# Developing an Autosteering of Road Motor Vehicles in Slippery Road Conditions

Natalia Alekseeva

Department of Information and Computer Science
Graduate School of Science and Engineering
Doshisha University

# Table of Contents

*Abstract*

*Acknowledgements*

*List of Abbreviations*

*List of Figures*

*List of Tables*

## CHAPTER 1: Introduction

1.1.Background and Motivation..... 1

1.2.Objectives..... 3

1.3.Related researches..... 4

1.4.Thesis Contributions..... 4

1.5.Thesis Overview..... 5

1.5.1 Research Framework (Chapter-2)..... 5

1.5.2 Upgrading SAF for PID and PD controllers with tuning parameters (Chapter-3)..... 5

1.5.3 Upgrading SAF for PID and PD controllers with prediction (Chapter-4)... ..... 6

1.5.4 Relaxed structure of PD controller analytical model, developed heuristically via the GP – RM-GP (Chapter-5)..... 6

1.5.5 Evaluation of the GP-RMEP controller with extended parameters obtained via GP (Chapter-6)..... 6

1.5.6 Summary, Conclusion, and Future Work (Chapter-7)..... 7

## CHAPTER 2: Research Framework

2.1. Model of the Autonomous Car..... 9

2.1.1. Real-time Feedback Control Mechanism..... 9

2.1.1.1.Acceleration and brake control of the car keeping desired speed..... 9

2.1.1.2.Steering angle control..... 11

2.1.1.2.1 Physical and mechanical constrains. Maximum lock angle ..... 15
2.1.1.2.2 Mechanical constrains (features). Ackerman principle ..... 16
2.1.1.2.3 Mechanical constrains. Delay ..... 17
2.1.2. Environment ..... 18
2.1.3. Perception ..... 18
2.1.4. Reaction ..... 18
2.2.Testing and Simulation Software Environment ..... 19
- Credits ..... 19
- Platforms ..... 19
- Features ..... 19
2.2.1.Simulation architecture ..... 20
2.2.2.Simulation parameters ..... 23
2.2.2.1.The Car ..... 23
2.2.2.2.Test Track ..... 25
2.2.2.3.Automation test runs ..... 27
2.2.2.4.Race target and critical speed ..... 28
2.3 TORCS Requirements ..... 29
2.3.1 Software ..... 29
2.3.1.1 OpenGL ..... 29
2.3.1.2 GLUT or FreeGLUT for Linux (OpenGL Utility Toolkit) ..... 30
2.3.1.3 Libpng ..... 30
2.3.1.4 Checking library availability ..... 30
•Checking OpenGL/DRI ..... 30
•Checking GLUT ..... 31
•Checking Libpng ..... 31
2.3.2 Hardware ..... 31
2.4 Evolutionary computation ..... 32
2.4.1 Algorithm loop of the search of the reproduction population ..... 34
2.4.2 Genetic Programming (GP) Approach ..... 37
2.4.3 Evolution aim ..... 39
2.4.4 Estimation ..... 39
2.5 Summary ..... 41

**CHAPTER 3: Upgrading SAF for PID and PD controllers with parameters tuning**

3.1. Materials and Methods ..... 43
3.1.1. Algorithm Summary and Components ..... 44
3.1.2. Tuning parameters with brute force ..... 47
3.2. Summary and Discussion ..... 51

**CHAPTER 4: Upgrading SAF for PID and PD controllers with prediction**

4.1. Algorithm Summary and Components ..... 54
4.1.1. Human behaviour during the race. Human prediction tactic ..... 54
4.1.2. Projection of the geometry position of the Car ..... 56
4.1.3. Simulators facilities for driver prediction ..... 59
4.2. Results ..... 61
4.2.1. Comparison with the classical approach ..... 61
4.2.2. Features obtained ..... 64
4.2.2.1. Time needed to return on the lane ..... 65
4.2.2.2. Critical speed increasing. New trajectory features ..... 67
4.2.2.3. Safe distance between car and obstacle ..... 68
4.3. Discussion ..... 70
4.3.1. Prediction time distance ..... 70
4.3.2. Special shape of the turning trajectory ..... 72
4.3.3. Performance and Validation ..... 75
4.3.4. Future Work and Summary ..... 76

**CHAPTER 5: Relaxed structure of PD controller analytical model, developed heuristically via the GP – RM-GP**

5.1. Materials and Methods ..... 80
5.1.1. Algorithm Summary and Components ..... 80
5.1.2. Algorithm Convergence ..... 81
5.2. Discussion ..... 83
5.2.1. Performance and Validation ..... 86
5.2.1.1. Future Work ..... 87

## CHAPTER 6: Evaluation of the GP-RMEP controller with extended parameters obtained via GP


6.1. Materials and Methods ..... 90

6.2. Experiments and Results ..... 91

6.2.1. Algorithm Convergence ..... 92

6.2.2. Efficiency comparison of the proposed controllers ..... 93

6.2.3. Universal (robust, general) equations ..... 93

6.3. Discussion ..... 95

6.3.1. Performance and Validation ..... 95

6.3.2 Driving with noisy input data

6.3.3. Oscillation analysis and validation ..... 97

6.3.3.1.Car modelling ..... 99

6.3.3.2.Evolutionary structure ..... 100

6.3.3.3.Oscillation suppression ..... 100

6.3.3.4. Inducing an artificial oscillation ..... Error! Bookmark not defined.

6.3.3.5.Possible reasons for such oscillations producing ..... 103

6.3.3. Summary and Future Work ..... 106


## CHAPTER 7: Summary, Conclusion, and Future Work


7.1. Summary and Conclusion ..... 109

7.2. Future work ..... 112

**Bibliography**……115

**Appendix**

XML GP Schema for Evolving Driving Agent ..... 118

XML GA Schema for Optimising Weight Coefficient of Power Spectrum ..... 119

Program Source Code ..... 120

**Publications**

Journal Papers ..... 121

Conference Paper ..... 121

# Abstract

In the nearest future, the human driver is viewed as a reliable backup even for the fully automated road motor vehicles (cars). Indeed, the driver is assumed to swiftly take the control of the car in cases of suddenly occurring (i) challenging environmental conditions, (ii) complex unforeseen driving situations, or (iii) degradation of performance of the car. However, due to the cognitive overload in such a sudden, stressful takeover of the control, the driver would often experience the startle effect, which usually results in an unconscious, instinctive, yet incorrect response. An extreme case of startle is freezing, in which the driver might be incapable to respond to the sudden takeover of control at all.

The possible approaches to alleviate the startle during the takeover of control (i.e., the automation startle) include an offset- (i.e., either early- or delayed-), gradual yielding the controls to the driver. In the cases considered above, however, these approaches are hardly applicable because of (i) the presumed unpredictability of the events that result in the need of takeover of control, and (ii) the severe time constraints of the latter.

Conversely, the *objective* of our research is to propose an approach of minimizing the need of yielding the control to the driver in challenging environmental conditions by guaranteeing an adequate automated control in these conditions. Focusing on slippery roads as an instance of challenging conditions, and steering control as an instance of control, we aim at developing such an automated steering that controls

the car adequately in various road surfaces featuring low friction coefficients without the need of driver's intervention.

In order to develop such an automated steering we employed an in-house evolutionary computation framework – XML-based genetic programming (XGP) – which offers a flexible, portable, and human readable representation of the evolved optimal steering functions. The trial runs of the evolved steering functions were performed in the Open Source Racing Car Simulator (TORCS), which features a realistic, yet computationally efficient simulation of the car and its environment.

The obtained experimental results indicate that due to the challenging dynamics of the unstable car on slippery roads, neither the canonical (tuned) servo-control (as a variant of PD) nor the (tuned) PID-controller could control the car adequately on slippery roads. On the other hand, the controller, featuring a relaxed, arbitrary structure evolved by XGP outperforms both the servo- and PID controllers in that it results in a minimal deviation of the car from its intended trajectory in rainy, snowy, and icy road conditions. Moreover, the evolved steering that employs anticipated perceptions is even superior as it could anticipate the imminent understeering of the car at the entry of the turns and consequently – to compensate for such an understeering by proactively turning the steering wheels in advance – well before entering the turn. The obtained results suggest a human competitiveness of the evolved automated steering as it outperforms the commonly used alternative steering controllers proposed by human experts.

The research could be viewed as a step towards the evolutionary development of automated steering of cars in challenging environmental conditions.

# List of Abbreviations

AI : Artificial Intelligence

API : Application Programming Interface

CAD : Computer-Aided Design

GLUT : The OpenGL Utility Toolkit library

GP : Genetic Programming

GPL : General Public License

GP-RMEP : Relaxed Model with Extended Parameters set constructed via Genetic Programming

GPU : A Graphics Processing Unit

MPC : Predictive Control Model

OpenGL : Open Graphics Library

PD : Proportional Derivative controller

PID : Proportional Integral Derivative controller

PNG : Portable Network Graphics

PPD : Predictive Proportional Derivative controller

RM-GP : Relaxed Model constructed via Genetic Programming

RPM : Red Hat Package Manager

SAF : Steering angle function

TORCS : The Open Racing Car Simulator

UDP : User Datagram Protocol

# List of Figures

*Fig- 1* Rear-driving car schema ..... 13
*Fig- 2* Kinematics of turning the wheels of the front axle ..... 14
*Fig- 3* Maximum steering angle ..... 15
*Fig- 4* Ackerman wheel position (left) and parallel wheel position (right) ..... 16
*Fig- 5* The architecture of the competition software ..... 21
*Fig- 6* Simulated car in TORCS lateral side view ..... 23
*Fig- 7* Hook-type test track ..... 26
*Fig- 8:* Fitness landscape and the location of the population of four candidate-solutions of a given (current) generation. The evolution attempts to find a new population of candidate solutions that are presumably “higher” in the fitness landscape. The new population is obtained by recombining (via genetic operations – crossover and mutation) the candidate solutions of the current generation. ..... 34
*Fig- 9:* Flowchart of genetic programming and simple imitation of the mechanism of natural selection ..... 35
*Fig- 10* Fitness function evaluation Flowchart ..... 36
*Fig- 11* Driving the car steered by SAF in TORCS ..... 36
*Fig- 12* Crossover operation example ..... 38
*Fig- 13* Mutation operation example ..... 38
*Fig- 14* Steering wheel control system ..... 39
*Fig- 15* PID controller feedback control system ..... 43
*Fig- 16* Servo-control model of steering as a PD steering controller. The SAF, defining the steering angle δ is implemented as a sum of the proportional- (P) and derivative (D) terms of the error – the deviation e from the center of the lane. ..... 45
*Fig- 17* Car under servo model control parameters ..... 47
*Fig- 18* Sequential search of values combination $k_1$ and $k_2$. F – is a value of the estimation function (left), the behavior of the car on a wet road corresponding to the values found for the combination of parameters (right). The dashed line is the desired trajectory of the car. ..... 47
*Fig- 19* Sequential search of values combination $k_1$ , $k_2$ and $k_3$. The lowest values of the estimation function are marked with lilac colour. ..... 49
*Fig- 20* Car with tuned PID controller parameters trajectories ..... 50
*Fig- 21* Trajectories of the car under the PD controller in the different road conditions (on the slippery road, $\mu = 0.5$ (wet surface) – red curve, on the dry road – blue curve) ..... 52
*Fig- 22* The effect of the car is shifting "outward" relative to the desired trajectory during maneuvering while entering a turn that leads to losing control and stability

after some time. The effect is seen more clearly than in previous images due to the increased wetness of the road surface (μ = 0.3) ..... 55
*Fig- 23* Predicting the lateral deviation of the car $e_{pred}$ ..... 58
*Fig- 24* Trajectory of the car predicted position on the road, $\mu$= 0.3 ..... 58
*Fig- 25* Track segment sample ..... 60
*Fig- 26* Track segmentation schema ..... 61
*Fig- 27-1* Car trajectories on the track tuned with prediction SAF of standard PD controller for friction coefficients μ=0.5 (a-c), μ=0.3 (d-f), μ=0.1 (g-i) respectively. The blue curves correspond to the trajectories controlled SAF with prediction (PPD controller), red – original SAF ..... 62
*Fig- 27-2* Car trajectories on the track tuned with prediction SAF of standard PD controller for friction coefficients μ=0.5 (a-c), μ=0.3 (d-f), μ=0.1 (g-i) respectively. The blue curves correspond to the trajectories controlled SAF with prediction (PPD controller), red – original SAF ..... 63
*Fig- 28* Dynamics of the steering angle for PD (red curve) controller and PPD (blue curve) controller. Environment conditions: μ = 0.3, $0.95V_{cr}$ ..... 64
*Fig- 29* Trajectories of the car exceeded the critical speed for friction coefficient $\mu$=0.1 with target speed equal to $1.05VCR$. The blue curve correspond to the trajectory controlled by PPD controller, red – original PD controller. ..... 67
*Fig- 30* (i) Smooth, uniform and short PPD controller trajectory, ..... 68
*Fig- 31* Distance error (left picture) and yaw angle error (right picture) with $0.95V_{cr}$ and 0.3μ. ..... 69
*Fig- 32* Different types of the car trajectory behavior depending on the prediction time. From (1) to (3) this duration becomes longer ..... 70
*Fig- 33* Target quality function convergence of the PPD controllers under the different speed levels and friction values (μ=0.1, 0.3, 0.5) depends on time prediction. ..... 71
*Fig- 34* Rotation curve. The transition between straight part (blue) and circular part (green) is spiral curve (red). Picture is taken from "https://cifrasyteclas.com/clotoide-la-curva-que-vela-por-tu-seguridad-en-carreteras-y-ferrocarriles/ ..... 73
*Fig- 35* Trajectory of the first left and right turns combined with clothoid spiral. Both turns demonstrate gradually increasing turning radius. ..... 74
*Fig- 36-1* Fitness convergence characteristics in the process of evolution for more 20 independent runs of GP evolving the SAF of GP-RM controller for friction coefficient μ=0.3 , $0.85V_{cr}$(a) ..... 81
*Fig- 36-2* Fitness convergence characteristics in the process of evolution for more 20 independent runs of GP evolving the SAF of GP-RM controller for friction coefficient $\mu$=0.5, $0.85V_{cr}$ (b) ..... 82
*Fig- 36-3* Fitness convergence characteristics in the process of evolution for more 20 independent runs of GP evolving the SAF of GP-RM controller for friction coefficients μ=0.3 (a), μ=0.5 (b), μ=0.8 (c) respectively. The bold curves correspond to the average, minimum and maximum values in each generation. The best fitness

of the PD (obtained via a complete enumeration of the values of their parameters) is shown as red horizontal line. 82
*Fig- 37* The dynamics of the deviation from the center of the lane on the snowy ($\mu$ = 0.5) – road. 87
*Fig- 38* The dynamics of the steering angle on the icy (μ = 0.5) – road. 88
*Fig- 39* Fitness convergence characteristics of 20 independent runs of GP evolving optimal steering function for friction coefficient $\mu$=0.3 (a), $\mu$=0.4 (b), $\mu$=0.5 (c), $\mu$=0.6 (d), $\mu$=0.8 (e) and $\mu$=1.0 (f), respectively. The best fitness of the servo control model, obtained via a brute-force search is shown as a horizontal red line. 92
*Fig- 40* Comparative performance of the most general SAF, evolved for friction coefficient μ=0.5 when tested on tracks with different friction coefficients 94
*Fig- 41* The dynamics of the steering angle (top) and the deviation from the center of the lane (middle) and Θ angle – deviation from the desired trajectory angle(bottom) of the car steered by the most general SAF on the track with friction coefficient μ= 0.5 96
*Fig- 42* Steering angle (left) and deviation from the center of the lane (right) of the understeering car controlled by SAF of a tuned PD controller without- (dashed line) and with an artificially introduced oscillation (solid line). 103
*Fig- 43* Slip angle of a turning wheel α (left), The dynamics of the friction coefficient as a function of the slip angle of the tires (right) 104
*Fig- 44* Turning of a well-controllable (left) and understeering (right) car. The different (Ackermann) steering angles $\delta_L$ and $\delta_R$ of the understeering car (right) result in different slip angles $\alpha_L$ and $\alpha_R$ of the tires, which, in turn, yields an asymmetric braking 105

# List of Tables

*Table- 1:* The steering parameters ..... 16
*Table- 2:* Description of the available commands to simulated car ..... 22
*Table- 3:* The feature of simulated car ..... 24
*Table- 4:* Description of the car sensors ..... 24
*Table- 5:* Main features of the test track ..... 26
*Table- 6 :* Speed of the car during the fitness trial on the test track with different friction coefficient ..... 28
*Table- 7* Main parameters of GP applied for evolving SAF ..... 38
*Table- 8* Experimental result for constructed SAF ..... 51
*Table- 9* Track segment parameters information ..... 60
*Table- 10* Time needed to return to the middle of the lane from deviation, sec ..... 66
*Table- 11* Distance to the obstacle for each parameters combination, m ..... 69
*Table- 12* Target quality function of steering controllers for each parameters combination ..... 75
*Table- 13* Target quality function of steering controllers *F* for each parameters combination split by addends corresponding to deviation from the center of the lane and lateral acceleration respectively. ..... 76
*Table- 14* SAF produced by GP and PD controller's SAF quality estimation function values *F* ..... 83
*Table- 15* Time needed to return to the middle of the lane from deviation, sec ..... 84
*Table- 16* Distance between the edge of the car and the obstacle ..... 85
*Table- 17* Experimental Results on Evolution of SAF ..... 93
*Table- 18* Experimental results on evolution of SAF with penalty for steering oscillations. Obtained from 32 evolutionary runs of GP ..... 102

# Chapter 1

# Introduction

## 1.1. Background and Motivation

For the development of various branches of science and economics, as well as for ensuring the safety of citizens, the solution of transport problems in conditions not suitable for humans is of particular importance. A special role and perspective in solving this problem has autonomous controlled vehicles.

Autonomous vehicle - a vehicle equipped with an automatic control system that can make decisions about movement without the direct participation of a human driver in this process. Self-driving cars have many advantages and benefits that become even more pronounced when a vehicle of this type interacts with itself on the road. However, when interacting with a human driver, it has several advantages: (i) the ability to transport goods in hazardous areas, during natural and technological disasters, in a dynamically changing environment; (ii) more efficient use of road capacity due to centralized traffic flow management, which also leads to savings in time, financial resources, as well as fuel consumption; (iii) expanding the possibilities of using the car for people with disabilities; (iv) minimization of road accidents and the number of human victims in them due to the exclusion of cases when they were caused by factors that affect a person but are not affected by the computer – in a view of the often changing conditions, the work of drivers requires a large number of control actions and a constant concentration of attention, which contributes to rapid fatigue and cognitive overload. According to the researches [1] even qualified drivers with long experience after 4-5 hours of continuous driving, fatigue reduces attention and the number of control errors increases 1.5-2 times, which increases the likelihood of small and medium traffic accidents. Beside that, the use of self-driving cars also eliminates such causes of accidents as stress or so-called startle effect[2][3] which is one of the most famous, common and well-studied[4][5][6] reasons of the crashes, alcohol and the lack of necessary trained and extreme

driving skills , which are especially necessary in the conditions of the wet road, which will be discussed further in this work.

Active development of autonomous vehicles by leading automakers began in the 80s of the 20th century. The objects of research in this area are passengers cars, trucks, agricultural machinery, rescue equipment, as well as various factory and production vehicles.

All modern car manufacturers compete in the development of the autonomous vehicles, the successes of industry giants are especially noticeable. However, in view of the fact that research works themselves are often protected from disclosure of trade secrets, research results are hardly published in the open access. It is also affected by the fact that innovative technical solutions, software and hardware systems for autonomous vehicles in many countries are classified as dual-use products.

The efficiency and safety of passengers of any self-driving machine depends on the characteristics and specific implementation of its control model. Choosing a behavior model, calculating and simulating all the necessary dynamics of processes, including such as the necessary clutching force, the vehicle's speed or the angle by which the steering wheel must be turned to ensure the stability of the vehicle's position on the track, is all the work of the controlling model. In addition, a good realization of the advantages of self-driving cars cannot be achieved without taking into account various limitations of such parameters as the speed and delay of sending commands to the mechanical parts of the car, the accuracy of measuring and transmitting data from sensors, and others.

The work is dedicated to solving the scientific problem associated with ensuring the safety of self-driving cars.

## 1.2. Objectives

Currently, control models can be divided into two categories. The first in its calculations relies on data received by it from external auxiliary infrastructure - sensors built into the road, Internet data on weather or road loads, and others. The second category includes more modern and independent control models. They make all decisions on their own, relying solely on the data on the external environment that it receives from cameras, sensors, radars and satellite navigation systems that are installed in the car itself. Improving this approach in the future involves the development of integrated automation of vehicle systems. Naturally, in their calculations they (both approaches) do not rely on the knowledge and skills of a person, except for some cases that vary in each model when control is completely transferred to the human driver. It should be noted that the obvious drawback of this (second) approach is the fact that the quantity and quality of the data obtained in this way can vary greatly depending on the time of day, weather conditions and the case (visibility, for example, may fall at night, in rain or fog, or if the car is a short distance from the truck, which will obscure the cameras). Minimizing the required set of sensors is one of the most interesting research topics in this field. Today, the symbiosis strategy of driving a car with a human driver is also popular with the transfer control to the controlling function in special modes - skidding, parking, driving on a highway or in traffic. Some of the mentioned modes, such as automatic parallel parking, are now available from many of the leading manufacturers that we mentioned, while others, like navigating while driving on the motorway, are still in a research or testing state and are less common. Probably the most impressive results in the field of creating a self-driving car were achieved by RoboSV and Google, thanks to the complexity of their approach.

Analysis of the data available today demonstrates the presence of certain problems that stand in the way of a large number of studies at the stage of development and determination of the requirements for a control model. Among these problems, one can list the following -

significant calculation errors associated with a high error and delay in interpreted data received from external devices through third-party devices and a lack of resources necessary for the continuous receipt, storage, processing and accounting of constantly and very quickly changing data in sufficient quantity and with sufficient reactivity. Given that these parameters are directly related and ensure the adequacy of the control model, the main goal in which the safety of passengers and traffic motion, the development of a model that would not require attracting a large amount of resources and at the same time would not suffer functionally, is relevant.

Thus, the main objectives of this work is to ensure the safety of an autonomous vehicle in a given external environment - on a slippery road - by developing a control model that is able to compensate for the existing drawbacks of data collection technologies.

## 1.3. Related researches

In the process of testing and analyzing the resulting models, we came across some issues that led to additional research. Since these studies are not directly related to the stated research topic, we consider it appropriate to take them outside the chapters devoted entirely to the development of control models, but we consider it necessary to bring them in this work in a separate chapter inside Chapter 4 – in Discussion – Oscillation analysis and validation.

## 1.4. Thesis Contributions

Based on the results of studies for practical use in the development of self-driving transport control systems, several high-security vehicle control models were developed and implemented. To confirm their effectiveness and safety, these models were tested in a well-known and used simulator that allows you to simulate the dynamics of a self-driving car and evaluate the impact of control delay on traffic safety in various road conditions. The

demonstrated results made it possible to recognize the declared safety benefits for the operation of self-driving cars for these models. Also, some of the results of the thesis can be used for educational purposes extreme driving training. Chapter 5 analyzed non-native tactics of driving on slippery roads, which showed the high efficiency and theoretically can be applied by the human driver after appropriate training to achieve the greatest passenger safety and vehicle stability. The results were also published at conferences and in journals.

The list of publications is available in appendix Publications and Conference Papers

## 1.5. Thesis Overview

### 1.5.1 Research Framework (Chapter-2)

Chapter 2 is entirely devoted to the formulation of the research problem and is divided into several parts. The first part is a description of the presentation of the mathematical model of a self-driving car - a system of its environmental interaction through data collection and reaction corrections of its condition. This description leads us to the adoption and brief description of a common type of controller for a car model. The second part is a description of the presentation of the environment provided by the widely used simulator. The third part is devoted to the exchange of data between the car model and the simulation of the environment using a special software application.

### 1.5.2 Upgrading SAF for PID and PD controllers with tuning parameters (Chapter-3)

Chapter 3 tells about the first step of our research, in which we began to improve the existing classical management model. The improvement was in the analysis and selection of the best parameters of the existing model. Parameter tuning itself was carried out in two ways

– brute force the values in a reasonable range and using genetic programming, which instead of constant values picked up some composition of variables.

### 1.5.3 Upgrading SAF for PID and PD controllers with prediction (Chapter-4)

The second step of the study was the upgrade of the current method by introducing into it the components of predicting the position of the car in the future on the road based on an analysis of the behaviour of a professional human driver. This chapter presents the corresponding tests of the new method and comparing its results with the base method. Based on the collected data, individual characteristics of the obtained method were identified and analysed, which became the key to a safer driving style.

### 1.5.4 Relaxed structure of PD controller analytical model, developed heuristically via the GP – RM-GP (Chapter-5)

This Chapter 5 is devoted to the next step of our research, which was to develop a new, more complex model based on the classical one. In this section, it was not the model components that were complicated, but its structure. Having made the assumption that in difficult conditions for driving a slippery road, the classic steering model is not flexible enough, we applied genetic programming and constructed a new formula for controlling the steering angle from components of the classic formula.

### 1.5.5 Evaluation of the GP-RMEP controller with extended parameters obtained via GP (Chapter-6)

The logical continuation of the thought described in Chapter 5, we considered an increase in the number of components in the equation that controls the rotation of the steering wheel. A tool for modeling equations was also genetic programming. The best of the results

obtained are presented in the work, both for specific environmental conditions, and recognized as universal by testing on simulations.

### 1.5.6 Summary, Conclusion, and Future Work (Chapter-7)

Chapter7 is the last chapter of this thesis. This chapter present the summary of our study, some of our achievement and conclusion, and finally ends by discussing possible direction of this research.

# Chapter 2

# Research Framework

## 2.1. Car Model of the Autonomous Car

Autonomous vehicles produced by different manufacturer have already traveled many hundreds of kilometers, and the creators of this technology claim that it will help reduce the number of congestion on roads, increase traffic capacity and, importantly, make traffic safer without driver intervention. The grounds for such allegations are that, according to statistics, about 90% of all traffic accidents occur with the participation of a person.

### 2.1.1. Real-time Feedback Control Mechanism

The mathematical model of a car is a subject, on the one hand, equipped with a set of sensors that allow receiving data from the environment, and on the other hand, having a certain control center or controller inside, responsible for data analysis and decision making. This reactive model interacts with the simulator during the race, which will be discussed in Chapter 2.2. Here we will describe some important elements of internal control - acceleration, brake and steering wheel control, which are used to move the car along some desired trajectory. It consists of two main modules: (i) the steering module and (ii) the proportional (P) module of the vehicle speed controller.

#### 2.1.1.1 Acceleration and brake control of the car keeping desired speed

Driving autonomous car while moving along a trajectory even on a dry road should combine flexibility and smoothness of correspondence between controlling the steering angle with acceleration and braking to maintain movement at the target speed. Especially, if a moving or stationary obstacle appears along the route, the control system must quickly make a decision and safely stop the vehicle on time, avoiding collisions with the obstacle and not aggravating

the accident situation on the road for other cars, or induce extreme driving (for example, controlled drift), which will provide a safe trajectory of movement.

To control the speed of movement, a PID controller was considered. However, in practice, PID controller does not give a significant advantage compared to the simple proportional controller, since the vehicle has some inertia when speeding and braking (for derivative component) from the one hand, and could change speed very quickly (for integral) from the other. Also, P-Only control can be ideal when applied to the inner loop of so-called cascaded control loop architecture. Cascade control is a system involves the use of two controllers with the output of the first controller providing the set point for the second controller, the feedback loop for one controller nestling inside the other (In our case we have for example acceleration controller which use the output of the speed controller). Such a system can give a improved response to disturbances [7][8][9]. Thus, to control the speed, the following proportional controller was chosen:

$$u = K_p(v_{trg} - v) \tag{1}$$

Where $v$ is current speed, $K_p$ – scaling proportional coefficient and $v_{trg}$ – is target desired speed.

Setting the controller coefficients is, in the general case, a rather non-trivial task due to the large number of degrees of freedom of the system. In recent years, the development of computer technology and methods of numerical optimization has led to the fact that finding the parameters of the regulators has been greatly simplified. In our case, the selected model of the simulated car influenced the selection of parameters - this is a car of the Mercedes SLK MD brand. Knowing its technical description and having configuration files of settings, we took the corresponding values of many parameters, including constants, which are responsible for the transmission and processing speeds of signals. We chose this car model

because of its increased stability - for example, its center of gravity is only a quarter meter from the ground compared to more than half a meter in ordinary car, lightweight design and the ability to test the car at height speeds and prпy engine loads that are unreal for ordinary cars, but convenient for different experiments and model testing.

#### 2.1.1.2 Steering angle control

As mentioned earlier, we selected a rear-wheel drive vehicle for research. The design of the rear-wheel drive is that the engine is located in front, but longitudinally to the length of the car (Figure 1). He drives his torque to the rear wheels through a long driveshaft. Among its features, attention should be paid to the following:

- High performance is characteristic of the rear-wheel drive, since the probability of slipping of the rear wheels is very small, due to the fact that during acceleration, inertia transfers a significant part of the mass to the drive wheels, i.e. back.

- In rear-wheel drive cars, due to the distribution of heavy mechanical components along the entire length of the car, the weight is distributed evenly, which gives better handling. Also, rear-wheel drive allows splitting the responsibilities of the wheels for driving and turning between the front and rear, whiling in the front - wheel drive these functions are performed by one set of wheels.

- On a slippery road, the car on the rear wheel drive is easier to drive, but it gets into it gets into a skid (oversteer) also easier than front-wheel drive cars. To stop the skid, driver just need to release the gas pedal.

- The simplicity of the rear-wheel drive design allows the steering system to turn the front wheels to a wider angle, because the front wheels are not leading. This reduces the turning radius of the car.

Another advantage of rear-drive cars is that the engineering solutions allows the drive shaft to transfer more torque (than front wheel drive). Therefore, this configuration is often is used in heavier- or more powerful cars. The skid of rear-wheel drive is usually an oversteer, because the rear wheels are overloaded with (i) longitudinal grip requirements (traction) and (ii) lateral grip requirements (turning). The vector sum of these two components on the rear wheels may exceed the available grip, and it would happen earlier than that of the front wheels as the latter are not subjected to the longitudinal (traction) component. The skid of front-wheel drive is usually understeer, because the front wheels are overloaded with (i) longitudinal grip requirements (traction) and (ii) lateral grip requirements (turning). The vector sum of these two components on the front wheels may exceed the available grip, and it would happen earlier than that of the rear wheels (as they don't have the longitudinal traction component).

Therefore, the question is: *Which kind of instability – understeer or oversteer – is easier for the driver to counter?* There is an opinion that understeer (pertinent to the front wheel drive) is easier – just reducing the speed would be enough. However, the understeer increases the drag forces, and it naturally results in reducing the speed. Also, if the car hits an obstacle during understeer, it happens with the front of the car – which is well protected (engine room as a buffer, airbags, etc.). In case of oversteer in rear-drive cars, not just lifting the accelerator, but also an *appropriate steering response* is needed to counter for the spinning, oversteering car (in understeer, the car does not spin, it just goes straight). If the car hits an obstacle during understeer, it may happen with the side or even rear of the car – which are not as well-protected as the front. Therefore, despite the engineering- and traction-related advantages of the rear-drive car (as you explained well in the thesis), the car featuring a rear-wheel drive is *more challenging* to drive on slippery roads, as it requires both a swift- and

precise response form the driver which is not always possible – due to factors such as lack of expertise of the driver, stress, startle effect, etc..

