2015).

This project does not need to capture the longitudinal dynamics of the vehicle as these would be controlled by another controller, Therefore, it can be stated that the system needs to capture 2DoF, which is the lateral motion and the rotation about the z-axis. Below is the SAE standard vehicle axis system where the yaw rate and lateral acceleration can be seen.

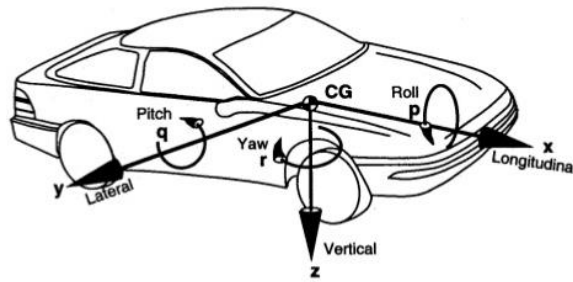


Figure 6 - Vehicle Coordinate System (SAE, 2019)

The lateral dynamics of a car are captured by these equations of motion.

$$m(\dot{v}_y + v_x r) = F_{yf} + F_{yr} \quad (17)$$

$$I_{zz} \dot{r} = a F_{yf} - b F_{yr} \quad (18)$$

$$F_{yf} = C_f \alpha_f \quad (19)$$

$$F_{yr} = C_r \alpha_r \quad (20)$$

$$\alpha_f = \delta - \frac{v_y + ar}{v_x} \quad (21)$$

$$a_r = -\frac{v_y - br}{v_x} \quad (22)$$

The equations can then be derived into state-space representation. (Davin, 2011)



$$\begin{bmatrix} \dot{v}_y \\ \dot{r} \end{bmatrix} = \begin{bmatrix} -\frac{C_f + C_r}{mv_x} & \frac{l_r C_r - l_f C_f}{mv_x} - v_x \\ \frac{l_r C_r - l_f C_f}{I_{zz} v_x} & -\frac{l_r^2 C_r + l_f^2 C_f}{I_{zz} v_x} \end{bmatrix} \begin{bmatrix} v_y \\ r \end{bmatrix} + \begin{bmatrix} \frac{C_f}{m} \\ \frac{l_f C_f}{I_{zz}} \end{bmatrix} [\delta] \quad (23)$$

As discussed earlier the measured outputs of the vehicle will be  $a_y$  and  $r$ , as these can be directly measured with sensors, so an additional state is added to the plant. On the basis that the centripetal acceleration is equal to equation 24. (Rajamani, 2006)

$$a_y = v_y + v_x r \quad (24)$$

$$\begin{bmatrix} \dot{v}_y \\ \dot{r} \\ \dot{a}_y \end{bmatrix} = \begin{bmatrix} -\frac{C_f + C_r}{mv_x} & \frac{l_r C_r - l_f C_f}{mv_x} - v_x & 0 \\ \frac{l_r C_r - l_f C_f}{I_{zz} v_x} & -\frac{l_r^2 C_r + l_f^2 C_f}{I_{zz} v_x} & 0 \\ 1 & v_x & 0 \end{bmatrix} \begin{bmatrix} v_y \\ r \\ a_y \end{bmatrix} + \begin{bmatrix} \frac{C_f}{m} \\ \frac{l_f C_f}{I_{zz}} \\ 0 \end{bmatrix} [\delta] \quad (25)$$

When the plant model is defined, then the plant can be analysed using the MATLAB functions (*step*, *stepinfo* and *damp*) to investigate the poles, damping ratio, rise time and so forth. This can assist in the process of choosing the values of  $Q$  and  $R$  for the LQR and the weights of each different states. (Brunton, 2019) It also shows the characteristics of the plant which can assist in understanding the results from the final controls problem.

#### Vehicle Model

As mentioned in the literature review, the tire model is linearized as seen in equation 12 and 13. This is the biggest drawback of the bicycle model and the non-linear tire behaviour quite visible in figure 9. Therefore, controller testing and evaluation will be done both with a linear vehicle model and a non-linear model, while maintaining the linear internal plant model.

The non-linear model has the added dynamics of weight transfer and a Pacejka MF 6.2 tire model. This is done to evaluate the performance of the controller on a vehicle model which as close to the formula student car as possible. The tire model will include tire parameters for the LCO 16-inch tire, which will be on the OBR20 car. The weight transfer equation used are.

$$F_z \Delta = \frac{(a_x - v_y r) * m h}{(a + b)} \quad (26)$$

$$F_{zf} = \frac{(m * g * b)}{(a + b)} + F_z \Delta \quad (27)$$

$$F_{zr} = \frac{(m * g * a)}{(a + b)} - F_z \Delta \quad (28)$$

The Pacejka 6.1 equations used are.

$$F_x = (D_x \sin[C_x \arctan\{B_x k_x - E_x(B_x k_x - \arctan(B_x k_x))\}] + S_{vx}) * G_{xa} \quad (29)$$

$$F_{yp} = D_y [C_y \arctan\{B_y a_y - E_y(B_y a_y - \arctan(B_y a_y))\}] + S_{vy} \quad (30)$$

$$F_y = G_{yk} F_{yp} + S_{vyk} \quad (31)$$

The full list of equations and parameters are described in Besselink *et al.*(2010). The advantages of the model are combined slip and relaxation length amongst others.

### Obtaining Model Data

OBR20 has not been manufactured yet, so to obtain the parameters the CAD model will be used. Obtaining moment of inertia of the car in CAD tends to be inaccurate, due to the nature of a Formula Student CAD model. This is often because not all the components in the model are given the correct material, with some being under and over approximated by the software due to the correct material properties not being available. The OBR18 mass analysis showed a 10 % difference from CAD to the actual weight of the car because of these factors.

### Moment of Inertia and Centre of Gravity

To obtain the moments, CoG height etc. an approximation can be made with the OBR18 car. This car is a combustion car, but the layout of the two mean that both have the heaviest load on the rear wheels. To obtain these dynamic parameters of the OBR18 car, it was brought to Cranfield Impact Centre. Here the university has a 3-axis moment of inertia rig available.



Figure 7 - OBR18 Moment of Inertia Testing

As the 2020 car will be the first car with in-hub motors, there will be some distinct differences in the vehicle dynamics of the car. (Omar, 2015)

### Tire model data

The tire test data is obtained from the Formula Student SAE Tire Data Consortium. This data then must be fitted to a tire model, which as described earlier is the Magic Formula 6.2. This is done through Optimum Tire, which is a software for managing tire data and has function to export data to tire model parameters. Longitudinal result (Figure 10).

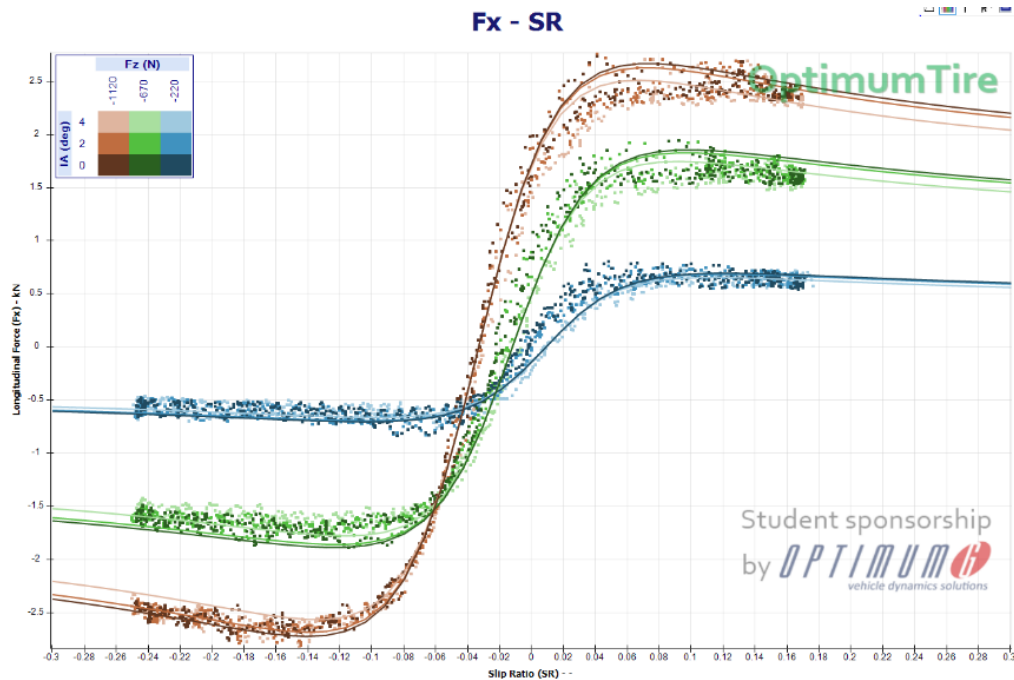


Figure 8 - Optimum tire  $F_x$  Fitting for LC0 FS Tire

## Validation of the Model

Two different methods will be used for the validation, which has been chosen to be a derivatives analysis and trackside data correlation.

A derivatives analysis tries to predict the behaviour of the vehicle dynamics, by dividing the time up into steps and then assuming constant behaviour across the step. This analysis is described in Balkwill (2017), where the relevance of the model especially for a simple dynamic system are elaborated.

The second method will be to validate these results against trackside data, which is only available for the OBR18, so a deviance is expected, but the purpose is to validate that the behaviour of the non-linear model is correct. The downside is that the plant can never be fully validated using this method.

## Analysing Plant and Vehicle model

When analysing the plant and vehicle model in controls, it's to familiarize with the system dynamics. This project isn't aimed at optimising the plant, which would be done through vehicle dynamics, but assumes that a vehicle dynamics department in the formula student team has optimised this. In controls one wants to analyse the plant model for tuning and determining the weights of the states.

## 4.4 Choice of controllers

As the literature showed, both LQR and MPC can be good choices for optimal control. It has therefore been chosen to go forth with both and then evaluated the results, in order to finally be able to choose the optimal controller for the formula student team.

Initially a linear LQR will be used, which will be created in MATLAB using the *LQR* command. This linear optimisation will only work for a constant velocity, which is of course not what one would see around a racetrack, in which the velocity trace is shown below (Figure 11).

