School of Information Technologies

Department of Computer Systems

Mark Chernyaev

Digital Twin for ABS control models

Master's Thesis

Aleksei Tepljakov
Ph.D.
Professor

Saleh Alsaleh
MSc
Early Stage Researcher

TALLINN 2021

Infotehnoloogia teaduskond

Arvutisüsteemide instituut

Mark Chernyaev

Digitaalne kaksik ABS-i juhtimismudelitele

Magistritöö

Aleksei Tepljakov
Ph.D.
Professor

Saleh Alsaleh
MSc
Early Stage Researcher

TALLINN 2021

Declaration of Originality

Declaration: I hereby declare that this thesis, my original investigation and achievement, submitted for the Master's degree at Tallinn University of Technology, has not been submitted for any degree or examination.

Deklareerin, et käesolev diplomitöö, mis on minu iseseisva töö tulemus, on esitatud Tallinna Tehnikaülikooli magistrikraadi taotlemiseks ja selle alusel ei ole varem taotletud akadeemilist kraadi.

Mark Chernyaev

Date: May 9, 2021

Signature: .....

Abstract

The anti-lock braking system (ABS) is an extremely important part of the car systems. It guarantees preventing of the wheel-lock, when driver presses the brake. Moreover, this system must provide a controllable behaviour of the car on different surfaces. Hence, it requires additional development of the control systems, which will automatically be applied for specific model of braking depending on the type of road.

Another common problem is the cost of operations in the industrial field. Since technologies are continuously evolving and becoming more available, now it is possible to combine them into one complex system to solve field specific tasks. For example, Digital Twin (DT) is a combination of modelling, data transmission and software development. It makes possible to explore consequences that was caused by changing parameters of the digital object before working with a real one.

In addition, this paper will explain the importance of using Virtual Reality (VR) in the modern technologies development process. This paper outlines the principles of using Digital twin with Virtual Reality and future opportunities for studying and industrial fields. Finally, the paper will provide a project description and current results related to this topic.

The thesis is in English and contains 47 pages of text, 4 chapters, 26 figures.

Annotatsioon

ABS-pidurid on väga oluline osa igal autol. Süsteemi eesmärk on mitte blokeerima autoratas, millal autojuht vajutab pidureid. Lisaks, selline süsteem peab pakkuda autode juhtimise stabiilsus erinevatel pinnal. Seetõttu on vaja teha rohkem uuringud kontrollsüsteemidele, missugused saavad valida kindel pidurdusmuudel automaatselt.

Teise probleem on operatsioonide maksumus tööstusvaldkonnas. Kuna tehnoloogiaid arenevad pidevalt ja muutuvad kättesaadavamaks, praegu on võimalik need kombineerida ühele keerulise süsteemisse, et lahendada valdkonnaspetsiifilised ülesanded. Näiteks, digitaalne kaksik on kombineeritud modeleerimisest, andmeedastusest ja tarkvaraarendusest. See süsteem aitab uurida tagajärjed, mis võivad toimuda pärast parameetride muutumist reaalsel objektil.

See uuring selgitab vajalikuks kasutamiseks virtuaalne reaalsus tänapäeva tehnoloogiate arendamise protsessis. Lisaks, see töö visandab digitaalse kaksiku koos virtuaalse reaalsusega kasutamine põhimõtted ning tuleviku võimalusi õppimise ja tööstuslik valdkonnas. Lõpuks, autor kirjeldab oma projekt seotud lõputöö teemaga ning saadud tulemusi.

Antud lõputöö on inglese keeles ning sisaldab teksti 47 leheküljel, 4 peatükki ja 26 joonist.

Nomenclature

2D	2-dimensional
3D	3-dimensional
ABS	Anti-lock Braking System
AR	Augmented Reality
BP	Blueprint
CPU	Central Processing Unit
DT	Digital Twin
GPU	Graphics Processing Unit
HDD	Hard Disk Drive
HDMI	High Definition Multimedia Interface
HMD	Head Mounted Display
IDE	Integrated Development Environment
MR	Mixed Reality
OLED	Organic Light-Emitting Diode
OS	Operating System
PC	Personal Computer
RAM	Random Access Memory
RPM	Revolutions per minute
SSD	Solid-State Drive
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UE4	Unreal Engine 4
UMG	Unreal Motion Graphics

USB Universal Serial Bus

VR Virtual Reality

VS Visual Studio

Contents

List of Figures

1. Introduction

ABS in cars was implemented in late 70's. Nevertheless, car manufacturers still cannot provide an ideal model of the system. The main problem for the system is a ground environment, where small changes of the surface may have a massive impact on the control. Furthermore, ABS should be able to respond rapidly, which might restrict some possible solutions. Fortunately, some companies provides an ABS for laboratory works and tests. In particularly, it became possible to control the system by transmitting the data between PC and the board in ABS via MATLAB/Simulink. [1]

Since advanced technologies in Industry 4.0 have quite accurate sensors and transmits a numerous of data, it is important to avoid the damage of devices. [2] In this case, Digital twin is a perfect solution, which provides engineers with an exact same object in digital environment, where destruction of the whole setup will not cost much resources. Moreover, it might help with achieving a specific goals. As a result, DT could be used in both educational and industrial fields.

Nowadays, game engines are becoming more popular and usable in industrial fields. [3] Additionally, it has a considerable support of different devices such as Android/iOS smartphones, game consoles, Virtual Reality (VR) headsets, Augmented Reality (AR) and Mixed Reality (MR) devices. Also, high resolution graphics plays a significant role in a virtual world and allows to see small details in a good quality. In order to create an VR application, developer should choose the engine. Today, two of the most popular are Unreal Engine 4 (UE4) and Unity. Further, UE4 will be used as a main game engine.

The paper is organized as follows. In Section 2, the literature review is presented and describes basic information about topic related researches. Then, Section 3 is dedicated to the ABS Digital Twin development process. Finally, current progress and future possibilities discussion will be provided in Section 4.

2. Literature Review

Digital Twin

Digital twins was introduced more than 15 years ago. However, the definition of it has been changed through the period. In [4], authors describe the history of DT. Moreover, it explains the importance of using Digital twins in industrial fields, how it is connected with different types of modelling. Also, numerous examples of the real researches in DT was presented. As a result, it is an extremely informative paper for everyone, who is interested in the topic of DT.

A basic description of Digital Twin principles covered in numerous references to different articles that are all related to DT is presented in [5]. Moreover, it outlines challenges, which are a common for every DT project. It also provides a list of enabling technologies that usually play a key role.

More detailed description of DT foundational properties was published in [6]. It expands DT state-of-the-art and makes several points about the value of the Digital Twins concept. In addition, the article gives several examples highlighting the importance of having DT concept.