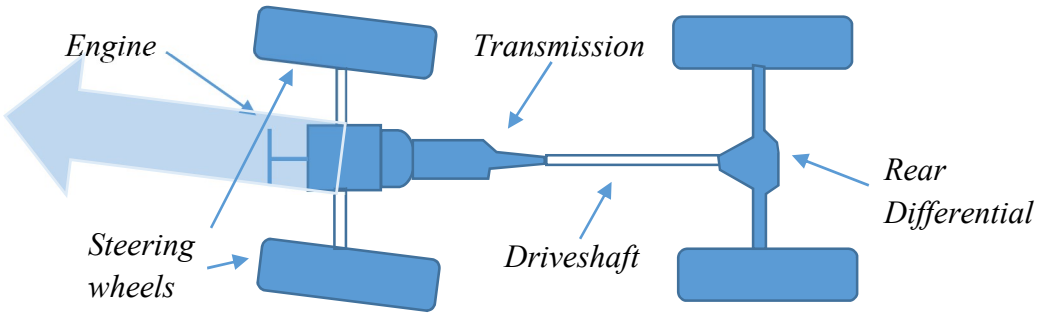



***Fig- 1*** Rear-driving car schema

The steering pole of such a machine (the projection of the kinematic center of rotation on the longitudinal axis) is located on its rear axle, between the non-rotatable rear wheels (Figure 2, point *P*). The distribution of rotation angles for the first axis is defined as follows. It is assumed that the angle of rotation of one of the wheels of the front axle, for example, right wheel $\delta_{right}$ (see Figure2), is determined by the result of the calculation of the controller. This value also takes into account all delays and restrictions of various kinds, which will be discussed later. Then the angle of the left wheel is calculated by the equation 2.

$$tg(\delta_{left}) = \frac{L}{D+\frac{Base}{2}} \tag{2}$$

Where *Base* is the wheel track, *L* is the wheel base, and *D* is distance between the instantaneous rotation center O and the axis of the rear axle of the car, in other words, the radius of rotation.

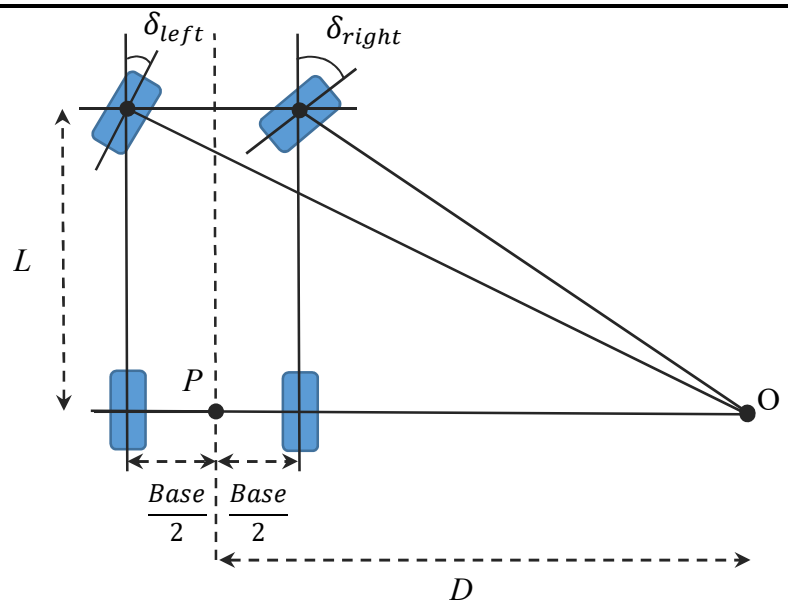



***Fig- 2*** Kinematics of turning the wheels of the front axle

The following trigonometric dependencies are visible from the Figure 2:

$$tg(\delta_{right}) = \frac{L}{D-\frac{Base}{2}} \tag{3}$$

That means that distance can be calculated as follows:

$$D = \frac{L}{tg(\delta_{righ})} + \frac{Base}{2} \tag{4}$$

And accordingly, the value of the left wheel rotation angle could be expressed as follows:

$$tg(\delta_{left}) = \frac{L}{\frac{L}{tg(\delta_{right})}+Base} = \frac{L*tg(\delta_{right})}{L+Base*tg(\delta_{right})} \tag{5}$$

The steering wheel controller which calculates the steering angle $\delta_{right}$ value consists of several stages of information processing. The first stage is angle value calculating itself. It depends on the specific analytical equation that was adopted in this model. In our work, we tested different models derived analytically and heuristically in Chapters 3-6. The most safe

and fast steering controller model was the goal of our research and its expression was changed during the analysis and testing of the autonomous vehicle. The obtained value of the turning angle undergoes changes at the second stage of imposing physical and mechanical constraints.

#### 2.1.1.2.1 Physical and mechanical constrains. Maximum lock angle

The maximum angle of lock (Figure.3, part 3) is the angle which the wheels adopt at the full steering angle (left and right) relative to the axis of symmetry (Figure 3, part 2). This determines the vehicle's turning circle radius. The maximum angle of lock is measured with the relevant internal cornering wheel fully locked (Figure 3, part 3). Depending upon the steering trapezium, the external cornering wheel (Figure 3, part 1) must have the same negative difference, within a specified tolerance.

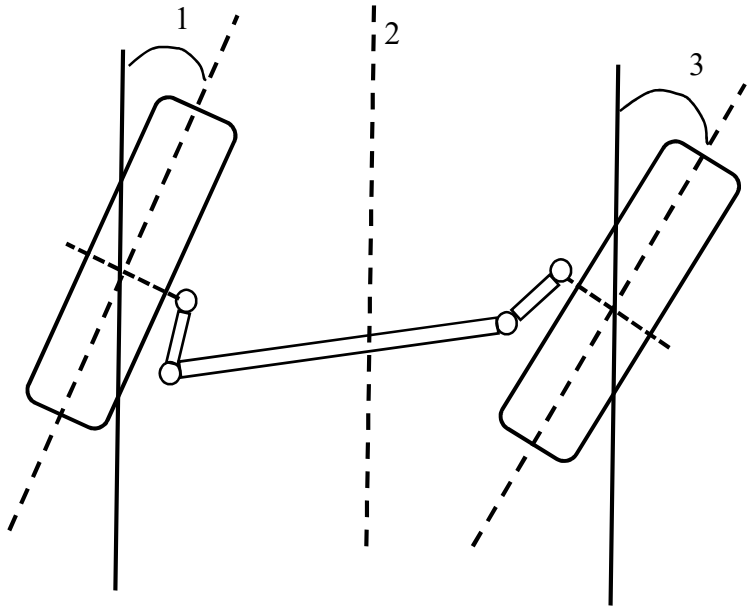



***Fig- 3*** Maximum steering angle

For example shown in the picture, we could take following parameters:

***Table- 1:*** The steering parameters

| | |
|---|---|
| Maximum angle of lock, right (2) | 34° |
| Angle of lock, left (3) | 32° ± 1° |
| Difference (1) | 2° |
| Tolerance | ± 1° |

Tested car has rear wheel drive with the maximum steering angle 35° and for the wheel difference responsible the Ackerman effect.

### 2.1.1.2.2 Mechanical constrains (features). Ackerman principle

Rudolf Ackerman is the person who developed the steering system for horse drawn carriages, and we use his name to describe the angle of the inner wheel relative to the outer wheel, when the wheels turn all the way to the farthest left or right. Typically, when the front wheels turn all the way, the inner wheel has a larger angle of rotation than the outer wheel.

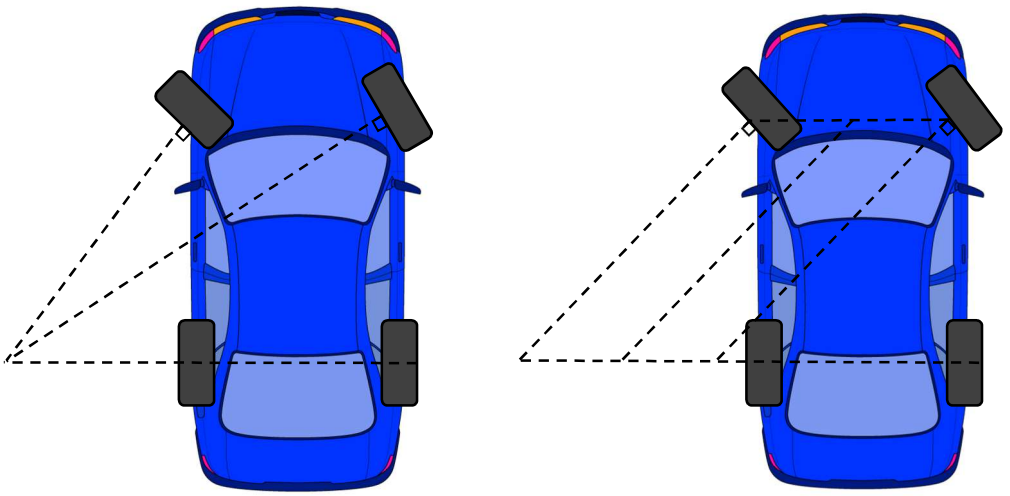


***Fig- 4*** Ackerman wheel position (left) and parallel wheel position (right)

If you extend the center line of each wheel to the point where they intersect and measure this angle, this will be the angle of Ackerman. Ideally, for perfect control, the Ackerman angle should intersect on the center line of the rear axle.

In wide corners, the front wheels do not turn very right or left, the inner wheel does not turn at a steeper angle than the outer wheel, and the Ackerman angle is not very wide. In tight corners, the inner wheel turns the baud at a steeper angle than the outer wheel, and this is called the Ackerman effect. The steering system with levers is an approximate way to obtain the Ackermann effect, and it is sufficient for car models due to wheel slippage, deflection of the side walls of tires and other factors. Ackerman's small angle (carried out by lengthening the Ackerman traction or by using external mounting holes) will provide a more aggressive corner entry with the possibility of oversteer in the middle of the turn when most of the weight is on the outside wheels. A larger Ackerman angle (by shortening Ackerman traction or using internal mounting holes) will provide more predictable and smooth control.

#### 2.1.1.2.3 Mechanical constrains. Delay

Besides steering angle changing obtained as a result of controller model calculations and further mechanical limitations and also influence of the vehicle's design and functional modifications, another factor introduces significant corrections to the value of the steering angle – it is an automatic delay of signal transmission. This is delay of a command from the computing module to the steering wheels itself. For our research, we chose the value of this parameter close to the real in 100 ms [10][11].In our further research, we must consider that in a feedback control system, any delay can lead to a potentially unstable or oscillating system  according Nyquist-Mikhailov criterion. This should lead to the fact that in the early

stages of testing controllers there will be problems similar to those faced by radio-controlled models of cars, namely oscillations.

### 2.1.2. Environment

The environment of the car is the outside world with road, obstacles, weather conditions and etc. Information about the world control model receives through sensors.

### 2.1.3. Perception

Since our experiments do not imply a human driver, only sensors are responsible for the perception of information about the environment. On the basis of the data provided by them all calculations of the computing module are performed. This data is the position of the car on the road relative to the center of the lane and edges and also potential obstacles (GPS positioning), the data themselves about the obstacle and its location (camera), weather conditions (coefficient of friction) and others. In the conditions of arrival, this data is constantly updated and the speed of their transmission is one of the safety parameters. As mentioned above, one of the goals of our study was to reduce the number of sensors necessary for safe driving. In particular, in the methods of Chapters 5-6, we do not use data from the camera, which due to various circumstances (fog, turn, the truck goes closely ahead) can be deactivated, thereby increasing the danger to the passenger.

### 2.1.4. Reaction

The reaction involves the mechanism of the environmental impact of the control model. Based on the data obtained, after appropriate calculations for the specific control model and adoption of corrections, the initiatives recognized by the model as needed are carried out. Such initiatives could be, for example, acceleration, braking, turning and others. These

actions affect the environment, which captures by the sensors. Thus, a feedback system is obtained.

## 2.2. Testing and Simulation Software Environment

We used a software simulator called The Open Racing Car Simulator (TORCS)[12][13] for collecting test data and validating our models. This simulator is traditionally used for competitive races of cars controlled by an AI with cars with manual control from the keyboard. In addition, it allows to replace manual control with a self-developed controller.

- **Credits**

TORCS is developed by Eric Espié, Christophe Guionneau and other contributors. The official TORCS site is http://torcs.sourceforge.net/ . The source code of TORCS is licensed under the GPL ("Open Source").

- **Platforms**

TORCS is a highly portable multiplatform car racing simulation. Supprted platforms are Linux - all architectures, 32 and 64 bit, little and big endian, FreeBSD, OpenSolaris and Windows 98/XP/2000 (32 and 64 bit).

- **Features**
  - Opportunity to choose from 42 different cars. Their models described by configuration files, that you could update or change by your decision.
  - 30 tracks also with flexible configuration,
  - More than 50 implemented controllers to race against.
  - Height level of the realism of simulation of car dynamics including quite simple damage model, collisions, tire and wheel properties (springs, dampers, stiffness, etc.), aerodynamics (ground effect, spoilers, etc.) and other.

### 2.2.1. Simulation architecture

TORCS is a separate open-source independent application that allows to simulate various conditions for cars in racing or single player mode. This application also allows you to implement your logic for driving cars by recording in special separate sections of the program code. These sections of code are called bots. They are compiled as separate modules that are loaded into main memory during a race. This application architecture delimits the simulation data of the environment, the car and the car controller, which allows you to add new control models to the simulation without violating the rules and laws of the simulation itself.

In fact, there is no separation between bots and the modeling engine. This means that bots that implement various control models can theoretically have full access to all data structures that determine the route and the current race status, and, as a result, each bot can use different information for its driving strategy. However, thanks to the mentioned modularity, using the interface of each bot, we can easily select, limit and dose the data that each model receives. That is, we can provide the model with the opportunity to analyze the complete state of the race (for example, the structure of the track, the position of obstacles, speed, etc.), plan our actions or give a minimal idea of them, in order to understand how significant this or that parameter is for our model.

Another feature of the TORCS architecture is software that emulates the exchange of data between the environment and the car. This is achieved by structuring TORCS as client-server applications: bots are launched as external processes connected to the race server via UDP (User Datagram Protocol) connections. In addition, this contributes to the appearance of a delay in real-time data transmission. Each tick of the game corresponds to approximately 20 ms of simulated time, and each tick server sends data from the sensors to the bots that control the machines. If within 10 ms the server does not receive control responses from the bot through the client (both because of unreliability of UDP or any client problem), then it is

forced to continue to consider its previous reaction relevant. Due to this, the desired level of abstraction is achieved, at which the bot (driver) and server codes do not know anything about each other and only exchange data, according to the specified protocol. The software architecture is shown in Figure 5. The code of each bot, like the simulation, is located on the server, but accesses the simulation through the client. Accordingly, each bot on the server listens on the port of the race server. Also, each server bot associates itself with a specific client with which it establishes a connection. Then, when the race begins, each server bot sends the current sensory information to its client and waits for action until 10 ms has passed (in real time). Each game tick corresponding to 20 ms of simulated time, the server updates the race status, which is sent back to clients.

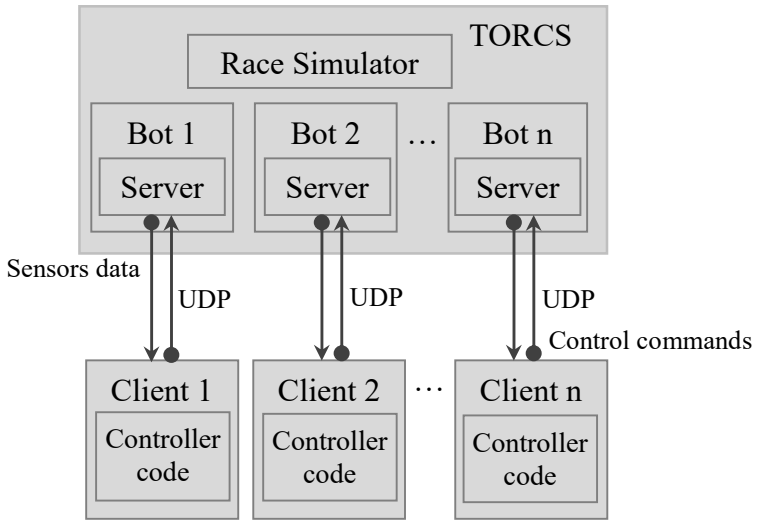



***Fig- 5*** The architecture of the competition software.

Sensors data from simulation contains a lot of data about the current car outside and inside state to controller's analysis. Regarding the first point, it is car position on the road in Cartesian coordinate system, the angle of deviation of the direction of the car from the axis of the road, the time and the distance traveled from the start of the race and the distance from the start point. Sensors each tick measure the distance from the car to the edge of the road in five directions and the speed of the car in three dimensions, etc. Regarding the second point,

the sensor data also contain the state of the vehicle itself. Among this data are the spinning velocity of each wheel, fuel supply, the degree of car damage, the rpm, and others.

The control commands that are transmitted to the bot to execute them in the car belong to a rather typical and limited set. This set includes commands to turn the steering wheel, to control the gas and brake pedals and the gearbox. There are also additional meta-commands related to managing the simulation, such as a request to restart a race, which are sent in the event of a race conclusion or a car crash. Table 2 shows this commands and their description.

***Table- 2:*** Description of the available commands to simulated car

| **Name** | **Description; Range** |
|---|---|
| Focus, deg | Focus direction to measure the distance between car and the road edge; *[-90, 90]* |
| accel | Virtual gas pedal (0 means no gas, 1 full gas); *[0, 1]* |
| brake | Virtual brake pedal (0 means no brake, 1 full brake); *[0, 1]* |
| clutch | Virtual clutch pedal (0 means no clutch, 1 full clutch); *[0, 1]* |
| gear | Gear value. (-1 is reverse, 0 is neutral and the gear from 1 to 6); *{-1,0, ..,6}* |
| steering | Steering value: -1 and +1 means respectively full right and left, that corresponds to an angle of 0.366519 rad; *[-1, 1]* |
| meta | This is meta-control command: 0 do nothing, 1 ask competition server to restart the race; *{0, 1}* |

An important feature of this simulator for us is that the races are not held in real time, since the execution of bots in some of the approaches that we have tried may take several years in search of some way to go along the track. When the graphics mode is turned off, TORCS can process the data of one race, which lasts two minutes in about one second. Thus, the data

collected for the number of races that take a year to view can be collected and processed in just a few hours.

Among the limitations of the simulator, one can mention the following - since bots must be compiled as a loadable module of the main TORCS application written in C ++, it is most convenient to implement models in C ++.

### 2.2.2. Simulation parameters

In this section we provide data on the parameters of the environment and simulated transport, which have the greatest impact on the race process and, accordingly, which are of the greatest interest to us. We chose a car from among the sports cars in order to be able to freely experiment with its speeds on the one hand and, on the other hand, to test a wider range of control models on a car with increased stability. In contrast to this, we chose the track of increased complexity.

#### 2.2.2.1. The Car

As stated earlier, TORCS has a built-in extensive collection of car models, whose parameters and behaviour on the track, tied to physics and dynamics, it emulates with adequate accuracy[14][15]. In addition, a mechanism for adding descriptions of new cars by users was implemented in TORCS.


***Fig- 6*** Simulated car in TORCS lateral side view

In all our experiments to avoid collisions, we used the same CLK DTM sportive rear-driving car model. Its appearance is shown in Figure 6.

The main parameters of this model are presented in the Table 3 below.

***Table- 3:*** The feature of simulated car

| **Feature** | **Value** |
|---|---|
| Model | CLK DTM |
| Mass, kg | 1050 |
| Length, m | 4.76 |
| Width, m | 1.96 |
| Height, m | 1.17 |
| Front and rear weight repartition | 0.5 and 0.5 |
| Height of centre of gravity, m | 0.25 |
| Coefficient of friction of tires | 1.0 |
| Drivetrain | Front engine, rear wheels drive |

Among the previously mentioned in the description of the simulation of car sensors, we would like to highlight the following (Table 4):

***Table- 4-1:*** Description of the car sensors

| **Name** | **Description, Range** |
|---|---|
| distFromStart | Distance between the car and the start line along the track line |
| distRaced | Distance between the car and the start line |
| fuel | Fuel supply; |
| gear | Car gear position,<br>• -1, if reverse motion,<br>• 0, if neutral,<br>• 1 to 6, other. |
| RPM | Car engine parameter – rotation per minute, in numbers; |
| speed_X | Car speed along the X axis, in km per hour; |
| speed_Y | Car speed along the Y axis, in km per hour; |
| speed_Z | Car speed along the Z axis, in km per hour; |
| $\theta$ angle | Angle between the car wheel direction and the centre of the track axis, in rads $[-\pi,+\pi]$ |
| Level damage | The total damage received by the car since the start of the race as a result of collisions with other cars or the roadborders. If the manual chosen limit is exceeded, it counts as a car crash, in points; |
| curLapTime | The stopwatch time for the current lap, in sec; |

***Table-4-2:*** Description of the car sensors

| | |
|---|---|
| lastLapTime | The stopwatch time for the last completed lap, in sec; |
| racePos | Position in the race with respect to other cars – In our case it is forever equal 1, in numbers; |
| trackPos | The distance between the car center of mass and the axis of the track. It is normalized relative to the track width and could take the following values:<br>• 0, if the center of the car is on the axis,<br>• -1, if the center of the car is on the right edge of the track,<br>• +1,if it is on the left edge,<br>> 1 or <-1, if the car flew off the track, in numbers; |
| wheelSpinVel | • 4 sensors translating the rotation speed of the wheels, in rads per sec; |

### 2.2.2.2. Test Track

Tracks in TORCS are available as a ready-made collection, it is easy to find a description for each track and make changes to it, for example, about the quality of the coating, incline level or other physical properties. It is also quite simple to create a new track for your test case by adding a file with a description of it into the special folder "../runtime/tracks/surfaces.xml". Track is defined as a static object in simulator's environment. The track consists from tree type patterns: "Straight" segment, "Right Curve" segment and "Left Curve" segment. The layout of the track we testing our car on is a variant of the so-called "fish hook" [16]. It consists of two sectors: Sector 1, including a short straight followed by a left turn, and Sector 2 including a single, long 180° right turn. The track is illustrated in Figure 7, and its main characteristics are given in Table 5. A feature of this two sequential turns is quick and sharply changing the track curvature, forming an area in which the car is very likely to go in skid (firstly understeering which will cause the oversteering in the Sector 2). We have exacerbated this instability of the road even harder with low traction along the entire route. In our experiments, the friction coefficient was in the range from 0.1 to 0.8. The coefficient of friction $\mu$ between the tires of the car and the surface of the considered track could be set in

TORCS to value corresponding to rainy ([0.8 - 0.5]), snowy (0.5 - 0.3) or icy (0.3 - 0.1) conditions.

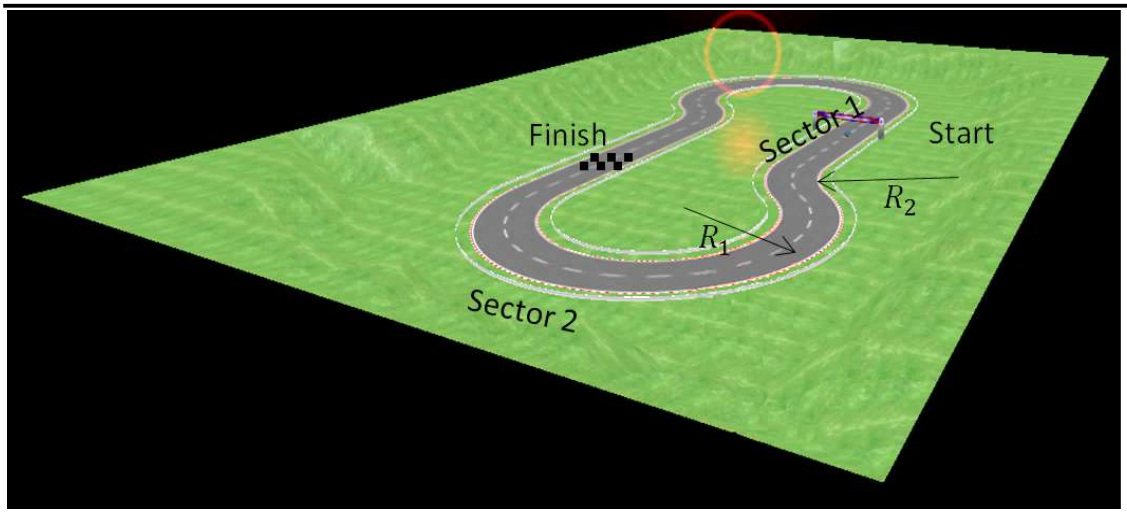



***Fig- 7*** Hook-type test track

Such a difficult track was chosen by us for two reasons. The first is for adequate testing of the control model of the car behavior, and the second is for training some control models that we obtained as a result of genetic programming (Chapters 5- 6).

***Table- 5:*** Main features of the test track

| **Name** | **Value, m** |
|---|---|
| Total length | 300 |
| Length of sector 1 | 90 |
| Length of sector 2 | 210 |
| Radius of turn 1, $R_1$ | 50 |
| Radius of turn 2, $R_2$ | 50 |
| Lane width | 20 |
| Surface on the track | asphalt |
| Surface out of the track | grass |
| Barrier height | 1.0 |
| Barrier width | 0.1 |

### 2.2.2.3. Automation test runs

Using an autonomous control system has many advantages in the field of vehicle testing. In addition to significantly improving the accuracy of the test and safety, allowing to manage the autonomous control system of the tested vehicles, this makes it possible to use completely new tests that would be impossible with the human driver because of their obvious danger or inapplicability. Using an autonomous vehicle control system and a virtual testing environment, it is possible to perform many types of dynamic tests, including rollovers and collisions, without endangering a human driver. These types of tests can reproduce real life situations, for example, when a car moves at high speed on a slippery road in low visibility conditions and runs the risk of colliding with other participants in the movement or flying off the track. Excluding a driver from a vehicle control system is beneficial for several more reasons. Firstly, the autonomous control system is not subject to fatigue and can function around the clock without accumulating cognitive overload and errors, and also allows for repeatability of parameters when testing a car. The ability to reproduce experimental conditions allows us to obtain more accurate indicators during endurance testing on cobblestone tracks, dirt paths, etc.

The use of an autonomous vehicle control system also makes it possible to conduct tests continuously, which can significantly reduce the total test time. Secondly, this eliminates the specific negative impact on the health of the human driver. For example, when testing on a cobblestone track, drivers are exposed to prolonged harmful effects of vibration and shaking. Thirdly, in the future, the autopilot system can adopt the experience of a better driving style, which can be easily translated to a larger number of vehicles, thereby increasing the overall traffic safety.

### 2.2.2.4. Race target and critical speed

Working with the simulation of a racing car, we did not limit its speed artificially by the rules of use at 20 - 30 km per h as usual [17][18]. In fact, we allowed the car to use its speed to learn extreme maneuvers (for example, at high-speed sharp turns), counting only the critical speed of passage of the turn through centrifugal force,

$$F_c = \frac{mV_{cr}^2}{R} \tag{2}$$

Where $m$ is the mass of the car, R is the radius of the turn and $V_{cr}$ is a speed at which the centrifugal forces during a steady-state cornering become theoretically equal to the friction force

$$F = \mu m g \tag{3}$$

Where μ is the overall coefficient of friction and g is the gravitational acceleration). At the considered traveling speed of 0.85 the car inherently suffers from intermittent instability (due to the yaw inertia both in the entry- and exit of corners, dynamics lateral weight transfer in corners, etc.) that we intend to counter by the use of the steering controller. The speeds of the car during the race on the test track featuring different friction coefficients that provides a different level of the slippery condition are shown in Table 6.

***Table- 6 :*** Speed of the car during the fitness trial on the test track with different friction coefficient

| **#Road Condition** | **Friction of Tires, $\mu_t$** | **Friction of Road Surface, $\mu_s$** | **Overall Friction, $\mu = \mu_t \times \mu_s$** | **Critical Speed, m/sec** | **Speed of the Car (0.85 of the Critical), m/sec** |
|---|---|---|---|---|---|
| 1 | 1.0 | 1 (dry) | 1 | 22.13 | 18.82 |
| 2 | 1.0 | 0.8 (dry) | 0.8 | 19.79 | 16.8 |
| 3 | 1.0 | 0.6 (rainy) | 0.6 | 17.15 | 14.5 |
| 4 | 1.0 | 0.5 (rainy) | 0.5 | 15.65 | 13.3 |
| 5 | 1.0 | 0.4 (snowy) | 0.4 | 14 | 11.9 |
| 6 | 1.0 | 0.3 (icy) | 0.3 | 12.12 | 10.3 |

In addition, we took into account in the model through simulation the influence of dynamic effects on the car, such as side gliding.

## 2.3 TORCS Requirements

### 2.3.1 Software

#### 2.3.1.1 OpenGL

The graphical component of the TORCS application is implemented using the OpenGL [6] software interface. This is platform-independent API that supports all major operating systems (Windows, Linux, Mac OS) and allows to work with a large number of different GPUs. It applies for writing applications that use two-dimensional and three-dimensional computer graphics, which includes more than 300 functions for drawing complex three-dimensional scenes from simple primitives. It is used to create computer games, computer-aided design (CAD), virtual reality, visualization in scientific research. On the Windows platform, it competes with Direct3D.

Among the main features of the architecture of the OpenGL specification, you can specify a single API for different types of adaptation of various 3D accelerators, as well as the absence of a noticeable difference in the capabilities of hardware platforms achieved by the implementation of the missing functionality using software emulation.

So, we need a working OpenGL / DRI driver, development tools including gcc and additional header files for the libraris GLUT, GLU, XFree86, libc, and OpenGL. OpenGL is most often supported by the distribution. For a Linux system, this is usually solved by checking the "development machine" parameter in the distribution settings.

#### 2.3.1.2 GLUT or FreeGLUT for Linux (OpenGL Utility Toolkit)

The main goal of mentioned OpenGL is rendering two-and three-dimensional graphics. At the same time, this API does not create windows for rendering at all, reading input from the user, and other similar and highly dependent on the specific operating system work, so TORCS use the GLUT cross-platform library for these purposes. The most important features worth noting are:

- Creating an application window
- Application window control functions
- Poll keyboard and mouse
- Functions for drawing various geometric shapes
- Functions for creating context menus
- GAMEMODE (quick switch to full screen mode)

#### 2.3.1.3 Libpng

Libpng was written as a companion for the PNG specification (raster graphics) as a way to reduce the time and effort required to support the PNG file format in applications.

Libpng was designed to manage multiple sessions simultaneously, to be easily modifiable, to be portable to the vast majority of machines (ANSI, 16-, 32- and 64-bit) and easy to use. The ultimate goal is to facilitate the adoption of PNG format in any way possible.

#### 2.3.1.4 Checking library availability

Below are the test commands (for the shell console), verifying that the system contain the necessary requirements.

- **Checking OpenGL and DRI**

  Start XFree86, open a terminal and run as normal user (the $ means the prompt of a normal user, # the prompt of root).

```
$ glxinfo | grep direct
```

An answer indicating that the OpenGL is installed is as follows:

```
direct rendering: Yes
```

Otherwise you have to check your OpenGL setup.

- **Checking GLUT**

For RPM (Red Hat Package Manager – open-source and free package management system for Linux distributions), run the following command:

```
$ rpm -qa | grep glut
```

An answer indicating that the GLUT is installed on your PC up to a specific package names is as follows:

```
mesaglut-3.4.2-42
mesaglut-devel-3.4.2-42
```

Otherwise you have to install GLUT. It is important to check if you have installed glut.h. If in response to the previous request only one package is listed, then you can check the presence of the glut.h header file in it with following command:

```
$ rpm -ql PACKAGE_NAME | grep glut.h
```

where PACKAGE_NAME takes value *mesaglut* or *mesaglut-devel* in our case, without version numbers. If the result is an empty string, then the header file is need to be installed.

- **Checking Libpng**

For RPM based distributions, run the following command:

```
$ rpm -qa | grep png
```

An answer indicating that this library is installed up to a specific package name is as follows:

```
libpng-2.1.0.12-153
```

Otherwise you need to install it and its header files named like png.h.

### 2.3.2 Hardware

TORCS requirements include quite old hardware with low performance by today's standards, due to the long-standing start of development and support for backward compatibility. Thus, for the correct operation and launch of TORCS, we need

1) 3D accelerator with OpenGL support for our platform,

2) Processor with 800 MHz or more,

3) 256 MB RAM and,

4) Nvidia GeForce 2MX AGP or better (or similar Kyro, etc.).

Our z-buffer should have a depth of 24 bits or more to avoid failures. On the official website of the TORCS robot tutorial, application tests for various systems [7] are commented on, which are useful to familiarize with when installing the application.