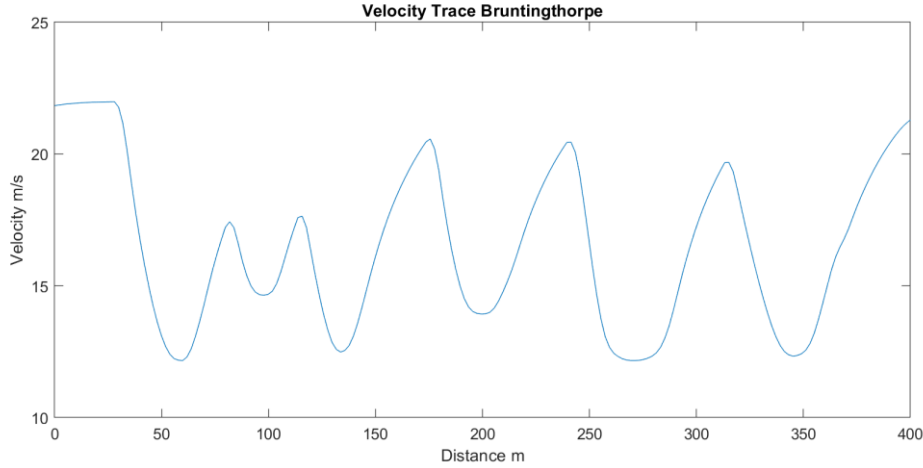


Figure 9 - Bruntingthorpe Velocity Trace Data from Optimisation

The LQR optimisation requires a linear plant, which in our plant model requires a constant velocity. To combat this issue, Gain-scheduling can be used. (Wang Z. *et al.*, 2018). This has been shown to be able to control non-linear behaviour such as missiles, with accurate trajectory tracking. (Azizi *et al.*, 2013). The optimisation will happen at velocity from 0-25 m/s as this project is regarding Bruntingthorpe and the data (Figure 11) shows that the maximum velocity will be around 22 m/s. With the plant model dynamics analysis, it will be possible to determine the velocity intervals necessary. The gain-scheduling will be controlled by a *lookup table* in Simulink.

It has been chosen that the MPC will be linear, since the computational demand of the MPC is quite high. More advanced types exist like non-linear and adaptive, but as seen in Liniger (2014) it requires a powerful computer to run. The ETAS 910 does not have the same computational power and the MPC will have to share the load with the rest of the control system. Whether this is a viable option compared to the LQR, will be based on the controller's ability to cope with the varying velocity vehicle model compared to the constant velocity plant model. To aid in the process of creating the controller the Simulink MPC block will be used.

### Evaluation of controllers

The controller performance can be evaluated using RMSE, which is a good measure if large outliers are undesirable. (SQUARK, 2019) Simultaneously the MAE value will be looked at to get wider range of results, as this value isn't sensitive to large outliers as is the case with RMSE. These methods of evaluating controller's performance have shown to give a good overview of the performance, which is hard to evaluate from plots. (Wang Z. *et al.*, 2019)

$$RMSE = \sqrt{\int_0^{tf} [ref(t) - measured(t)^2] dt} \quad (32)$$

$$MAE = \sum_0^{tf} [ref(t) - measured(t)] \quad (33)$$

These two measurements will evaluate the tracking error, but not the effort used. If a controller uses a large amount of steering angle to achieve these values, then it will be at higher slip angles and that will put more energy into the tire (Balkwill, 2018) and therefore it will degrade faster (Li *et al.*, 2011). To evaluate the effort the IACA equation is used. (Wang *et al.*, 2019)

$$IACA = \sqrt{\int_0^{t_f} [u(t)]dt} \quad (34)$$

These values will be used to evaluate the total performance of the LQR, LQRGS and LMPC in regard to simple inputs such as step (constant velocity), sinusoidal (constant velocity) and following the optimal trajectory traces for a full lap simulation.

#### 4.5 Computational demand analysis

Another strategy that will need to be determined is to make code run in real time. For a controls system to be relevant, your code will need to run faster than the command is needed. (Salazar, 2017) The method used will be check the code run time, using the built-in feature tic-toc in MATLAB. This will run the simulation and time how long it takes to simulate the event.

As mentioned before, this is also the reason why the MPC will be kept to a linear plant system, as the computational power of the ES910 is limited.

To get the code to run at real time, an iterative method will be used to evaluate different methods of formulating the problem with different combinations of MATLAB and Simulink code. Another method that will be used is to push as much of the code to be run offline, thereby easing the computational requirement.

#### 4.6 Testing Strategy

Two types of testing will be carried out to evaluate the performance of the code. The first method will be software in loop testing. This is simple running the code with pre-defined trajectory with all the code being run on the same hardware. This will be done on a surface pro 4, with all the running being done in MATLAB using:

tic, sim ('Name of Simulink code'), toc

Second part of the testing will be doing hardware in loop (HIL) testing, with an ETAS ES910 that will serve to run the model of the car, with the controls being run on a computer.

### 5. Results and Discussion

#### Optimal trajectory results

Data was taken from the a test around Bruntingthorpe, which was then filtered using *filtfilt*.

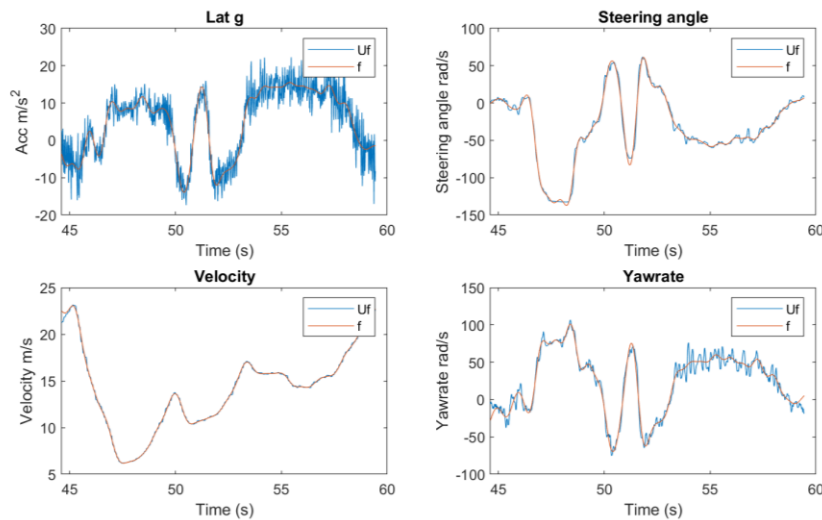


Figure 10 - Filtering input data ( $U_f$  = unfiltered,  $f$  = filtered)

As mentioned in the methodology, the track data is the basis for the optimisation. The reasoning for the filtering becomes apparent when looking at the unfiltered data in figure 12, which is especially noisy in the lateral sensor data. From this the lateral and longitudinal data was separated and plotted against velocity, for the ggvs plot as shown in figure 13.

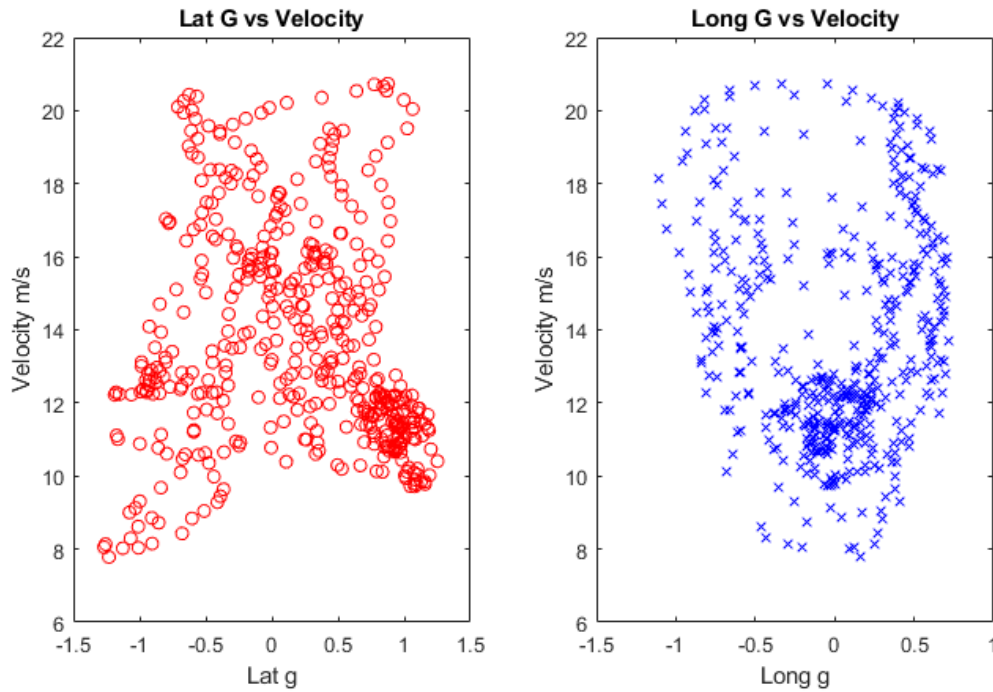


Figure 11 – Lat. and Long g vs. Velocity

From figure 12 the lateral and longitudinal g against velocity can be interpreted and a track map can be created.

Figure 12 - Track Map of Bruntingthorpe from Data and Google Earth

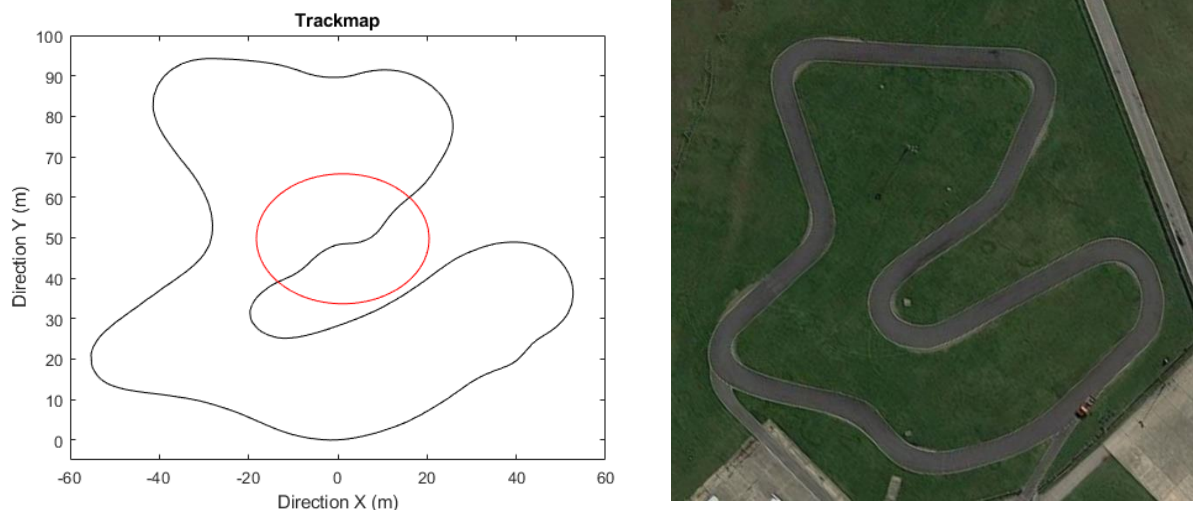


Figure 14 shows the outputted track from data compared to the real track. Some key observation can be made from the differences about how acquire the data for the optimisation. On the track map an area is highlighted by a red circle where a slalom can be observed, that isn't present on the real track. This comes from the team setting up cones, which the driver then made a slalom through. This is also present at  $Y = 90, X = 0$  and  $Y = 20, X = 40$ . The driver has taken a racing line, which also minimises the curvature of the corner. This does not affect this project, but when the controller needs to be used for the actual car, then one would ideally need two sets of data. One of the drivers going around the middle



of the track, to create the reference track and curvature. Secondly one with driver going full velocity to create the ggplot.