Virtual Reality

In [7], authors were aimed on using Virtual Reality in Industry 4.0. It based on current VR technologies, which may be used for different purposes such as medicine, learning, safety, training and industry-manufacturing. Moreover, it lists development platforms for certain use cases. For example, OpenSim and Second Life are mostly used for education and learning. On the other hand, the paper outlines disadvantages of using VR headsets, which could cause health problems.

In addition, Bellalouna [8] goes into more details of using VR in industrial training. First of all, it is evident that the difference between games and industrial application is crucial due to the number of transmitted data. While the game object can only

appear on the surface, in industrial every object should includes type of the material, weight and other parameters. In order to use CAD data in VR, it usually requires additional conversions and loss of data. Otherwise, the application will not be stable. The article also approves the importance of programming efforts to achieve real case scenarios in VR.

Yao et al. [9] describes autonomous driving based on VR and the information can be successfully used for a future development of the project. Since autonomous driving has to work with different algorithms, the functionality of ABS's DT might play one of the most important roles.

Oprea et al. [10] provided a paper related to VR, which is aimed on realistic grasping system. The information could be extremely useful especially in VR driving, where driver must interact with several objects at the same time such as driver's wheel and shifter.

Lastly, Sportillo et al. [11] has presented an actual investigation of using autonomous driving in a virtual environment. It creates an idea of improving Automated Driving System with a Digital Twin of ABS that will significantly increase the safety.

Modeling of vehicle dynamics

In order to build own system of the vehicle dynamics, Hlavaty et al. [12] published a research paper about the fuzzy control model for laboratory ABS created by INTECO. It is focused on designing of two controllers - PD and Fuzzy. As a result, PD controller has been unstable during lower speeds. Moreover, PID controller is not feasible for highly nonlinear models such as laboratory ABS. Hence, it was decided to use Fuzzy controller of Mamdami type, which is well suited for nonlinear systems. However, the final tests on real ABS have not been significantly improved compared with PD controller.

Also, Taixiong and Yage [13] had a closely related topic to the project, which might help with vehicle dynamics model development. Unfortunately, the article does not have a detailed enough description, but the main concept is understandable.

Challa et al. [14] introduced a relatively modern research, where can be found useful information regarding Rule-Based ABS algorithms. Moreover, it outlines the major difference between Model-Based Algorithms (MBAs) and Rule-Based Algorithms (RBAs).

To sum up reviewed literature, it becomes clear that there is a lack of papers related to ABS Digital Twin. Therefore, it outlines the importance of the project. Fortunately, there are numerous literature describing different control algorithms for car ABS that might be tested in the Digital Twin.

3. Project description

3.1. Development tools

This project is extremely depends on the equipment and a workspace. Fortunately, a laboratory was already equipped with necessary tools and required a few connections between devices before using them for tests. The list of used equipment in the project includes:

- Two personal computers (PCs);
- INTECO ABS object;
- Logitech G29 Driver's seat;
- Oculus Rift.

The major part of the work is related to software. The laboratory is aimed and prepared for using Unreal Engine 4 as a main game engine. Therefore, it will be basic tool in the further development. Moreover, Visual Studio 2019 is fully supported by UE4 in order to write a custom C++ code for additional functionality that might be missing in Blueprints. The last important software component of the project is Matlab/Simulink 2012b that provides powerful mathematical and physical functions, which are used to control a system in a real-time application.

Finally, the project description will be separated into several subsections, so it may possible to track the logical order of the finished work. Subsections 3.1.1–3.1.7 will include an equipment and software tools basic description, the purpose and other important features that were used in the project. The last subsections will be dedicated to the main workflow description.

3.1.1. Personal computers

Personal computers are widely used in industrial field and become a base for almost every type of application development. This project is not an exception and requires two-PC setup. One of the reason to use a second PC is caused by an ABS object constraints that will be explained in the subsection 3.1.2.

Unfortunately, this approach will create a more complex system, where additional network setup must be done. On the other hand, numerous manufacturers are using remote controls of the system by having more than two computers connected with each other. It is a common practice in autonomous vehicles to be able to send and receive the data remotely. [15] This type of communication provide more safety environment for testing new algorithms and devices. Therefore, the support of configurable network becomes crucial for the project.

Also, there are more limitations that should be considered such as operating system (OS) support and minimal requirements for using Unreal Engine 4 with Virtual Reality. In the case of OS, both PCs are running on Windows, but the versions are different. First computer is connected to INTECO ABS object and requires old drivers that are stable only on Windows 7. Second computer requires modern software and hardware that are used by UE4 with VR support, which includes Windows 10, fast processor, graphic card and memory.

The first PC specifications with installed UE4 that connected to Logitech G29 and VR:

- Central Processing Unit (CPU): Intel(R) Core(TM) i7-6700K 4.00GHz;
- Random Access Memory (RAM): 32 GB;
- Graphics Processing Unit (GPU): NVIDIA GeForce GTX 980 Ti;

- Hard Disk: 512 GB Solid-State Drive (SSD) and 2 TB Hard Disk Drive (HDD);
- Operating System: Windows 10 64-bit.

The second PC specifications with installed Matlab 2012b:

- CPU: Intel(R) Core(TM) i7-7700 3.60GHz;
- RAM: 16 GB;
- Display adapter: Intel(R) HD Graphics 630;
- Hard Disk: 256 GB SSD;
- Operating System: Windows 7 64-bit.

3.1.2. INTECO ABS

The model is created for research purposes in order to optimize braking effectiveness by implementing own control algorithms. [16] It is a simplified model and has several limitations. The first significant problem is the number of controlled wheels that is only one, while usually cars have at least 4 of them. Another problem might become a case of testing the system on different "road" surfaces. Since one of the wheels in the model replaces behaviour of the road, ABS object does not provide possibility to change it during tests. The last constraint is a lack of some real parameters such as tires friction, car mass and others.

However, those limitations are not critical for the project, because the main goal is to get a basic knowledge of the vehicle dynamics and create a realistic model based on the real parameters of the ABS object. Nevertheless, one of the possible solutions for illustrating four-wheeled car with INTECO ABS will be described further in details. Also, Section 3.1.7 will provide more information about the software that is able to process input/output signals of ABS.

The following features are ready-to-use in INTECO ABS (Fig. 1 [17]):

- Double-wheel (car wheel and "road" surface wheel) setup equipped with DC flat motor;

- Mount frame that provides support of the system;
- Two high-resolution measuring encoders;
- Testing cases with the car velocity range from 0 to 50 km/h;
- Observing the slip control under different "road" conditions (cannot be changed in real-time);
- ABS Control/Simulation Toolbox for Matlab/Simulink with basic control blocks;
- Two simple examples of the control algorithms based on relay controller.