## 2.4 Evolutionary computation

Fuzzy control is currently one of the most promising intelligent technologies to create high-quality control systems [19]. A common prerequisite for the applying of fuzzy control systems is, on the one hand, the presence of uncertainty associated with both the lack of information and the complexity of the system and the impossibility or inappropriateness of its description by traditional methods, and on the other hand, the availability of information about the object, the necessary control actions, disturbances etc. quality information. This makes it surprisingly suitable for the tasks of driving a self-driving car.

The task of driving a car includes a whole range of various limitations, both of a technical type (such as delays in transmitting commands, wear and tear of parts), and computational (such as various accounting errors or processing power). Almost imperceptible deviations in control resulting from these errors can lead to the loss of stability of the car and reduce the target control parameter - safety. The steering wheel control system is an object of control of high complexity with a complex mathematical model and a wide range of parameter changes. Traditional management methods, such as a PID controller, may not provide the required control quality. At the same time, it is advisable to use systems based on fuzzy logic in the absence of information or the high complexity of the control

object, while creating high-quality control systems. This paper presents a number of aspects related to control systems based on fuzzy logic implemented through genetic programming to implement a self-driving car model. In addition, the task is to minimize the arrival time for a given trajectory of movement along the race.

Evolutionary computing is based on the idea of applying Darwinian evolution to a computer program [20]. The goal of evolution is to improve the quality of poor decisions by random mutations until the problem is solved with the necessary accuracy. For ease of perception, the names of the mechanisms in this strategy (such as an individual, mutation, genes, generations, selection, survival) are also borrowed from biology. It should be clarified what in our case is understood by these terms. So, in our research, an individual is a function or equation that controls the movement of a car along a highway. It is a combination of terminals (genes). Various combinations of function terminals are implemented by selection, mutation and crossover operators. The track with its slippery conditions and turns is the environment, and the quality of its passage is the suitability of the individual.

Thus, natural selection is the process of forming a population, which contributes to the "survival" of individuals more adapted to the external environment and "elimination" of those individuals that have a reduced fitness for the external environment. In a long evolutionary process, the most successful gene combinations will survive and vary, which may make it possible to find very unexpected and bizarre solutions. The operator that forms the next generation population and implements the principle of natural selection in the genetic algorithm, called the selection operator.

The purpose of evolution is to maximize the fitness of the genome, to better match a particular environment. Despite the deep connection with biology, it can be considered as a pure optimization problem. We have a fitness estimation function and we need to find the

point of its maximum. The problem is that we don't know what the function looks like and can only make guesses by selecting (simulating) points around the current solution (Figure 8).

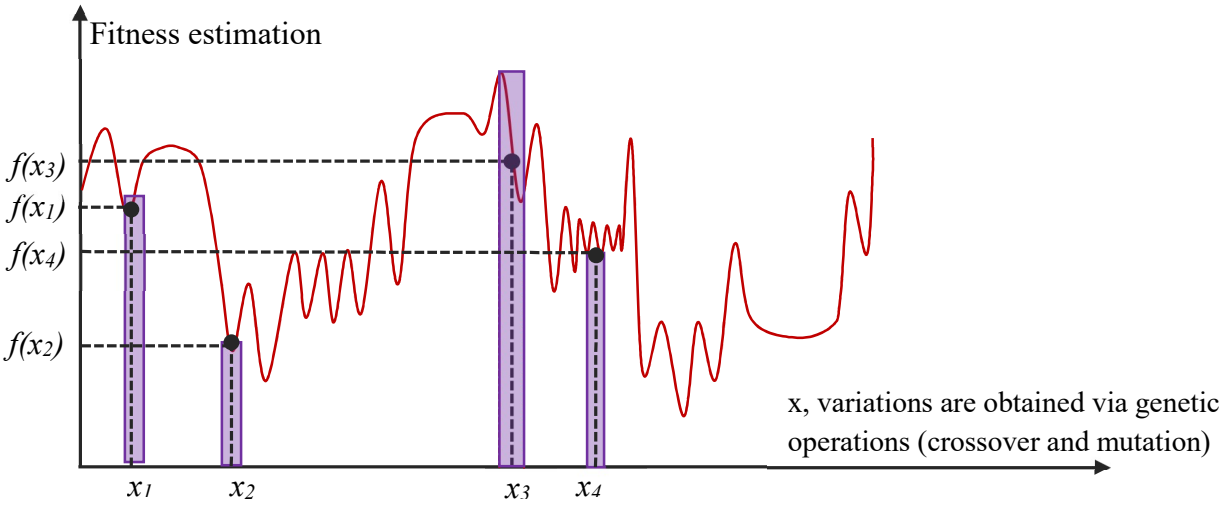



***Fig- 8:*** Fitness landscape and the location of the population of four candidate-solutions of a given (current) generation. The evolution attempts to find a new population of candidate solutions that are presumably "higher" in the fitness landscape. The new population is obtained by recombining (via genetic operations – crossover and mutation) the candidate solutions of the current generation.

Evolutionary computing do not guarantee the detection of a global optimum in polynomial time, because only using the full enumeration method allows to find a solution to global optimization. However, the genetic programming allows to choose a "reasonably good" solution in less time than other well-known deterministic or heuristic search optimization algorithms. Consider the generalized structure and each of the operators of evolutionary computing.

### 2.4.1 Algorithm loop of the search of the reproduction population

This is a search engine optimization algorithm that starts with an initial population $P^0$ (a set of functions $(t_1^0, t_2^0, \dots, t_n^0)$ ) and iteratively performs the following cycle of operations (Figure 9):

1) Estimation - calculation of fitness function values for any function
2) *Selection* - selection from a population of $P^i$ reproductive sets $R^i$(subsets of the functions).

3) Reproduction - generation of new functions from the reproduction set $R^i$ using combinations of the following operations:
    a. copying - creating identical copies of some or all functions from $R^i$
    b. *crossover* - constructing new functions by concatenating terms of those functions that are selected by copying from $R^i$;
    c. *mutation* - constructing new агтсешщты by substituting characters from the some alphabet of terms T in the selected positions of one of the functions.
4) Replacement - the formation at the next step (generation) of a new population $P^{i+1}$ by replacing some or all of the functions in $P^i$.

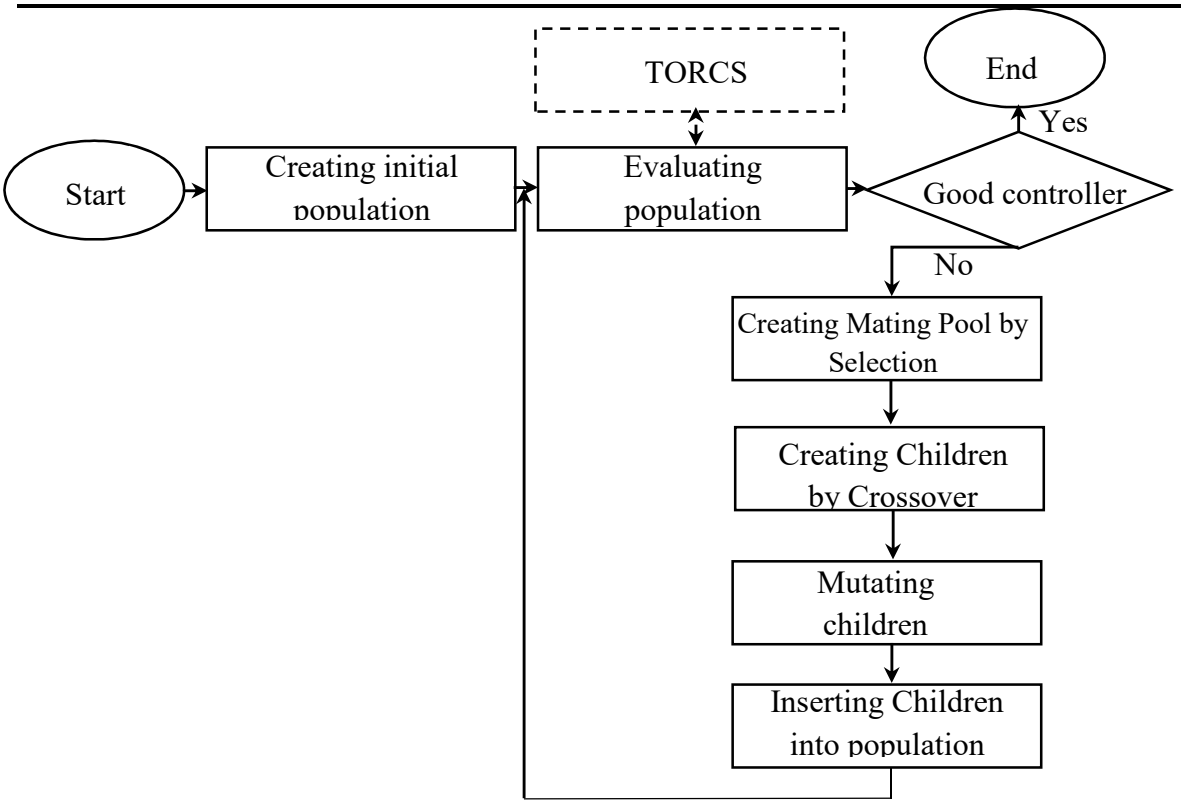



***Fig- 9:*** Flowchart of genetic programming and simple imitation of the mechanism of natural selection

Moreover, on the Figure 10 showed the process of the fitness evaluation (estimation function) via interaction between XGP and TORCS applications more detailed.

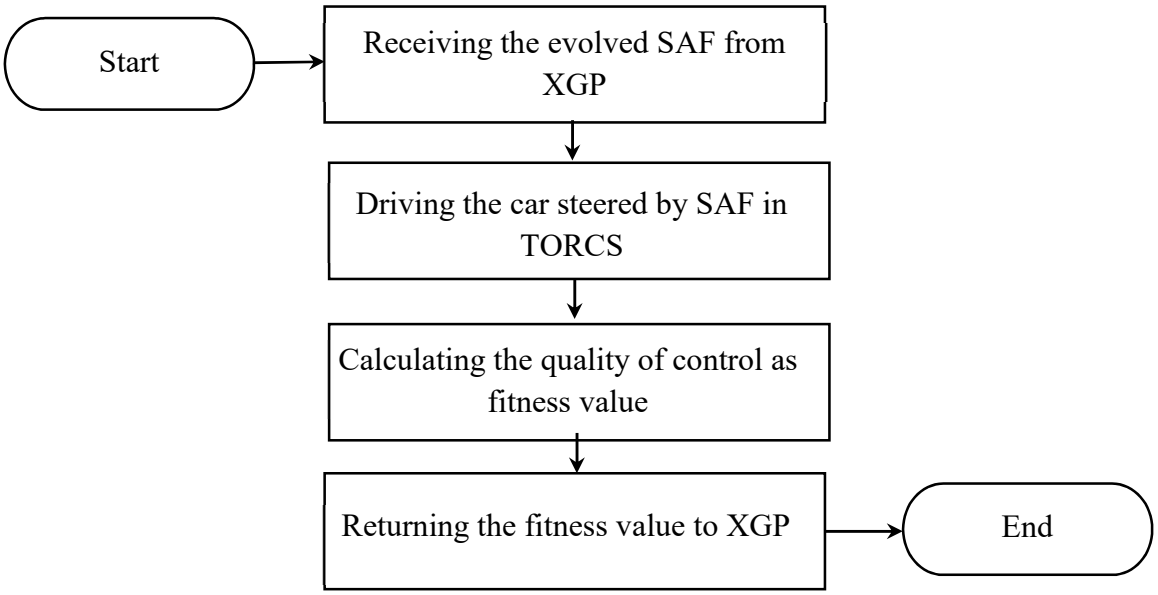



***Fig- 10*** Fitness function evaluation Flowchart

Thus, in Figure 11, there is an exchange of information between TORCS and XGP applications and calibration of the SAF function, depending on the result obtained using the fitness function.

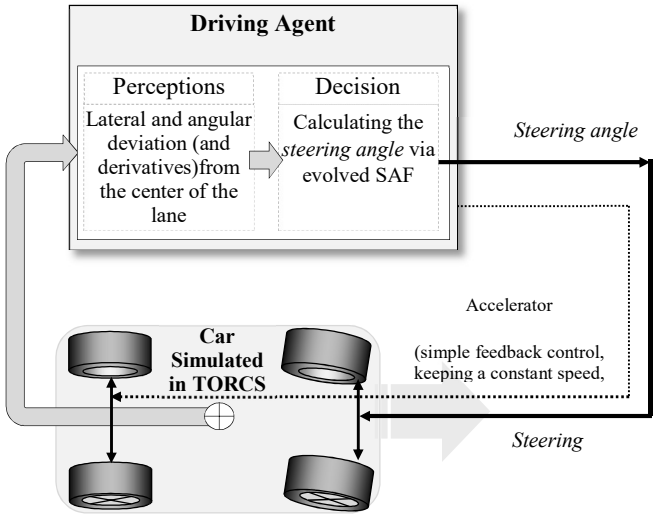



***Fig- 11*** Driving the car steered by SAF in TORCS

### 2.4.2 Genetic Programming (GP) Approach

Genetic programming is an evolutionary algorithm whose individuals are computer programs or functions - which is good coincides with our purpose. The first results using this technique were obtained in the 80s, and in the past decades, due to a sharp increase in the power of computers, interest in genetic programming as a means of automating software development has increased. Currently, using genetic programming, a number of results have been obtained that are superior to similar results obtained by people, for example, sorting networks, a quantum algorithm for the Grover problem, etc.

The key idea of genetic programming is to present the program at a fairly high level of abstraction, allowing for the specificity and structure of computer programs to be taken into account. The following describes genetic programming in the concept of John Koza. This method uses the representation of programs in the form of parse trees [21].

There are terminals in the leaves of the tree, and non-terminals in the internal nodes.
Previously mentioned main genetic operations – selection, mutation, and crossover in our project are implemented as follows.

- In genetic programming, the same *selection* methods are used as in evolutionary calculation.
- The single-point *crossover* for parse trees is defined as follows - a random subtree of the first individual is replaced by a random subtree of the second individual randomly selected from the mating pool. An example of a crossover operation is shown in Figure 12. The second descendant individual is created in a reciprocal way to that of the first one.
- *Mutation* is realized by replacing one subtree from newly created descendants with a randomly generated subtree (Figure 13).

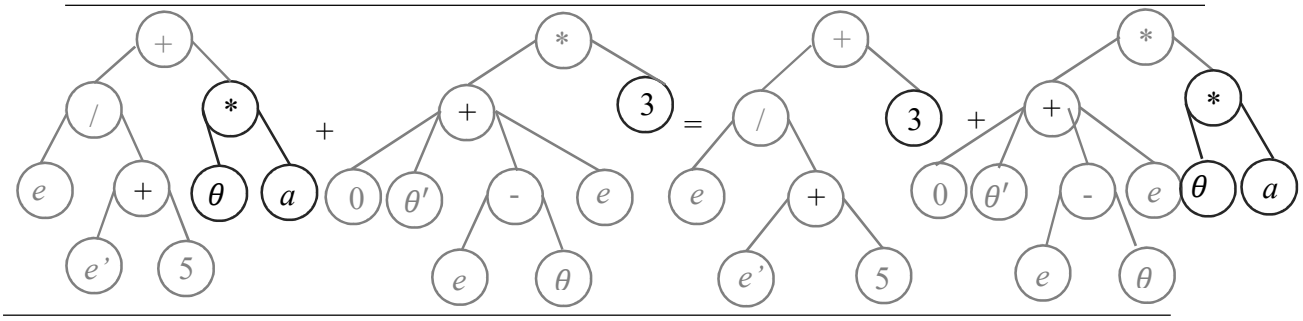



***Fig- 12*** Crossover operation example

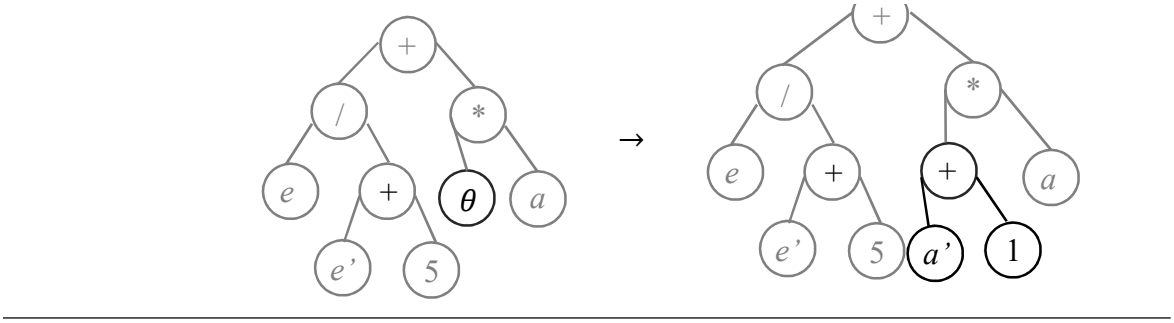



***Fig- 13*** Mutation operation example

The value of the main parameters of the adopted [22][23] evolutionary process can be seen in Table 7:

***Table- 7*** Main parameters of GP applied for evolving SAF

| **Name** | **Value** |
|---|---|
| Set of non-terminals (operations inside SAF, functional set) | { +, -, *, / } |
| **T**erminal **set** | {Variables defining the state of the car}:<br>• speed (*V),*<br>• steering angle (*δ),*<br>• lateral deviation (*e*) from center of the lane, its derivative ($\dot{e}$) and it integral ($\int e$),<br>• lateral acceleration(*a*),<br>angular deviation ($\theta$) , and the derivatives of the latter two (i.e., lateral *jerk* and *yaw rate*) |
| Population size | • 200 individuals |
| Selection | Binary tournament, ratio 0.1 |
| Elitism | Best 4 individuals |
| Crossover | Single point, ratio 0.9 |
| Mutation | Random subtree mutation, ratio 0.05 |
| Fitness value | Mean quadratic deviation from the center of lane + Mean quadratic lateral acceleration |
| Termination criteria | (#Generations>200) or (no fitness improvement for 16 consecutive generations) |

### 2.4.3 Evolution aim

Riding is not only a matter of finding current suitable values for the steering angle. This is a long-term task requiring an adequate steering response – steering angle function (SAF) in the event of an environmental change. In our study, we do not want to give evolution either the general structure of the SAF or its complexity, fully providing the choice of evolution. To do this, we will provide her with the input context variables of the car, and take the control SAF as the output.

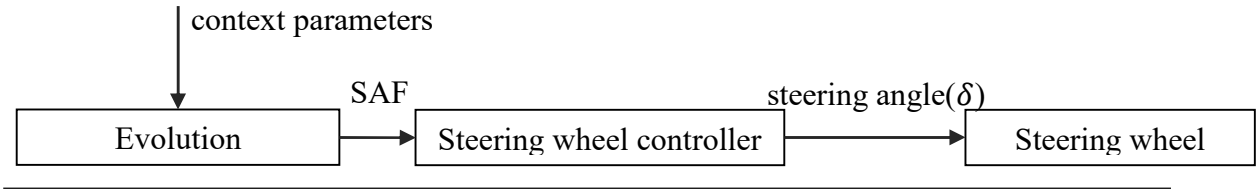



***Fig- 14*** Steering wheel control system

Steering angle $\delta$ - this is the entity controlled by the car, it behavior during whole race will be the goal of our evolutionary calculations. The evolutionary process will adapt the SAF so that the car can pass the track without any accidents.

### 2.4.4 Estimation

The suitability parameter should estimate the quality of the steering generated by the evaluated SAF during the test. The latter is implemented on a given test track (Figure 7) with a given coefficient of friction μ (Table 6) as follows: firstly, the simulated car is located in the initial position of the track — at the center of the lane – similarly to the realistic car position. Then the car slowly accelerates to a given constant target speed (see Table 6). From this point of view, in order to abstract the dynamics of the car from the effects of possible excessive traction or braking forces, the speed of the car is kept constant by means of a simulated cruise control (open loop). When the car accelerates to the target speed, the cruise control is activated and the steering switches to the advanced evaluated SAF. Each race frame

(with a sampling frequency of 40 Hz) according to this SAF, the rotation angle δ is recalculated. Technically, a reading formula is a parsing tree calculated for the current values of the parameters related to the state of the car.

The ride quality estimation function should encourage successful (leading to an accident-free ride) SAFs with better trajectories. By better trajectory we mean both (i) the exact movement along the center of the lane, and (ii) a quick and smooth return if it periodically deviates from it. At a fixed speed, compliance with these conditions leads to the search for the optimal control SAF which does not require additional time penalty parameters.

Thus, the estimated function of ride quality in our surveys is as follows:

$$F = P + C \times V_{l\_avr} \tag{4}$$

This is the weighted sum $P$ of (i) the area under the vehicle path around the center of the lane (as an integral of the absolute value of the transverse deviation e) and (ii) the average value of the transverse speed $V_{l\_avr}$ (as the integral of the absolute value of the transverse acceleration a) of the car. Lower compliance values correspond to better steering quality [23].

To make the task of driving a car difficult, but solvable, the target speed is set at 0.85 of the critical speed. At such a speed, the car by its nature suffers from instability - this is due to yaw inertia both at the entrance and exit from the turns, and the dynamics of lateral weight transfer in the corners [24]. We intend to extinguish these instabilities with the evolutionary value of SAF. A car moving at a speed higher than the established one is theoretically uncontrollable, and therefore there will not be such a SAF that can adequately drive the car. Similarly, a car moving much slower than the critical speed does not suffer from any instability, and its control can be adequately implemented by canonical servo control models.

## 2.5 Summary

The main idea of this study is that, having refused to consider the human driver as an adequate driver due to its cognitive vulnerability (see Chapter 1), we are looking for a controller that will be the safest way to control the movement of the car. Thus, we consider the controller-car union as a feedback system with a delay in signal transmission. Such a delay is fraught with the emergence of a potentially unstable system (Nyquist-Mikhailov criterion), therefore, in the case of accurate simulation, as in TORCS, the primary task for the controller is to suppress such instabilities and oscillations. One way is to analyse and tune the parameters of classic widely used models like proportional-integral-derivative (PID) (see Chapter 3). Another considered approach is the construction of the new control model using genetic programming, in conditions that train the suppression of oscillations. The proposed methodology is based on various simulation tools and software environments. These frameworks and tools are indicated in this chapter, and the next Chapter 3 of this dissertation discusses the details about the first of proposed approaches.

# Chapter 3

# Upgrading SAF for PID and PD controllers with parameters tuning

## 3.1. Materials and Methods

The PID controller was invented back in 1910. After 32 years, in 1942, Ziegler and Nichols developed a methodology for its adjustment. After the advent of microprocessors in the 1980s, the development of PID controllers is increasing. The PID controller is the most common type of controller. About 90-95% of the regulators [] currently in operation use the PID algorithm. The reasons for such a high popularity are the simplicity of construction and industrial use, clarity of operation, suitability for solving most practical problems and low cost. Among PID controllers, 64% are single-loop controllers and 36% are multi-loop []. Feedback controllers cover 85% of all applications, direct controllers - 6%, and cascaded controllers - 9% [].

The simplest automatic control system with feedback is shown in Figure 15.

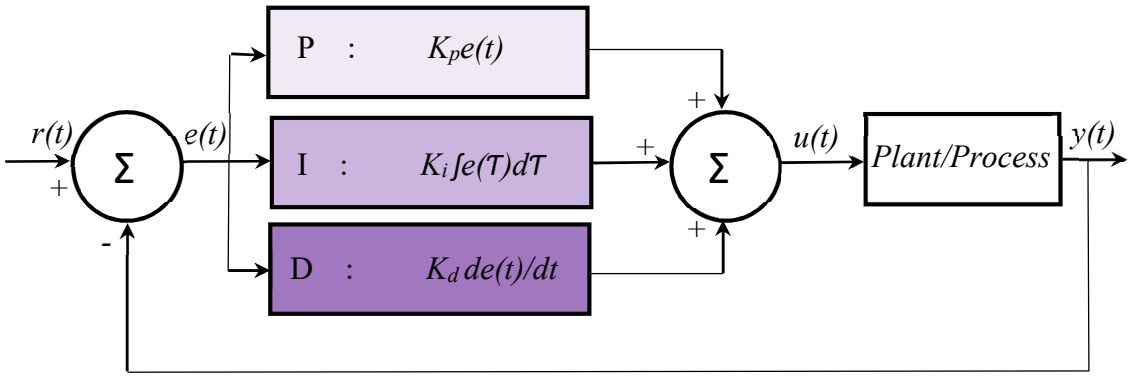



***Fig- 15*** PID controller feedback control system

The system controls the quantity *y(t)*, that is, it outputs the quantity *y(t)* to the externally specified value *r(t).* An error *e(t)* is supplied to the input of the PID controller, the output of the PID controller is the control action for some process (for the control object) controlling the value of *y(t)*. The output variable and controller *R* is described by the expression

$$y(t) = Ke(t) + \frac{1}{T_i}\int_0^i e(t)dt + T_d \frac{de(t)}{dt} \qquad (5)$$

where $t$ is time, and $K$, $T_i$, $T$ is a proportional coefficient, integration constant and differentiation constant, respectively, then such a controller is called a PID controller.

In a particular case, the proportional, integral, or differential components may be absent, and such simplified controllers are called I, P, PD, or PI controllers. An important aspect is the correct estimation of PID constant parameters.

### 3.1.1. Algorithm Summary and Components

The first step of our research will be to look at the already known servo control model as a kind of PD controller. This is not an obvious step, so let's take a closer look at it.

Steering servo controls are a widespread way to control autonomous vehicles and one of the methods we are exploring. The servo control is carried out by the steering wheel and represents the approach of determining the instantaneous value of the angle of rotation. According to this model, the steering function δ is considered as a linear combination of the following two scaled accessory components: the angle $\theta$ between the longitudinal axis of the car and the desired path (say, the direction of the lane) and the distance e between the geometric center of the car and the center of the lane (as shown in the Figure 16):

$$\delta = k_1 e + k_2 \theta \quad (6)$$

Where $\boldsymbol{k_1}$and $\boldsymbol{k_2}$ – are the scaling coefficients.

Taking into account that for the short time period $dt$ and small values of angle $\boldsymbol{\theta}$ the following expression is true:

$$\theta \approx de/dx = de/(V\,dt) \quad (7)$$

When the speed in (7) is constant – for very short time period we could assume that – the previous equation (7) could be rewritten as:

$$\theta \approx kv\,de/dt = kv\,e' \quad (8)$$

And here *e'* is the first derivative of the lateral deviation of the car from the middle of the lane. Putting this new value of angle $\theta$ in (6), we obtain a representation of the servo model in which it is already easy to see the PD controller form:

$$\delta = k_1 e + k_2 (k_v e') = k_1 e + k^*_2 e' \quad (9)$$

This equation (9) represents the steering servo model as a typical PD controller. Its output function *y(t)* transfers the control variable, which is the rotation angle δ. It represents the sum of the proportional (P) and derivatives (D) terms of the error - the deviation of the car from the center of the lane e. Since the desired value of the process is the desired deviation from the center of the lane and tends to 0, the absolute value of the error e is only the measured value of the process.

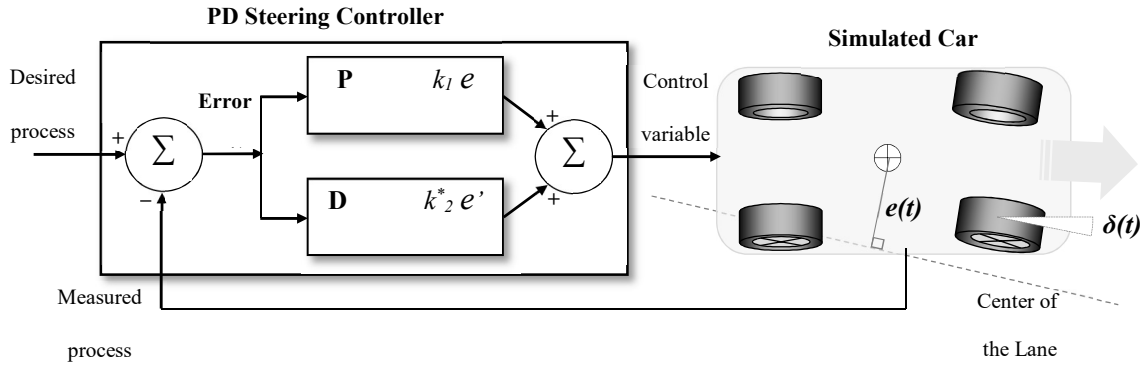



***Fig- 16*** Servo-control model of steering as a PD steering controller. The SAF, defining the steering angle δ is implemented as a sum of the proportional- (P) and derivative (D) terms of the error – the deviation e from the center of the lane.

The Controller cycle attempts to reduce the error value by constantly adjusting the rotation angle δ, which, as planned, will lead to the trajectory as close to the center of the lane as possible (Figure 17). This model could be extended in the same way to the PID model. This is the second controller we investigated. The PD model is extended with the help of an additional integral term. The reason for the inclusion of the integral term in the model is that

it can provide additional steering control for the car, which is stronger the longer the car does not return to the center of the lane. Such behavior is typical, for example, for a situation of skidding on corners under slippery conditions. In our experiments, time duration of 2 seconds was chosen in order to preserve the influence of the components on the one hand and, on the other hand, not to take into account outdated information in it. Thus, we provide greater model responsiveness and potentially better vehicle stability.

$$\delta = k_1 e + k^*_2 e' + k_3 \int e\, dt \quad (10)$$

Now, we can discuss the mechanism of tuning the coefficients in these equations from the point of view of PD and PID controllers and servo model.

The scaling coefficients from (6) are calculated as follows:

$$k_1 = \frac{STEER\_FACTOR}{\text{Steer lock angle}}, \quad (11)$$

$$k_2 = \frac{1}{\text{Steer lock angle}} \quad (12)$$

Steering lock angle (Figure 3, Table 1) are usually in the range [30 ° ~ 45 °]. Varying the values of the coefficients $k_1$ and $k_2$ affects the speed at which the car approaches the center of the lane. This servo control model is designed to control the car in such a way as to minimize both the average absolute deviation of the geometric center of the car from the middle of the lane and the average absolute lateral acceleration during this test. Coefficient scaling allows one of these properties to be strengthened, usually to the detriment of the second.

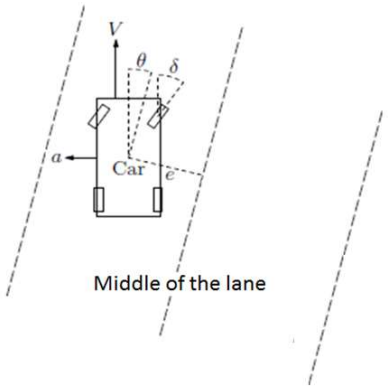



***Fig- 17*** Car under servo model control parameters

PID control coefficients must be set in advance and cannot be changed during the course of the experiment. Finding and adjusting the coefficients requires experience and is a complex and time-consuming process. Recommendations for tuning are given in special technical and scientific literature and are mainly reduced to some versions of enumerating values in areas of potential interest (tangential method, Ziegler-Nicholson method [25][26][27]). Here we give the tuning options for the PID controller by the brute force search method.

### 3.1.2. Tuning parameters with brute force

The results of a complete enumeration of combinations of parameter values are shown in Figure 18 for the PD model as a two-dimensional case, and Figure 19 for the PID model - a three-dimensional case. As expected, in the studied areas for which the numerical values are reasonable, we can notice the process of approximating the combination of parameters to the extremum. Steering delay that was taken into account is 100 ms.