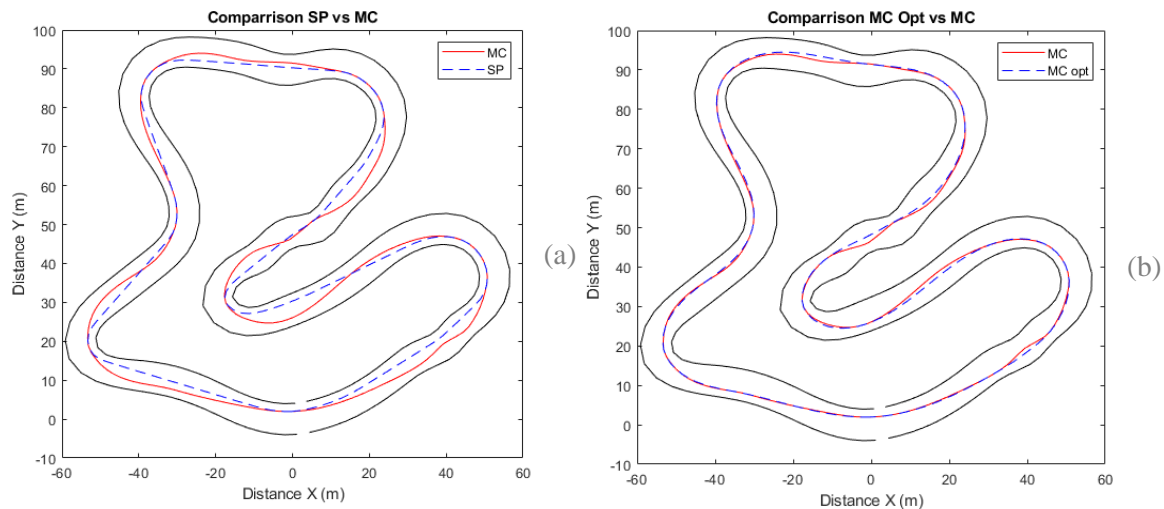


Figure 13 - Minimum curvature vs shortest path (a) Minimum curvature vs Opt (b)

Figure 14 show the results from the optimisation done in Python, where the results have been divided up into MC vs SP and MC vs MC opt. The principles of shortest path and minimum curvature are apparent in the figure. The shortest path optimisation takes the most direct route between corners in the straightest line possible. When looking at the minimum curvature trace, the problem from having the slalom as described earlier shows to affect the results. As the optimisation is done using the reference curvature for the minimum curvature, the slalom shows up as small corners, which the shortest path does not contain. To combat this one can, turn to minimum curvature optimised, which is an iterated optimisation, where the trajectory no longer follows the slalom but has found the more direct route.

Table 3 - Optimal Trajectory Results

Performance	Shortest Path	Minimum Curvature	Minimum Curvature IQP (Fastest)
Distance (m)	413.4	425.12	424.85
Avg. Velocity (m/s)	11.9	11.8	13.35
Lap time (s)	34.71	36	31.82

Above are the performance parameters for which optimisation gave the optimal trajectory for this track and vehicle, which in literature showed will be different at each track and will vary with different vehicles. As expected, one can see that shortest path optimisation creates a path that is 11.45 meters shorter than the nearest estimation. The average velocity shows the impact of the slalom, as the average velocity is 0.1 m/s lower than the shortest path, while being 11.72 meters longer. This is reflected in the lap time, which is 1.3 seconds slower. This is a 4 % difference in lap time, which at FSUK 2019 is the difference between 4<sup>th</sup> and 11<sup>th</sup> position in the sprint event (FSUK, 2019) and between 1<sup>st</sup> and 2<sup>nd</sup> place in the driverless sprint event at FSG. (FSG, 2019)

These results show good correlation to what other papers have found to be the better optimisation for lap time minimisation. (Cardamone *et al.*, 2010) (Heilmeier *et al.*, 2019).

## Plant model validation and analysis

The derivative analysis (Balkwill, 2018) is put against the state-space representation, to see if the yaw rate response is equal.

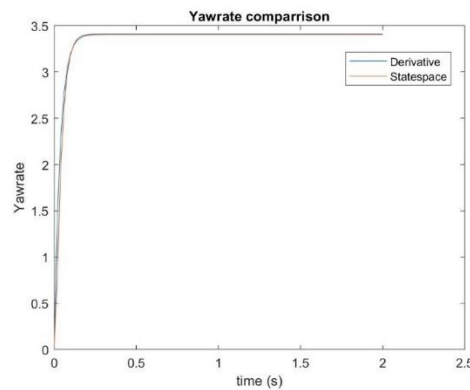


Figure 14 - Derivative vs State space yaw rate from step

Figure 16 shows that the state-space representation is derived correctly, and the response of the system is equal to that of the derivatives analysis. From this the project can now be moved forward treating the vehicle as a plant model, to analyse the system for the controller inputs. This will provide the characteristics of the plant model and can give an idea for initial gains of the system.

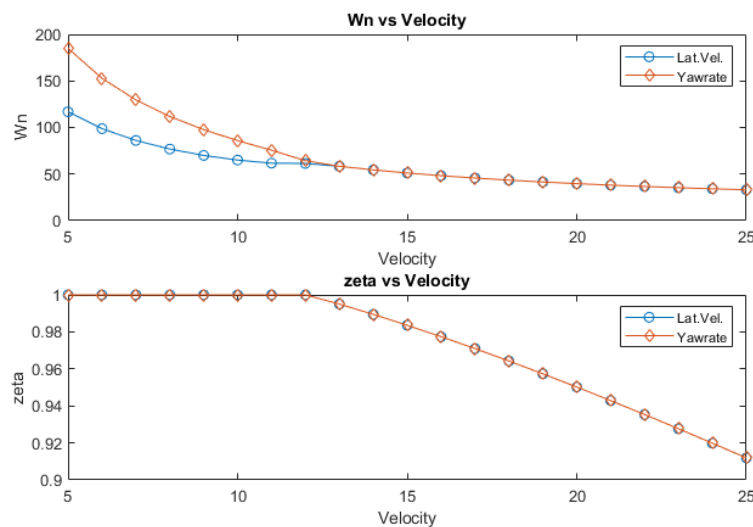


Figure 15 - System Characteristics Analysis

When looking at the characteristics of a system, then damping and damping ratio of the system can show the behaviour to inputs over a range of velocities and show how nonlinear the system is. This is important since the plant is going to assumed linear around a certain velocity. Figure 16 show that the system is critically damped up until 12 m/s, where the system then tends towards underdamped, which would mean that the system will overshoot and take longer to settle at higher velocities.

The trend of being critically damped at lower velocities and then branching off to being under or overdamped follows literature. (Balkwill, 2018) it was decided to only investigate to 5 to 25 m/s as the velocity around Bruntingthorpe are within these limits. From this it has been chosen to set the plant model for the LQR and MPC to be at 10 m/s, as here  $\zeta = 1$ , which when looking at the curve is the only value of  $\zeta$  that persist for more than 1 point.



As mentioned there will also be a gain-scheduled LQR, which will have an updating plant model dependent on the velocity. From figure 17 it becomes clear why this quality of changing plant models inside the controller is attractive as the damping ratio changes almost 10 % through the chosen range.

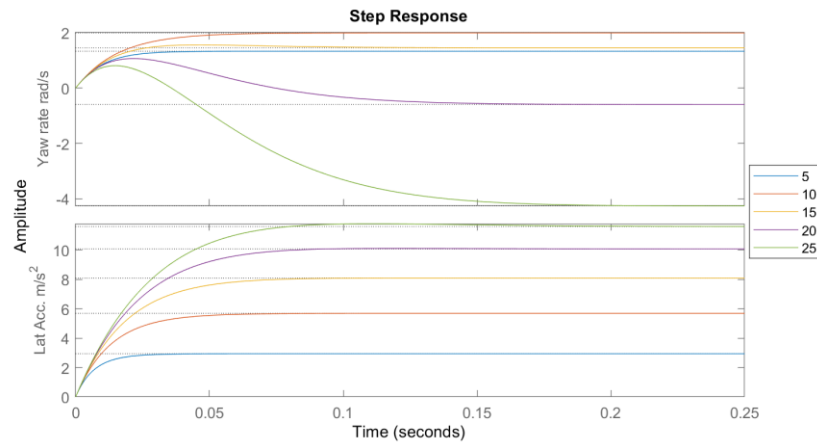


Figure 16 - Step Response Characteristics

Now that some of the characteristics of the plant has been determined, the focus can now be shifted to the open loop reactions of the plant. Figure 17 shows the performance of the plant to a step-input using the MATLAB function *step*, at velocities 5 to 25 m/s. The plant builds yaw rate from 0 to 10 m/s as the velocity builds up with the same steering input. The trend then declines until 20 m/s, here the plant becomes unpredictable in open loop, ending with a negative yaw rate. This will mean that at higher velocity, the effort of the controller will be higher to keep the car stable.