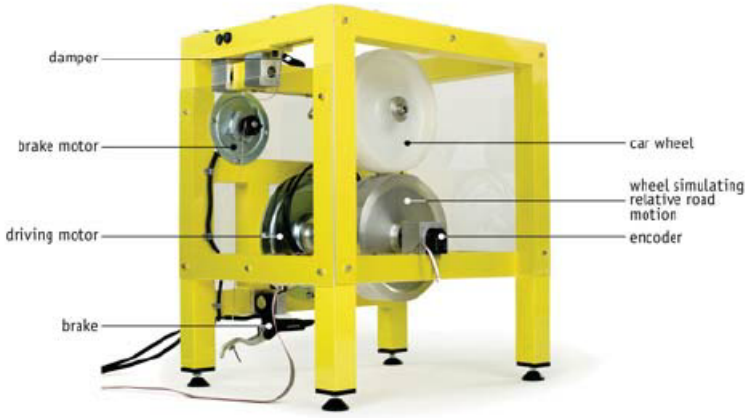

Figure 1. ABS device by INTECO

3.1.3. Logitech G29 Driver's seat

Today's cars usually consist of different complex control systems. However, from a driver perspective, every car movement is controlled via pedals, shifter, handbrake and steering wheel. In order to achieve a more immersive experience, modern market provides device setups for driving virtual car. One of those devices will be used in the project since the laboratory already have it. Moreover, it will significantly improves the way of testing ABS Digital Twin. Also, no papers was found related to using Logitech G29 for industrial tests such as movement control.

Unfortunately, this setup does not include any algorithm for realistic pedal and steering wheel behaviour. In this case, brake pedal has no force feedback, which

happens during hard braking. It lacks of handbrake and ready-to-use steering wheel force feedback control. Hence, additional development is required to get a proper feedback, which will not be in the scope of current project.

On the other hand, the full setup of Logitech G29 (Fig. 2) is supporting PlayStation consoles and Windows 10. The common features of the device:

- Steering wheel:
  - 900 degrees lock-to-lock rotation;
  - Hall-effect steering sensor;
  - Dual-Motor Force Feedback;
  - Overheat safeguard.
- Pedals:
  - Nonlinear brake pedal;
  - Self-calibrating.
- Shifter:
  - Six speeds push-down reverse gear.

Figure 2. Logitech G29 setup

3.1.4. Oculus Rift

During the last decade, Virtual Reality has made a huge impact on almost every field including gaming industry, military, manufacturers and medicine. It becomes more

available for everyone and expands human the possibilities. There are numerous researches and development projects related to VR. The market provide different setups for Virtual Reality including Oculus Rift, HTC Vive, Playstation VR and Samsung Gear VR.

Fortunately, laboratory had Oculus Rift CV1, which was released in 2016. It has been the first commercially released device in the Oculus Rift lineup. Moreover, it is perfectly suits to the goals of the project since it is widely supported by different platforms and game engines. The basic Oculus Rift setup (Fig. 3 [18]) includes:

- Virtual Reality headset;
- Two touch controllers;
- Two sensors.

Oculus Rift CV1 headset specifications:

- Display: PenTile Organic Light-Emitting Diode (OLED) 2160x1200 (1080x1200 per eye) @ 90Hz;
- Sound: Integrated 3D audio headphones;
- Input: Six degrees of freedom through USB-connected IR LED sensor;
- Controller input: Xbox One game controller or Oculus Touch motion tracked controllers;
- Connectivity: High Definition Multimedia Interface (HDMI) 1.3, Universal Serial Bus (USB) 3.0, USB 2.0;
- Mass: 470 g.

Figure 3. Oculus Rift setup

It is important to notice that touch controllers are not going to be used in the project. The reason is an inconvenience of using them during steering wheel rotation. Two sensors are usually used for moving around in a specific area with VR headset. However, such movement will not be supported in a current version of the project. Hence, in-game camera will have fixed position, but a free rotation angles.

3.1.5. Unreal Engine 4

Todays game engines are widely used not only in a gaming industry, but in manufacturing [19] and filming. The aim of those engines is to simplify development process in order to achieve more realistic picture and objects behaviour in a digital environment. It is closely related to a physics, 3D modelling and programming. Fortunately, modern game engine tools are focused on allowing developer to rapidly implement a complex systems. Those tools are saving a lot of time and provide some basic flexibility. Moreover, it is possible to implement own complex system from a scratch, which might be integrated later in the game engine.

Unreal Engine 4 is a free to use game engine that includes a numerous functionality for almost every field. In order to work with it, developer should have a basic undertanding of tools and C++, which is a main programming language for UE4. One of the UE4 features is a built-in physics for a wheeled vehicle. [20] Therefore, it becomes a base for the project. More details about used tools will be provided in

the following chapters.

3.1.6. Visual Studio 2019

Visual Studio (VS) is one of the most popular Integrated Development Environment (IDE) for various programming languages. It supports such languages as C, C++, C#, Visual Basic and more. As an IDE, it plays different roles in a programming. First of all, it is a text editor, where programmer can write a code and save the output in a specific file format. Secondly, it provides an interface for compilers, which would not require to write a numerous commands into the console to get a final program file. Lastly, VS might be used an environment for a program debugging. There are more functionality in IDE, but it will not be used in the case of this project.

Since UE4 is able to process C++ programs as an additional functionality, they will be first written and compiled in VS. As an example, TCP/UDP connection would require an explicit programming code, because UE4 does not have tools for it. The most of built-in UE4 functionality is written in a different program files, which can be seen and modified by a developer.

3.1.7. Matlab/Simulink

Matlab is a development environment for a technical field that consists of a numerous program packages in order to produce a mathematical or physical calculations. Simulink is a graphical extension for Matlab, which mostly focused on modeling and analyzing different systems. [21] Also, ABS is a real-time system that is perfectly fits to Simulink modeling. [22] Hence, it becomes a crucial for INTECO ABS object and the project.

In addition, Simulink supports signal processing for ABS object using INTECO drivers. Unfortunately, the latest drivers available drivers are stable on Matlab 2012b. Therefore, Simulink model will be based on the old version that might not be supported on a newer one. Moreover, Simulink has an ability to establish UDP connection between devices using standard block elements that represents a program functions mostly implemented in C language. This feature will play a key role in a

communication between devices that is actively used in the project.

3.2. Phase one

The initial goal of the project was to research and setup basic components that will be used later in a real-time application. Therefore, it was decided to provide a basic wheel control movement for ABS object. Subsections 3.2.1–3.2.4 will outline each component that is a crucial for the entire system. Also, subsection 3.2.5 lists the results that were achieved during development.