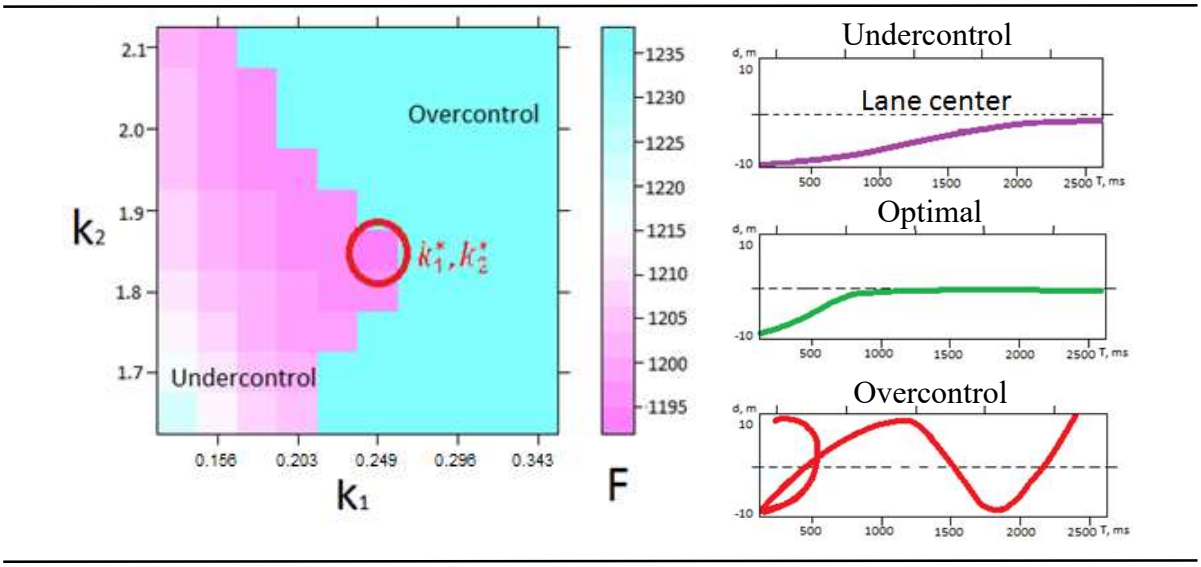



***Fig- 18*** Car under servo model control with different parameters

On the Figure 18 on the right part showed the trajectories of the car with corresponding to various typical parameters from the left part from the figure. From top to down – the insufficient consideration of the PD parameters due to the low value of the $k_1$ and $k_2$ coefficients, until overreaction to their changes due to their too large coefficient values**.** This is a trade off between fast controller response with smooth car trajectory and stability on the wet road. Optimal values we found of them are:

$$\begin{cases} k_1 = 0.240 \\ k_2 = 1.889 \end{cases}$$

The obtained parameters slightly increase the influence of the derivative part of the controller (sharpness of change in the amount of turning) in the calculations, which allows both parameters to be taken into account most fairly and affects driving in the best way.

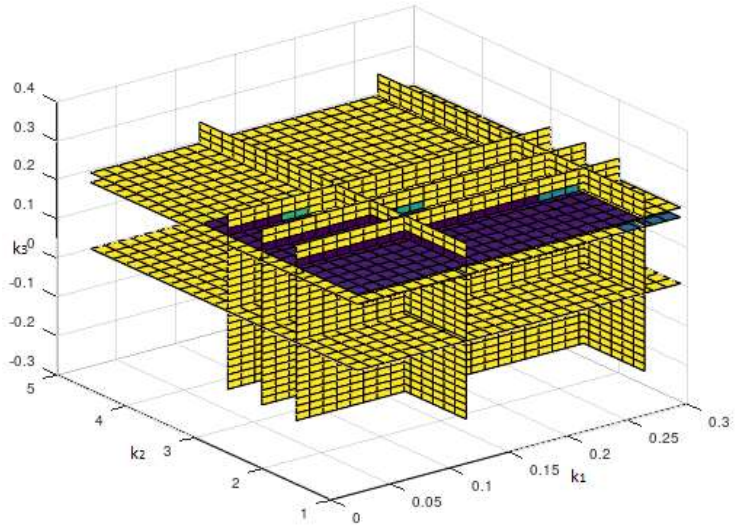



***Fig- 19*** Sequential search of values combination $k_1$ , $k_2$ and $k_3$. The lowest values of the estimation function are marked with lilac colour.

As mentioned earlier, the third parameter is responsible for enhancing steering in the event of a prolonged deviation of the car from the desired path. In the current conditions of a slippery road and an almost critically high vehicle speed (Table 6), i.e. instability of the car, it was expected that this parameter will make a big impact in the control model. As a result of iterating in the over 2500 values search space, the following optimal values of the coefficients for PID control model were found:

$$\begin{cases} k_1 = 0.113 \\ k_2 = 2.433 \\ k_3 = 0.048 \end{cases}$$

However, as we see on the one hand, according to the estimation function values presented in Table 8, and on the other, in terms of the value of the corresponding coefficient $k_3$, the PID model does not have significant advantages over the PD (classical servo control model). This may mean that under the created conditions, the increased reactivity of the model through the integral term gives rise to even greater instability in the process of steering back to the middle lane and most likely provokes skidding in the opposite direction like overcontrol (Figure 18,

green curve). It is easy to verify this propagation by assigning a larger value to the coefficient at the integral term and tracing the vehicle path (Figure 20).

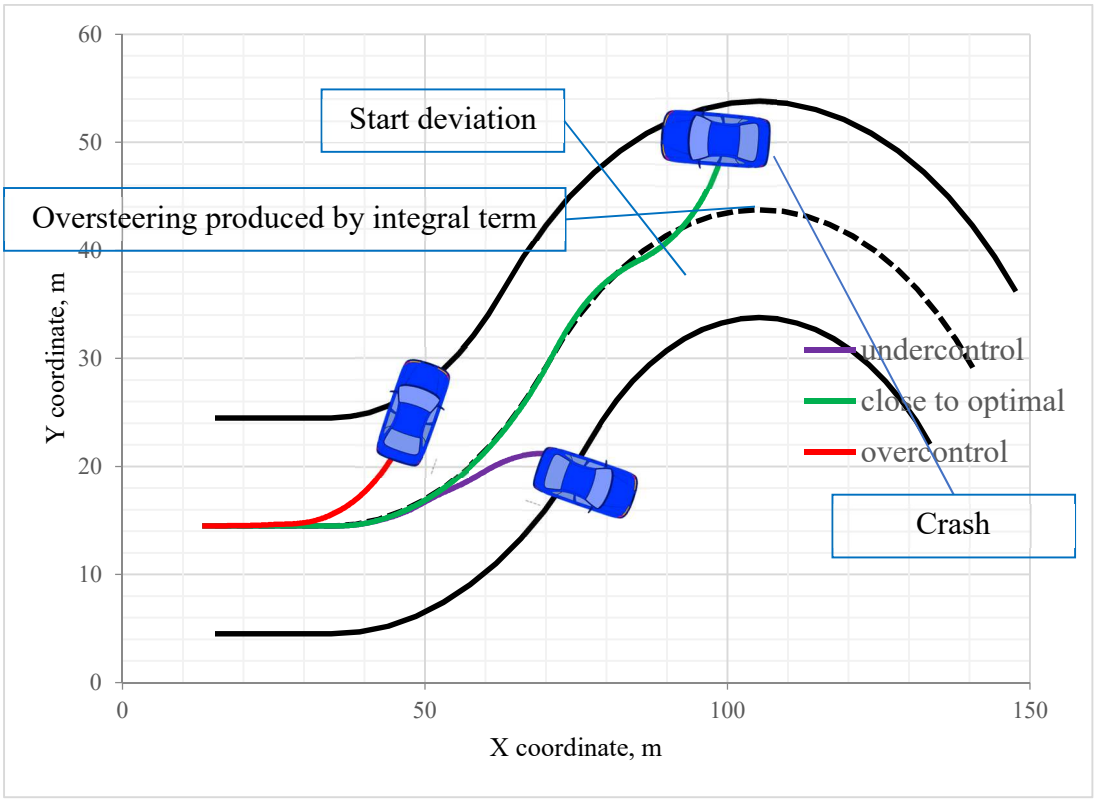



***Fig- 20*** Car with tuned PID controller parameters trajectories

On the picture above we displayed several selected paths of the car controlled by the PID controller, which are of interest due to their characteristic behavior. The violet curve corresponds to a search region that is close to the best in terms of decreasing coefficients (regions are showed on the Figure 18), red - in terms of increasing, and green - for fixed best coefficients $k_1$ and $k_2$ and with increased $k_3$.

Thus, in future experiments, we do not specifically consider these two models (PD and PID) separately, assuming the advantages and characteristics of their behavior to be nearly ambiguous in our conditions.

***Table- 5*** Experimental result for constructed SAF

| **Friction coefficient $\mu$** | **Tuned PD model** $\delta = k_1 \times e + k_2 \times \theta \approx k_1 \times e + k^*_2 \times e'$ | | **Tuned PID model** $\delta \approx k_1 \times e + k^*_2 \times e' + k_3 \times \int e$ | |
|---|---|---|---|---|
| | **Tuned values of $k_1$ and $k^*_2$** | **Fitness value** | **Tuned values of $k_1$, $k^*_2$ and $k^*_3$** | **Fitness value** |
| 0.6 | 0.247, 1.866 | 661 | 0.229, 2.055, 0.0321 | 546 |
| 0.5 | 0.186, 2.244 | 687 | 0.113, 2.433, 0.048 | 584 |
| 0.4 | 0.113, 3.378 | 765 | 0.150, 3.189, 0.0013 | 702 |
| 0.3 | 0.332, 2.055 | 1693 | 0.1257, 4.512, 0.0415 | 1212 |

## 3.2. Summary and Discussion

Thus, in this chapter we examined the classic servo model as a PD controller, and its extension to the PID controller. Studying their premium performance in the selected conditions through simulation, we selected the best parameters for them. The advantage of these control models is the simplicity of implementation and rather fast reactivity, which however leads to an increase in the dynamic instability of the car in wet conditions (Figure 20). This model well imitates the steering behavior of a human driver and, like the latter, provides good steering quality on dry roads and much worse on the slippery road (Figure 21). It should be noted that safe driving on wet roads under PD controller is achieved mainly at the expense of the speed of the transport, which is the easiest and most undesirable due to unpromising way. With appropriate settings for the scaling coefficients (depending on the specific characteristics of the physical model of a particular car) on dry roads, servo control can achieve steering behavior that is very similar to the behavior of a human driver with adequate cognitive ability. In the case of wet roads and bad weather conditions, a person needs additional extreme driving skills to maintain safety. This idea gives rise to a desire to optimize the PD controller in a similar way, to “instill” additional skills into it. However, as far as we know, there are no documented studies of the applicability of the PD servo control

model for automatic driving in wet, snowy or icy road conditions. Thus, in the next chapter we will consider a method based on the PD approach, but including some heuristics designed to simulate in some way the human ability to analyze the road.

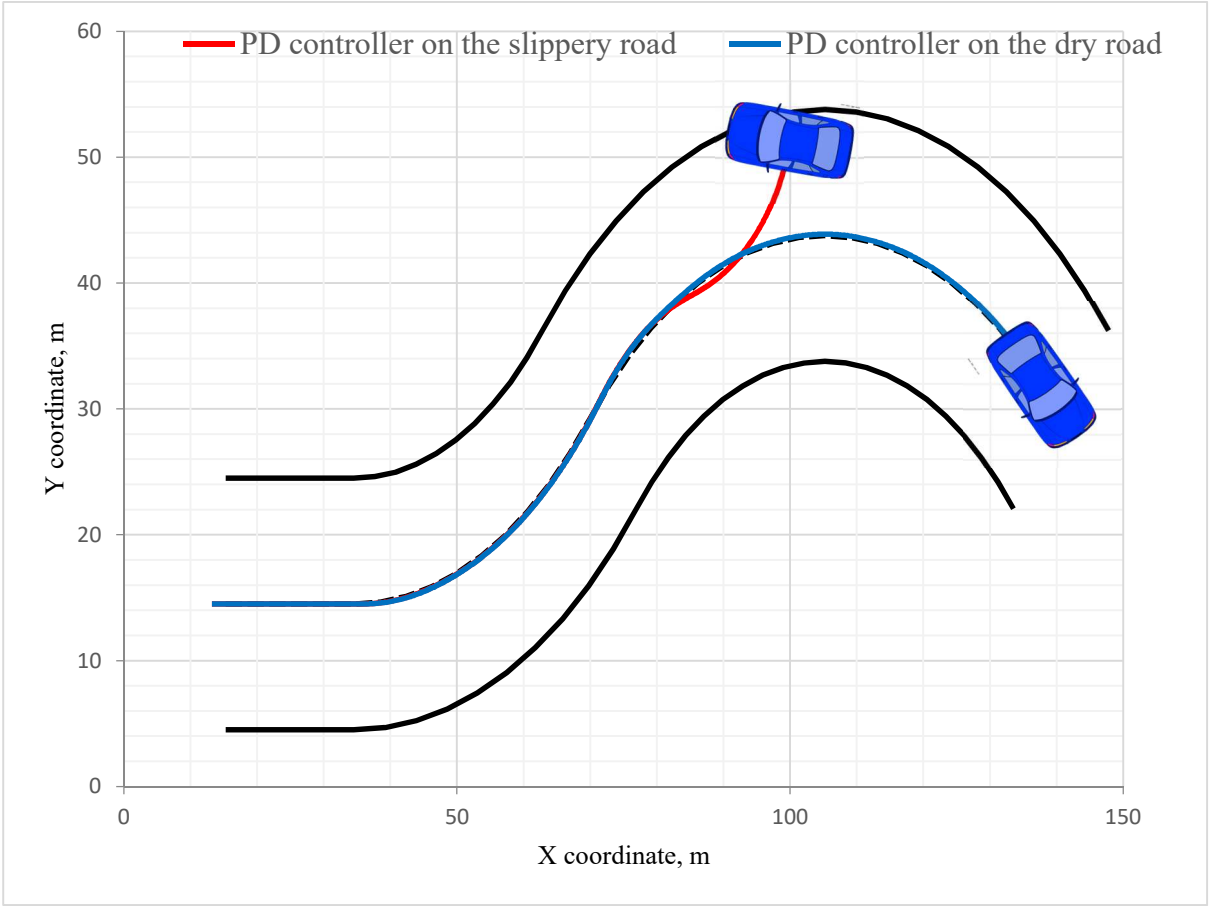



***Fig- 21*** Trajectories of the car under the PD controller in the different road conditions (on the slippery road, $\mu$ = 0.5 (wet surface) – red curve, on the dry road – blue curve)

# Chapter 4

# Upgrading SAF for PID and PD controllers with prediction

## 4.1. Algorithm Summary and Components

In this chapter, we will focus on a review of the development and analysis of an extended version of the canonical PD controllers, which are known to provide good steering quality only on non-slip roads. However, as was shown in previous chapter, on slippery roads, due to poor stability and controllability during the turning, the car starts suffers from understeer and, after applying correction mechanisms, from oversteer. This leads to a decrease in the quality of control and security and makes such controllers inapplicable in their classic form.

### 4.1.1. Human behaviour during the race. Human prediction tactic

The association between driving a vehicle through the PD of a controller and a person is based on the fact that both control models accept approximately one set of input data - the angular and linear deviations from the desired trajectory[28]. However, when driving a car as a human driver, the steering adaptation to various road conditions (dry, wet, snowy, etc.) occurs dynamically and taking into account the characteristics of the car (for example, length, width, weight, etc. ) The accuracy of the tuning and ride safety depends on the driver's experience and skills. Optimization of these parameters by a human specialist often requires deep knowledge both in control theory and in vehicle dynamics. On the other hand, automatic parameter tuning may require the use of heuristic approaches, which are notorious for their long runtime even when using significant computing power [29]. Thus, the style of an experienced driver is difficult to simulate for PD and PID controllers due to their rigid structure with few variables. In addition, the reactivity mechanism of these controllers calculates the steering output as a direct result of the combination of the currently perceived lateral and angular deviation of the vehicle from its assumed ideal trajectory (which is in all our experiments the centre of the road for convenient). Since these deviations are used as a discrepancy for correcting the position in the steering feedback control of the car, the non-

zero error value obtained during the movement of the car in the turn will lead to the trajectory of the car, which is always shifted “out” of the corner, which is fraught with a whole bunch of negative effects. Consider a situation where a turn is initiated by the appearance of an obstacle along the route: under the control of the PD and PID controller, the car will inevitably bypass the obstacle at a distance that will go farther than the intended ideal path (Figure 22), which, in turn, leads to an increased risk of collision with another obstacle, departure to the oncoming lane, etc.

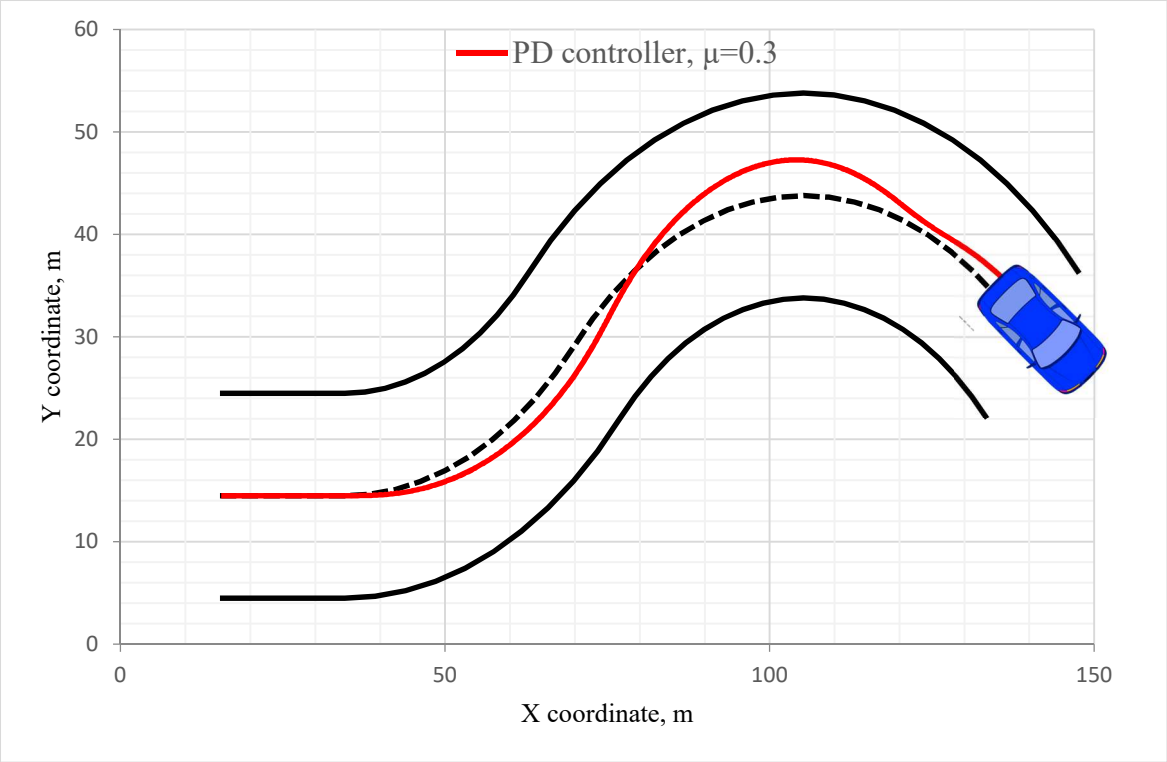



***Fig- 22*** The effect of the car is shifting "outward" relative to the desired trajectory during maneuvering while entering a turn that leads to losing control and stability after some time. The effect is seen more clearly than in previous images due to the increased wetness of the road surface (**μ** = 0.3)

In addition, the reactivity mechanism, combined with the purely mechanical delay in the transmission of commands to control the steering of the car (which in our experiments was selected as the standard [30] 100 ms) on slippery roads leads to a significant accumulation of the delay in the entire control loop, which cannot be fully compensated by the predictive (derivative ) component of these controllers.

Realizing the insufficiency of such compensation for delays, we developed a different algorithm for controlling the steering wheel of a self-driving car.

Another approach to improving the PD controller is to add a prediction mechanism. It is worth mentioning here that predictive models have already been widely used in relation to autonomous vehicles [31], but only also in conditions of non-slippery roads. One of the most widely used methods is the predictive control model (MPC) [32] and its modifications [33],[34]. This method, however, has some obvious disadvantages that hinder its applicability to many cases. The first of these is the high computational load, which rapidly [35] grows with an increase in the predicting area. In addition, the predictive model itself is very complex and includes integral calculations. Of course, this complexity increases even more if we take into account the slippery surface of the road and the updated dynamics of the vehicle.

### 4.1.2. Projection of the geometry position of the Car

Thus, we made small changes to the original PD controller, which, according to our idea, should imitate the behavior of the human driver with some degree of certainty. He sees an obstacle in front of him in advance, and, knowing about it, uses a maneuver to avoid it. The corresponding change in the structure of the controller consists in replacing one of the values of its members with its predicted value (see Figure 23, Figure 24).

$$\delta = k_1 e_{predicted} + k_2 \theta \tag{13}$$

Where $\theta$ is angle between the car direction and the road and $e_{predicted}$ – distance between the predicted car centre and the road calculated by following (14) – (17):

$$e_{predicted} = \mathrm{F}(x_{predicted}, y_{predicted}) \tag{14}$$

$$x_{predicted} = x_0 + V_x t = x_0 + Vt\ cos\alpha \tag{15}$$

$$y_{predicted} = y_0 + V_y t = y_0 + Vt\ sin\alpha \tag{16}$$

$$\alpha = \theta + A_{road} \quad (17)$$

Here $\boldsymbol{x_{predicted}}$ and $\boldsymbol{y_{predicted}}$ are predicted coordinates of the position of the car, $\boldsymbol{A_{road}}$ is the angle of the road at the point that is closest to the car, $F$ – function calculating the distance between the car and the center of the lane, $\boldsymbol{V, V_x, V_y}$ – the speed of the car, and its two orthogonal components, respectively, $\boldsymbol{t}$ – the predicting time interval, and α is the angular deviation of the car from the center of the lane .

This controller we called Predictive PD controller (PPD controller).

As the replaced term, we chose the lateral deviation component. We decided to predict only one of the components of the linear combination, both for simplicity and for the purity of the experiments, so that the new results were affected by a single change in the structure of the control formula. Moreover, using the predicted value of only one perceived variable related to the state of the car — a lateral deviation from the center of the road — we have demonstrated that the quality of driving on slippery roads can be significantly improved using the same set of the perception information of the controller, in case of the obtaining the maps of the upcoming road. According to the idea, the changes made - replacing the current deviation of the car from the desired trajectory with a deviation of its future position, if the car retains its motion vector - should compensate for the flaws (caused by delay) of the controller. In addition, such changes will not cause the addition of new variables or changes in the structure of the controller, like classical methods like MPC, that would inevitably increase the computational load of the controller.

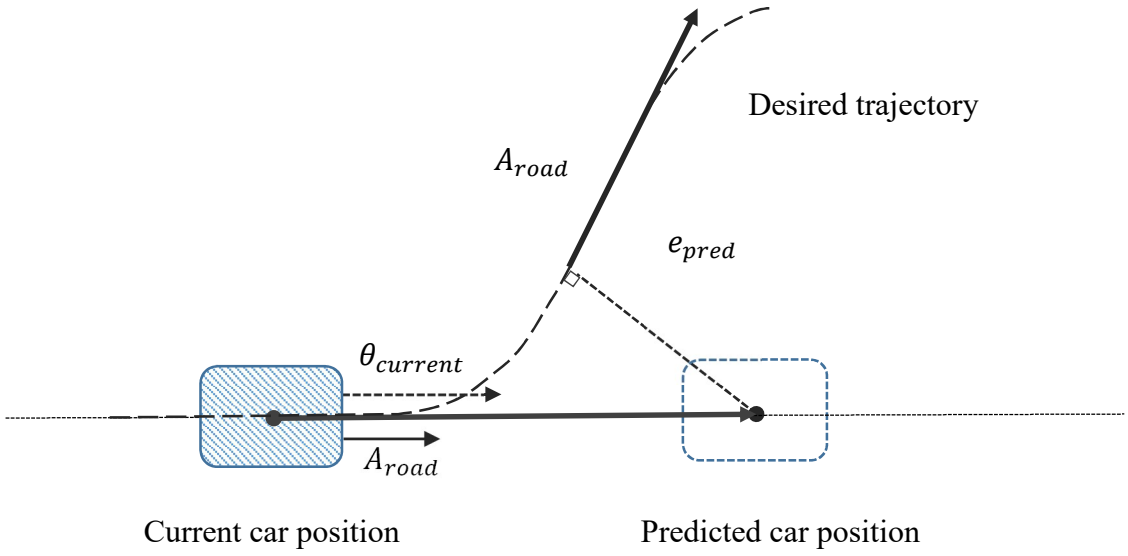



***Fig- 23*** Predicting the lateral deviation of the car $e_{pred}$

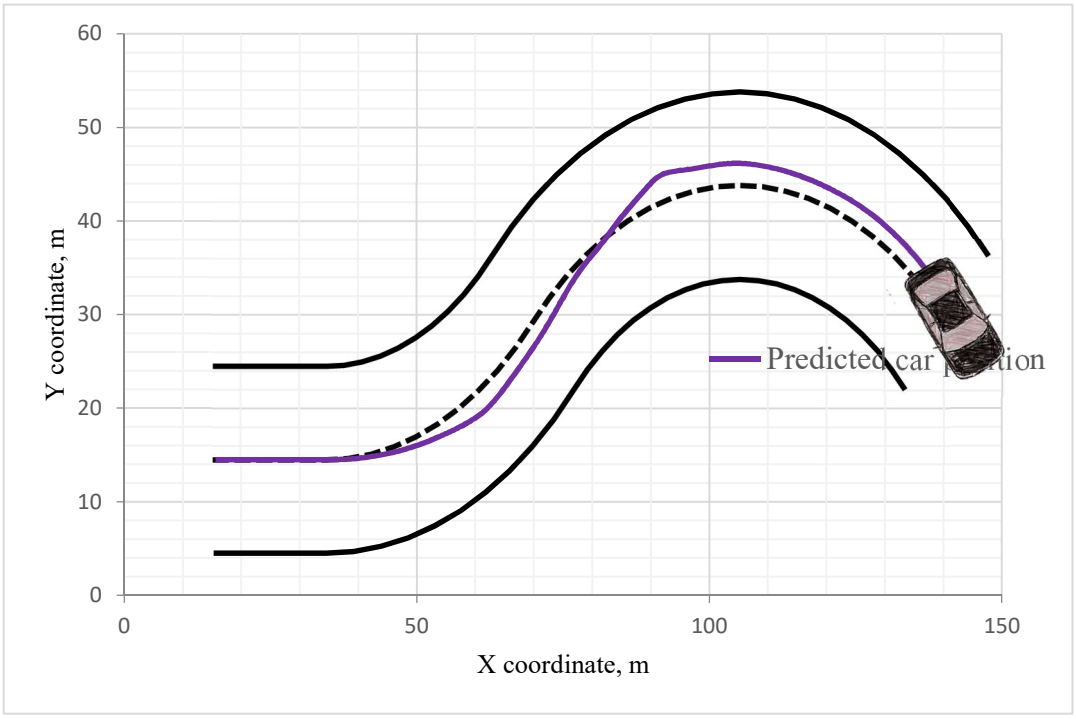



***Fig- 24*** Trajectory of the car predicted position on the road, $\mu$= 0.3

### 4.1.3. Simulators facilities for driver prediction

This method has one additional feature that is important to consider when choosing a control model for a self-driving car. This feature is the presence of "vision" in the controller. This feature is caused by the imitation of a professional human driver racing style and is realized by transmitting information to the controller about the upcoming section of the road, as if he had vision and the ability to interpret a seen or updated map of a short part of road ahead. Despite the potential profitability of the method, it imposes some limitations that did not exist before in PD controller model. Among these limitations is a sharp decrease in the effectiveness of the method in deteriorated environmental conditions - rain, fog, dark night, a truck close in front, etc., i.e. its dependence on additional information compared to the canonical method. As a possible measure to mitigate this drawback and increase security, a combination of this method with the canonical one, i.e. you can switch to it only if the conditions are suitable for the correct maintenance of the new controller needs, and use the PD controller by default. In addition, in our experiments, the weather vision conditions are ideal, and the car participates in the race alone. Therefore, the implementation of the proposed measure is not necessary, and we will consider the method and observe the results for its "pure" form, not claiming it to be complete, but only for an analysis of its main idea.

From a technical point of view, the TORCS simulator provides the ability to manually access and manipulate track data, so we implemented a method that collects and sends data about the upcoming track segment (see the Table 9 and Figures 27-1, 27-2) to the controller's decision center in real time for a moving car. Based on the foregoing, the question naturally arises of the "range" of the prediction - how much data is provided to the controller at a time, and how effectively it can use it. We tried to give answers to these questions in the following paragraphs.

***Table- 6*** Track segment parameters information

| Name | Value |
|---|---|
| seg.length | Length in meters of the middle of the track |
| seg.width | Width in meters of the segment (if constant width available from description file ) |
| seg.startWidth | Width in meters of the beginning of the segment |
| seg.endWidth | Width of the end of the segment |
| seg. lgfromstart | Length of begining of segment from starting line |
| seg. radius | Radius in meters of the middle of the track (>0) |
| seg. radiusr | Radius in meters of the right side of the track (>0) |
| seg. radiusl | Radius in meters of the left side of the track (>0) |
| seg. arc | Arc in rad of the curve (>0) |
| seg. center | Center of the curve |
| seg. vertex[4] | Coordinate of the 4 corners of the segment |

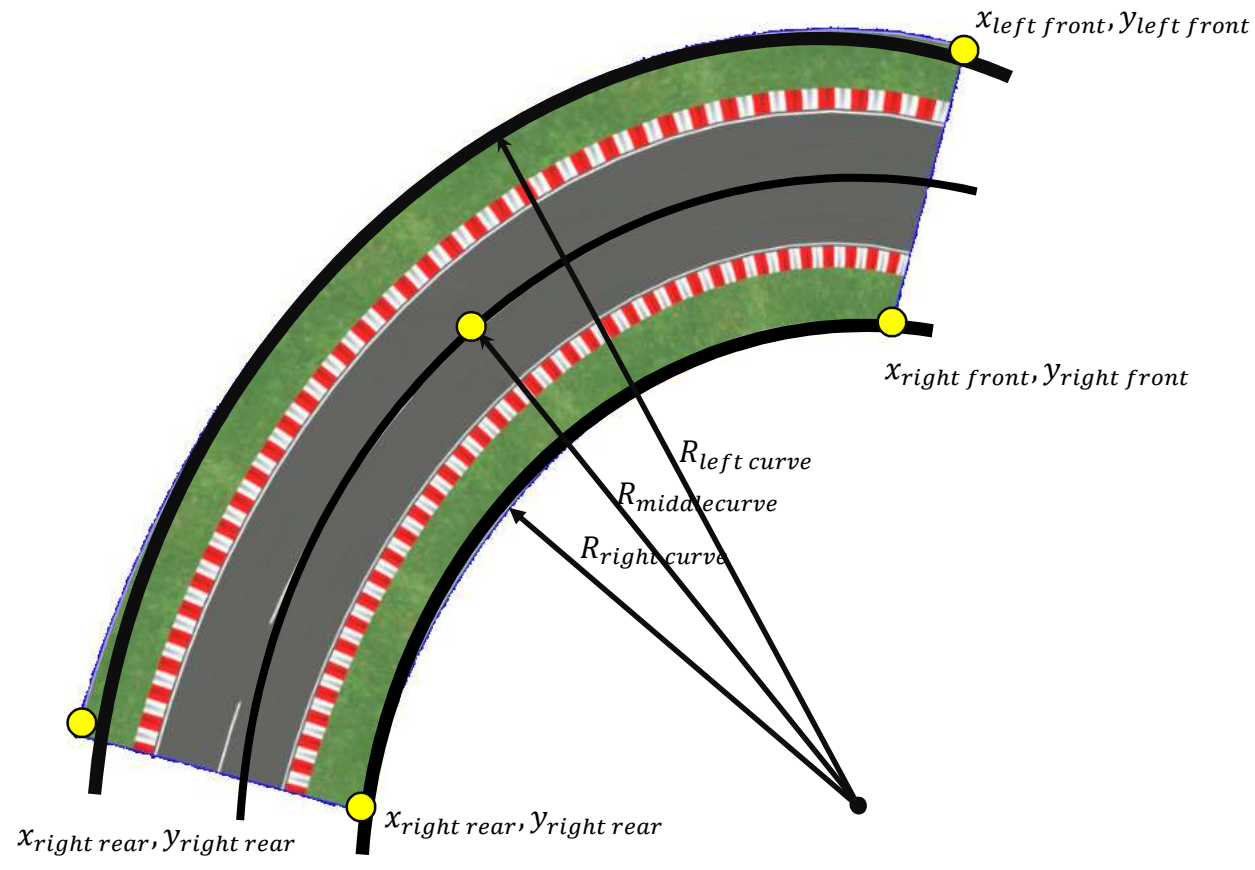



***Fig- 25*** Track segment sample

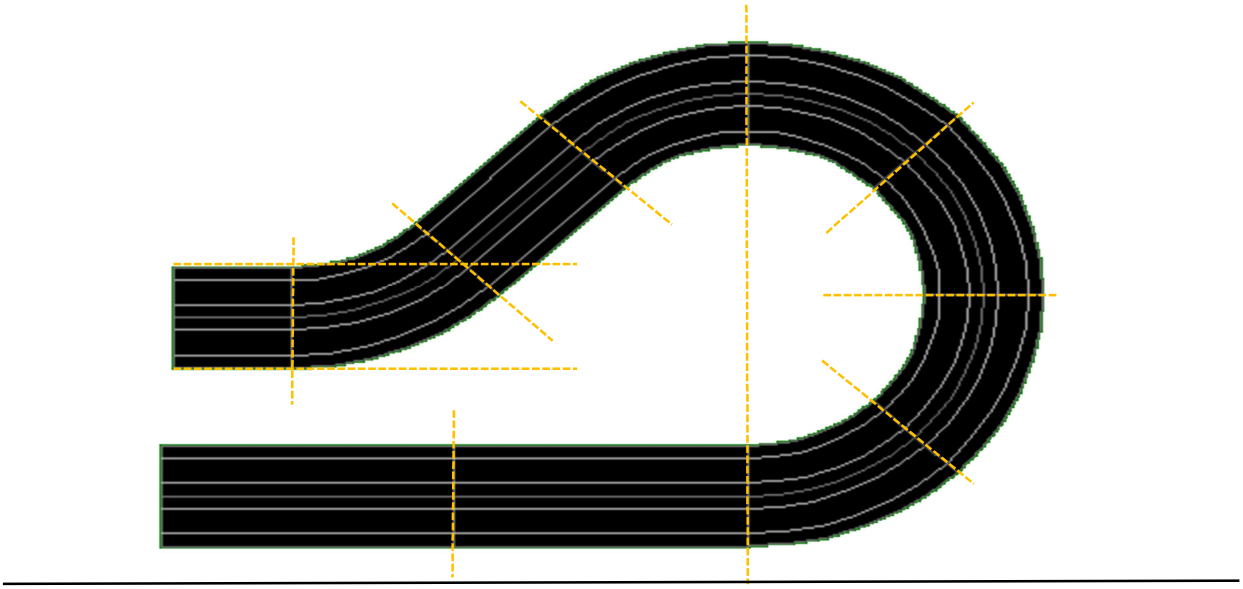


***Fig- 26*** Track segmentation schema

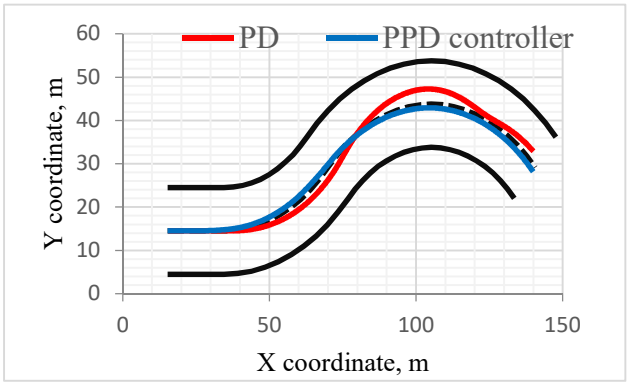
## 4.2. Results

The result of our investigation of the proposed approach was its implementation in the TORCS simulation environment in parallel with the classical method by writing a robot agent in C ++. This agent exchanges data with TORCS, receiving vehicle sensors data, and sending control commands to the car's steering wheel. This robot represent a control model idea.