Table 4 - Results Step Response

Velocity m/s	Rise time (s)	Poles of system
5	0.0205	$[-1.1693 + 0.0e + 0.0i]$ $[-1.8499 + 0.0e + 0.0i]$
10	0.0344	$[-65.04 + 0.0e + 0.0i]$ $[-85.92 + 0.0e + 0.0i]$
15	0.0185	$[-50.3214 + 9.2318i]$ $[-50.3214 + 9.2318i]$
20	0.0619	$[-37.74 + 12.378i]$ $[-37.74 + 12.378i]$
25	0.0818	$[-30.1928 + 13.59i]$ $[-30.1928 + 13.59i]$

Above are two key parameters for evaluating the performance of the plant. The rise time is used to see what the quickest response the system can have. As it can be seen the quickest time is at 15 m/s, where the rise time is 0.0185 seconds. To ensure that all reactions of the plant are capture, then the sampling rate must be lower than this. Looking at the poles one can see that the system is stable throughout the velocity range and will therefore always find a plateau. (Burton, 2019)

## Vehicle model validation

The validation results of the non-linear and linear vehicle model (figure 18), are showing good correlations. Some key difference between the model and the track data make up the difference in yaw rate. Firstly, the vehicle models are based on the 16-inch OBR20 LCO tire, which according to TTC data should produce higher forces then R25B tire which were on the OBR18. Another factor would be that the tires that were used for this test are used which is seen to decrease the performance (Li *et al.*, 2011). A final reason for the difference is that the tire models assume the perfect surface, which at Bruntingthorpe isn't the case although the surface is purpose-built asphalt. Considering these

parameters, then the derivations are acceptable and most importantly for this project, the trend and behaviour of the vehicle models replicate that of the car.

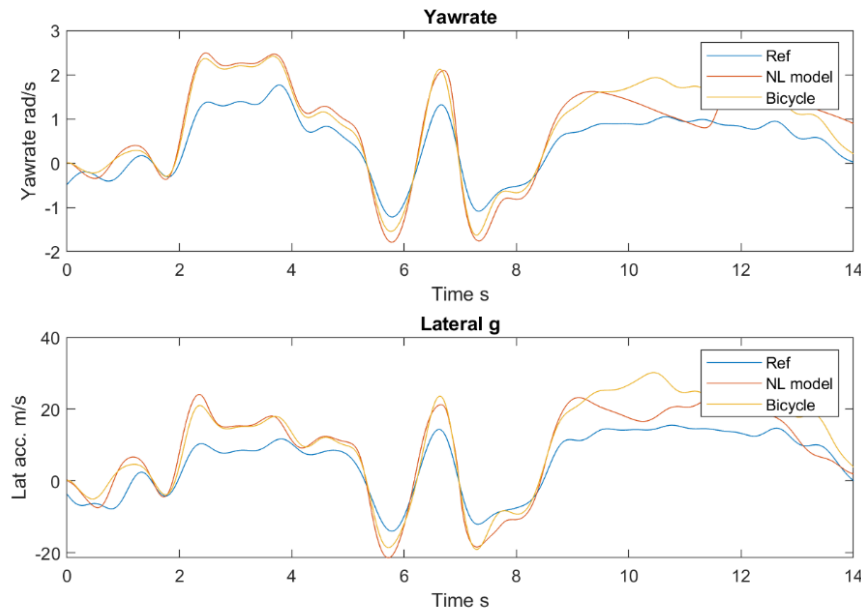


Figure 17 - Validation of non-linear model against OBR18 data

## Controller results

During these simulation the weights of  $Q$  and  $R$  were not changed for the LQR as this wouldn't happen in an event at competition, the same is said for the MPC gains. This will provide non-ideal condition for the MPC and LQR at lower and higher velocities then 10 ms. But that is the purpose of this paper to find the optimal controller for a formula student team and therefore the test always need to reflect the end purpose. The values were however changed between the Step/Sinusoidal input and the track simulation as this would be a possible in between events.

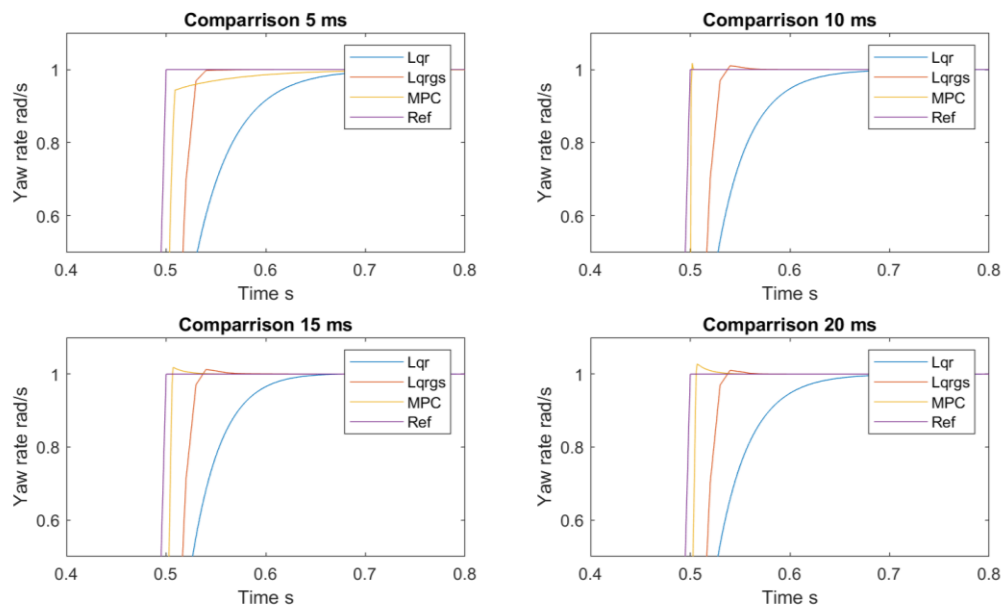


Figure 18 - Controller Comparison Step Reference (Step val. = 1 step time = 0.5 s)

Figure 19 shows the resulting yaw rate to a step input for 4 different constant velocities. Focusing on the 10ms step results, which is where the plant model for the MPC and LQR is equal to the vehicle model, it's possible to observe the ideal controller performance. As expected, the MPC shows an almost perfect performance, with only a small overshoot. This is expected as it will look ahead of the reference signal and can therefore anticipate the step. The LQR and LQRGS work on a full feedback loop and will therefore only react after the step has happened.

For higher velocities the MPC is still showing better results than the other two controllers, but the overshoot slowly increases. Only at lower than 10 m/s is the MPC performance worse than the LQRGS, which would be because of the mismatch in plant and vehicle model, causing the controller to provide a steering angle which is lower than what is needed. At 5 m/s the MPC never gives the correct input as it always underpredicts the necessary control input, therefore giving the slow response.

In general, the LQR controller shows the worst performance, which could have been changed for the 10 m/s with a more aggressive Q value, and the performance could have matched LQRGS. But this provided a large overshoot in the other velocity responses, The Q value were therefore chosen after which one gave the overall best performance between the 4 responses.

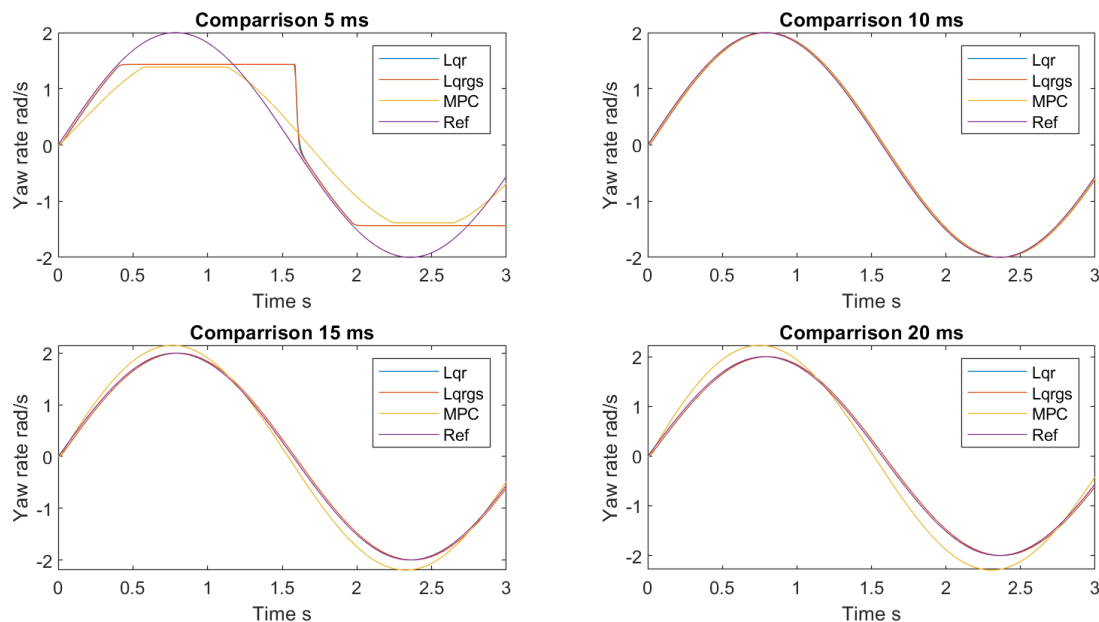


Figure 19 - Controller Comparison Sinus Reference (Sine val. = 2 Freq. = 2 rad/s)

Figure 20 shows the response to the sinusoidal input. Looking at the response of the ideal 10 m/s, the response of all 3 controller show extremely good performance in comparison to the step input. This stems from the fact that the step input is the most aggressive input a controller will see. Looking at the 5 m/s response is showing all controllers are struggling to achieve the desired yaw rate. This give an interesting insight into what the controllers do when it cannot achieve the desired the target. Here all controllers keep the control input at maximum, until the reference decreases again. But as with the step, the feedback loop controller takes longer to react then the MPC, which is anticipating the change in target yaw rate.

From the first two test it's possible to see the limitation of having fully linear controllers, as the actual vehicle model changes velocity and therefore dynamics change. The results show that lower velocity gives a worse performance, compared to having a higher velocity then the plant. This can then be

correlated back to figure 17 that showed the characteristics of system across the velocity range. Here the natural frequency of the system increases rapidly for lower velocities, where at higher velocities it is consistent but with a changing damping ratio. From this it can be concluded that a changing natural frequency on the model will have a greater effect on the performance, than the changing damping ratio as first assumed.