3.2.1. Initial preparations

Since there are no existing public papers related to the topic, it was decided to start with a simple model of ABS control system. The idea was to make a system that allow user to control ABS device via a real-time application, where control comes from Logitech G29 Driver's pedals. The application itself is based on Unreal Engine 4, which supports Logitech G29 and Virtual Reality environment for a further work. Unfortunately, ABS device requires an elder operating system while UE4 has to be run on a newer one, which leads to use of a two personal computer setup. Hence, the connection for the data transmission must be additionally installed. Figure 4 shows basic elements of an initial setup.

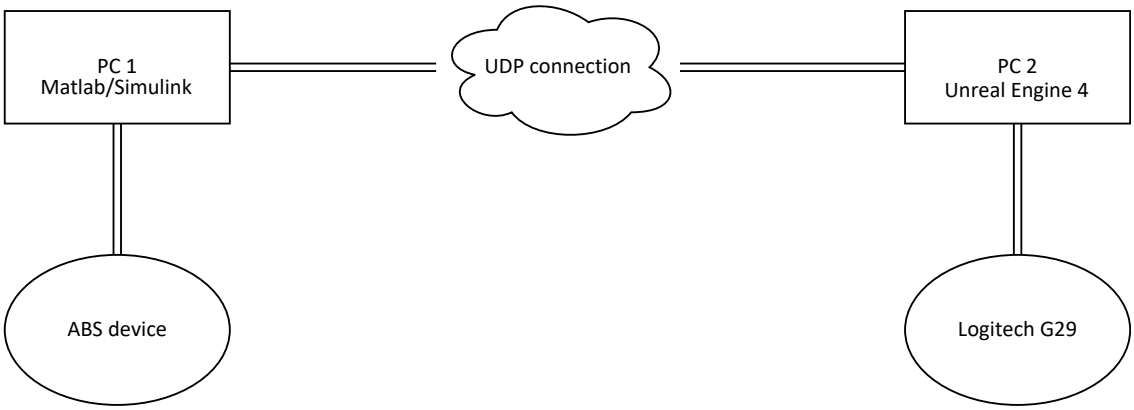

Figure 4. First version of communication setup between hardware and software components

3.2.2. First real-time application

After the first preparations, the next step was establishing a logical behaviour for the application that gathering signals from the pedals. There were two different ways of completing this task. First option is a development of a Blueprint in Unreal Engine 4, which behaves similar to scripts with elements of programming. Second option is writing a pure code in C++. The first approach was chosen, because it is more convenient, faster and allows developer to see execution nodes during debugging. Finally, the application had a functionality that is illustrated in Figure 5.

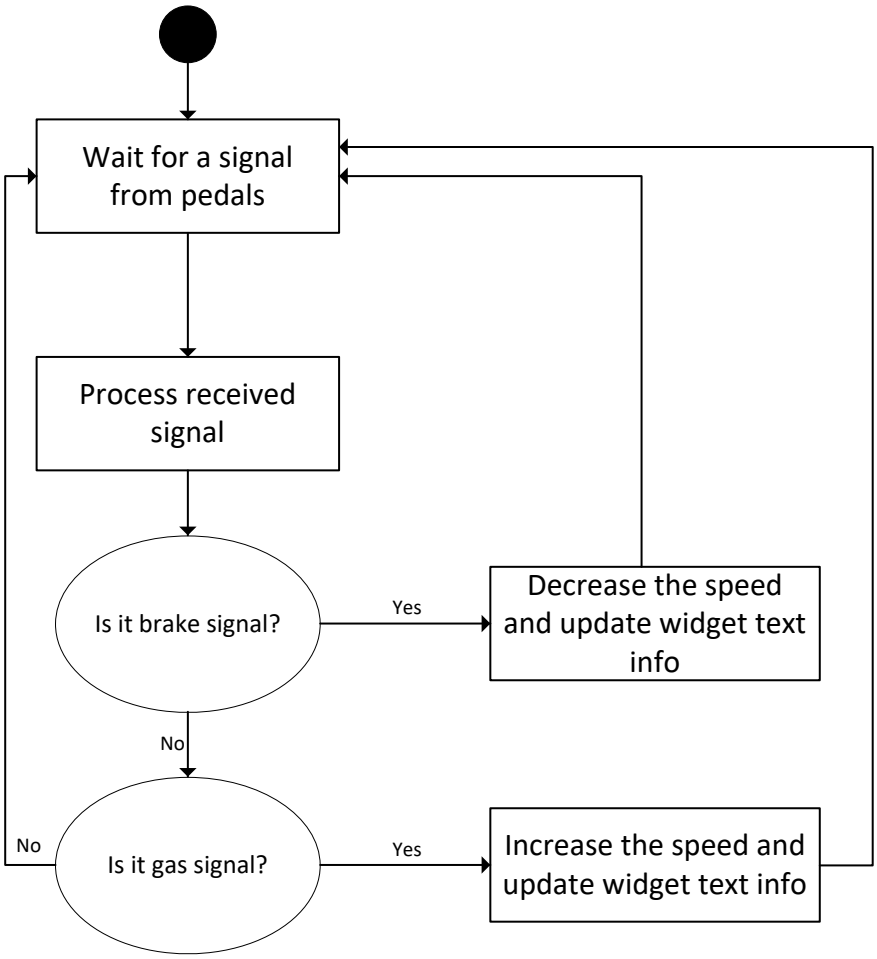

Figure 5. Chart flow of processing the signal from pedals

3.2.3. ABS Device in a digital environment

The major objective of current project is a Digital Twin of ABS provided by INTECO for studying purposes. There are several benefits of having DT for this model. First of all, the construction of the device is quite heavy and large for a small room. Secondly, it grants no protection from a hazardous behaviour such as breaking down mechanisms during experiments. Thirdly, the device is not easy replaceable due to taking a lot of time to fix or getting a new one. Lastly, the model is limited by a single wheel, which almost makes impossible to create and test ABS algorithms simultaneously for a car with four wheels.

On the other hand, laboratory ABS has numerous advantages such as double-wheel model, where one wheel represents a road surface and another is a representation of a car wheel. Moreover, it is equipped with DC flat motor, which may be used in the car velocity range from 0 to 50 km/h. Also, two high-resolution measuring encoders allow to observe an important data. For example, car, wheel and bump positions with a velocity of the car and the wheel. In addition, INTECO provides ABS Control/Simulation Toolbox for Matlab/Simulink environment that significantly increase a speed and flexibility of creating new algorithms for ABS. (Official web-page link)

At this point, the project task was to learn the basic functionality of ABS device, which later will be used in a more complex system. Since the main goal is DT, it also requires 3D model of the device. In this case, it was created in Unreal Engine 4 from several basic objects and textures. The final combination of those objects represents ABS 3D model with a ratio 1 to 1 (Fig. 6).