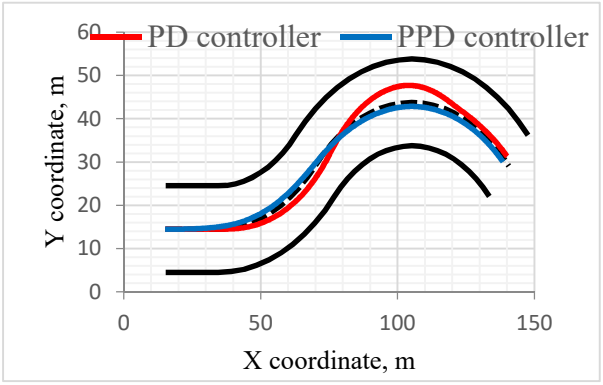
### 4.2.1. Comparison with the classical approach

To begin with, for each state of the road surface shown in Table 6, we developed a new analytical equation for the control model and checked them with different levels of target speed (for speed levels, see paragraph 2.2.2.4 and Table 6). We selected the scaling factors $k_1$ and $k_2$ in accordance with their previous best performance from Chapter 3, Table 9. This was done for the following reasons: we wanted to spend the optimization efforts of the evaluation function only to improve the perception of the controller. That is, to optimize only one parameter associated with the prediction in the controller, while fixing all the others. Thus, we launched a simulation to prediction PD controller for various environmental conditions. All algorithms are tested on the same road and with the same estimation quality function $F$,

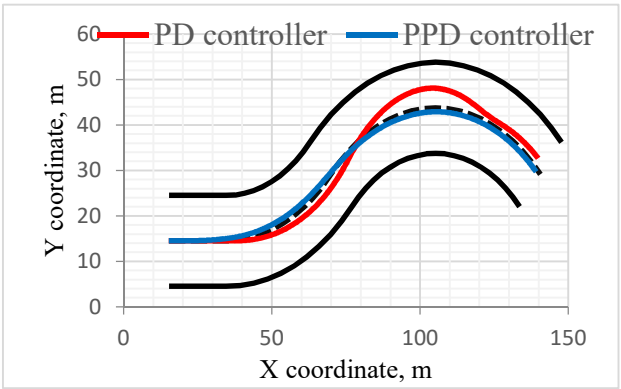
so we can correctly compare them. In these experiments, the prediction parameter $t$ from equations (15)-(16) was manually selected and varied in the range from 0.5 to 2.5 seconds for each controller equation. The resulting vehicle paths are shown in Figure 27.



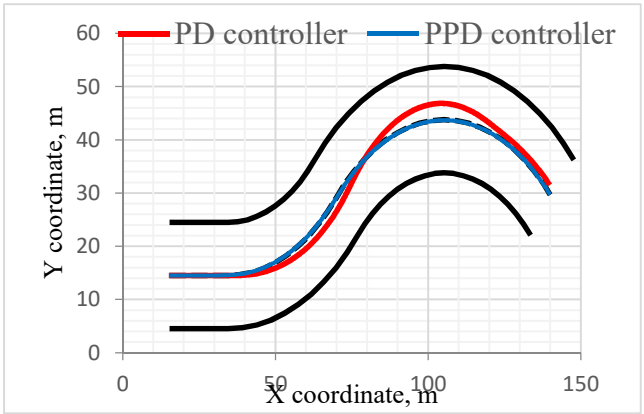

μ = 0.5, 0.85 $V_{cr}$ (*a*)



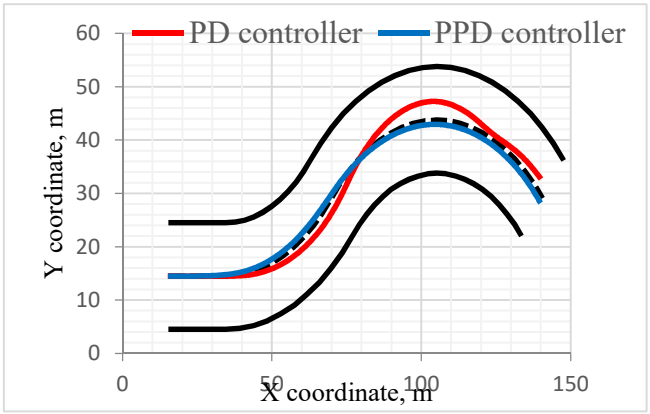

μ = 0.5,0.9 $V_{cr}$ (*b*)




μ = 0.5, 0.95 $V_{cr}$ (*c*)

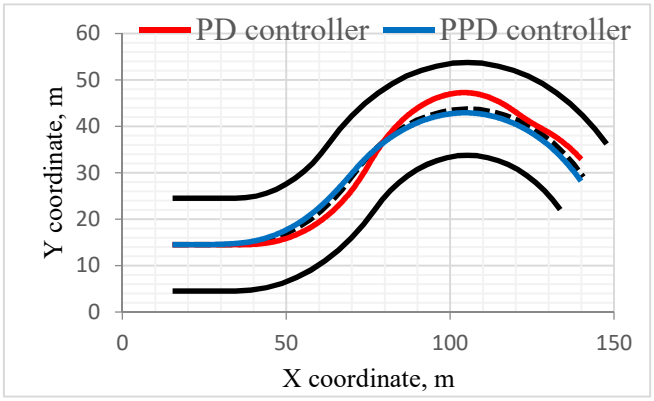



μ = 0.3, 0.85 $V_{cr}$ (*d*)

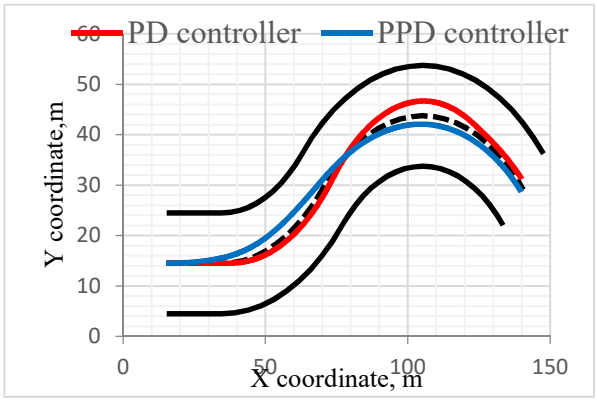



μ = 0.3, 0.9 $V_{cr}$ (*e*)

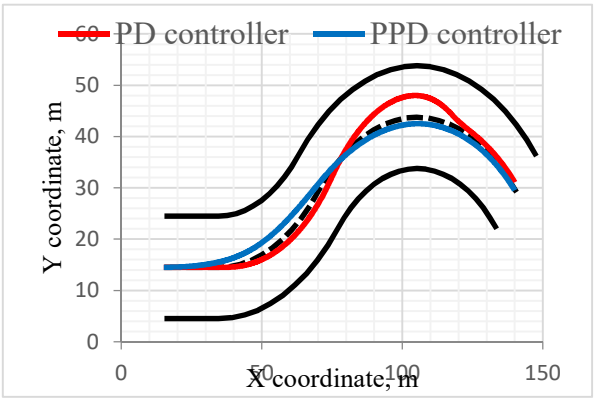
***Fig- 27-1*** Car trajectories on the track tuned with prediction SAF of standard PD controller for friction coefficients μ=0.5 (a-c), μ=0.3 (d-f), μ=0.1 (g-i) respectively. The blue curves correspond to the trajectories controlled SAF with prediction (PPD controller), red – original SAF



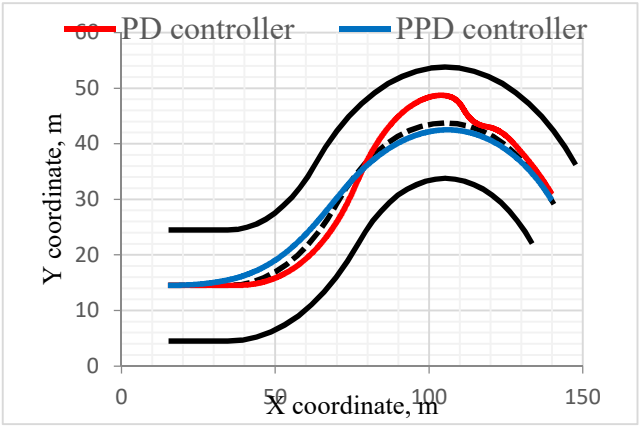

μ = 0.3, 0.95 $V_{cr}$ (*f*)




μ = 0.1, 0.85 $V_{cr}$ (*g*)




μ = 0.1, 0.9 $V_{cr}$ (*h*)




μ = 0.1, 0.95 $V_{cr}$ (*i*)

***Fig- 27-2*** Car trajectories on the track tuned with prediction SAF of standard PD controller for friction coefficients μ=0.5 (a-c), μ=0.3 (d-f), μ=0.1 (g-i) respectively. The blue curves correspond to the trajectories controlled SAF with prediction (PPD controller), red – original SAF

Visually estimating the vehicle trajectories presented in Figure 27 for each road condition, it can seen the following: the trajectory provided by the PPD controller version was better (i.e. closer to the center of the lane and smoother) in all test cases. In addition, as can be seen in Figure 28, which shows the changes in the steering angle for the parameters μ = 0.3, 0.95$V_{cr}$, the new controller provides smoother control than the classical PD controller with a lower steering amplitude. This driving style is more comfortable for both passengers and car mechanics. Also, this characteristic corresponds to the stability of the driving style. We analysed the smoothness of each function as the number of sign changes in the approximate first derivative: $d = \Delta\sigma/\Delta t$. The results were as follows PD controller - 218 changes, PPD controller - 98 changes. This makes the PPD controller more stable than the PD.

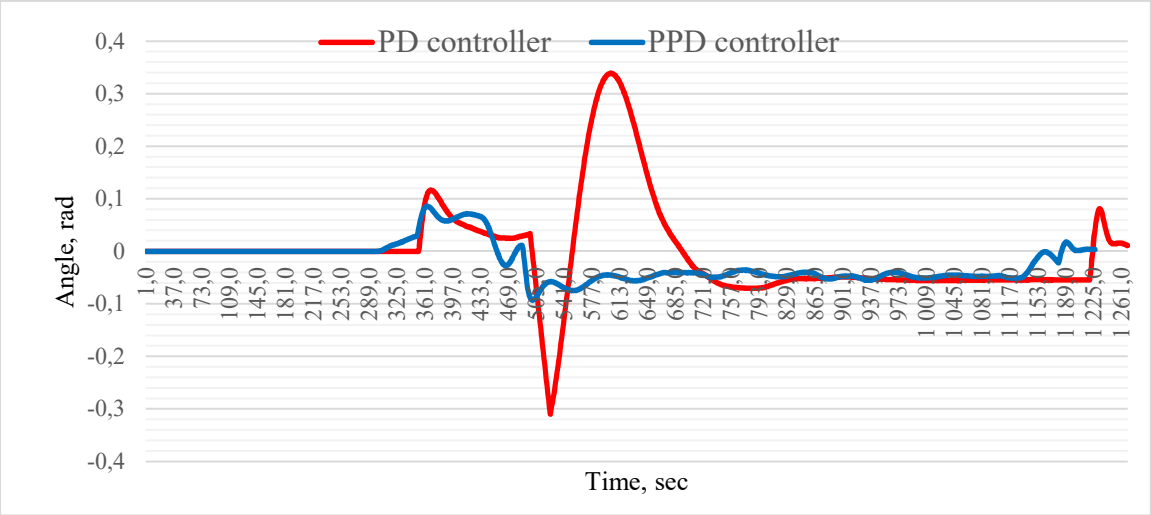



***Fig- 288*** Dynamics of the steering angle for PD (red curve) controller and PPD (blue curve) controller. Environment conditions: μ = 0.3, 0.95$V_{cr}$

## 4.2.2. Features obtained

It should be noted that the trajectory obtained as a result of the experiments is very remarkable not only from a "quantitative" point of view, but also from a "qualitative" one. The fact is that the behaviour of this trajectory differs fundamentally from the behaviour of

the trajectory of the classical method in around a turn. If the first of the methods considered by us suffered at the time of the turn, first due to a understeering as a result of a delay in assessing the situation and delay of the steering wheel command transmission, and then from a oversteering achieved as a result of attempts to return to the lane as quickly as possible (which is reasonable, because ensures the car's safety), which, as can be seen in Figure 27 and others, led to a deviation from turning from its outside, the new method is completely devoid of this drawback. On the contrary, anticipating an imminent turnaround, the car began it in advance, both (i) reducing the total deviation and (ii) making it from the potentially safer "inside" side of the turn. In the following paragraphs, we present some considerations in favour of the advantages of the new trajectory.

#### 4.2.2.1. Time needed to return on the lane

In conditions of intensive traffic flows moving at high speeds, the correct and clear interaction of vehicle drivers is of great importance. One of the main measures to ensure the safety of traffic flow is the implementation by all drivers of traffic rules regarding navigating and maneuvering. The exact location of vehicles within the width of their row and the exclusion of sharp turns has a big impact on the ability to increase vehicle speeds and ensure traffic safety. In the conditions of our single car on the track simulation, this means observing the car's trajectory along the line of the middle of the lane with the smallest and shortest deviations.

One of the most difficult and dangerous maneuvers is the overtaking maneuver. The most common form of overtaking is overtaking with leaving the row and returning to the same row, which is comparable to a double turn on our track. Overtaking from the point of view of the stability of the car is dangerous, because when it is performed, the car twice describes a curve with a small radius and a center of rotation, located either on the left or on the right. Small

radius and increased vehicle speeds during overtaking increase centrifugal force. Roughnesses and inclines along the road with high centrifugal force can cause the car skid and crash. Navigation and maneuvering rules recommend to do overtaking maneuver in the shortest possible time to keep the car safe.

And the first of the features of the trajectory provided by the PPD controller is a faster return to the middle of the lane. This is quite noticeable in the images of this trajectory due to the shorter blue curve, especially during the second, right turn. As shown in Table 11, for each case under consideration, the time required to return to the desired path decreased by 4–11%, which corresponds to 10–20 meters of movement at a speed from Table 6 (close to critical speed for stability) and can be crucial for safety in case of avoidance of collision with obstacles.

***Table- 7*** Time needed to return to the middle of the lane from deviation, sec

| #Road Condition | Overall Friction Coefficient μ | Speed is 0.85 of the critical one | | Speed is 0.9 of the critical one | | Speed is 0.95 of the critical one | |
|---|---|---|---|---|---|---|---|
| | | PD | PPD | PD | PPD | PD | PPD |
| 1 | 0.5 | 14.72 | 13.71 | 14.56 | 13.57 | 14.44 | 13.35 |
| 2 | 0.3 | 18.47 | 17.59 | 18.3 | 17.89 | 18.19 | 17.1 |
| 3 | 0.1 | 37.51 | 34.58 | 35.59 | 32.17 | 34.31 | 30.48 |

These times demonstrate how quickly the car returns from the position in which it turned out during the turn and trying to maintain its controllability. The parameter is directly related to the safety of the driver and all participants in the traffic flow. Larger values can correspond to both a noticeable distance to the desired trajectory and an uncomfortable orientation of the car. Both cases can cause the following complications, leading, for example, to turning the vehicle in the oncoming traffic lane.

### 4.2.2.2. Critical speed increasing. New trajectory features

Another feature of the proposed approach, which we noticed during the car trajectory analyzing and which increases its safety, is shown in Figure 29. In this figure, it can be seen that a car controlled by the PD controller loses control and crashes. This is understandable enough, since it follows at a speed exceeding the critical allowable by five percent.

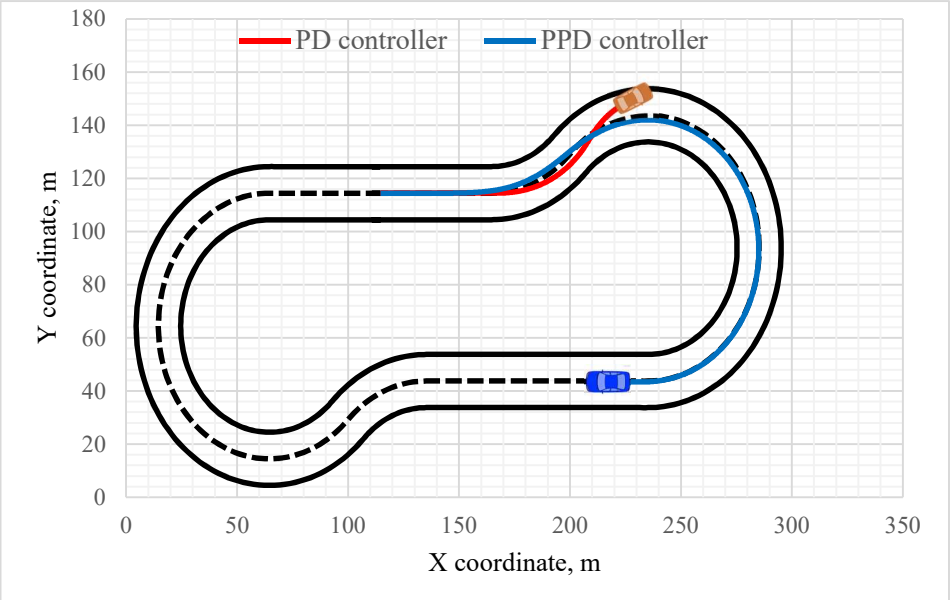



***Fig- 29*** Trajectories of the car exceeded the critical speed for friction coefficient $\mu$=0.1 with target speed equal to $\mathbf{1.05}\boldsymbol{V_{CR}}$. The blue curve correspond to the trajectory controlled by PPD controller, red – original PD controller.

An interesting feature here is the trajectory of the car under the control of the PPD controller- as we can see, with identical environmental parameters, when moving at a speed higher than critical, it does not lose control, but successfully completes the race without losing stability. Judging by the trajectory of the car, it even still has some margin of stability. The reason for this is the method of calculating the critical speed - as shown in paragraph 2.2.2.4, equation (2), Table 6, the critical speed of the vehicle during a turn is proportional to $\sqrt{R}$, where $R$ is the radius of the turn curve (see details in Figure 25). The predicting model begins to turn earlier, which leads to smoothing and an increasing the $R$ and, as a result, to a proportional increasing in the critical speed, a faster and safer driving style. From the point of view of the objective quality function, this effect leads to a decrease in the parameter characterizing the

instability of the car - the second addition in equation (4). This is confirmed by the data from Table 11.

### 4.2.2.3. Safe distance between car and obstacle

The above-mentioned manoeuvring rules to reduce the danger also require avoiding sharp turns, striving to make the line of departure for overtaking and returning to the row after overtaking as uniform and smooth as possible, which is quite achieved, as we see with our control model (Figure 30). Also the second property of the model found is the increased distance to a potential obstacle during a turn, that is, the moment of minimum vehicle stability.

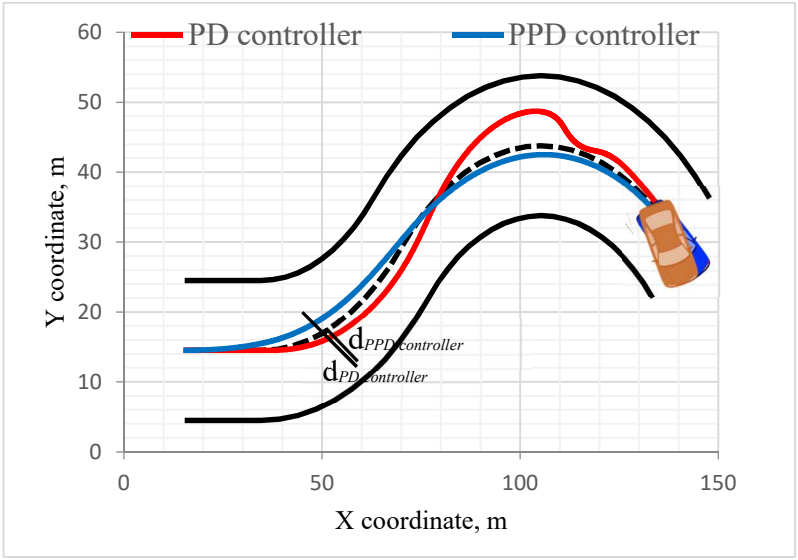



***Fig- 30*** (i) Smooth, uniform and short PPD controller trajectory,

(ii) distance between car and obstacle for both model ($d_{PD\_controller}$ and $d_{PPD\_controller}$), μ 0.1, 0.95 $V_{cr}$

The model that includes a prediction part makes the car begin to turn earlier. This not only makes the trajectory smoother and increases the radius of the turn, but also allows to avoid collision with the potential cause of the turn - a turn of the road or an unexpected obstacle.

According to Table 12, the average distance to the obstacle increased by 35%, which is also a safety parameter.

The numbers given in the Table 12, as well as the value of the estimation quality function indicate the deviation of the results obtained by applying each method from the desired ones, i.e deviation the car trajectory from the ideal path along the center of the lane.

***Table- 8*** Distance to the obstacle for each parameters combination, m

| #Road Condition | Overall Friction Coefficient μ | Speed is 0.85 of the critical one | | Speed is 0.9 of the critical one | | Speed is 0.95 of the critical one | |
|---|---|---|---|---|---|---|---|
| | | PD | PPD | PD | PPD | PD | PPD |
| 1 | 0.5 | 8.88 | 12.14 | 8.73 | 11.81 | 8.34 | 11.4 |
| 2 | 0.3 | 8.09 | 10.81 | 7.93 | 10.79 | 7.91 | 10.74 |
| 3 | 0.1 | 8.03 | 10.87 | 7.89 | 10.8 | 7.86 | 10.07 |

The movement along this trajectory without deviations caused by instability is the desired result and the deviation on it is equal to zero. In other words, these numbers can be interpreted as the amount of method errors.

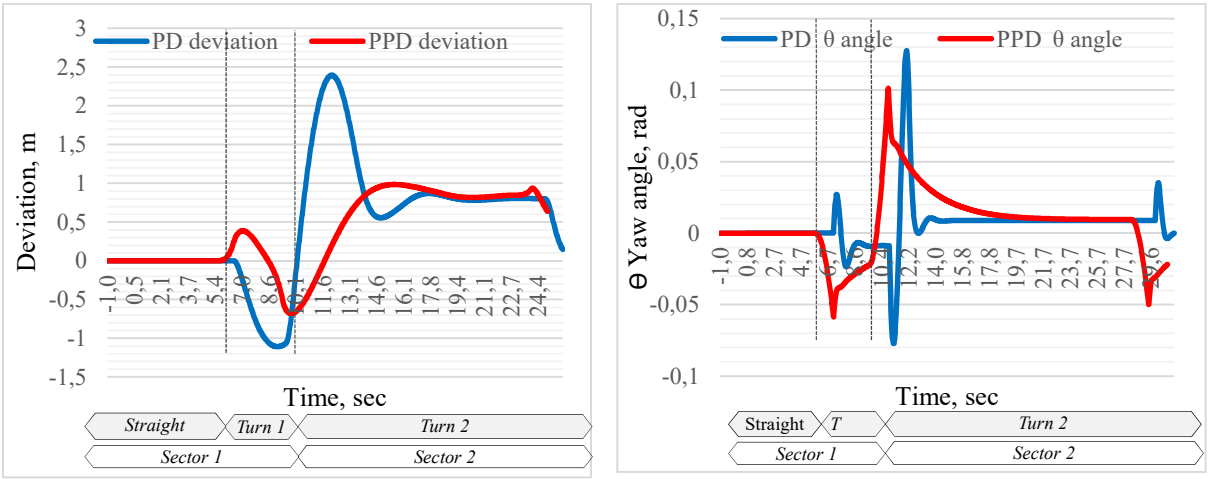



***Fig- 31*** Distance error (left picture) and yaw angle error (right picture) with $0.95V_{cr}$ and 0.3μ.

In Figure 31, you can see the dynamically changing behaviour of the deviation and yaw angle errors for both methods in the path segment of interest to us (during the turn). As a result of a visual analysis of the data presented, it is clear that in this sense PPD controller works better than PD controller and has the smallest of both amplitude changes, which ensures its stability and vehicle control.

## 4.3. Discussion

Despite the promising results of this method, there are several considerations that we believe deserve to be mentioned in connection with this method and its limitations and features. In addition, this section we provide an additional numerical analysis of the results obtained taking into account the values of the estimated quality function.

### 4.3.1. Prediction time distance

Another task that was announced, but has not yet been considered, is the search for the value of the optimal prediction time. In other words, it is the problem of finding such a parameter for equations (14) -(17) in which the estimation function of the model *F* returns the best result. In each solution obtained and demonstrated in the "Results" section, this parameter was selected manually, as a result of some enumeration of values.

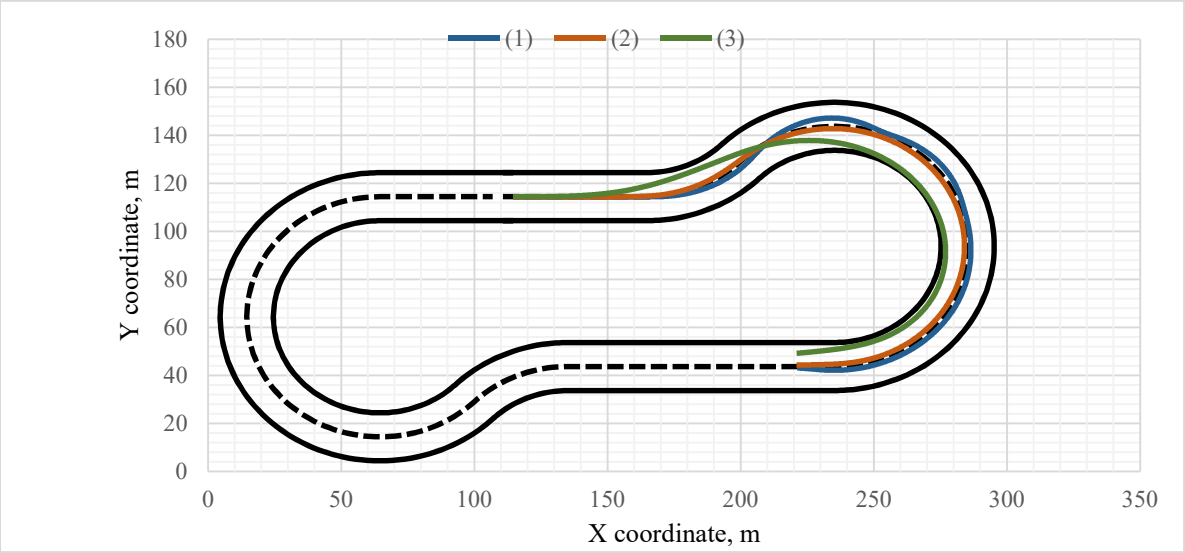



***Fig- 32*** Different types of the car trajectory behavior depending on the prediction time. From (1) to (3) this duration becomes longer

We allowed ourselves to stop and take the current number of *t* parameter as a solution if it turned out to be (i) already better than the PD controller solution and (ii) better than all the previous ones tried during the search enumeration process. Certainly, this could lead to the search for only a locally better solution, but since we were interested in a qualitative comparison of the new approach with the original one, even a local (not the best) solution for the new approach satisfied us. However, the question itself, what is the best prediction time is quite intriguing and deserves some additional study. Firstly, to search for it, it is necessary to take into account the fact that the range of possible solutions cannot be very large - depending on the configuration, the range of solutions does not exceed [0.8 ... 1.8] seconds.