Evaluating these initial results, it becomes clear that when the plant and vehicle model are equal, then the performance of the MPC is superior. The prediction aspect helps to improve the response time with rapid inputs, while having less overshoot than an aggressive feedback loop controller. It can also be seen that the added complexity of gain scheduling on the LQRGS helps to keep the performance consistent for all velocities. Looking back to figure 11 the minimum velocity around Bruntingthorpe is 12 m/s, which means the simulation won't reach the lower velocity, where the MPC and LQR show the worst performance.

Looking at literature (Abirami, 2014) the trend seen above is quite similar in that MPC shows incredible performance, with the LQR following. But as stated in the uniqueness of the project most test is performed with constant velocity and updated plant, so these results showing un-ideal performance are quite unique.

Below are the results from the optimisation traces, while using the linear bicycle model as the vehicle model. The controllers are following the desired traces consistently throughout the velocity ranges. Which as discussed earlier was anticipated due to the performance at higher velocities.

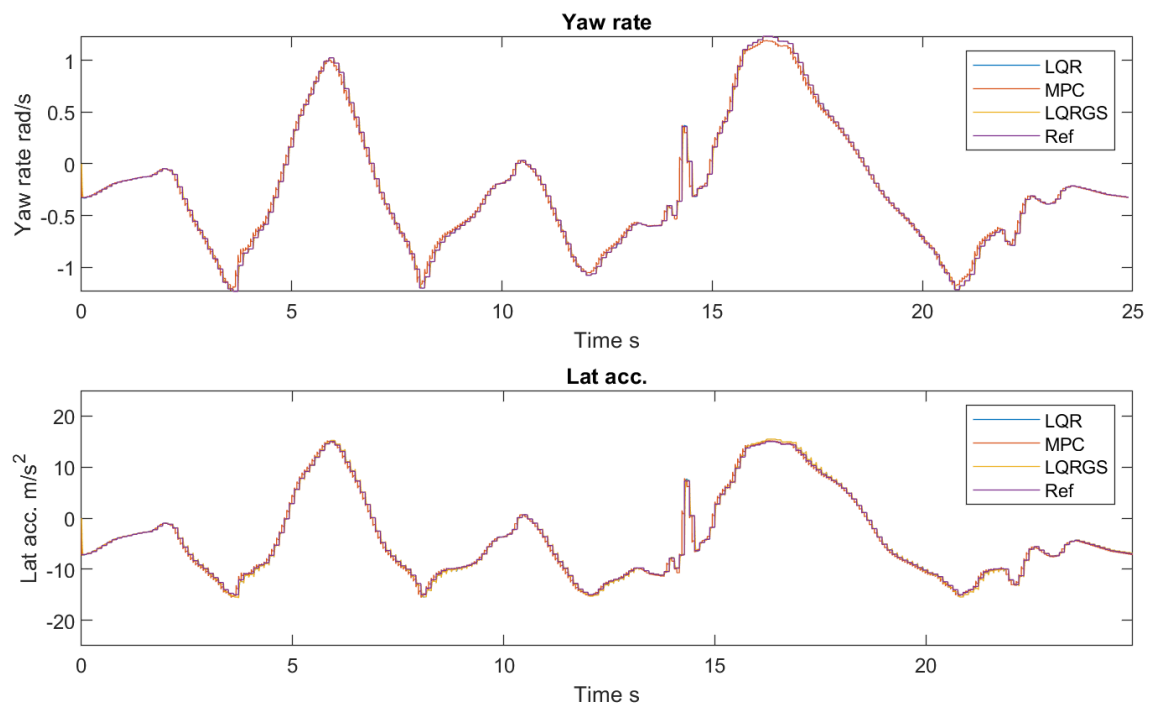


Figure 20 - Linear Vehicle Model Results, All Controllers

As mentioned in the methodology, it was found that RMSE, MAE and IACA were good performance indicators on controller performance. These results can be seen below.

*Table 5 - Linear Vehicle Model Results and computation demand*

Value	LQR	LQR-GS	MPC
RMSE	0.03	0.0301	0.0202
MAE	0.0095	0.0091	0.0134
IACA	84.24	26.64	25.19
Avg. Simulation Time in seconds	0.73	0.8	0.95
Simulated Event Length: 10 seconds			

Looking at the RMSE, then the prediction of the MPC minimised the large errors in the general trend of the simulation, but the MAE is higher than both LQR controllers.

As discussed in the methodology, the RMSE punish large outliers more than the MAE equation. This means that in general the MPC does a worse job at following the trace compared to the LQR, but it compensates in the that it has fewer large outliers. Thinking about the car going around the track, then this is a desirable quality as small errors will provide less performance, but a large error can set the car going off the track.

Looking at the IACA the LQRGS and MPC have similar effort levels, with the MPC coming out on top, due to the prediction capabilities of the controller. The LQR controller shows to have a larger effort than the others, which can be explained from the tuning of the controller. When the simulation of the controller was performed each one was tuned to give the minimum RMSE and MAE. This was done because having high values could mean that the car goes off track, where a high IACA values would only impact the tire degradation as described in the methodology.

The performance from the first test with a linear plant show that all 3 controllers would be acceptable choices for a formula student team with the best option currently being the LQRGS. This is due to lower computational time (table 6) compared to the MPC and the lower IACA compare to the LQR.

Figure 22 show the results of simulation with a non-linear vehicle model. This is where the Linear controllers LQR and MPC really struggle with controlling the vehicle. One of the major contributors is the reaction of the plant to the steering input, which in the linear bicycle mode had linear tire model that no matter the slip angle would provide a higher force, therefore enabling the controllers to be tuned aggressively. These aggressive tuning parameters showed to make the non-linear vehicle model spin out, so the use of the steering input had to punished 20 times more than in the linear test.

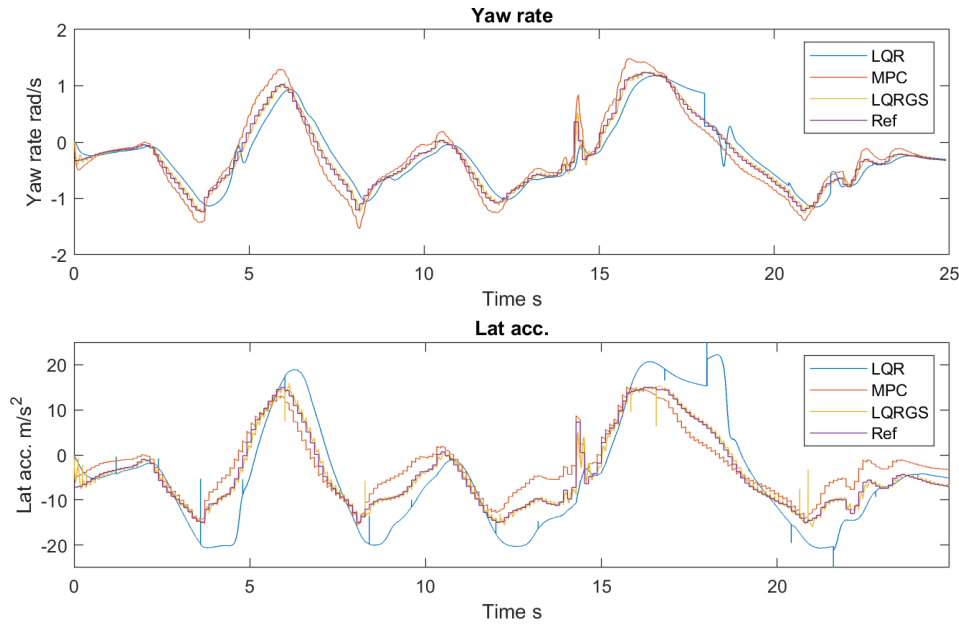


Figure 21 - Non-Linear Vehicle Model Results, All Controllers

As mentioned in this project, one of the unique aspects is also testing these controllers on a non-linear vehicle model and just from comparing Figure 21 and 22, one can see how important it is to do these investigations, as the linear vehicle model provided false prediction of performance. It also becomes clear why non-linear controllers such as the LQRGS are desirable, which does exist for MPC as AMPC and NMPC. But as mentioned in the methodology the formula student team is limited by only having a single ETAS ES910 to run the entire controls system of the car, which includes torque vectoring, BMS etc.

Table 6 - Non-Linear Vehicle Model Results

Value	LQR	LQR-GS	MPC
RMSE	0.1936	0.0656	0.7002
MAE	0.1524	0.0417	0.5456
IACA	20.87	24.15	29.87

Looking again at the RMSE, MAE and IACA it is possible to get a clear view of the performance of each controller. As mentioned above, the controls input had to be punished more, which can also be confirmed by the smaller IACA values compared to linear results. Only the MPC has a larger IACA but looking at the RMSE and MAE it can be concluded that the controller can no longer predict the behaviour of the plant and looking at figure 22 the MPC hardly ever follows the reference trace. As with the linear model, it is again the LQRGS that shows the best performance.

From the results in table 5 and 6 one can see that a linear controller with linear vehicle model (w. changing velocity), can provide sufficient performance. But when the vehicle model turns heavily non-linear, then a non-linear controller is needed. Even though the plant model in the LQRGS has a linear tire model, the non-linearity of the velocity change provides enough information to maintain good control.

## 6. Conclusion

In this project relevant papers and literature have been investigated to gain a foundation for what optimal trajectory and control is and what is required to fulfil these. For optimal controllers it was found that the optimality was based on an optimisation problem that was solved with a specific plant model, that represents the system that it trying to be controlled.

The equations necessary for a 2DoF linear bicycle model, that could represent the lateral dynamics of the vehicle where found and the advantages and disadvantages where investigated and discussed. A state-space representation was derived to be used as a plant model for the controllers.

It was found that an optimal trajectory could be found with either minimum curvature or shortest path and further improvements with iterations were found. These where tested in Python using an open-source code as a basis, which was then adapted to fit the requirements for a Formula Student car. The outputted trajectory was compared, and it was concluded that the optimised minimum curvature optimisation gave the optimal path, for Bruntingthorpe.

Considering the dynamics of a formula student car and the computational demand, it was decided to proceed with 3 different controllers. The controllers were MPC, LQR and LQRGS, which all completed the optimal controller requirement while showing to have low computational requirement. It was evaluated that these would control two vehicle model, a linear bicycle model with changing constant and changing velocity and would then be tested on non-linear model, that would capture a non-linear MF6.2 tire model and weight transfer.

Each of these controllers were optimised for the different test, but while keeping the linear controller's plant completely linear at one velocity, as this was concluded to be the most realistic testing environment. It could be seen that between the bicycle model and the non-linear model, it was necessary to punish the control input to the vehicle model.