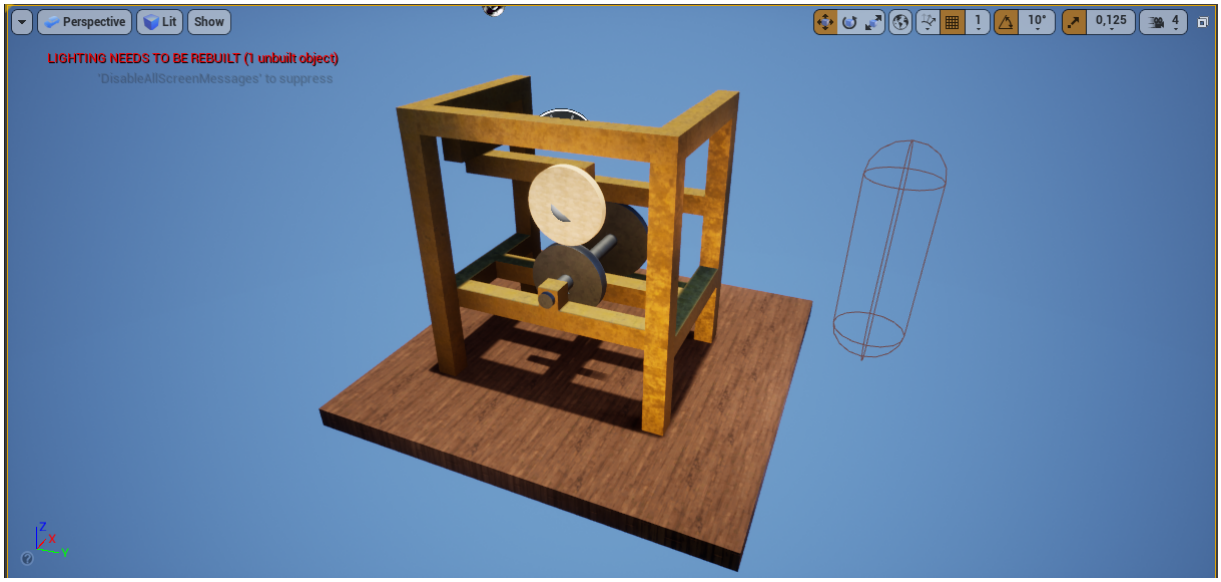

Figure 6. 3D model of ABS in Unreal Engine 4 Editor

3.2.4. UDP Connection

The data transmission between the real-time application and ABS device using two PC setup requires establishing of a network communication. There can be used two types of connection: User Datagram Protocol (UDP) and Transmission Control Protocol (TCP). The choice has fell on UDP, because the real-time application depends on a high speed. Moreover, since the workflow takes place locally in a laboratory, there is no need in additional packages control. [23]

The connection itself based on plugins for Matlab/Simulink and Unreal Engine 4. It is written in C++ and uses Windows Sockets API. The plugin consists of a class with different data types and functions. Class variables are used for containing an actual data that is received by specified function and sending as a formed data package. Also, plugin allows to configure IP addresses and ports inside the programs (Fig. 7), which makes it more flexible.

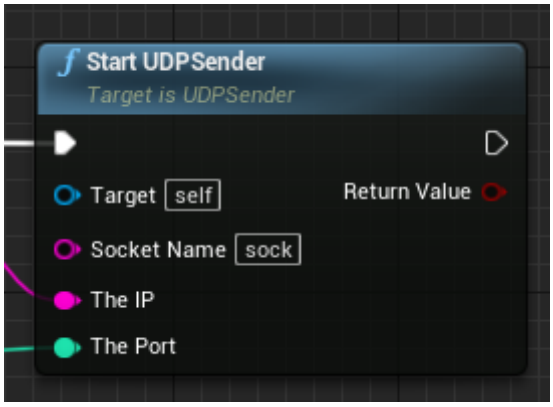

Figure 7. UDP configuration block in UE4

3.2.5. Results of the initial setup

Now, it was a time to connect all parts into a single complex system. The following functionality was achieved in the final version of UE4 application:

- In addition to gas/brake control from pedals was added keyboard support in order to test and debug the project without using an actual Driver's seat;
- Gas signal now obtaining more accurate data that shows the position of pressed pedal based on its angle. For example, 0 — pedal is released, 0.5 — half pressed pedal, 1 — pedal is fully pressed;
- Established UDP connection;
- Extended text information. For example, now it gathering actual data from ABS device such as brake's value and wheel velocity in RPM via network communication;
- Pedals from Logitech G29 Driver's seat are able to accelerate ABS device's wheel and send brake signal;
- Added 3D model of ABS to the project and expanded by a visual animation of rotational wheels that show a realistic behaviour of ABS device.

Moreover, several changes were made in Matlab/Simulink model:

- Established UDP connection;

- Added relay controller for braking system that is provided by INTECO;
- Processing received signals for relay controller from Unreal Engine 4 application;
- Sending feedback data to UE4 such as ABS device brake statement and current wheel's velocity.

3.3. Phase two

The first step showed and opened the possibility to use basic controls for ABS object. However, it does not give much information about the "real" system behaviour. Currently, the only way of observing data output is to watch graphs in Simulink model or text data on the screen. Therefore, it was decided to make a more immersive testing environment. This step is mostly focused on making the application more friendly to any user and flexible for further researches. [24]

3.3.1. Adding plots

To improve system's output observation, it is better to have all important information on a single screen. Hence, all the plots from Simulink model must be transfered to UE4 application interface. Unfortunately, game engine does not have tools for making charts, but there are existing free to use plugins shared by other developers. One of those called Kartan Charts, which is based on Unreal Motion Graphics (UMG). It is able to print out a real-time data as an UI element (Fig. 8).

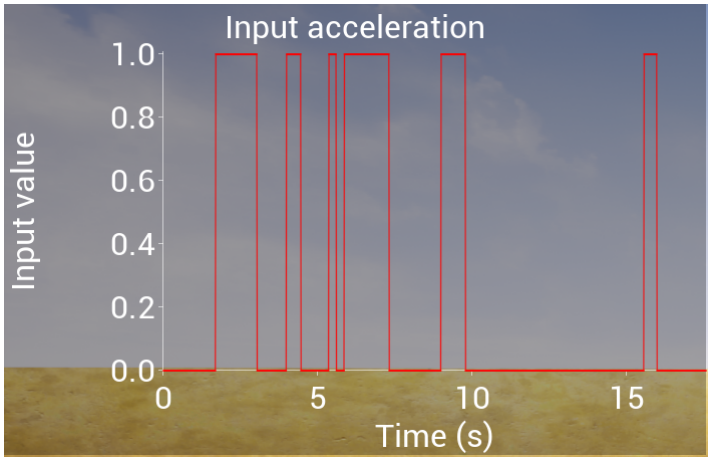

Figure 8. Acceleration input plot

Another feature of this plugin is an ability to change line color in a run time, so it will outline the moment, when the car received a brake pedal signal. Figure 9 [25] shows incoming brake signal as a green line, while red line is a released brakes state. This plugin is a powerful tool for the most developers needs since it is almost fully customizable including the values range for each axis, font, size, color of elements and more. Finally, the plugin eliminates the need of moving to another PC to observe the plots output during debugging.