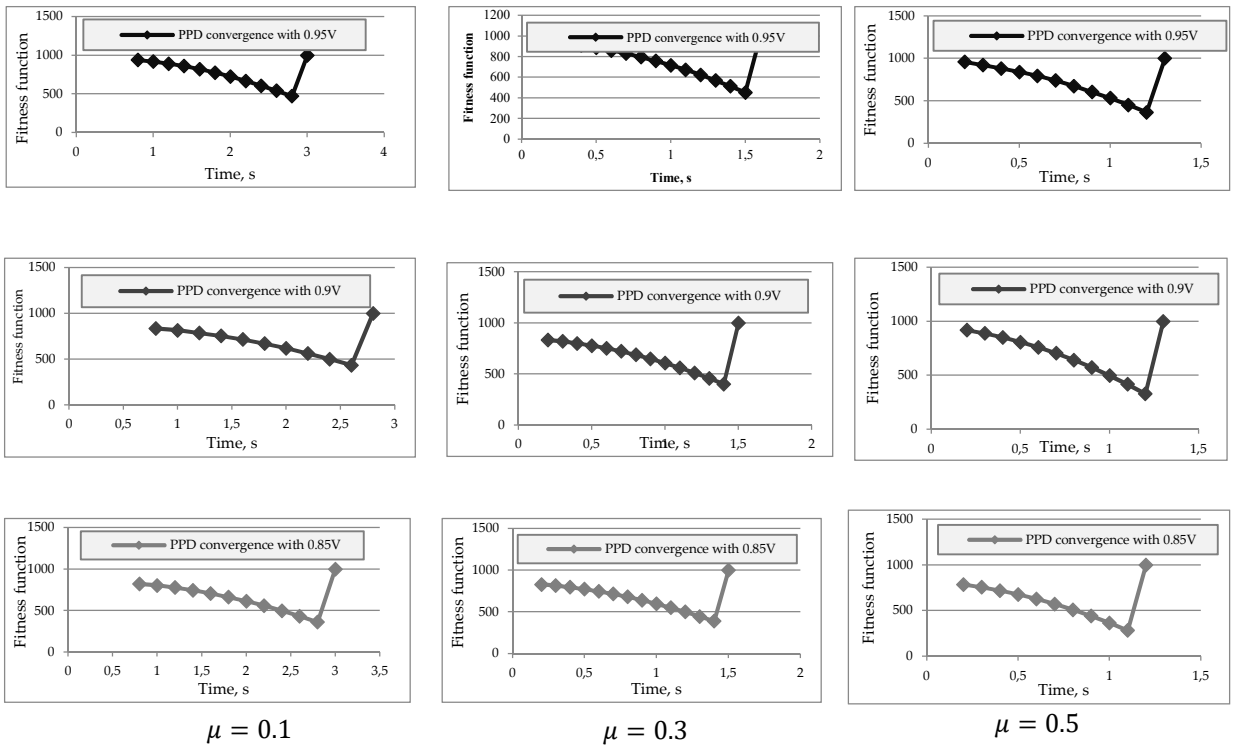



***Fig- 293*** Target quality function convergence of the PPD controllers under the different speed levels and friction values (μ=0.1, 0.3, 0.5) depends on time prediction.

Too short a prediction time does not allow to achieve the effect described in Section 4.2.2, and leads to the type of trajectory (1) in Figure 32. In contrast, a too long period provides changes in the trajectory that make it too far from the desired one, as shown in trajectories (3), Figure 32. Therefore, we ran a series of experiments enumerating the value of the prediction

parameter in the area of interest. Figure 33 shows the dynamics of the enumeration process for each of the conditions of the race. We began the search with a short prediction time provided by the trajectory (1), and gradually increased the time until the result first corresponded to the trajectory (3) and then became completely unacceptable, i.e. leading to a car accident. (in the context of calculations, in this case, the quality estimation function becomes 1000 as seen in Figure 33). The validity of our calculations is confirmed by the fact that the optimal values of the forecast time cannot be less than those with which we start, because they will turn into the results of the PD controller, and they can also be not more than those that lead to a car accident already due to its remoteness from reality. We also note a curious effect, that an increase in the target velocity along with a decrease in the friction coefficient increases the optimal predicting time. In other words, in less stable system conditions, the controller needs more extensive prediction in order to adjust the vehicle's trajectory as early as possible.

### 4.3.2. Special shape of the turning trajectory

Another feature of the obtained trajectory is related to its shape. Earlier, in section 2.2.2.2 devoted to the description of the characteristics of the track, we mentioned that we chose our test track as one of the most difficult to race. Indeed, it consists of a straight segment, which is replaced by a sharp turn, which in turn passes sharply into another turn. This configuration of the road is extremely uncomfortable for the driver and a big challenge for our controllers. The main part of the discomfort lies in the fact that usually during the first part of the route — the straight line — the driver accelerates and starts cornering at maximum speed, where he loses control stability especially on the slippery road. In the real world, they try to avoid this form of roads by connecting straight and turning segments during construction with a special

spiral shape segment of the path that allows you to gain a turn gradually, and whose radius of curvature decreases in proportion to the distance traveled. Such spiral passages were originally introduced on railways for safety reasons. They have also been implemented on motorways in recent years. The mathematical form of the spiral differs in different projects [36]. One of the common forms is the Euler spiral or clothoids [37]. In India, the usual transition curve is a third-order hyperbole, and in Germany, autobahns are designed as a continuous series of linked clothoids without tangential sections or circular curves [38].

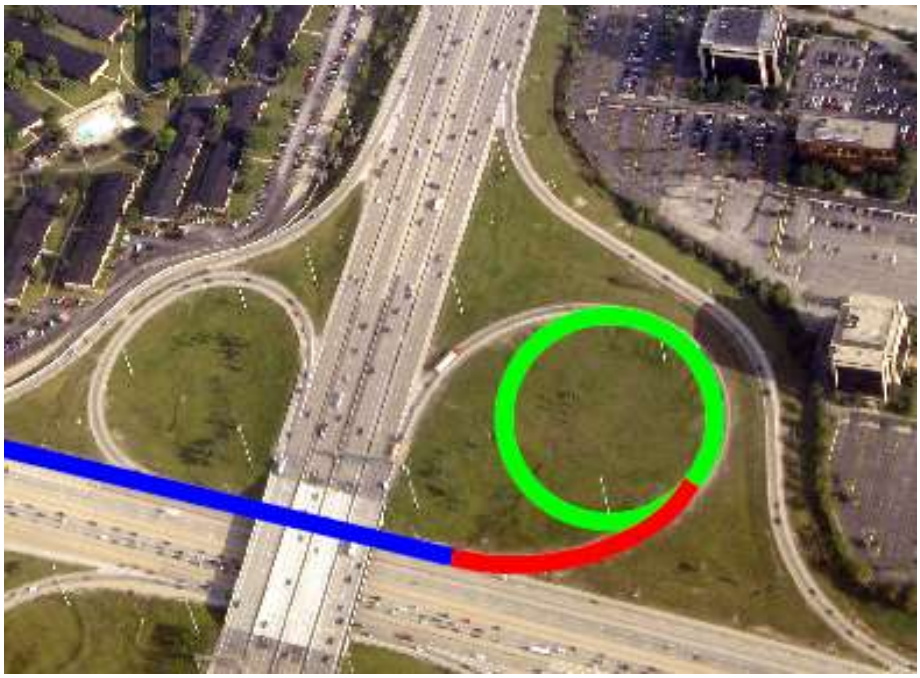


***Fig- 34*** Rotation curve. The transition between straight part (blue) and circular part (green) is spiral curve (red). Picture is taken from "https://cifrasyteclas.com/clotoide-la-curva-que-vela-por-tu-seguridad-en-carreteras-y-ferrocarriles/

In the Figure 34 - one of such sections of the road is visible. A spiral curve is a geometric element that can be added to a regular curve and provides a gradual transition (the red part in Figure 34) from moving in a straight line (blue part in Figure 34) to moving in a circle (green part in Figure 34).

A clothoid is uniquely defined as:

- Coordinates and heading from which to start: ($x_0$, $y_0$, $\theta_0$)
- Its length $L$
- Its linear curvature function, which is determined by two coefficients ($k_0$, $k_1$)

With these five values vector ($x_0$, $y_0$, $\theta_0$, $L$, $k_0$, $k_1$) we can estimate the position and direction of the clothoid ($x(s)$, $y(s)$, $\theta(s)$) at any point s in the region [$0$; $L$]. We can do this by solving the following equations (18) – (20):

$$x'(s) = \cos\theta(s) \quad (18)$$

$$y'(s) = \sin\theta(s) \quad (19)$$

$$\theta'(s) = k_0 + k_1 s \quad (20)$$

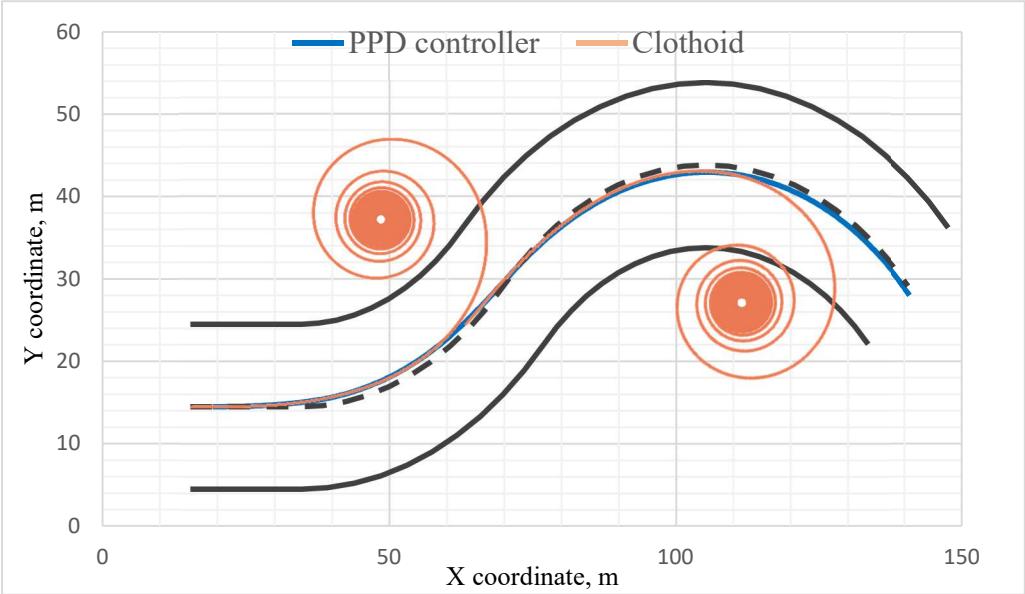



***Fig- 305*** Trajectory of the first left and right turns combined with clothoid spiral. Both turns demonstrate gradually increasing turning radius.

Noticing the visual similarity of the trajectory of the car under the control of the controller's PPD with the clothoid, we combined them in one section. The clothoid showed in Figure 35 was built by solving the above system of differential equations wrote below with *(0, 0, 0, 100, 0, 0.1)* parameters and turned around center.

### 4.3.3. Performance and Validation

Regarding the quantitative assessment of the quality of the trajectories presented in the "Results" paragraph, the values of the estimation function are presented in Table 13 below.

***Table- 9*** Target quality function of steering controllers for each parameters combination

| #Road Condition | Overall Friction oefficient μ | Speed is 0.85 of the critical one | | Speed is 0.9 of the critical one | | Speed is 0.95 of the critical one | |
|---|---|---|---|---|---|---|---|
| | | PD | PPD | PD | PPD | PD | PPD |
| 1 | 0.5 | 685 | 298 | 711 | 334 | 843 | 364 |
| 2 | 0.3 | 1693 | 383 | 1801 | 417 | 1854 | 458 |
| 3 | 0.1 | 1659 | 408 | 1717 | 432 | 1782 | 471 |

As can be seen from the results shown in Table 13, the PPD controller is superior to PD in terms of the estimation quality function in each of the considered conditions. Thus, the new controller has a characteristic path close to the desired path, but with a large turning radius and stability. It is appropriate here to recall the structure of the target quality function — it consists of two terms, the first of which characterizes the deviation of the vehicle's trajectory from the desired one, and the second the deviation from the desired direction. Thus, most likely, an increase in the stability of the model qualitatively means a decrease in the second term. In order to make sure of this, we analyze the contents of Table 14.

However, the data from Table 14 above demonstrate even more than proposed only stability parameter improvement, but also deviation parameter decreasing. This means that the features of the new model affected improving each additive component of the estimation quality function without reducing the quality of the second of them, while it is common to improve one at the expense of the other, the so-called "zero amount sum". Those the closer the vehicle is to the center of the lane, the stronger forces acting on it during the turn and the higher the value of the instability parameter and contrary- stable control requires a slow

change in the yaw angle of the car, which provokes a lag in the trajectory and deviation.

***Table- 10*** Target quality function of steering controllers *F* for each parameters combination split by addends corresponding to deviation from the center of the lane and lateral acceleration respectively.

| #Road Condition | Overall Friction Coefficient μ | Speed is 0.85 of the critical one speed | | Speed is 0.9 of the critical one speed | | Speed is 0.95 of the critical one speed | |
|---|---|---|---|---|---|---|---|
| | | PD | PPD | PD | PPD | PD | PPD |
| 1 | 0.5 | 239+446 | 79+219 | 255+456 | 103+231 | 288+555 | 110+254 |
| 2 | 0.3 | 779+914 | 186+196 | 823+978 | 215+201 | 870+984 | 252+206 |
| 3 | 0.1 | 1241+418 | 346+62 | 1291+426 | 367+65 | 1305+477 | 405+66 |

However, in this case, the new model demonstrated improvements in both parameters, which indicates a qualitative improvement in the model, in contrast to the simple parameter tuning in the previous Chapter 3.

### 4.3.4. Future Work and Summary

The proposed predictive PD controller (PPD controller) overcomes the main drawback of PD controllers, namely, the reactivity of their control behaviour.

In our approach, supporting the computational efforts of the controller at the same level as in PD controllers, i.e. avoiding (i) complex calculations and (ii) adding new variables, the PPD controller shows the best control quality both by increasing its accuracy and improving its stability.

As already mentioned in paragraph 4.2.1, we predicted only one of the components of the PD controller, although it seems quite natural to also predict the yaw angle. Earlier, we also explained that the reason for this decision is the sequencing and atomicity of the changes, which facilitate the development and analysis. However, in terms of the discussion, it seems appropriate to us to mention one more argument in favor of adopting such changes, namely,

the desire to achieve the professional human driver behaviour to follow. Unlike the PD controller, people use prediction of the future position of the car to control their natively without artificially setting such a goal. However, according to the studies we know [39] of the human brain and the properties of its perception of the environment, when dealing with a lot of information, it suffers from cognitive pressure. In other words, while driving at high speed, a person is forced to process a constantly and concentrated large amount of rapidly changing information (about other cars, approaching bends and road junctions, traffic lights, people crossing the road, etc.) which leads, in addition to fatigue, to narrowing the field of view. In this state, the driver loses the ability to predict the yaw angle, managing only the position prediction. This, among other things, opens up potential opportunities for further extension of the model, which has not so hard perceptual limitations and provokes further research in the development of possibly a more accurate model that predicts both parameters.

Also according to studies [40], the best and smoothest transition curve to be used as a path section is a clothoid. The fact that the PPD controller we developed came to the same conclusion with its results, as well as their comparison with the results of another controller once again indirectly indicates the effectiveness of the new controller.

However, we still have some doubts about this approach, namely that the servo control of the PD may be inadequate in such conditions (slippery road surface and high speed), due to the high complexity of the car dynamics. From general considerations, it seems reasonable that an adequate model should include some another variables in addition to lateral and angular deviations from the desired trajectory. Also, to describe and take into account the dynamics of the car on a slippery road, additional data describing speed and amount of change of the car state may be needed. Such complexity may not be sufficiently expressed by the relatively simple PD servo control model and even its predictive version, which do not track the internality of changes in acceleration, angle and moment, speed and other parameters.

Moreover, in case of the PD servo control model is unacceptable as a steering model in such slippery conditions, it is not clear what complexity and design the alternative model should have and how to develop it - this is also an open question. Such problem statement is typical characteristic of fuzzy logic problems.

# Chapter 5

# Relaxed structure of PD controller analytical model, developed heuristically via the GP – RM-GP

## 5.1 Materials and Methods

A rough implementation of the scenario of car movement on a slippery road based on the PID control algorithm revealed the imperfections of the original approach - to ensure acceptable transport stability due to the insufficient adequacy of the model based on the assumption of a linear nature of the dependence of the input and output variables of the control process. Despite the good results of the modified method, we are interested in the question of non-linearity and non-stationarity of the process, the complexity of constructing and identifying its model by traditional methods based, for example, on differential equations, as well as the fact that the driver “feels” the car much faster and many times faster makes the start of movement, rather than than the PID or PD controller. These considerations led us to choose the mathematical apparatus of fuzzy logic for constructing an autonomous car control system on a slippery road. This technology allowed us to formalize the non-verbal experience of the driver and implement it in a fuzzy steering wheel control controller.

### 5.1.1 Algorithm Summary and Components

To solve the above problems of canonical PD controllers, in which the control output signal is calculated as a weighted sum of control errors and their derivative, in our previous study we proposed a PD controller with a predictive component, although in the end we expressed doubts about the sufficiency of such a step. Thus, in this chapter we present the method we developed, which is based on the representation and additive structure of the PD controller, but does not have clear ideas about scaling coefficients.

Thus the controller we developed (i) has, in a sense, the relaxed structure of its analytical model (RM), developed heuristically through the GP. We designate it as the GP-RM controller.

The rationale for the adoption of GP in particular and fuzzy logic as a whole is based on our desire to cover in the model the dynamics of a vehicle of an unstable car under slippery road conditions, and even take into account its potential nonlinearity. We provide the GP with the opportunity to develop a new approach to driving that would effectively counteract its directional instability through long-term evolution and selection of options. Thus, SAF is automatically designed using a computer system by simulating evolution and in a manner similar to the evolution of species in nature described in sections 2.4.2 and Table 7.

The main characteristics of the adopted method - genetic representation, set of functions, set of terminals, suitability assessment and genetic operations are also presented in paragraph 2.4.3 onwards.

### 5.1.2 Algorithm Convergence

We set up several series of experiments. For each combination of the conditions from Table 6, its own SAF was developed. To obtain one successful SAF, 18 to 25 independent runs of the evolutionary process were required.

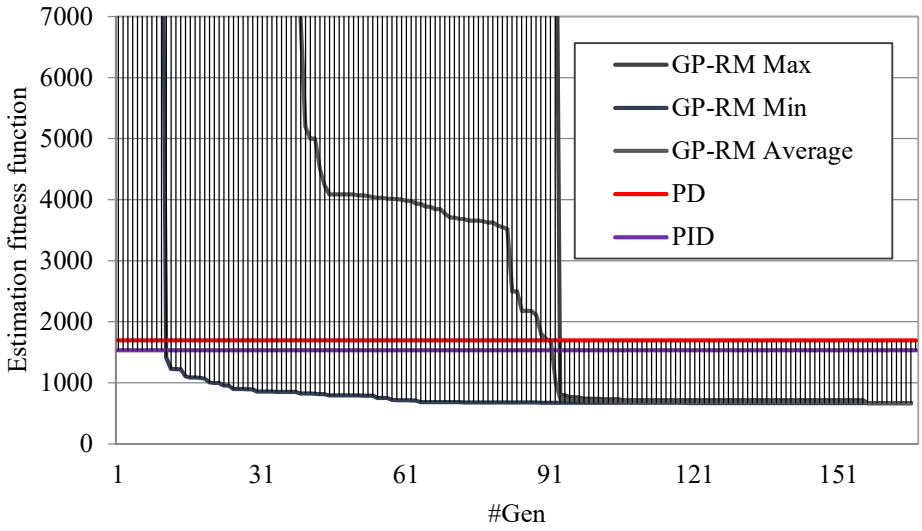



***Fig- 316-1*** Fitness convergence characteristics in the process of evolution for more 20 independent runs of GP evolving the SAF of GP-RM controller for friction coefficient $\mu$=0.3 , $0.85V_{cr}(a)$

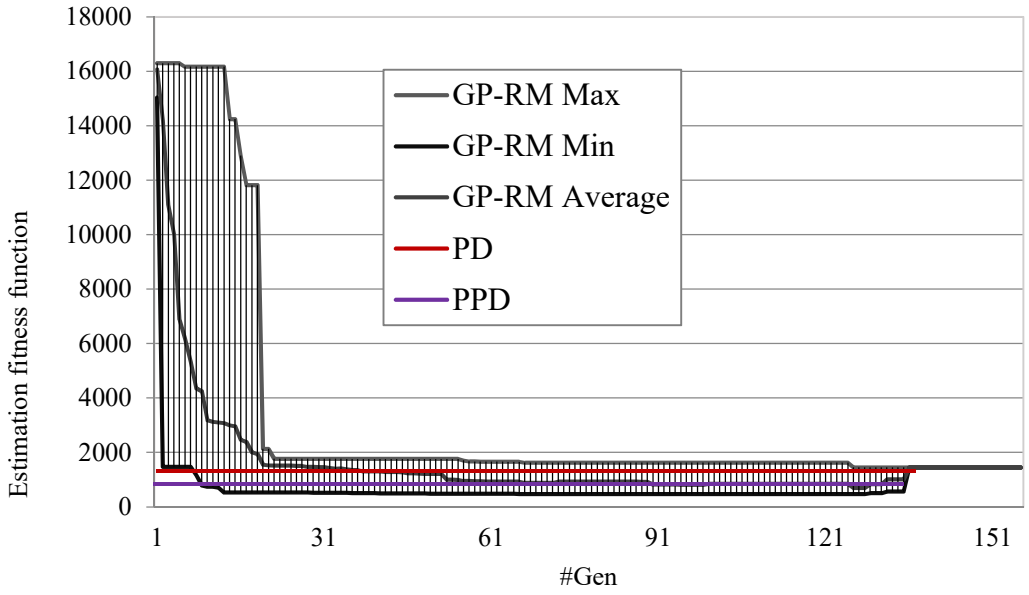
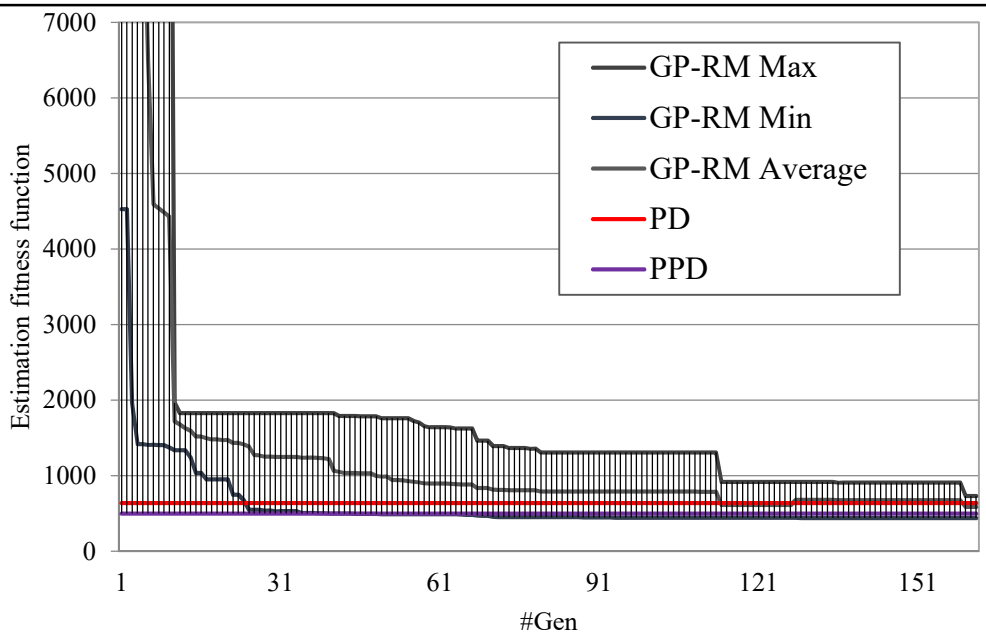



***Fig- 326-2*** Fitness convergence characteristics in the process of evolution for more 20 independent runs of GP evolving the SAF of GP-RM controller for friction coefficient $\mu$=0.5, 0.85$V_{cr}$ (b)




$\mu$=0.8, 0.85$V_{cr}$ (c)

***Fig- 336-3*** Fitness convergence characteristics in the process of evolution for more 20 independent runs of GP evolving the SAF of GP-RM controller for friction coefficients μ=0.3 (a), μ=0.5 (b), μ=0.8 (c) respectively. The bold curves correspond to the average, minimum and maximum values in each generation. The best fitness of the PD (obtained via a complete enumeration of the values of their parameters) is shown as red horizontal line.

Despite the fact that it took about 1 second to simulate a single car race without a visual part (For the consumption of computational resources by the evolutionary process in various graphical modes, see paragraph 2.2.1), such launches took some time, taking into account that there were 200 individuals in each run and an average of about 180 generations. Thus, obtaining each SAF took about 10 hours of evolution process without taking into account delays caused by the system load. The results as a set of convergence dynamics are presented in Figure 36. You may notice that in all the given road conditions, the suitability of the best developed via GP SAF converges to values that are better (i.e. lower) than the values of the best tuned PD controller. For reasons of time saving, we conducted an experiment for only three values of the coefficient of friction μ, including its smallest and most difficult value for the controller (μ=0.3) and at the single criticality level of speeds ($0.85V_{cr}$), wishing first of all to understand not detailed, but qualitative characteristics of the designed SAFs.

## 5.2 Discussion

The best suitability of the proposed controllers constructed via GP are compared with the PD controllers in Table 15 below:

***Table- 11*** SAF produced by GP and PD controller's SAF quality estimation function values *F*

| | | **Speed is 0.85 of the critical one** | | | **Speed is 0.9 of the critical one** | | | **Speed is 0.95 of the critical one** | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |

The numerical results of estimating the effectiveness of SAFs in Table 15 show that the controller, developed through the evolution, has higher suitability on the test track in all tested race conditions. You can also notice that the steering quality for the best developed

GP-RM controller is even better than that of PID on icy roads (μ = 0.3). On the contrary, on dirty, rainy roads (μ = 0.8) this difference is not so noticeable (526 against 442).

Also, for the correct comparison and estimation of methods, we made similar measurements of the safety parameters mentioned in the previous chapter. The experimental results are presented below in Tables 15 and 16. According to the information in Table 16, the new method is comparable with the best results of the predictive PPD method. A curious detail is the fact that with a higher coefficient of friction of 0.5, the new method shows slightly less stable results (13.71 seconds before returning to the line of the ideal path of the controller PPD against 14.18 seconds of the controller developed using genetic programming).

***Table- 12*** Time needed to return to the middle of the lane from deviation, sec

| #Road Condition | Overall Friction Coefficient μ | Speed is 0.85 of the critical one | | | Speed is 0.9 of the critical one | | | Speed is 0.95 of the critical one | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | PD | PPD | GP-RMEP | PD | PPD | GP-RMEP | PD | PPD | GP-RMEP |
| 1 | 0.5 | 14.72 | 13.71 | 14.18 | 14.56 | 13.57 | 12.81 | 14.44 | 13.35 | 13.52 |
| 2 | 0.3 | 18.47 | 17.59 | 17.18 | 18.3 | 17.89 | 17.19 | 18.19 | 17.1 | 16.9 |
| 3 | 0.1 | 37.51 | 34.58 | 33.14 | 35.59 | 32.17 | 31.42 | 34.31 | 30.48 | 30.16 |

With an increase in the complexity of the environmental conditions and cross-country ability of the route — a decrease in the coefficient of friction, it becomes noticeable that the new model outperforms its predecessor. The second safety parameter - the distance to obstacles also demonstrates (detailed result in Table 17) the overwhelming superiority of the new model in all road conditions. Such results look promising, so the next step in their analysis has naturally been to assess their suitability in terms of computational cost. Earlier, when we designed the PPD model, we said that one of its advantages over analogues (like PCM) is its simplicity and processing speed. It should also be clarified here that we are only interested in

the calculation time spent by the final SAF formula, and not the time spent on its creation, which in our case took several hours. Due to the delays in transmitting commands inside the

car in the simulator, it is possible to judge whether the formula's complexity caused the controller to delay processing of the commands by the vehicle's path and the absence of oscillations, jerks, and other artifacts in it. Thus, despite the complexity of the formula, the simulator of the selected car still copes with the computational load. Looking at the structure of the obtained decisions of the SAF, it is difficult to understand which part is responsible for which actions of the car, since in most cases the decisions obtained using the GP are too complex to be understood by humans.

***Table- 13*** Distance between the edge of the car and the obstacle

| #Road Condition | Overall Friction μ | Speed is 0.85 of the critical one | | | Speed is 0.9 of the critical one | | | Speed is 0.95 of the critical one | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | PD | PPD | GP-RMEP | PD | PPD | GP-RMEP | PD | PPD | GP-RMEP |
| 1 | 0.5 | 8.88 | 12.14 | 12.33 | 8.73 | 11.81 | 12.07 | 8.34 | 11.4 | 11.84 |
| 2 | 0.3 | 8.09 | 10.81 | 11.6 | 7.93 | 10.79 | 11.38 | 7.91 | 10.74 | 11.14 |
| 3 | 0.1 | 8.03 | 10.87 | 11.23 | 7.89 | 10.8 | 11.16 | 7.86 | 10.07 | 10.9 |

Moreover, it is often difficult to even determine whether the considered part of the equation is generally significant from the point of view of the contribution to the car's behaviour on the track or is it the so-called non-coding element of the genome, whose functions are connected only with the evolutionary process and are still not fully understood. Thus, any analysis of the developed SAF does not seem objectively possible and reasonable. This means that we can't explain exactly why and how those SAF works, and therefore cannot guarantee its future operation under any conditions, but only reasonably assume that the function "trained" in the

complex test conditions we have adopted should adequately respond to them changes, since it most likely contains multiple elements that are responsible for checking the conditions of movement and the reaction to its changes. However, there is also a hypothetical probability of a situation in which evolution, in order to minimize efforts, will create a SAF that will consist of elements encoding (but may not be identifiable in visual analysis) the constant movement of the car along an ideal trajectory without reactions to changes in external conditions. In fact, this is not controlled through the fitness function since it is quite difficult to come up with a check for such a situation and evolution would be completely pleased and satisfied to find such a SAF. To eliminate such solutions, it is enough to run them under other speed or weather conditions, i.e. this is approximately equivalent the track changing or testing it on a different track. However, despite the unreadable structure of the SAF, we can argue that SAF initially implements proportional-derivative (PD) control in a way in which both (i) direct proportional parameter values and (ii) their derivatives are included in its code.

After obtaining this results, we run an another set of experiments. They based on the same idea as RM-GP approach, but applied to the PID instead PD controller SAF. We assumed that the best results for both SAF methods should not differ in estimation driving quality. This assumption is based on the conclusions described in work [41] that the integral term may be obsolete in some nonlinear PID controllers. Actually, analysing the best steering equations obtained in the new series of experiments we found that the integral term included in the GP term set, which is also the lateral deviation integral, is often not included in the SAF equation, and the estimation function has numerical results similar to those in the main series of experiments values, which is consistent with our assumptions.

### 5.2.1 Performance and Validation

From the point of view of evaluating the method, we are interested in the trajectory of the car as its deviation from the ideal trajectory. Unlike previous controllers, its position on the track

is more close to ideal and its path itself does not seem to be informative, as visual analysis does not allow to see a special difference between the models, especially on roads with more stable pavement (μ=0.8, μ=0.5). Thus, let us consider in Figure 35 the behavior of cars driven by the best GP produced controller for the rainy (μ=0.5) condition. As these figures show, the lateral deviation of the car driven by the best developed steering function model is much lower than that of the best servo control (PD) and PID solution, especially during a steady turn in left turn 1 (between 3 and 5 seconds of the trial period) and right turn 2 (between 5 and 10 seconds). In addition, during the transition between these two corners - about 10 seconds after the start of the test - the steering becomes much smoother and more stable. The difference in the dynamics of the steering angle (Figure37) and the deviation from the center (Figure 38) of the vehicle's lane was in one of the most difficult – rainy (μ = 0.5) - road conditions, which are given here.