From the test it can also be concluded that the LQRGS is the optimal path controller for the Formula student team. This should be used in collaboration with the optimised minimum curvature path, to achieve the best performance at the competition.

## 7. Future improvements

As this project is OBR first investigation into autonomous controls, a big focus was on getting reliable and low computational requirement controllers, that could be created using the available MATLAB and Simulink programs, to ensure that everyone in the team could use and make changes to the controls system.

For future work, it would be interesting to investigate the exact requirements of the whole controls system, in order to be able to point to an exact computational demand that the steering controller could use. This could enable the use of AMPC and NMPC, which in the literature showed incredible performance, but comes with a heavy computational demand.

Neural network and learning algorithms could also be implemented for the endurance part of the competition, where the controller could be adjusted to learn faster routes or if it hits a cone and would then back off the velocity in that part of the track in the next lap.

For the testing of the controllers it could be improved by having the controller runs passively, while a driver goes around the track. By doing this one could get an idea of how the controls system reacts to real input data. As could be seen in the imported data from testing, it can be noisy and before letting the controls system drive the car, then this could indicate if the system is processing the data correctly and how these effects the performance of the controllers.

It could also be interesting to test the controllers on a full vehicle dynamics model, to investigate which exact parameters affect the performance of the controllers most.

Future improvements could also involve having the trajectory optimisation run online in the background, but would require coding work to use the online input from the state-estimator to constantly iterate the optimal trajectory as new data comes in.



## References

- Abirami S, H.Kala H, P.B.Nevetha P, B.Pradeepa B, R.Kiruthiga R and P.Sujithra P (2014) Performance Comparison of Different Controllers for Flow Process. *International Journal of Computer Applications*. 90 (19), 17-21.
- Anderson J and Ayalew B (2018) Modelling minimum-time manoeuvring with global optimisation of local receding horizon control. *Vehicle System Dynamics*. 56 (10), 1508-1531.
- Andersson A and Nasholm E (2018) Fast Real-Time MPC for Fighter Aircraft. Master degree. Linköping University.
- FSUK (2019) [Online] Available at: [https://www.imeche.org/docs/default-source/1-oscar/formula-student/2019/results/fs\\_uk\\_2019\\_class1\\_result\\_sprint.pdf?sfvrsn=2](https://www.imeche.org/docs/default-source/1-oscar/formula-student/2019/results/fs_uk_2019_class1_result_sprint.pdf?sfvrsn=2) (accessed 18/ 09/ 19).
- FSG (2019) *FSG: Results FSG 2019*. [Online] Available at: <https://www.formulastudent.de/fsg/results/2019/> (accessed 18/ 09/ 19).
- Ghasemzadeh Ebli H, Khani A and Azizi A (2013) Gain Scheduling Controller for Missile Flight Control Problem by Applying LQR. *JWEET*.
- Balkwill J (2018) *Performance Vehicle Dynamics: Engineering and Applications*. Butterworth-Heinemann
- Basenese (2019) Why Driverless Cars Are Here to Stay. [Online] Available at: <https://www.wallstreetdaily.com/2017/09/05/driverless-cars-stay/> (accessed 03/ 07/ 19).
- Besselink I, Schmeitz A and Pacejka H (2010) An improved Magic Formula/Swift tyre model that can handle inflation pressure changes. *Vehicle System Dynamics*. 48 (sup1), 337-352.
- Borrelli F, Falcone P, Keviczky T, Asgari J and Hrovat D (2005) MPC-based approach to active steering for autonomous vehicle systems. *International Journal of Vehicle Autonomous Systems*. 3 (2/3/4), 265.
- Brunton S (2019) *Control Bootcamp: Linear Quadratic Regulator (LQR) Control for the Inverted Pendulum on a Cart*. [Online] Available at: [https://www.youtube.com/watch?v=1\\_UobILf3cc&list=PLMrJAKhIeNNR20Mz-VpzgfQs5zrYi085m&index=15&t=0s](https://www.youtube.com/watch?v=1_UobILf3cc&list=PLMrJAKhIeNNR20Mz-VpzgfQs5zrYi085m&index=15&t=0s) (accessed 18/ 09/ 19).
- Carvalho A, Gao Y, Gray A, Tseng H and Borrelli F (2013) Predictive control of an autonomous ground vehicle using an iterative linearization approach. 16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013).
- Casanova D (2000) On minimum time vehicle maneuvering: The theoretical optimal lap. Ph. D. Cranfield University.
- Cardamone L, Loiacono D, Lanzi P and Bardelli A (2010) Searching for the Optimal Racing Line Using Genetic Algorithms. *IEEE*. (978-1-4244-6297-1).
- Davin E (2011) Parameter identification of a linear single track vehicle model. Traineeship Report. University of Eindhoven.
- Heilmeier A, Wischniewski A, Hermansdorfer L, Betz J, Lienkamp M and Lohmann B (2019) Minimum curvature trajectory planning and control for an autonomous race car. *Vehicle System Dynamics*, 1-31.
- Li Y, Zuo S, Lei L, Yang X and Wu X (2011) Analysis of impact factors of tire wear. *Journal of Vibration and Control*. 18 (6), 833-840.
- Liniger A, Domahidi A and Morari M (2014) Optimization-based autonomous racing of 1:43 scale RC cars. *Optimal Control Applications and Methods*. 36 (5), 628-647.

Marques L (2016) Optimal Lap Time for a Race Vehicle. Masters degree. Tecnico Lisboa.

MathWorks (2019) Understanding Model Predictive Control, Part 1: Why Use MPC?. [Online] Available at: <https://www.youtube.com/watch?v=8U0xiOkDcmw> (accessed 04/ 07/ 19).

Rajamani R (2006) *Vehicle Dynamics and Control*. Springer

Roborace (2019) *The World's FASTEST Autonomous Car / Robocar Guinness World Records / Roborace*. [Online] Available at: [https://www.youtube.com/watch?v=6WmprzuTc\\_U&t=199s](https://www.youtube.com/watch?v=6WmprzuTc_U&t=199s) (accessed 18/ 09/ 19).

Salazar M, Balerna C, Elbert P, Grando F and Onder C (2017) Real-Time Control Algorithms for a Hybrid Electric Race Car Using a Two-Level Model Predictive Control Scheme. *IEEE Transactions on Vehicular Technology*. 66 (12), 10911-10922.

Stankovic J (1988) Misconceptions about real-time computing: a serious problem for next-generation systems. *Computer*. 21 (10), 10-19.

Squark (2019Fr) *Mean Absolute Error or MAE / Squark*. [Online] Available at: <https://squarkai.com/mean-absolute-error-or-mae/#.XXeY0TZKiUk> (accessed 18/ 09/ 19).

Wang L (2010) *Model predictive control system design and implementation using MATLAB*. London: Springer

Wurm S (2019) Model based software development and validation [Oxford Brookes Racing Seminar] Oxford Brookes University. March 2019.

Wang Z, Montanaro U, Fallah S, Sorniotti A and Lenzo B (2018) A gain scheduled robust linear quadratic regulator for vehicle direct yaw moment Control. *Mechatronics*. 51, 31-45.

Wu S, Chiang H, Perng J, Chen C, Wu B and Lee T (2008) The Heterogeneous Systems Integration Design and Implementation for Lane Keeping on a Vehicle. *IEEE Transactions on Intelligent Transportation Systems*. 9 (2), 246-263.

## Appendix A – Supplementary Results

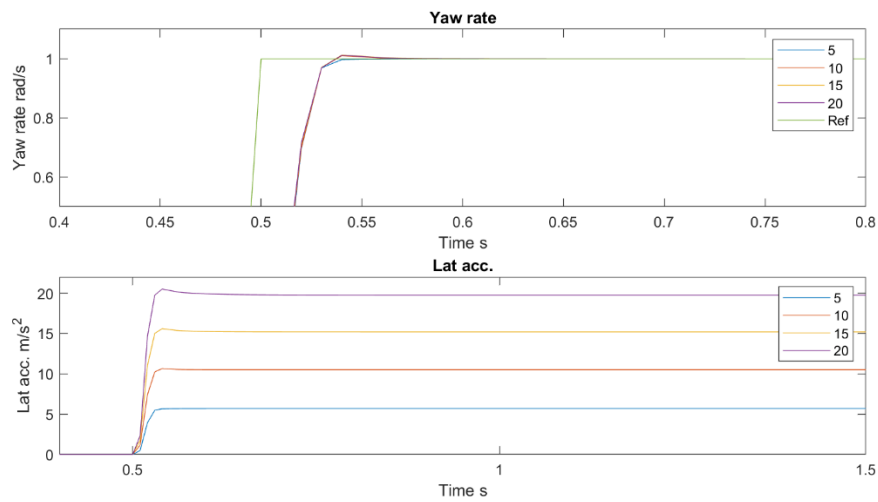


Figure 22 - All Gain-Schedule LQR Results (Step val. = 1 step time = 0.5) – Start of Appendix

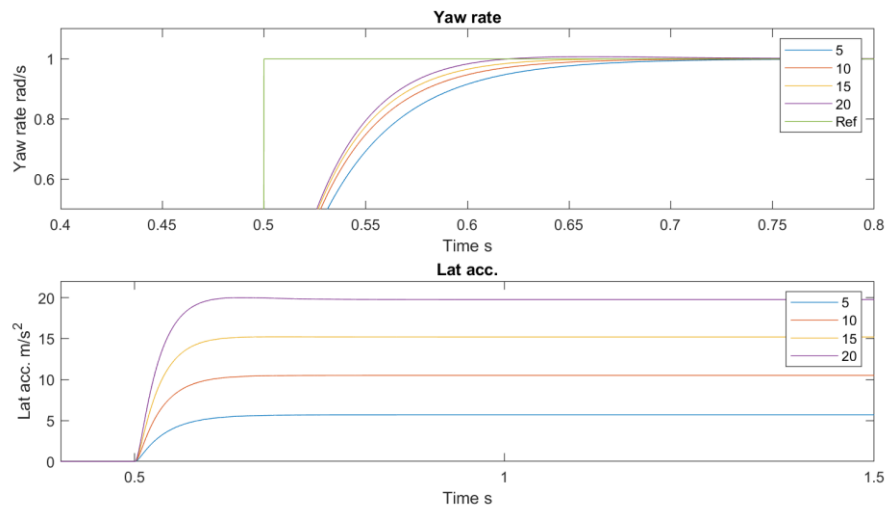


Figure 23 - All LQR Results (Step val. = 1 step time = 0.5)