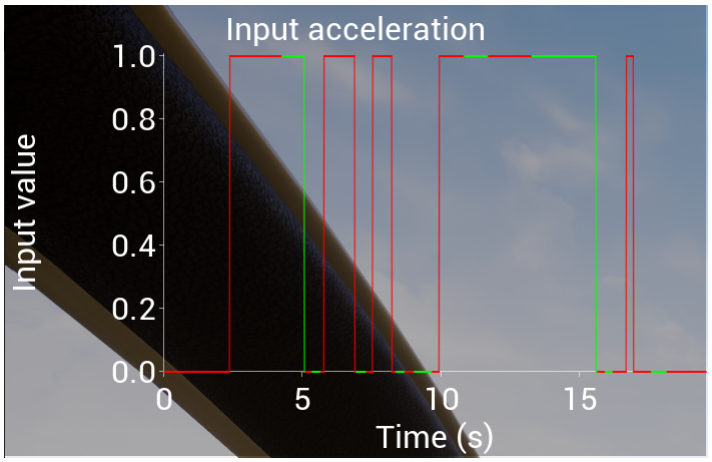

Figure 9. Visualization of the acceleration input with brake signals using a modified version of the KantanCharts plugin

3.3.2. Immersive environment

Previously, it was discovered that current INTECO ABS object has a wheel number constraint. However, normally ABS cannot be tested without a car. At this point, there is an option to recreate a car, which would use a single braking signal for all wheels. In reality, it is absolutely wrong approach, but a single INTECO ABS setup leaves no alternatives. Therefore, the car will be added to the application using default physical control parameters provided by UE4. Also, some changes to the default parameters must be done in order to get a realistic behaviour. It is important to mention that the project will simulate behaviour of ABS object, so in-game car movement might not look like in a real car. The reason of divergence is in the implementation of ABS object, which is not allowing to make any change in the force generated by the upper wheel.

In addition to the car, a new testground must be created. For current goals, it would be enough to have a straight road surface with walls around. Figure 10 shows the

final version of the testground.

Figure 10. Testground with a space to test car dynamics

3.3.3. User Inteface

Another important aspect of the user friendly environment is a User Interface. First of all, a common element of each application is a main menu that gives a basic understanding of what user can get inside the app. Currently, project application has 2 different working sections that are related to step one and two respectively (Fig. 11). It was done in this way due to having a massive rework between initial application and a final version. This menu is working in a dedicated game level scene, so it would not load extra objects that are activly used only on a specific scene. Also, the common principle of implementing UI is to switch between several widgets. This is done by changing a visibility for a specific widget, which is usually triggered by some event.

Figure 11. Main menu of the application

Secondly, the first version of the app required changing IP address and port number for UDP connection manually inside a Blueprint. Now, it is possible to do in the main menu as well (Fig. 12). Moreover, every element of the menu is fully controllable using only keyboard keys. In contrast to default input lines, UDP plugin helps with checking the data for errors such as wrong IP structure input.

Figure 12. IP and port setup menu

However, it does not check port number for correctness, so additional check was implemented using BP elements. The logic is to get an actual string from the widget, which later becomes converted to the integer (Fig. 13). If the value was not numeric

then conditional blocks will never give a “true”. Also, the number has a limited range of values (9900-9999) in order to prevent using ports that are already occupied by different communication networks.

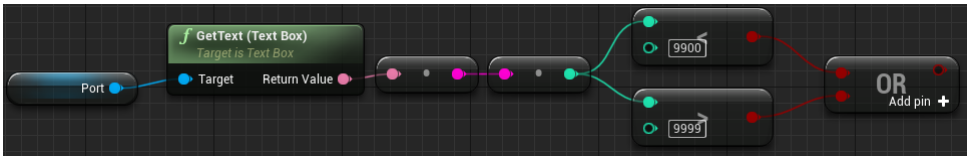

Figure 13. Port number checking logic in BP

Unfortunately, it might become inconvenient for a user to enter IP and port every time, when application is launched. Therefore, UE4 provides additional tool that creates a slot with saved game data. It is based on the object that inherits a base class from selected Blueprint (BP). In this case, the object has own function, which gets three input signals: IP as a string, Port as an integer and Save Game Instance as an object itself (Fig. 14). Those slots should also have a personal slot name and user index to get uniqueness. As an output, the function provide a boolean that tells if the operation was succeed or not.

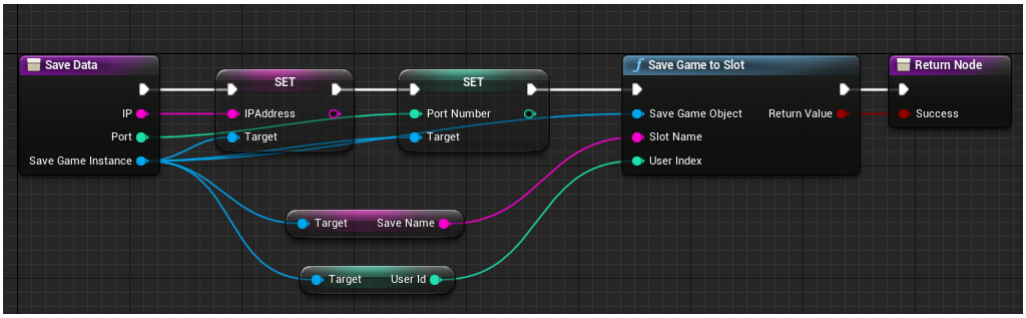

Figure 14. Save Data function structure in BP

Lastly, the project has different plots that might be used by the end user, but there was no option to enable them in a run time. Hence, UI easily helps in this situation and makes things more understandable. Figure 15 shows one of the latest versions of graph selection menu. Also, it shows numerous plots that Simulink model provides as an output.