#### 5.2.1.1 Future Work

Due to the method we have chosen, the obtained SAFs are complex for perception and understanding, redundant functions, overloaded operators and terminals. Of course, this affects the computational cost of the steering wheel control function, which increases

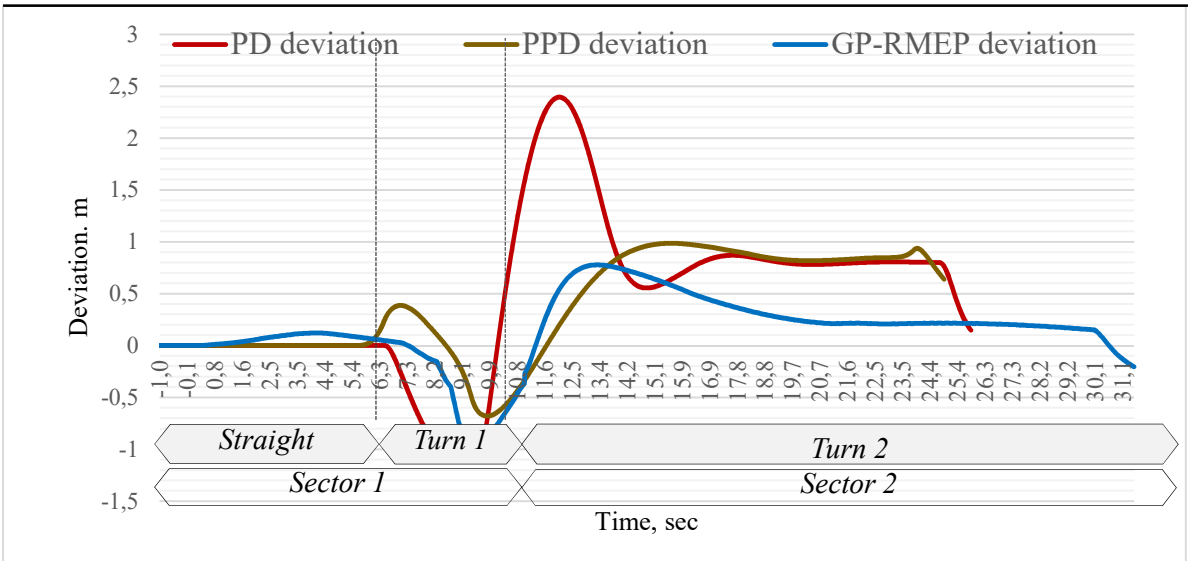



***Fig- 37*** The dynamics of the deviation from the center of the lane on the snowy ($\mu$ = 0.5) – road.

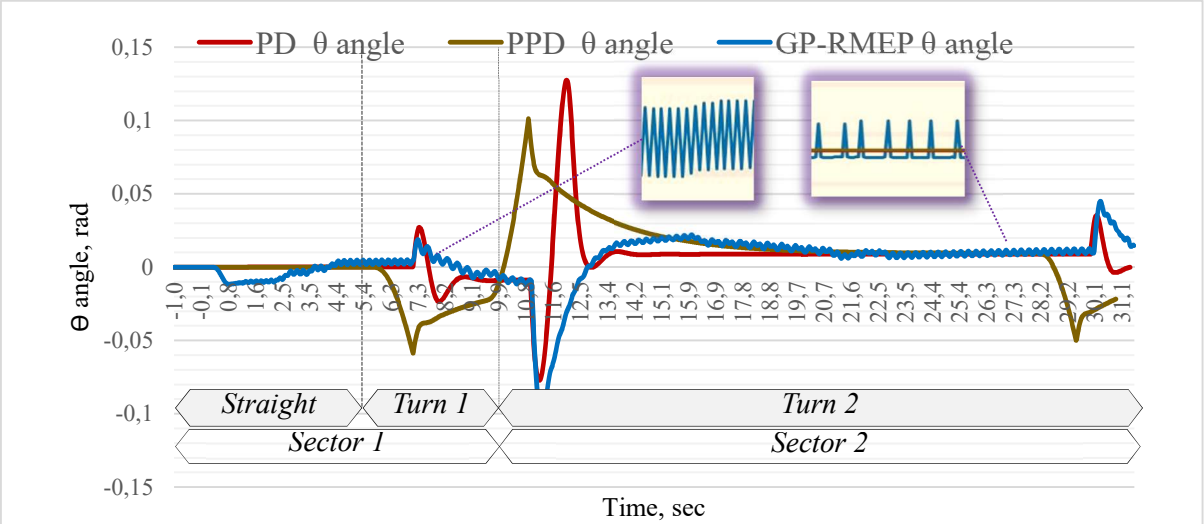



***Fig- 38*** The dynamics of the steering angle on the icy (μ = 0.5) – road

compared to a conventional PD and PID controller. Also, despite the general tendency to improve the quality and safety of driving (Figure 37), we observed a specific steering control effect that arose during the evolution - a very frequent (2-5Hz) steering direction change (Figure 38). The change is so frequent that it does not have time to have any noticeable effect on the trajectory of the car and is noticeable only during the tracking yaw angle. The reason for these oscillations can be equally the evolutionary find - a unique style that preserves the stability of driving, and the evolutionary "laziness" - in these experiments we did not impose restrictions and penalties on the evolution for additional turns during the race and it could maintain such an "redundant" driving style because it didn't see anything bad in it. However, the fact that the vast majority of successful SAFs and the vast majority of the best of them have this feature sounds quite promising for us and requires additional research.

Another consideration is the next logical step of our study - a complete rejection of the rigid structure of the known controllers in favour of evolutionary uncertainty and the ability to construct a new control model - an ideal artificial driver. The time of its evolution can be colossal compared to an ordinary human driver, practically we have an amazing opportunity to select a driver's model without the inhibitory limitations of the real world.

# Chapter 6

# Evaluation of the GP-RMEP controller with extended parameters obtained via GP

## 6.1 Materials and Methods

At the beginning of this chapter, devoted to the new GP method, it will be appropriate to recall that one of the main goals of our study was the complete automation of the driving process. In most modern models [42], used in practice, automatic control is combined with human control for emergency situations. Moreover, in the event of critical instability, control is transferred just to the human driver as the most reliable for resolving difficult situations. Such an approach, as we mentioned in Chapters 2 and 3, leads in practice to situations when an unprepared human driver finds himself in a difficult situation, often without the proper skills of extreme driving and an understanding of the context of the environment, the changes of which he did not closely monitor. Thus, in our current study, instead of focusing on maintaining an adequate cognitive load of human drivers while transferring control to them in difficult (slippery) road conditions, we intend to explore an alternative approach: to avoid transferring control to the driver at all. The objective for our work is that if the driver often struggles to take control of the car under normal driving conditions (and, according to the theory of risk homeostasis[43], drivers of cars equipped with sophisticated driving means are usually less focused on driving and excessive optimism [44]), it can be even more difficult for them to do this in difficult (for example, slippery) road conditions. Failure to transfer control implies that automatic driving is reliable enough even in difficult, slippery road conditions.

The purpose of our study is to study the possibility of developing a heuristically adequate control of a realistic simulated car in slippery conditions by simulating evolution using genetic programming (GP).

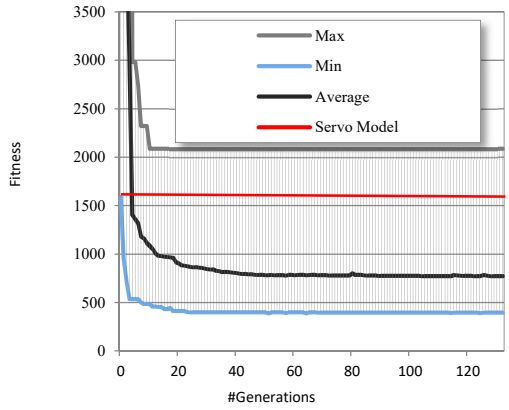
As we have shown in previous chapters, the canonical servo control model was not able to provide adequate driving on slippery (for example, wet, snowy or icy) roads. The extension of this model to proportional derivatives (PD), as well as proportional integral derivatives

(PID) of the steering controller and the controller with the prediction component (PPD) provided better vehicle handling and increased driver safety. Further modification of the model, its relaxed version with an arbitrary (and not additive, as in PD, PID and PDD controllers) - the structure of its analytical model, developed heuristically via GP, provides even more better vehicle controllability on slippery roads. In view of the results obtained, we assumed that servo control is not suitable in such conditions, because the model of the dynamics of a vehicle suffering from instability on a slippery road should be more loaded and complicated in that it includes additional variables - in addition to lateral and angular deviations from the desired trajectory, as in servo control - is related to the condition of the car than to a car moving along a normal, non-slippery road. Thus, we attribute the superiority of the developed relaxed model (GP-RM) to higher complexity - compared to what is assumed by the PD control - of the relationships between these variables in the physical model of an unstable sliding car.

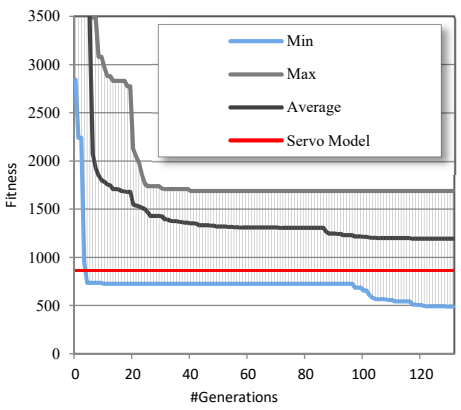
## 6.2 Experiments and Results

We expanded the number of the road conditions, supplementing them even more unstable (friction coefficient 0.1) and excluded the simplest ones cases (friction coefficient 0.8) from them due to its inconsistency for our research. To develop new SAFs with a completely unfixed structure and complexity, we used the GP and performed 20 independent evolutionary runs. In all runs, suitability constantly converges to the best (lower) values over 120 generations of GPs, after about 22,000 car tests for evolutionary run. The results of the suitability of the developed SAF best run and tuned servo control model (used as a reference in our study) are shown in Table 18 below.

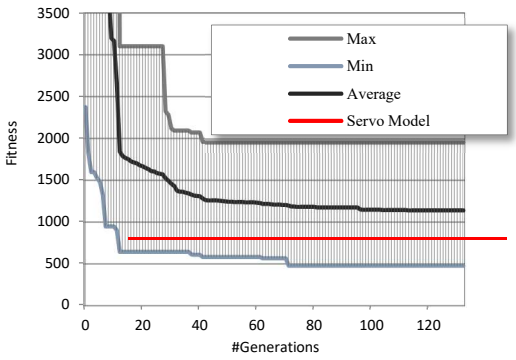
### 6.2.1 Algorithm Convergence



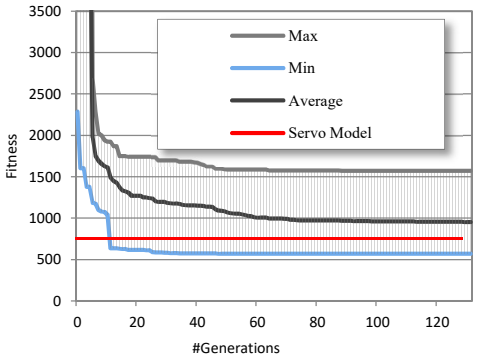

(a)
icy road conditions,
$\mu$=0.3



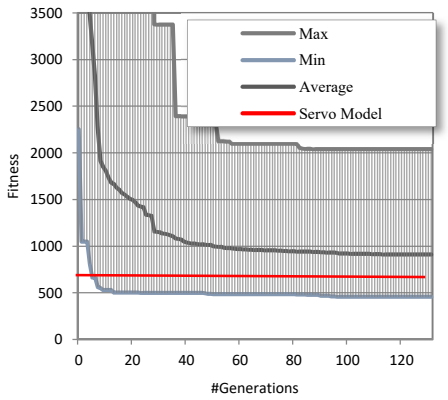

(b)
snowy road conditions,
$\mu$=0.4



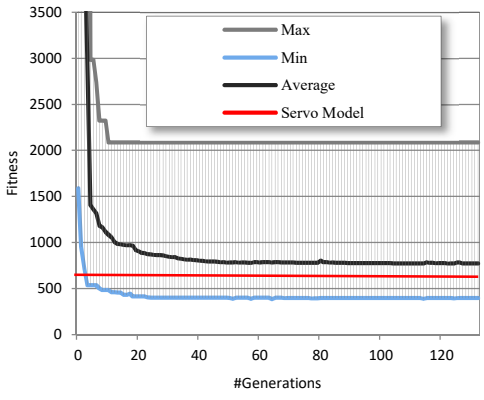

(c)
rainy road conditions,
$\mu$=0.5




(d)
rainy road conditions,
$\mu$=0.6




(e)
dry road conditions,
$\mu$=0.8




(f)
rainy road conditions,
$\mu$=1.0

***Fig- 39*** Fitness convergence characteristics of 20 independent runs of GP evolving optimal steering function for friction coefficient $\mu$=0.3 (a), $\mu$=0.4 (b), $\mu$=0.5 (c), $\mu$=0.6 (d), $\mu$=0.8 (e) and $\mu$=1.0 (f), respectively. The best fitness of the servo control model, obtained via a brute-force search is shown as a horizontal red line

The characteristics of the convergence of methods are presented above, from which it follows that a control model developed using evolution demonstrates better results compared to classical controllers with a fixed structure and complexity.

### 6.2.2 Efficiency comparison of the proposed controllers

***Table- 14*** Experimental Results on Evolution of SAF

| **Friction coefficient $\mu$** | **Fitness of best evolved SAF** | **Tuned servo-control model** $\delta = k_1 \times e + k_2 \times \theta \approx k_1 \times e + k^*_2 \times e'$ | |
|---|---|---|---|
| | | **Tuned values of $k_1$ and $k^*_2$** | **Fitness value** |
| 0.6 | **430** | **0.2472, 1.866** | **661** |
| 0.5 | **373** | **0.1864, 2.244** | **687** |
| 0.4 | **314** | **0.1135, 3.378** | **765** |
| 0.3 | **374** | **0.3322, 2.055** | **1693** |

Table 18 shows the numerical value of the advantages of the method developed using genetic programming over the best values provided by the PD controller with tuned parameters. Moreover, unlike the PD controller, whose control quality decreases with a decrease in the coefficient of friction, the evolutionary model does not demonstrate such a dependence on it.

### 6.2.3 Universal (robust, general) equations

Despite the high accuracy of the solutions obtained using GP, they have a whole list of serious drawbacks. The lack of generality is one of the most well-documented and significant such shortcomings[45] [46]. Indeed, we have no guarantees that a solution that works in specific conditions (at a fixed critical speed, on a specific shape of track and with a fixed coefficient of friction between tires and surface of the track and a fixed model of the car) and shows a good result will also be so or any effective in other conditions. We can only reasonably assume that the solution will probably work because of the evolution conditions

we made. This drawback imposes significant limitations in the areas of GP application related with people safety and hinders the applicability of GP to many real problems. Thus, one of our goal is certainly to develop a SAF that performs the car (almost) equally well in several suitability cases that meet various conditions, that is, a universal or robust SAF. Actually, since we created highly unpredictable and varied conditions for the evolution of the SAF, we

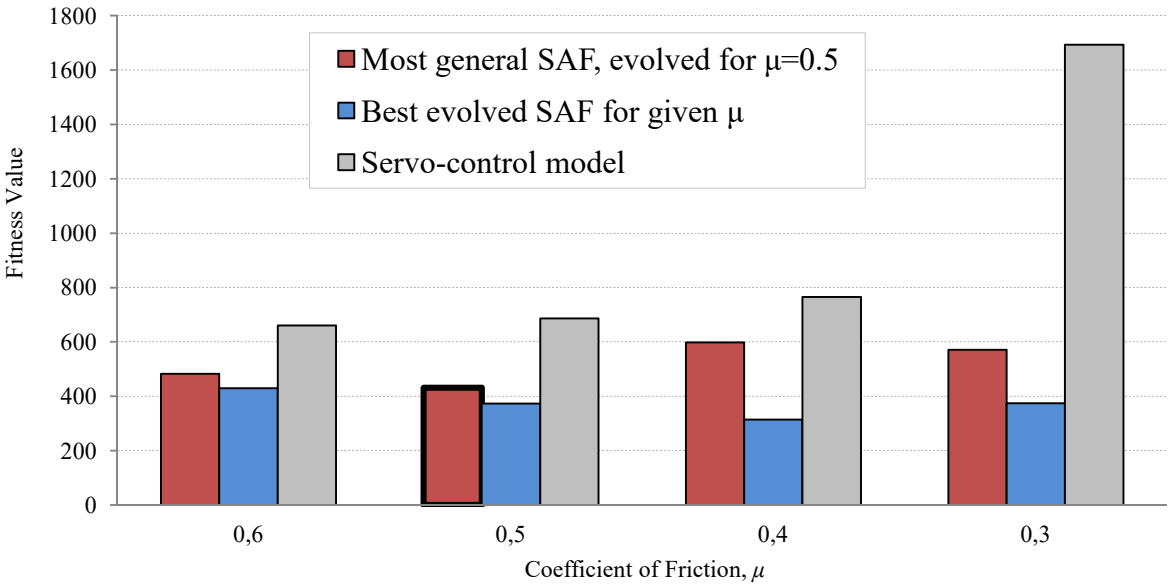



***Fig- 340*** Comparative performance of the most general SAF, evolved for friction coefficient μ=0.5 when tested on tracks with different friction coefficients

can assume with some degree of certainty that some of the received "successful" decisions take into account the dynamics of the car in some way and can be universal. All that remains in this case is to combine the pools of all successful SAFs obtained separately for each coefficient of friction - the degree of slip of the route, and test each of the SAFs for all conditions. The results of this procedure are presented in diagram on Figure 40.

Since the design strategy for the construction of the SAF for various conditions of critical speed and friction coefficients is no different, for visual convenience, we demonstrated the results of testing the SAF taken from a pool of the best solutions developed for conditions with a friction coefficient of 0.5 and a critical speed of $0.85V_{CR}$ on that same track with different friction coefficients. We chose μ equal to 0.5 due to the fact that the solutions in

these conditions are characterized by a qualitative deterioration in steering quality (it was with this coefficient that serious difficulties began to appear in classical models in our experiments). As shown in Figure 40, the suitability of such a solution for μ = 0.5 is 431 (worse than 373 best developed for the same friction coefficient, but still much better than 687 tuned servo model) and gradually decreases to 483 for μ =0.6 (compared to 430 of the best solution developed for the same coefficient of friction, and 661 of the servo model) and 571 for μ = 0.3 (compared to 374 of the best solution developed for the same coefficient of friction and 1693 servo models) .

## 6.3 Discussion

From the point of view of analysing the quality of the model, in this chapter we also give the dynamics of the angle of rotation of the steering wheel and the deviation of the trajectory from the desired car controlled by the robust SAF in Figure 41.

### 6.3.1 Performance and Validation

First of all, we note that the deviation of the obtained trajectory is even less than that of the previous relaxed via GP model. The car starts the test at the beginning of the initial straight line and slowly accelerating, it smoothly enters the left turn, deviating quite a bit (period 6.5 - 9.9 seconds, less than 0,1 m). Feeling a change in the curvature of the road, the car abruptly returns to the center of the lane (between 9.9 and 10 seconds), reducing the total time of deviation from the safe trajectory, and at the entrance to the right turn (10 -11.6 seconds), it is slightly left of the middle and slightly oriented right towards the turn (ϴ-angle deviation graph on the Figure 41). Unlike the previously considered reactive models, we do not see any delays in entering the turns along the trajectory while maintaining the same speed level. On the contrary, such an improved SAF improves the car's ability to negotiate a turn, by (i)

reducing the turn angle and, therefore, the turn speed at which the car should rotate duting the first left turning (since it already slightly oriented towards the turn), and (ii) by increasing the actual turning radius (and, therefore, reduce the lateral forces) of the car when cornering (since it is slightly outside the turn at its entrance).

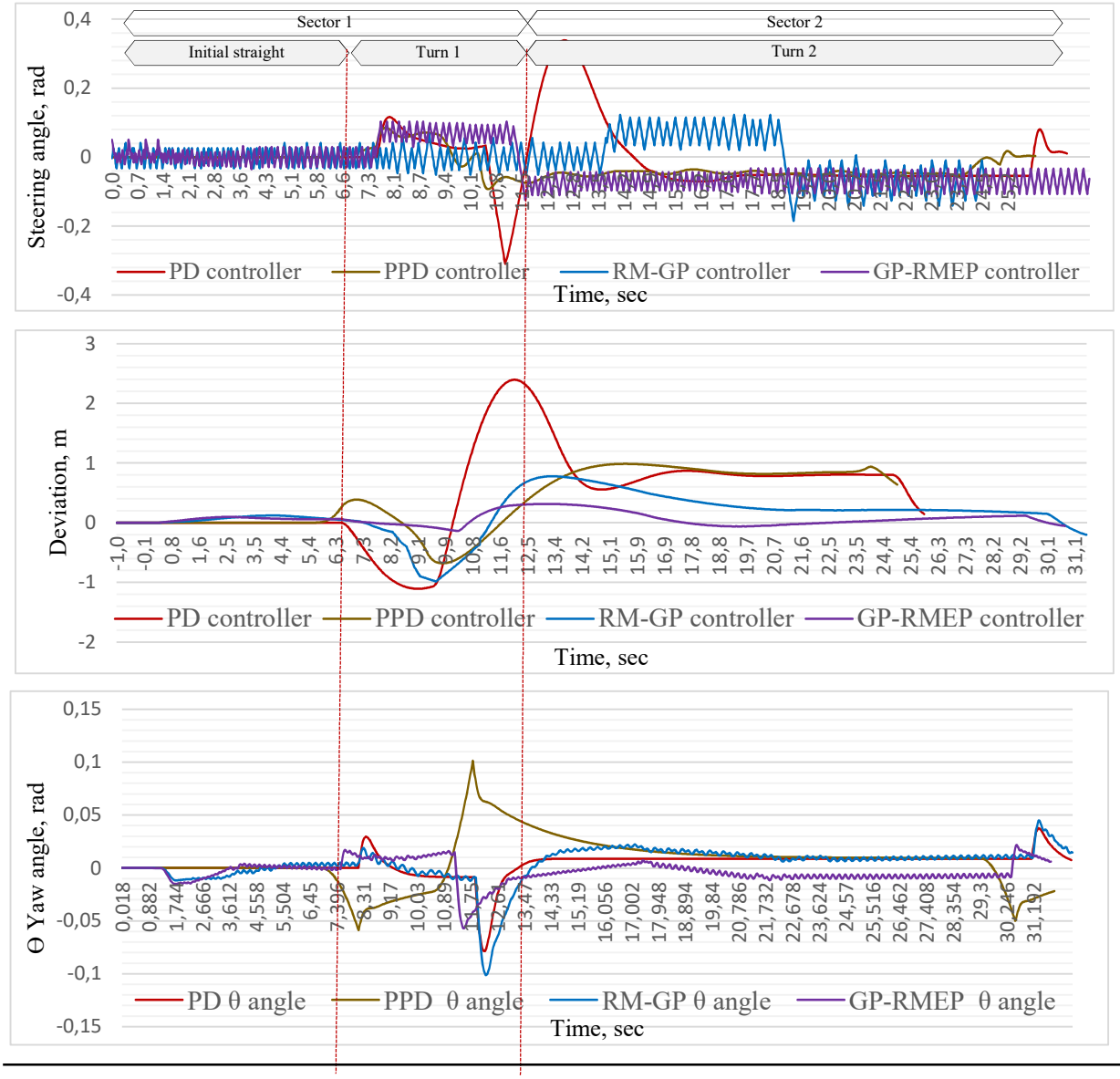



***Fig- 41*** The dynamics of the steering angle (top) and the deviation from the center of the lane (middle) and Θ angle – deviation from the desired trajectory angle(bottom) of the car steered by the most general SAF on the track with friction coefficient μ= 0.5

Interestingly, the return to the center of the lane after the start of the right turn is gradual and slower than we might expect. The steering controller, apparently, should know that a faster return to the center of the line is possible, but the advantage of such a return is that it has less area under the path of the car, and, therefore, better stability, will probably be negated in the future and therefore such tactics did not take root in individuals in the SAF during evolution. The reasons for the devaluation of such tactics could be either (i) subsequent fluctuations around the center of the lane — oscillations, or (ii) the car's unfavorable position for the beginning of future maneuvers. Those, short-term return to the center of the strip ultimately led to higher values of the penalty parameters of the fitness function than not the most advantageous, but temporary and intermediate shifted position during the second right turn.

### 6.3.2 Driving with noisy input data

Another important consideration in the light of the developed controllers is driving with noisy input data. The causes of noise can be associated with data transmission disturbances, defects in sensor components, as well as various unforeseen defects in sensor perception during a real drive (a sudden strong fogginess or a truck carcass, for example, will dramatically degrade the quality of data from the car cameras scanning the road ahead).With noisy and or incomplete data from sensors, there are different ways of making decisions. In aircraft control, for example, all sensors are duplicated several times and their readings are compared. In the event that any sensor demonstrates data that is different from the rest, it is excluded from consideration and accepted as unreliable. In automatic control of the car, such a system of duplication of sensors is absent. Therefore, we propose another option for dealing with noisy data. This method is also common among photo-processing and is also to compare and average the results of several controllers. Only in this case, instead of comparing the input data, we will compare the output - i.e. recommended steering angle in the opinion of several controllers developed by GP for current road conditions. From the entire list of controller

responses, we exclude all markedly different results (as in aviation tactics), and from the remaining take averages. The fact is that averaging can suppress noise without destroying detail, since it essentially increases the signal-to-noise ratio (SNR) of our data. Averaging data (both input and output) works on the basis of the assumption of an absolutely random nature of noise in the image. Accordingly, random deviations from the true data will gradually decrease as the average number of sensors (for input data) or controllers (output data) is averaged.

### 6.3.3 Oscillation analysis and validation

However, one of the most interesting and unexpected features of the advanced SAF's driving style is the behavior of the steering commands, which is the appearance of steering oscillations (Figure 41, top). After analyzing the number of sign changes in the approximate first derivative $d = \Delta\sigma/\Delta t$, as we did earlier, we got GP-RMEP - 349 changes (Recall that the PD controller had 218 changes, and the PPD controller had 98 changes). At first glance, this behavior is a typical malicious artifact, the result of any error or delay in the control system. In addition, during driving, driver-induced steering fluctuations are also usually considered harmful to driving and are often associated with loss of control of the car [47] due to inattentive driving [48] or even driving under the influence of alcohol [49] or other distorted perceptions driver and possibly health related reasons. However, such steering oscillations have their own easily recognizable specific characteristics. For example, the frequency of such oscillations usually lies in the range of about 0.2 Hz ~ 0.5 Hz [48] and is always manifested by corresponding oscillations (“weaving”) on the vehicle’s trajectory. In our case, however, steering vibrations do not lead to any trajectory oscillations, on the contrary, it is quite smooth and clear. In addition, the oscillations we observe in Figure 39 have a much higher frequency (2 Hz ~ 5 Hz). Without a doubt, such a high oscillations frequency gives an

additional load on the mechanical part of the steering wheel of the car, and also depletes tires faster.

Another consideration, besides the unusual steering command characteristic, indicating the importance of these oscillations is, as in the previous chapter, the fact that almost all advanced steering controllers with the best estimation quality function result have similar steering command shape. Based on these data, two big assumptions can be made. First: Oscillations do not carry any semantic load for the driving process. They are the result of a mistake, or a curiosity of the evolutionary process, which does not see harm in them and therefore has not got rid of them. Then an effective method of dealing with them will be the design of an adequate fitness function, which will fine for an excessive change of direction. The second assumption is even more exciting - the oscillations appeared as a result of evolution in each of the most successful individuals as a unique evolutionary technique, which somehow gave these individuals an advantage over their companions in speed or stability of control. So, if we expand the first big assumption, we get the following mutually exclusive hypotheses about the possible causes of the oscillations:

(i) unrealistic modeling of the car's rotation,

(ii) inadequate evolutionary structure,

(iii) neutral genetic code in the developed solutions,

(iv) beneficial effect oscillations detected by evolution.

Further, in the discussion, we will look closer to these assumptions.

#### 6.3.3.1 Car modelling

As for the first hypothesis, the behavior of the car during cornering is determined by the complex interaction of the friction forces acting on the tires and the chassis of the car with a given moment of inertia of rotation (yaw). On the other hand, the friction forces depend on

the coefficient of friction between the tires and the road, the sliding angle of the tires and the normal forces applied to them. These normal forces are the sum of the static load on the tire and the dynamic weight transfer. The latter is caused by longitudinal (for example, during acceleration or deceleration) or lateral (for example, when cornering) forces acting on the vehicle, elevation of the center of gravity, wheelbase (for longitudinal weight transfer) and axial track (for lateral weight transfer) [24]. However, all the factors mentioned are accurately modeled in TORCS [50][51]. Therefore, we could not accept this hypothesis.

#### 6.3.3.2 Evolutionary structure

The second hypothesis implies that the adopted GP, due to the lack of a set of terminals, cannot develop a solution that is resistant to delays caused by 100 ms control feedback and oscillations are a typical consequence of this problem. Such a situation is really possible if, for example, the set of GP terminals and non-terminals does not allow us to calculate the expected values of the corresponding parameters related to the vehicle dynamics, i.e. in it there are no projections (at least linear) of the values of these parameters in the near future based on the gradients of their changes. However, as we clarified in Chapter 3, all of these parameters (e.g., yaw angle, lateral deviation, etc.) and their derivatives (e.g., yaw rate, lateral speed, etc.) are included in the set of GP terminals. In addition, as shown in [52], an identical evolutionary structure was able to develop non-oscillatory steering of the same car model with much longer steering delays (up to 400 ms) on non-slippery roads. Thus, with a delay of only 100 ms and the model's ability to project the dynamic parameters of the car, such oscillations should not occur. Therefore, we reject this hypothesis too.

#### 6.3.3.3 Oscillation suppression

The third hypothesis assumes that the car is modeled correctly, and the adopted GP can, in principle, develop non-oscillatory SAF (that is, both the first and second hypotheses are not

true), however, for some reason, it does this despite that benefits from this is not observed. One of the possible reasons is that the steering oscillations resulting from the neutral genetic code [53] in the developed SAF do not have a detrimental effect on the quality of the steering, as estimated by the GP fitness function. Therefore, the evolutionary framework cannot apply any selection pressure to the oscillating SAF, and they remain. However, as mentioned earlier, these fluctuations would not be desirable in the real implementation of advanced SAF in the real world.

To test this hypothesis, we redesigned the GP fitness function to include a clear penalty for steering fluctuations:

$$F_{PSO} = F + k \times N_O = S_T + C \times V_{L\_AVR} + k \times N_O \qquad (24)$$

where $F$ is the initial fitness function, and $N_O$ is the number of steering oscillations recorded during the test, and $k$ is the selected scale factor. If this hypothesis is true, the newly developed SAF, suppressing fluctuations, must demonstrate at least the same (or even better) quality of control due to the lower sum of the first two additive components (corresponding to the initial value of $F$ suitability) and the absence of a large penalty for oscillations rotation function. However, as shown by the experimental results in Table 19 obtained from 32 evolutionary runs of GP (for the coefficient of friction $\mu = 0.5$), the best value $F = 462$ is achieved at the lowest scaling factor ($k = 0.03$), and, therefore, the lowest selection pressure against steering oscillations. This best value is higher (i.e. worse) than the previously obtained best and most reliable oscillating SAF (373 and 431, respectively).

***Table- 15*** Experimental results on evolution of SAF with penalty for steering oscillations. Obtained from 32 evolutionary runs of GP

| **Scaling coefficient *k*** | **Best fitness value** | |
|---|---|---|
| | $\mathbf{F_{PSO} = F + k \times N_O}$ | $\mathbf{F = S_T + C \times V_{L\_AVR}}$ |
| 0.03 | **590** | **462** |
| 0.05 | **615** | **497** |
| 0.1 | **702** | **588** |
| 0.2 | **855** | **754** |
| 0.5 | **883** | **770** |

This means that steering oscillations for some reason lead to better control of the car in understeering position and do not affect stable turns. Thus, we assume that steering oscillations have a positive (*rather than neutral*) effect on the quality of driving in understeering position (on the entrance of the first turn for example – Figure 41, 6.5 - 8.1 sec) and, therefore, can also reject the third hypothesis. On the other hand, we assume that the oscillations in the steady rotation state are indeed the result of a neutral code in the GP.

#### 6.3.3.4 Inducing an artificial oscillation

Thus, all the theories above were not confirmed, but found their complete refutation. This implies that the fourth hypothesis - "emerging steering fluctuations have a beneficial effect on driving" received at least indirect confirmation. To test this hypothesis, we conducted an additional experiment, artificially introducing one oscillation into stable driving without own oscillations of the steering. To do this, we selected a tuned PD controller and applied one sinusoidal disturbance during the car cornering and suffering with understeer (i.e., in the unstable position of the car at the entrance to the first left turn), assuming that if the Fourth hypothesis is indeed true, then the steering quality (for example, approximated as a deviation from the center of the lane) should show some improvement. As a result of manual selection of the corresponding sinusoidal control pulse parameters - time, period and amplitude - we

conducted a series of experiments and present here the best of the results obtained - for a pulse with an amplitude of 0.07 rad and a period of 0.6 s. Interestingly, these values are within the ranges "detected" by the GP, which is also an additional argument indirectly indicating the benefit of the oscillations found by evolution. So, in our experiment, we use artificial oscillation with selected parameters starting from 7.1 seconds, i.e., 1.6 seconds after the car enters into a turn and begins to experience understeer.