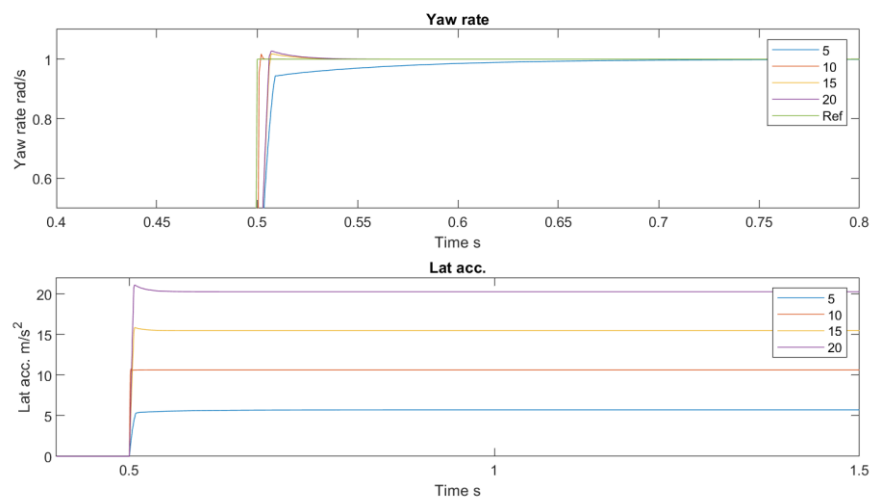


Figure 24 - All MPC Results (Step val. = 1 step time = 0.5)