Figure 15. Graph selection menu

3.3.4. Testground

The initial goal for the testground was to have a track with different road surfaces that would affect on ABS object behaviour. Unfortunately, practice showed that INTECO ABS is not able to safely process road surface changes in a real time. On the other hand, UE4 has an option to change the tires friction on a specific surface. Therefore, it might be useful for researchers with another ABS object. Nevertheless, the testground is an extremely important element for the project.

Even if INTECO ABS object cannot change road surface in the real time, it still shows a wheel behaviour that is close to the real one. Hence, user may observe approximate braking distance and acceleration that is happens with a fixed to place ABS object. In this case, the only way to get an output using different "road" surface is to replace the object's wheel manually.

3.3.5. Physics of the wheeled vehicle

As it was previously mentioned, UE4 provides its own version of car physics. Since the complexity of such systems is extremely high and requires a good knowledge of physics, it was more convenient to rework already existing system than create a new one. Moreover, ABS model is not a perfect, so there will be some inaccuracies

in calculations. In contrast to UE4 car, INTECO ABS does not count vehicle transmission setup. Therefore, the first major change of the car behaviour was making a vehicle with a one gear. Another change happened in engine setup (Fig. 16), where maximum velocity was increased via experiments in order to achieve a similar acceleration to ABS object. The final goal of UE4 car acceleration was to make it faster than in the object, so it became possible to add a real time speed limitation, which depends on the current ABS object speed.

Figure 16. Car mechanical setup

3.3.6. Acceleration and braking of the car

In a real physical model the process of acceleration and braking might be too complicated for a casual driver. To avoid the whole complexity of the system, UE4 provides a control of the car via different functions such as throttle and brake inputs. It is important to notice that default wheeled vehicle movement class does not allow to change car parameters in a real time, which is closer to the reality. Therefore, it makes debugging more complicated, because every changed paremeter will require a full restart of the application. Nevertheless, it is possible to reach a close to ABS object acceleration and braking behaviour. For debugging purposes, the project supports two versions of vehicle movement. The first is automaticaly activated, when UDP pluging receives the data from Simulink model. Hence, UE4 car will use acceleration and braking output signals from INTECO object (Fig. 17). The idea of using a branch block is to synchronize acceleration process of the car with ABS object.

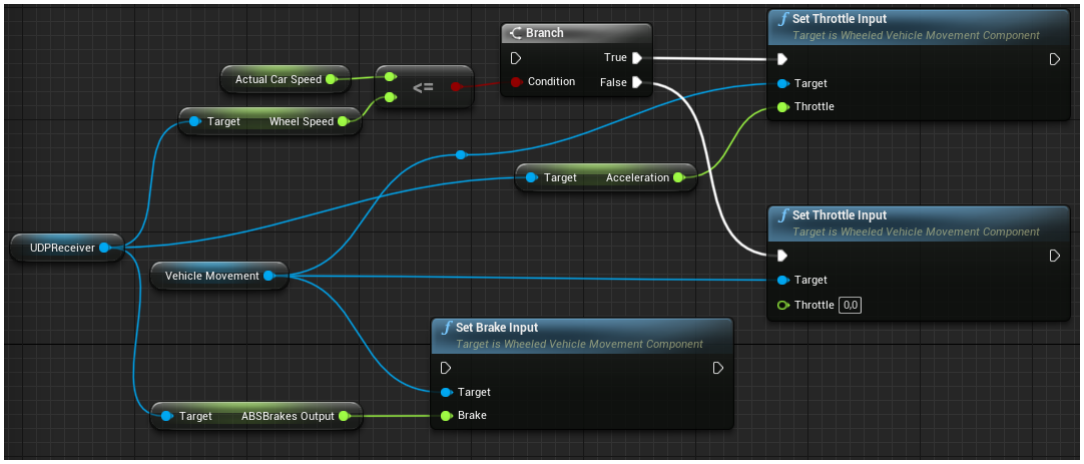

Figure 17. Acceleration and brake input behaviour in BP

Otherwise, the second version of the control system will be activated, which uses a straight input signals from the keyboard or Logitech G29 pedals. Moreover, it does not provide any stability of the car, because UE4 default model does not have an ABS. Unless the project has the data transmission between UE4 and Simulink, it would be useful to have a conditional block to set a maximum speed of the car. This is done by using engine velocity output in RPM, which make a control more accurate. Figure 18 shows that current engine speed is limited to 2000 RPM.

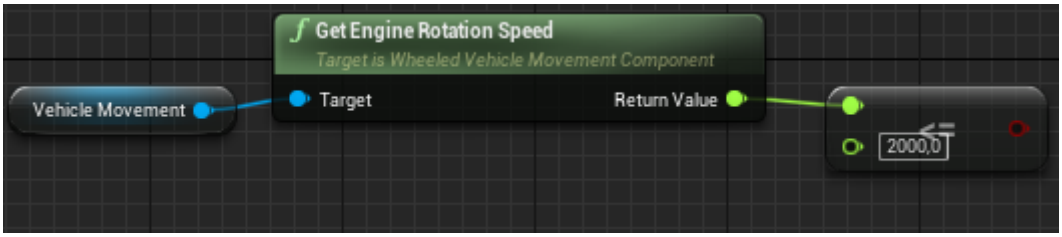

Figure 18. Speed limit in BP

The car can use a handbrake as well, but in the case of current project it will not play a key role. However, the application might be used for further researches with a different car control model, where handbrake become a crucial for the system. Figure 19 shows the basic input control setup for the handbrake that is based on project settings key bindings.