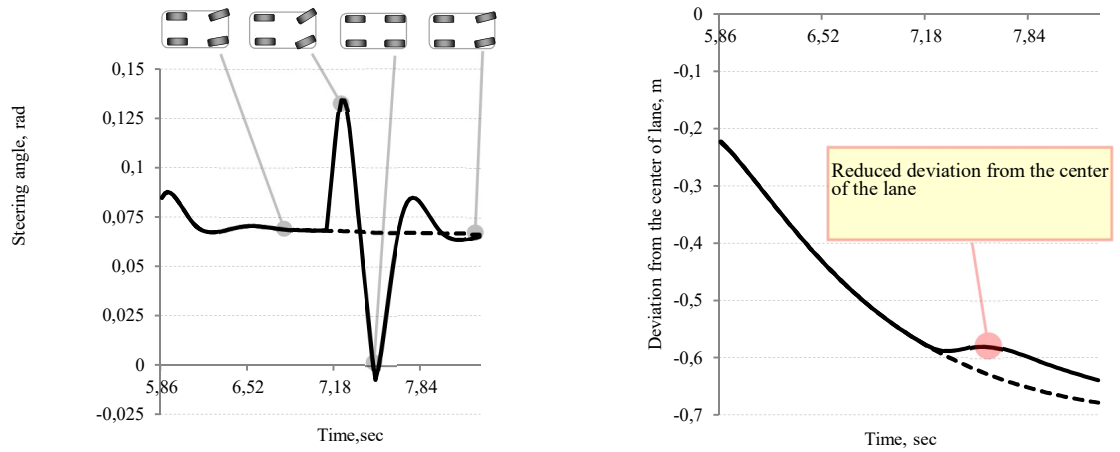



***Fig- 35*** Steering angle (left) and deviation from the center of the lane (right) of the understeering car controlled by SAF of a tuned PD controller without- (dashed line) and with an artificially introduced oscillation (solid line).

As shown in Figure 42, such a steering impulse reduces the deviation from the center of the lane by about 0.05 m, which is 8% of the whole deviation value. Accordingly, repeated pulse of oscillation would lead to an even greater decrease in deviation from the desired trajectory. These oscillations could not occur in the steering controller of the PD, because the a *priori* fixed structure of the mathematical model of the latter excludes the possibility of even "presenting" the beneficial effect of such oscillations.

### 6.3.3.5 Possible reasons for such oscillations producing

Since the fourth hypothesis was confirmed, the next logical step for us was to find out the reasons why the evolutionary driving style became so effective on slippery roads. First of all, it is worth noting that because of the frequency of changing the direction of steering, the

influence of oscillations does not directly extend to the trajectory, but the effect of these oscillations is experienced by car's tires. Thus, we are interested in the observing the distribution and influence of forces on the front tires of the car at a time when the car itself suffers from a understeering when entering the turn (as the most deviated from the desired trajectory during the mentioned moment). The first consideration is quite simple - constant oscillations support and ensure the deformation of the part of the tire that contacts the road surface (Figure 43, left). This means maintaining high values of the angle of slip precisely due to differences in tire orientation. The slip angle, in the turn, is proportional to the coefficient of friction, both in reality and in the TORCS simulation (Figure 43, right). Thus, it turns out that over the course of evolution, the model, experiencing a lack of stability of driving on slippery roads, found a way to independently increase the friction coefficient and thereby more successfully pass the race.

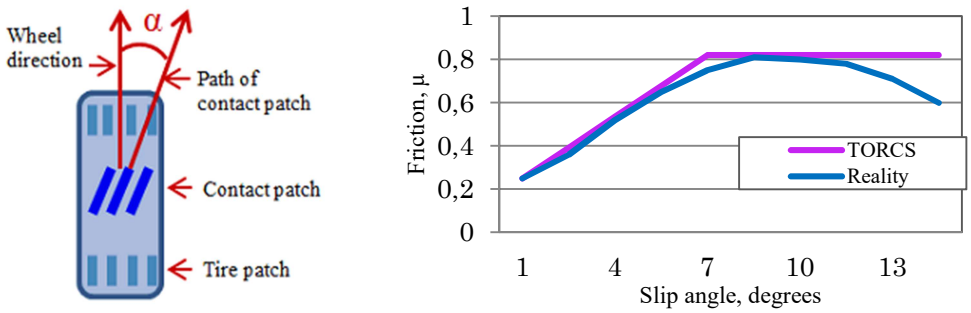



***Fig- 43*** Slip angle of a turning wheel α (left), The dynamics of the friction coefficient as a function of the slip angle of the tires (right)

The second consideration is a bit more complex and needs further clarification. First of all, I would like to recall the Ackerman effect in steering described in paragraph 2.1.1.2.2. In short, we note that it consists in different turning angle of the front inner and outer wheels of the car when passing a turn or curve (Figure 44). As shown in Figure 44 (left), in a stable car, the steering angles of the left (inner) and right (outer) tires are $\delta_L$ and $\delta_R$, respectively, where | $\delta_L$

|> | $\delta_R$ | thanks to the Ackerman steering principle [24]. Since the available friction forces are much larger than the lateral ones (depending on the speed, the estimated turning radius and the yaw momentum of the car), the slip angles of both tires ($\alpha_L$ and $\alpha_R$) are negligible. Therefore, the actual driving directions of both tires are almost identical to the estimated directions determined by the steering angles $\delta_L$ and $\delta_R$, and the actual turning radius of the car $R_a$ is almost identical to the estimated $R_i$.

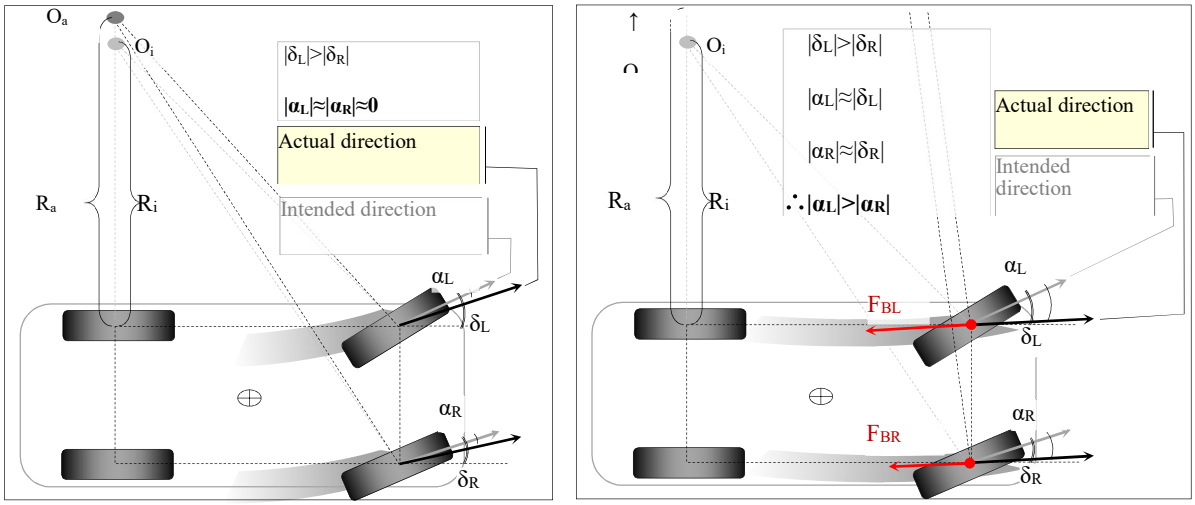



***Fig- 44*** Turning of a well-controllable (left) and understeering (right) car. The different (Ackermann) steering angles $\delta_L$ and $\delta_R$ of the understeering car (right) result in different slip angles $\alpha_L$ and $\alpha_R$ of the tires, which, in turn, yields an asymmetric braking

Conversely, the available friction of the front tires of a car with understeer (Figure 44, right) is less than the necessary lateral forces, and the tires slide in the forward direction. Therefore, the tire slip angles $\alpha_L$ and $\alpha_R$ are almost identical to the corresponding steering angles $\delta_L$ and $\delta_R$. Since the steering angle of the left tire $\delta_L$ is larger than that of the right tire $\delta_R$, the slip angle $\alpha_L$ will also be larger than $\alpha_R$. Since the resulting friction (braking) forces of the $F_{BL}$ and $F_{BR}$ tires depend (linearly) on the sliding angles $\alpha_L$ and $\alpha_R$, respectively, the friction force of the left (internal) tire $F_{BL}$ will be greater than the force of the right (external) one $F_{BR}$. Asymmetry of the braking forces $F_{BL}$ and $F_{BR}$ would lead to a torque around the yaw axis of the vehicle with understeer, which would facilitate its rotation in the direction of rotation. The

instantaneous peaks of the fluctuating steering angles $\delta_L$ and $\delta_R$ lead to the corresponding peaks in the highly asymmetric braking forces $F_{BL}$ and $F_{BR}$, which leads to better handling when cornering a car with understeer.

### 6.3.4 Summary and Future Work

Thus, we verified and proved the beneficial effect of steering oscillations on the controlling of a vehicle with understeer. However, it should be noted that we are aware that in our experiments we took as a given some assumptions that may not be fulfilled in some real applications. For example, we believed that the car will move at low speed on roads with a low coefficient of friction. Those, we deliberately limited the speed of the car to its critical speed during a turn. This was done to simplify the analysis of experiments and should be taken into account in the future. The results may turn out to be less or not applicable at all for high-speed (competitive) driving in conditions of high traction, since in this scenario, a significant dynamic weight transfer to the outer front tire will negate the effect of a lower tire slip angle. This will result in a higher (rather than lower) braking force applied to the vehicle's external tire. Indeed, the frictional forces applied to the front tires, in addition to the coefficient of friction and the sliding angle, also depend on the sum of the static and dynamic load of the tire. The latter, in turn, depends on the width of the axis, the elevation angle of the vehicle, and most importantly, on the lateral forces. These forces — being very small on slippery roads — will increase with the speed of negotiating more dangerous corners.

Moreover, the reliability of stable steering oscillations of 2-5 Hz will depend both on

(i) the power of the steering system, and

(ii) the resistance forces - mainly rotating tires.

As for the previous state - indeed, most modern cars are equipped with (hydraulic or electric) power steering. On the other hand, to minimize drag forces, lightweight, low-profile aluminum alloy wheels are required, which are usually related to sports cars. The reduced height of the side wall of low-profile tires is associated with increased stiffness of the latter, which, in turn, minimizes the damping effect of the tire on steering vibrations. Again, the proposed approach may not be applicable for high-speed driving, because the increased gyroscopic forces of the rotating wheels will strongly oppose the alleged rapid angular movements of the latter. In future experiments, it would be appropriate to test these assumptions and check the potential preservation of positive effects during the turn of the car.

Also in spite of the fact that we tested a car with specific parameters (a race car), providing, among other things, greater stability (for example, the height of its center of gravity is lower than usual), some of the found features and control tactics can be applied to regular cars. The height of the center of the gravity – together with the lateral forces – would determine the amount of the lateral weight transfer of the car on cornering, which, in turn, would affect the distribution of the normal forces on the tires. On slippery roads, due to the lower lateral forces applied to the cornering car (due to the lower friction coefficients), we assume that the lateral weight transfer would be negligible, regardless of the height of the center of the gravity of the car.

# Chapter 7

# Conclusions and Future Work

## 7.1 Summary and Conclusion

The initial pulse to this study was the fact that made a huge impression on me - namely, the fact that the vast majority of road crashes occur for reasons directly related to the "humanity" of the driver. Fatigue, inattention, unpreparedness, cognitive or physical overload are the main reasons why drivers cannot prevent a catastrophe in time and to which computer systems are practically not affected. The imperfection of existing systems that share control with the human driver during the most difficult periods of control has become another incentive and challenge to explore automatic control. This work aimed to find a control steering model that would be ready to adequately respond to any difficult weather conditions, putting safety of the driver and all road users as a priority. Starting our research with the study of existing steering models - classic reactive systems that evaluate their current deviation from the target trajectory and correct themselves, we encountered their shortcomings and found them to be inapplicable in the field of low friction coefficients on the road surface - that is, in severe weather conditions when the stability of driving is significantly reduced. As a result of the analysis of the models, we conducted a series of experiments designed to estimate their adaptive ability to various weather conditions by adjusting their coefficients. The scaling coefficients of these models allow you to adjust the reaction force of the model to specific deviations. As a result of the experiments, we found out that even the best representatives of the class of such models successfully managing maneuvering on a dry surface significantly lose stability and become dangerous on a slippery surface. This motivated us to try to create a new and better model. Bearing in mind that the existing most successful car driving models for dry roads were designed to imitate, in a sense, the human driving style, we assumed that we could achieve better results by more accurately imitating. Our assumption was to change the control model according to the following behavior - the human driver, having vision and seeing an obstacle in front of him or a change

in the curvature of the road, begins to adjust the car's position *in advance*, thereby facilitating future maneuvering and achieving greater car stability. So we included in the classic model a reaction not to the current position, as it was before, but to the position that the car will occupy *in the future* if it continues to drive in the same direction. This gave the system the ability to “see” in advance a turn or obstacle to go round it. In addition, supporting the concept of imitation of a human driver, we predicted only the position of the car on the road, but not its future orientation relative to it, keeping it current. This is due to the fact that with an intense load on perception, such as tracking a situation changing at a high speed on the road, human brain trying to keep the data processing at a high level narrows the field of view, "cutting off" part of the data [54]. Such changes to the classical model significantly improved its quality, which was confirmed by our experiments. Of course, it should be noted that we are not the first who included the element of prediction into the classical model, although the first to do so for the conditions of a slippery road. However, our approach differs from others (like already the classic MPC [55]) by some more features. In particular, we intentionally did not include new variables and data into the model (except for the terrain map, which allowed us to make predictions) and maximally reduced the computational load of the formula for safety reasons related to the speed of processing the control equation. And although this precaution may be unnecessary and not so significant for application in ordinary cars in comparison with the total delay time of signal transmission, we were able to obtain interesting results with minimal computational effort.

Despite the fact that the results of driving a car on slippery roads have already improved significantly, the idea of imitating nature inspired us to use fuzzy logic in determining scaling coefficients in the new model. The charm of this approach is that, having sufficient computing power, we are able to specify a set of terminals and, connecting the evolution and simulation of races, spend many millennia in the context of individuals (who are the new

control formulas) participating in the evolution to select the ideal driver. This driver will not be subject to fatigue, stress or illness, he will give a quick reaction that exceeds the reaction of an ordinary driver and he is not afraid of an accident. Thanks to many generations of evolution, we have received many successful models that have shown superior to all previous results in our study. Unfortunately, it was not possible to analyze the formulas obtained, not only because of their unusual complexity in appearance, but because of their evolutionary origin - they abound in a neutral code that is no different from the essential for driving a car. Thus, having obtained new results and analyzing them, we took the next step in the study. We decided to completely abandon the original structure of the reactive controller, leaving evolution to make this decision. According to the results of a new series of experiments with an expanded set of terminals, we got impressive results that were also expected to surpass all previous ones. Due to the impossibility to analyze the obtained formulas, we turned to a qualitative analysis of the results, namely, we were interested in the extremely unusual form of the steering function during the race. The fact is that it was all mottled with small and frequent oscillations (2-5 Hz). Such oscillations led us to the assumption that the evolutionary model in difficult conditions of slippery road tried to increase the friction coefficient on the tires, by maintaining the absolute value of the slip angle. In addition, evolution has found an effect that helps the car fit into a turn on a slippery road when it suffers from a understeer by maintaining the difference in angles slipping on the front tires (based on the Ackerman principle, which was taken into account in the simulator), which led to the difference in the normal forces that form the torque moment to the desired cornering. These effects are not native to the human driver, and even knowing about them, the human driver, if he is not a professional racer, is unlikely to be able to maintain such driving style. On the other hand, for an electronic model, this is relatively uncomplicated and can lead to a significant increase in road safety in difficult weather conditions.

We consider the results obtained as an important step towards the development of a fully automatic control system for driving a car, especially in the mode of maintaining control of the car in a slippery road.

## 6.1 Future work

Although we performed substantial amount of experiments to verify our proposed approach, there are still many areas that could be used to further improve this work. The continuation of this study may be an analysis of individuals obtained during the evolution process, a comparison of the most successful of them, and identification of a neutral code and driving factors that allowed these models to obtain superior results. This direction looks promising, from the point of view that even if in the future we abandon the model developed by evolution (for example, due to difficulties with its verification), the very features and techniques that it may have found — like oscillations from the ones we have analyzed — can be very promising and applicable in any other models. Of course, the very concept of developing a car control model based on the evolutionary process is not fully exhausted. We believe that the idea of combining the evolution of several car control systems at once is very promising. An excellent example would be the combined control system of an asymmetric brake system [56] and steering system. Undoubtedly, using such powerful control tools, evolution will be able to increase the safe speed of the route and improve vehicle control during sharp turns.

Other areas of future research may include manipulations with joint prediction of the position of the car on the road and its orientation. The imitation of the human driver, which we held earlier in our studies, was caused in this case by overloads of the human brain perception, which can be levelled by attracting additional computing resources. Another area of research and testing is the study of the behaviour of the obtained models on other paths - despite the fact that the path that participated in our experiments was quite complex, it can be assumed

that under the new conditions, the models found using evolution can demonstrate new interesting effects that previously weren't obtained. However, detailed studies and experiments are needed to verify this hypothesis.

# Bibliography


[1] R. TAY and D. KNOWLES, "Driver Inattention," *IATSS Res.*, vol. 28, no. 1, pp. 89–94, 2004, doi: 10.1016/s0386-1112(14)60095-9.

[2] K. H. R. J. de Boer, "Automation surprise," *Aviat. Psychol. Appl. Hum. Factors 7*, vol. 28.

[3] A. H. and H.-J. Hoermann, "Flying the needles: Flight deck automation erodes Fine-motor Flying skills among airline pilots," *Hum. factors 58*, p. 533, 2016.

[4] M. I. F. and M. T. Keane, "Why some surprises are more surprising than others: Surprise as a metacognitive sense of explanatory difficulty," *Cogn. Psychol. 81*, p. 74, 2015.

[5] M. S. Y. and N. A. Stanton, "Malleable attentional resources theory: a new explanation for the effects of mental underload on performance," *Hum. factors 44*, p. 365, 2002.

[6] A. Landman, "Managing Startle and Surprise in the Cockpit," Delft University of Technology, Delft, Netherlands, 2019.

[7] N. A. Anderson, *Instrumentation for process measurement and control, third edition*. 2017.

[8] K. S. Rao and R. Mishra, "Comparative study of P, PI and PID controller for speed control of VSI-fed induction motor," vol. 2, no. 2, pp. 2740–2744, 2014, [Online]. Available: http://www.ijedr.org/papers/IJEDR1402230.pdf.

[9] H. Dankowicz and R. Sandberg, "System Dynamics for Engineering Students: Concepts and Applications," *Proc. Inst. Mech. Eng. Part C J. Mech. Eng. Sci.*, vol. 225, pp. 1021–1024, Apr. 2011, doi: 10.1177/0954406211403548.

[10] V. Nikulin, A. Podusenko, I. Tanev, and K. Shimohara, "Regression Based Model for Autosteering of a Car with Delayed Steering Response," 2017, doi: 10.1109/DSAA.2017.16.

[11] L. Pantel and L. C. Wolf, "On the impact of delay on real-time multiplayer games," *Proc. Int. Work. Netw. Oper. Syst. Support Digit. Audio Video*, pp. 23–29, 2002, doi: 10.1145/507671.507674.

[12] D. Loiacono, L. Cardamone, and P. L. Lanzi, "Simulated Car Racing Championship: Competition Software Manual," no. April, 2013, [Online]. Available: http://arxiv.org/abs/1304.1672.

[13] C. R. Wymann B, Dimitrakakis C, Sumner A, Espi´e E, Guionneau C, "TORCS, the open racing car simulator, v1.3.5," 2013. http://www.torcs.org.

[14] E. Onieva, D. Pelta, J. Alonso, V. Milanes, and J. Pérez, *A modular parametric architecture for the TORCS racing engine*. 2009.

[15] L. Cardamone, D. Loiacono, and P. L. Lanzi, *Learning drivers for TORCS through imitation using supervised methods*. 2009.

[16] B. Wymann, "The TORCS Racing Board," 2017. .

[17] T. Vanderblit, "Autonomous Cars Through The Ages," *Wired*.

[18] M. Weber, "Where to? A History of Autonomous Vehicles," *Comput. Hist. Museum*, 2018.

[19] E. P. Klement, “Fuzzy Logic in Arti cial Intelligence CD { Technical Report 94 / 67 Technische Universitat Wien Institut fur Informationssysteme Abteilung fur Datenbanken und Expertensysteme,” no. July, 1997.

[20] S. S. Bremermann H. J., Roghson J., “Global properties of evolution processes. Natural automata and useful simulations.,” 1966, p. pp 3-42.

[21] J. R. Koza, “Genetic programming as a means for programming computers by natural selection,” *Stat. Comput.*, vol. 4, no. 2, pp. 87–112, 1994, doi: 10.1007/BF00175355.

[22] I. Tanev and K. Shimohara, “XML-based genetic programming framework: design philosophy, implementation, and applications,” *Artif. Life Robot.*, vol. 15, no. 4, pp. 376–380, 2010, doi: 10.1007/s10015-010-0857-9.

[23] N. Alekseeva, I. Tanev, and K. Shimohara, “Evolving the Controller of Automated Steering of a Car in Slippery Road Conditions,” pp. 1–19, 2018.

[24] T.D.Gillespie, *Fundamentals of Vehicle Dynamics*. 1992.

[25] P. Thomas R. Kurfess, “Getting in tune with Ziegler-Nichols,” *Control Eng.*, no. Academic Viewpoint column, p. issue, 28.

[26] J. B. Z. and N. B. Nichols, “Optimum settings for automatic controllers,” *ASME Trans.*, pp. 759–768, 1942.

[27] K. J. Å. & T. Hägglund, “Automatic Tuning of PID Controllers,” *Control Handbook, IEEE/CRC Press*, vol. Chapter 52, 1995.

[28] Y. Zhang, R. Hoogendoorn, M. Mazo, and H. Hellendoorn, “Steering Controller Identification and Design for Human-like Overtaking,” *Procedia Manuf.*, vol. 3, no. Ahfe, pp. 2526–2533, 2015, doi: 10.1016/j.promfg.2015.07.536.

[29] E. Sariyildiz, H. Yu, and K. Ohnishi, “A practical tuning method for the robust PID controller with velocity feed-back,” *Machines*, vol. 3, no. 3, pp. 208–222, 2015, doi: 10.3390/machines3030208.

[30] W. Chen, H. Xiao, Q. Wang, L. Zhao, and M. Zhu, *Integrated vehicle dynamics and control*. 2016.

[31] K. M. Moradi M., Johnson M., “Predictive PID Control,” *Springer, London*, 2005, doi: https://doi.org/10.1007/1-84628-148-2_13.

[32] S. H. Zak, “Fall 2017 An Introduction to Model-based Predictive Control ( MPC ) Basic Structure of MPC,” pp. 1–25, 2017.

[33] R. H. Middleton and G. J. Adams, *Modification of Model Predictive Control to Reduce Cross-Coupling*, vol. 41, no. 2. IFAC, 2008.

[34] K. Zhu and B. Chen, “Cross-coupling design of generalized predictive control with reference models,” *Proc. Inst. Mech. Eng. Part I-journal Syst. Control Eng. - PROC INST MECH ENG I-J SYST C*, vol. 215, pp. 375–384, Jun. 2001, doi: 10.1243/0959651011541076.

[35] M. Leuer and J. Böcker, “Switching strategy for Direct Model Predictive Control in power converter and drive applications with high switching frequency,” in *2015 6th International Conference on Automation, Robotics and Applications (ICARA)*, 2015, pp. 569–573, doi: 10.1109/ICARA.2015.7081210.

[36] K. Komoriya and K. Tanie, “Trajectory Design and Control of a Wheel-type Mobile Robot

Using B-spline Curve," in *Proceedings. IEEE/RSJ International Workshop on Intelligent Robots and Systems '. (IROS '89) 'The Autonomous Mobile Robots and Its Applications*, 1989, pp. 398–405, doi: 10.1109/IROS.1989.637937.

[37] H. Marzbani, R. Jazar, and M. Fard, "Better Road Design Using Clothoids," 2015, pp. 25–40.

[38] R. Lamm, B. Psarianos, and E. M. Choueiri, "A practical safety approach to highway geometric design. International case studies: Germany, Greece, Lebanon, and the United States."

[39] K. Zeeb, A. Buchner, and M. Schrauf, "Is take-over time all that matters? The impact of visual-cognitive load on driver take-over quality after conditionally automated driving," *Accid. Anal. Prev.*, vol. 92, pp. 230–239, 2016, doi: 10.1016/j.aap.2016.04.002.

[40] H. Marzbani, M. Simic, M. Fard, and R. Jazar, "Better Road Design for Autonomous Vehicles Using Clothoids," 2015, pp. 265–278.

[41] W. W. E.-M. Vagia, "The New Design Strategy on PID Controllers," Rijeka: IntechOpen, 2012, p. Ch. 11.

[42] M. Johns *et al.*, "Exploring shared control in automated driving," in *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2016, pp. 91–98, doi: 10.1109/HRI.2016.7451738.

[43] G. Wilde, "Risk Homeostasis Theory: An Overview," *Inj. Prev.*, vol. 4, pp. 89–91, Jul. 1998, doi: 10.1136/ip.4.2.89.

[44] N. A. Stanton and M. Pinto, "Behavioural compensation by drivers of a simulator when using a vision enhancement system," *Ergonomics*, vol. 43, no. 9, pp. 1359–1370, 2000, doi: 10.1080/001401300421806.

[45] J. Koza, M. Keane, M. Streeter, W. Mydlowec, J. Yu, and G. Lanza, "Genetic Programming IV: Routine Human-Competitive Machine Intelligence," Jan. 2003.

[46] K.Kinnear, "Generality and Difficulty in Genetic Programming: Evolving a Sort.," in *Proceeding of the Fifth International Conference on Genetic Algorithms*, 1993, p. San Mateo, CA, Morgan Kaufmann.

[47] R. Jagacinski and J. Flach, *Control Theory for Humans: Quantitative Approaches to Modeling Performance*. 2018.

[48] D. Sharma, I. Tanev, and K. Shimohara, "Detecting Driver-induced Steering Oscillations through Adaptive Thresholding of the Power Spectrum of Vehicle's Lateral Acceleration," *IEEJ Trans. Electron. Inf. Syst.*, vol. 138, pp. 254–262, Mar. 2018, doi: 10.1541/ieejeiss.138.254.

[49] M. West, "Drunk Driving," 2020, pp. 79–99.

[50] "TORCS - The Open Racing Car Simulator." .

[51] "TORCS robot tutorial." http://www.berniw.org/tutorials/robot/tutorial.html.

[52] V. Nikulin, A. Podusenko, I. Tanev, and K. Shimohara, *Evolving the autosteering of a car featuring a realistically simulated steering response*. 2018.

[53] T. Hu and W. Banzhaf, "Neutrality, Robustness, and Evolvability in Genetic Programming," 2018, pp. 101–117.

[54] D. Harris, *Engineering Psychology and Cognitive Ergonomics*. 2019.

[55] “Model Predictive Control for Autonomous and Semiautonomous Vehicles,” 2014.

[56] J. Huang, I. Tanev, and K. Shimohara, “Evolutionary development of electronic stability program for a simulated car in TORCS environment,” in *2015 IEEE Congress on Evolutionary Computation (CEC)*, 2015, pp. 1474–1481, doi: 10.1109/CEC.2015.7257062.

# Appendix

## 1. XML GP Schema for Evolving Driving Agent

```
<?xml version="1.0"?>

    <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">


    <xs:simpleType name="VAR">
      <xs:restriction base="xs:string">
        <xs:enumeration value="p_0"/>
        <xs:enumeration value="p_1"/>
        <xs:enumeration value="p_2"/>
        <xs:enumeration value="p_3"/>
        <xs:enumeration value="p_4"/>
        <xs:enumeration value="p_5"/>
        <xs:enumeration value="p_6"/>
        <xs:enumeration value="p_7"/>
      </xs:restriction>
    </xs:simpleType>

    <xs:simpleType name="CONST">
      <xs:restriction base="xs:integer">
        <xs:minInclusive value="0"/>
        <xs:maxInclusive value="10"/>
      </xs:restriction>
    </xs:simpleType>


    <xs:simpleType name="OP">
      <xs:restriction base="xs:string">
        <xs:enumeration value="+"/>
        <xs:enumeration value="-"/>
        <xs:enumeration value="*"/>
        <xs:enumeration value="/"/>
      </xs:restriction>
    </xs:simpleType>


    <xs:complexType name="STM2">
       <xs:sequence>
          <xs:element name="OP" type="OP"/>
          <xs:element name="STM" type="STM"/>
          <xs:element name="STM" type="STM"/>
       </xs:sequence>
      <xs:attribute name="ind" type="xs:integer" use="optional"/>
    </xs:complexType>

    <xs:complexType name="STM">
       <xs:choice>
          <xs:element name="STM2" type="STM2"/>
```

```
            <xs:element name="VAR" type="VAR"/>
            <xs:element name="CONST" type="CONST"/>
          </xs:choice>
        <xs:attribute name="ind" type="xs:integer" use="optional"/>
      </xs:complexType>

      <xs:element name="GP">
       <xs:complexType>
         <xs:sequence>
           <xs:element name="STM" type="STM"/>
         </xs:sequence>
         <xs:attribute name="ind" type="xs:integer" use="optional"/>
       </xs:complexType>
      </xs:element>

      </xs:schema>
```

## 2. XML tree structure processing class GP_expression

```
class GPAPI GP_expression
{
private:
   struct Expr_node
   {
      double value;
      int arity;
      int var_index;
      GPNodeType node_type;
   };

   void Make_Tree(XMLElement* xmlElement);
   void Insert_node(XMLElement* xmlElement, std::vector<Expr_node> & expr_vec);
   double eval_val(const Expr_node & e_node, IXgpContext* context);
   std::vector<Expr_node> expr_vec_;
   std::vector<Expr_node> adf_vec_;

public:
   GP_expression(XMLElement* xmlElement);
   double Evaluate(IXgpContext* context, bool adf_f);
};
```

## 3. tTrack structure in the Track header file

```
typedef
struct
        {
            const char *name; /**< Name of the track */
            const char *author;      /**< Author's name */
            char         *filename;   /**< Filename of the track description */
            void         *params;     /**< Parameters handle */
            char         *internalname; /**< Internal name of the track */
            const char *category;    /**< Category of the track */
            int          nseg;       /**< Number of segments */
```

```
    int         version;     /**< Version of the track type */
    tdble       length;      /**< main track length */
    tdble       width;       /**< main track width */
    tTrackPitInfo pits;             /**< Pits information */
    tTrackSeg   *seg;        /**< Main track */
    tTrackSurface *surfaces; /**< Segment surface list */


    t3Dd             min;
    t3Dd             max;
    tTrackGraphicInfo graphic;
} tTrack;
```

## 3. Program Source Code

The source code for the programs used in this study are available in the enclosed CD or found in the github repository of Socio-informatics laboratory.

# Publications

## Journal Papers

- **N. Alekseeva,** I. Tanev, and K. Shimohara, "*Evolving the Controller of Automated Steering of a Car in Slippery Road Conditions*", Algorithms, Special Issue on Algorithms for PID Controllers, 11(7), 2018, 17 pages
- **N. Alekseeva,** I. Tanev, and K. Shimohara, "*Steering Controller Utilizing the Predicted Position on Track for Autonomous Vehicles Driven on Slippery Roads*", *Algorithms* 2020, Special Issue Algorithms for PID Controller 2019, 13(2)*, 48,* 20 pages

## Conference Paper

- **N. Alekseeva,** I. Tanev, and K. Shimohara, "*On the Emergence of Oscillations in the Evolved Autosteering of a Car on Slippery Roads*", IEEE ASME International Conference on Advanced Intelligent Mechatronics, July 8 - 12, 2019, Hong Kong Science Park, Hong Kong, China, 8 pages (accepted)
- **N. Alekseeva,** I. Tanev, and K. Shimohara, "*Evolving a Single-variable Controller for Automated Steering of a Car on Slippery Roads*", Proc. of SICE AC 2018, 680 - 685, Sep. 2018