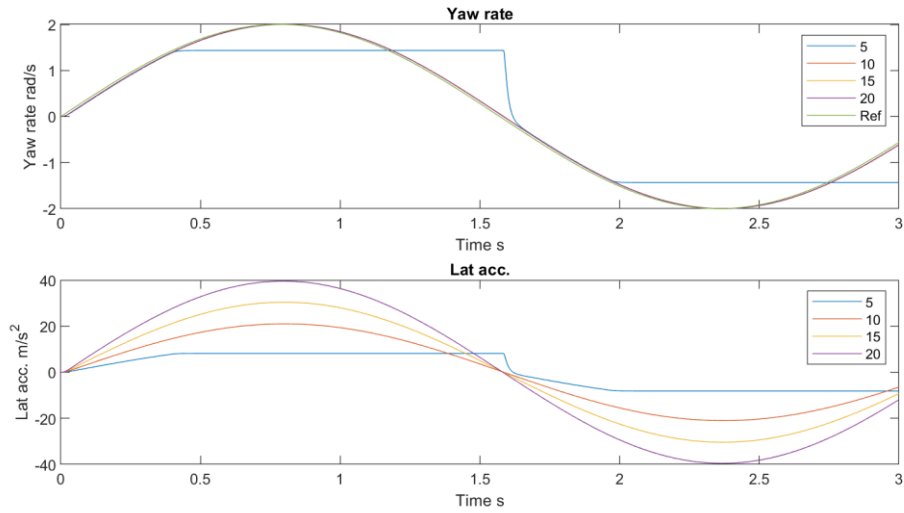


Figure 25 - LQR Sine Results

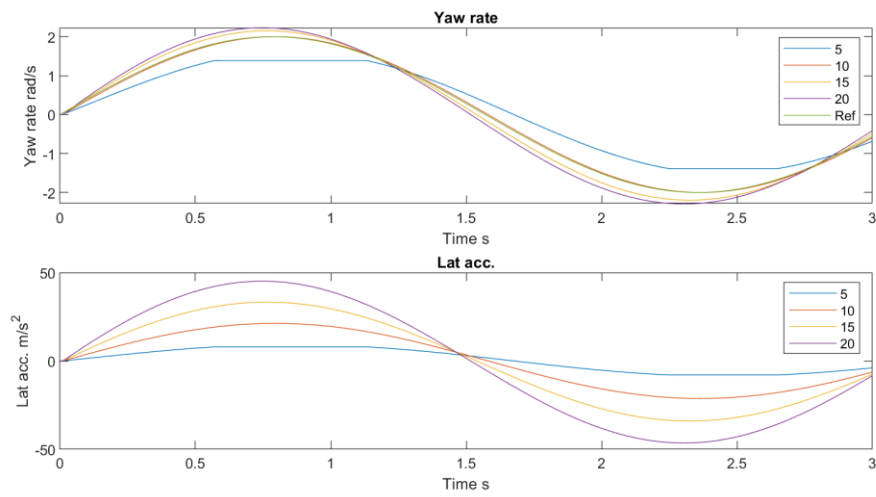


Figure 26 - MPC Sine Results

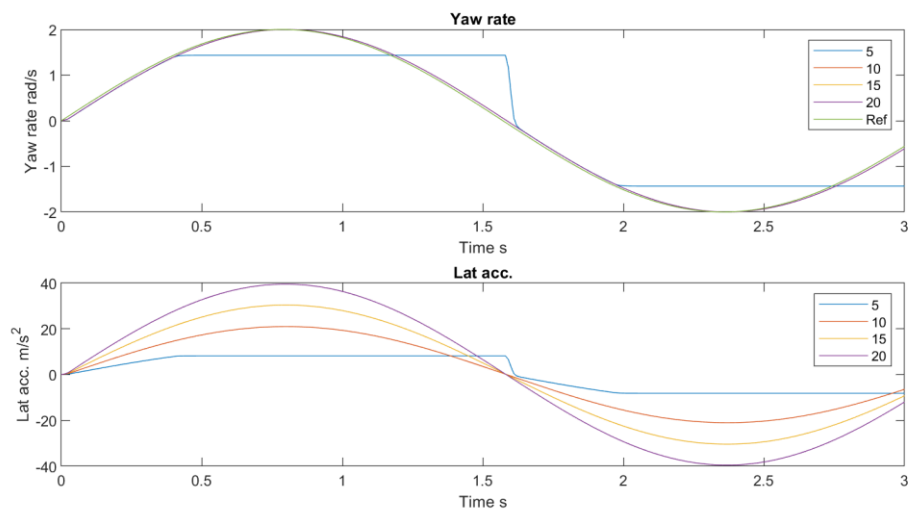


Figure 27 - LQRGS Sine Results

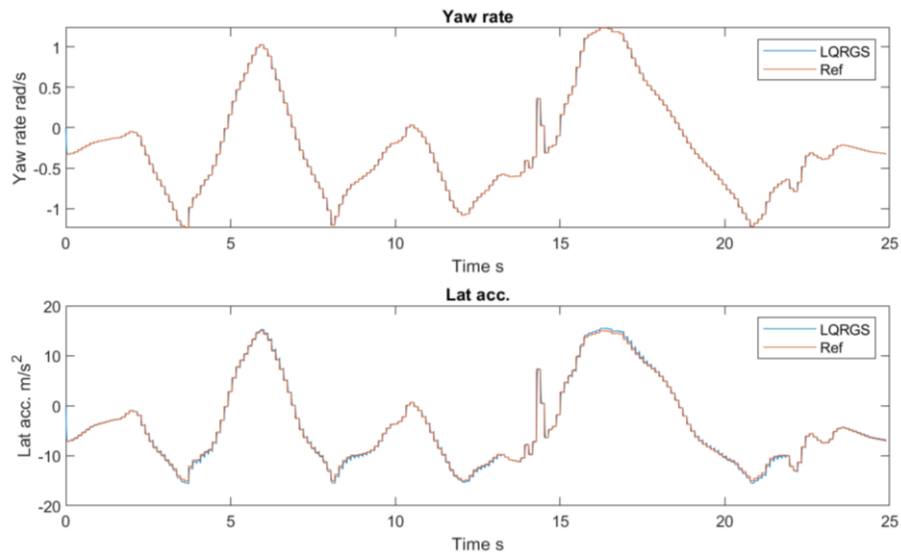


Figure 28 - Gain-Scheduled LQR Track Result Linear Vehicle model

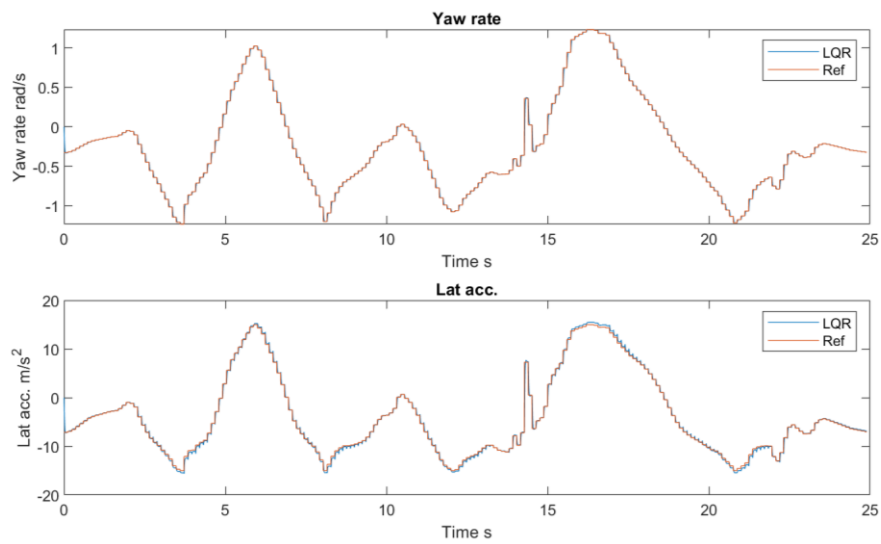


Figure 29 - LQR Track Result Linear Vehicle model

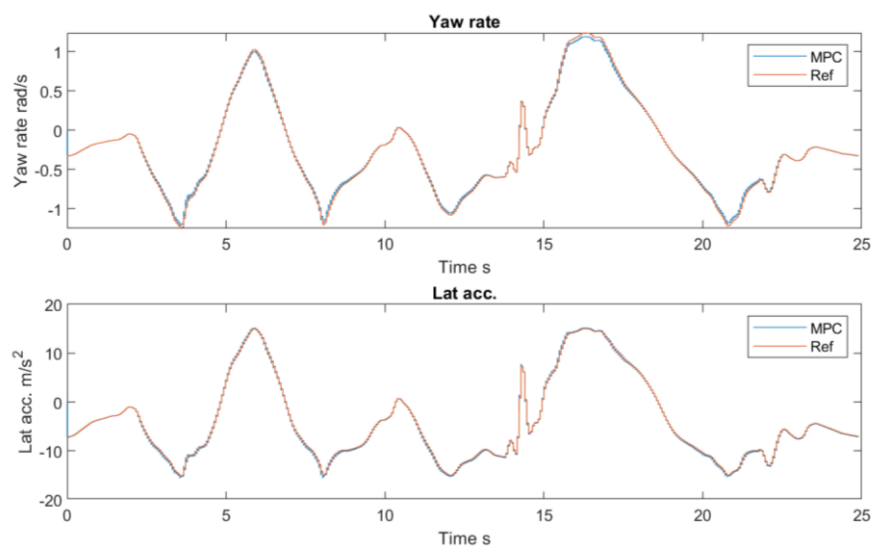


Figure 30 - MPC Track Result Linear Vehicle model

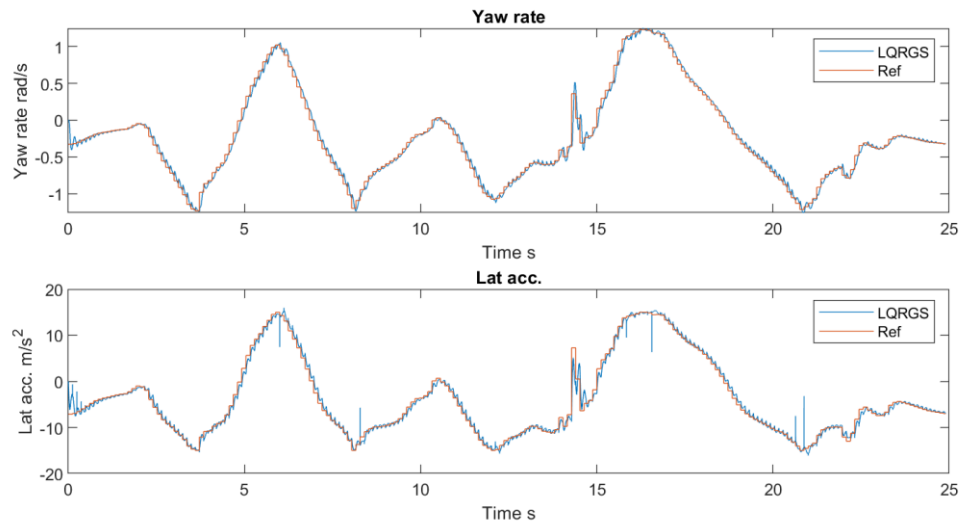


Figure 31 - LQRGS Results Non-Linear Vehicle Model

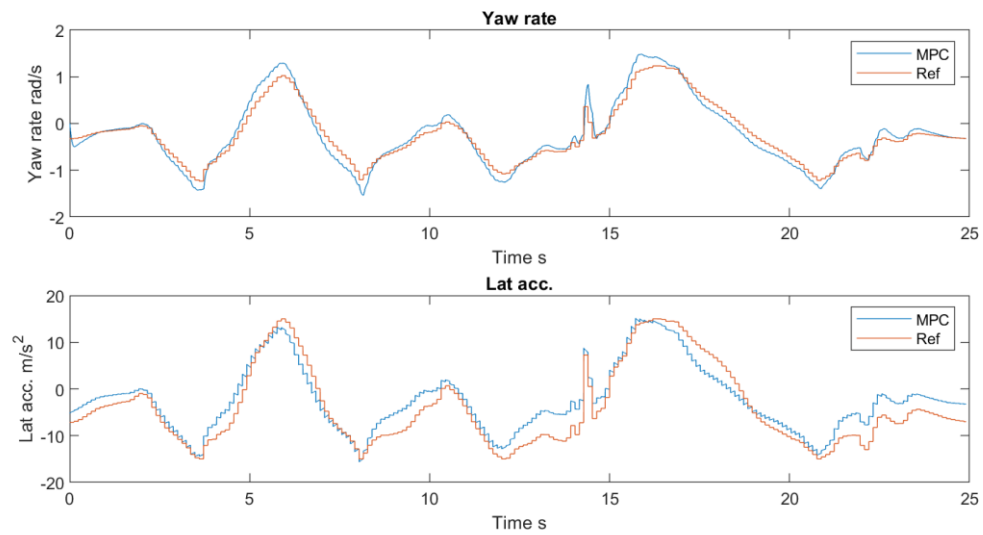


Figure 32 - MPC Results Non-Linear Vehicle Model

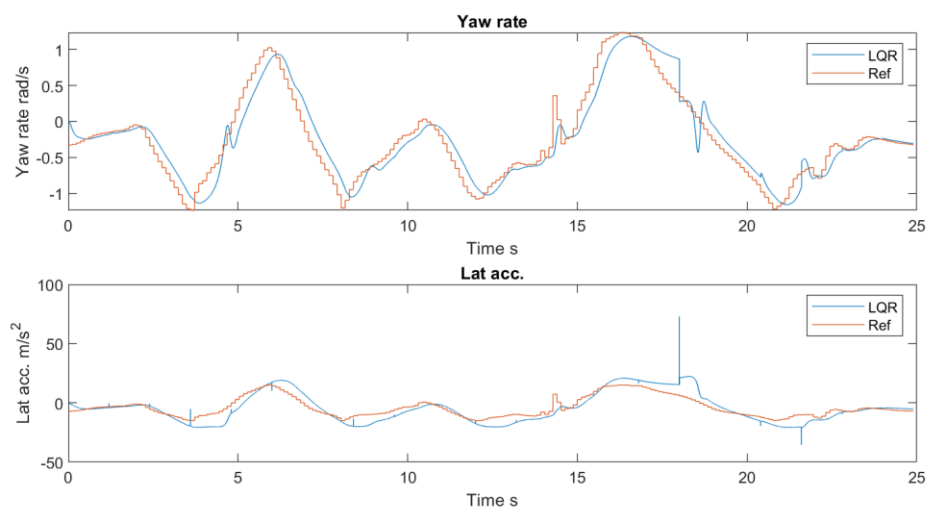


Figure 33 - LQR Results Non-Linear Vehicle Model

## Appendix B – Vehicle parameters

Table 7 - Vehicle Parameters

Vehicle Parameters Controller and Vehicle Model		
Parameters	Value	Unit
$m$	250	kg
$l_f$	0.753	m
$l_r$	0.723	m
$I_{zz}$	115	
$h$	0.3	m
min/max $\delta$	-/+ 0.43	rad

## Appendix C – Non-linear Simulink Code

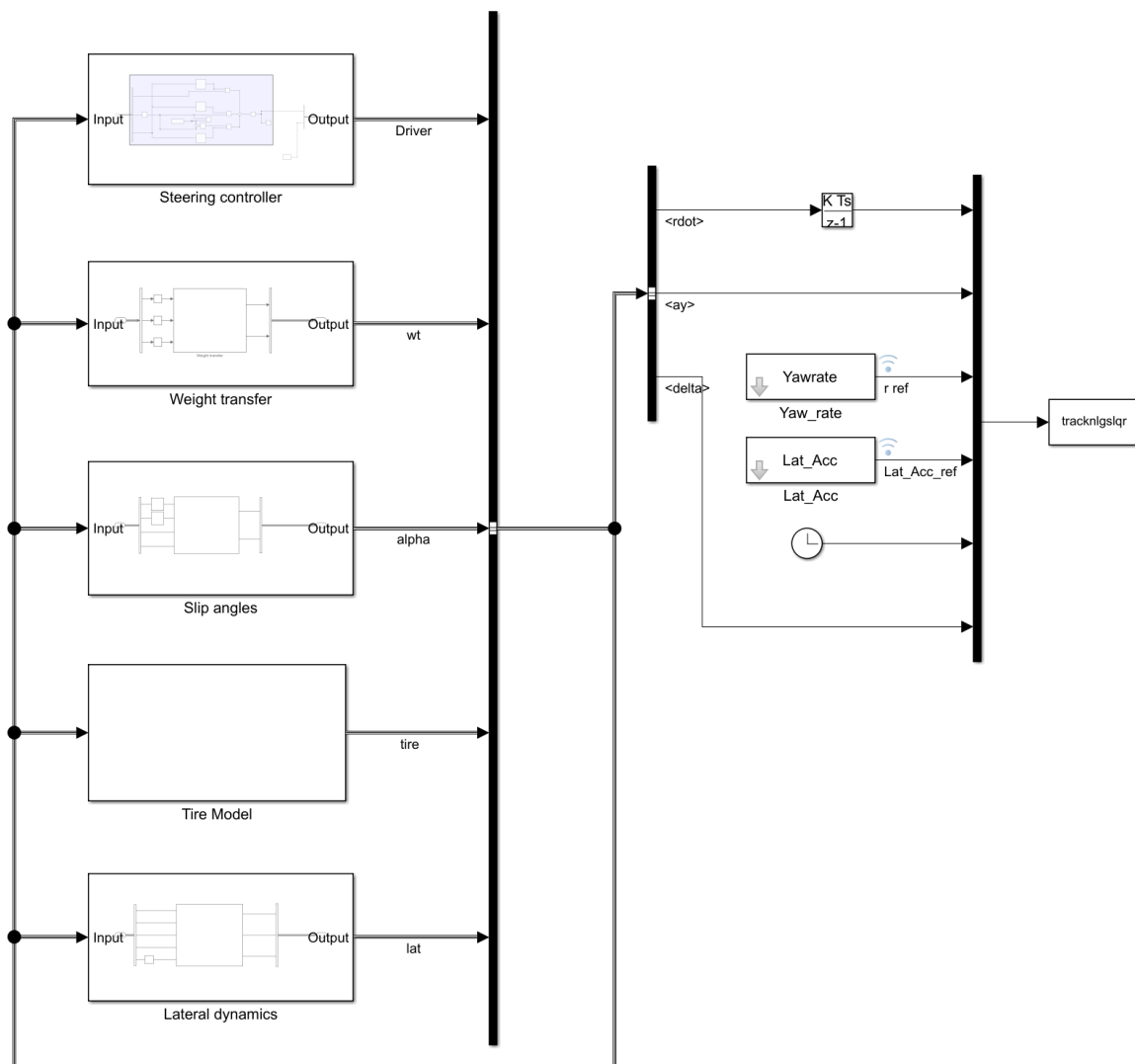


Figure 34 - Non-Linear Vehicle Model w. LQRGS Simulink Model