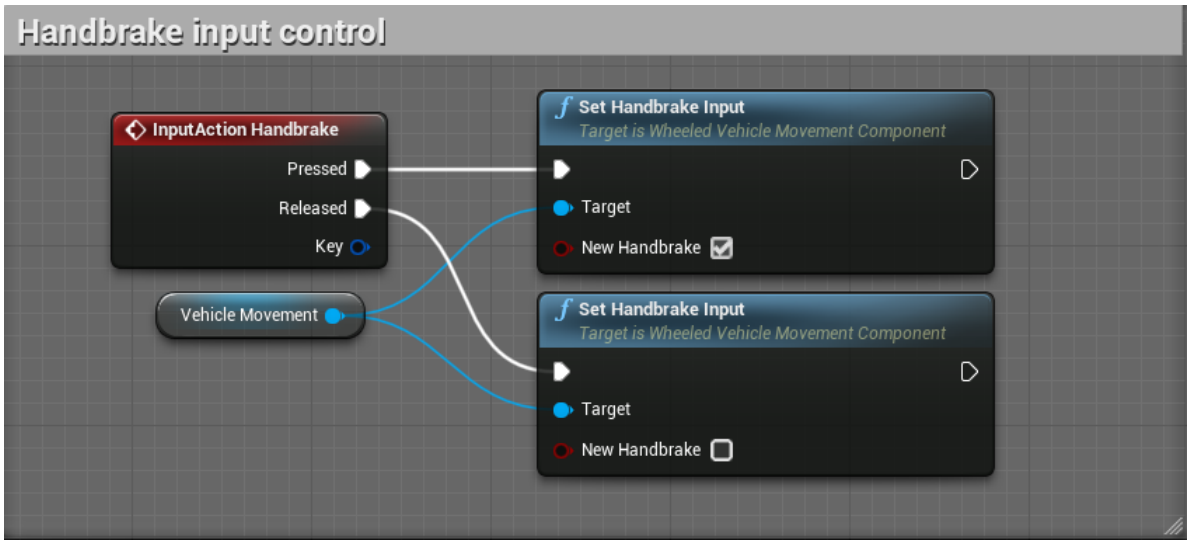

Figure 19. Handbrake intial control setup

3.3.7. Logitech G29 control

INTECO ABS object is capable of processing floating numbers, but the keyboard does not support analog output. At this point, Logitech G29 provide a good option to make a testing enviroment more immersive. Furthermore, it has a steering wheel, which produces analog output as well. Previously, it was mentioned that acceleration and brake inputs are getting their values from ABS object, but the object itself must receive an input signal. Hence, the initial signal for acceleration or braking must come from pedals. Figure 20 demostrates the sequence of processing control

data input for Logitech G29 pedals. To understand how does system works without connected ABS object, Figure 21 provides a flowchart of the system.

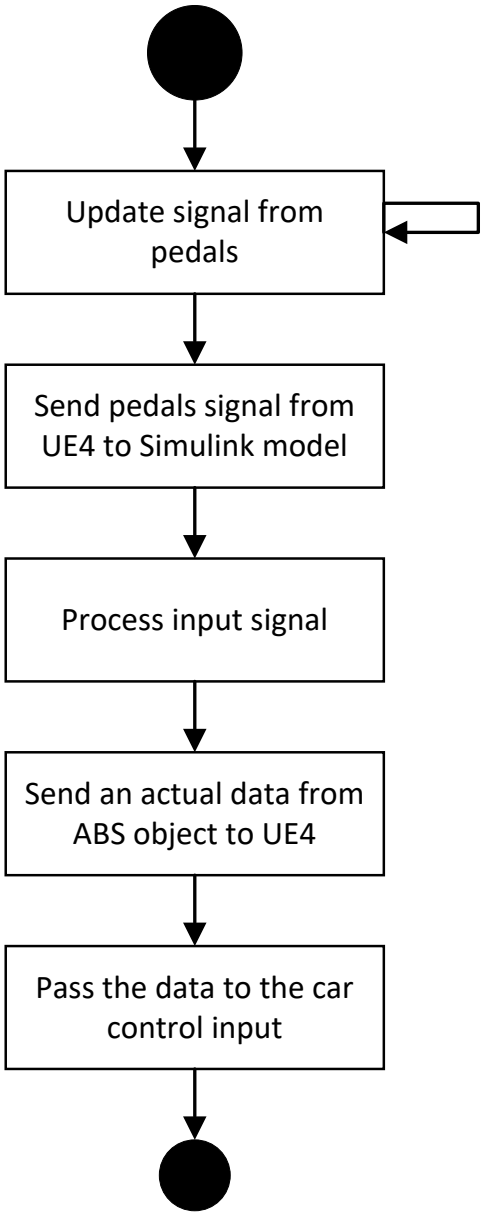

Figure 20. Flowchart of processing pedal signals

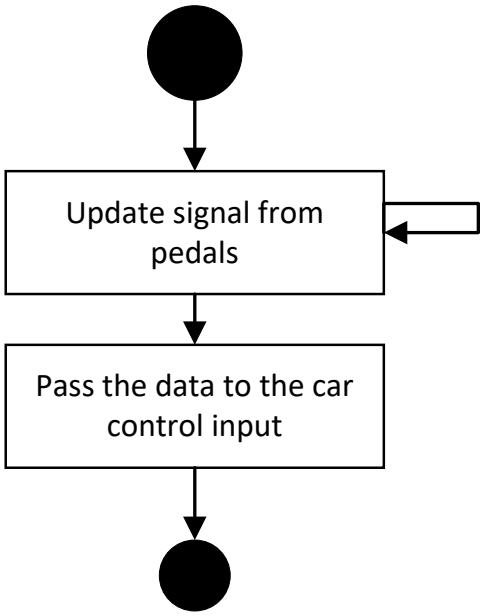

Figure 21. Flowchart of processing pedal signals without ABS object

In addition, steering wheel rotation control is not implemented in UE4. It requires additional efforts from developer to construct a realistic behaviour for the wheel. There are several points that should be considered during implementation. First of all, a real steering wheel is not just rotating, it has a limited rotation angle range. In a regular car this range is from -450° to 450°, which might be done by using default function called Clamp Angle. Secondly, rotation speed must be simulated due to using keyboard input as well. It is a common practice to slow down the rotation by a constant number, so it would avoid sharp wheel rotation. Finally, Logitech steering wheel rotation will look smooth in the application.

3.3.8. Additional car features

Since the project supports both VR and desktop versions of display, more functionality must be added to the camera view. VR features will be outlined in the following subsection. Firstly, camera must follow the car all the time. It is easy to implement, because UE4 has built-in camera objects that are crucial for almost every type of application and adding it to the car mesh already make it “stick” to the object. Camera behaviour for both X and Y axis is demonstrated in figures 22 and 23

respectively. Furthermore, the project provides two different camera positions: inside and outside the car.

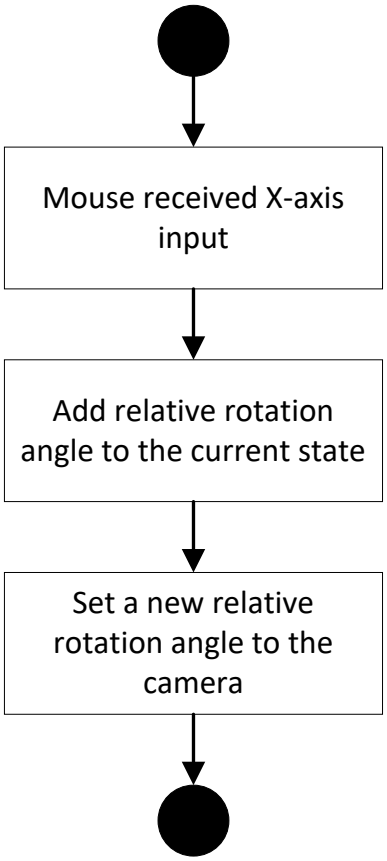

Figure 22. Flowchart of processing camera X-axis rotation

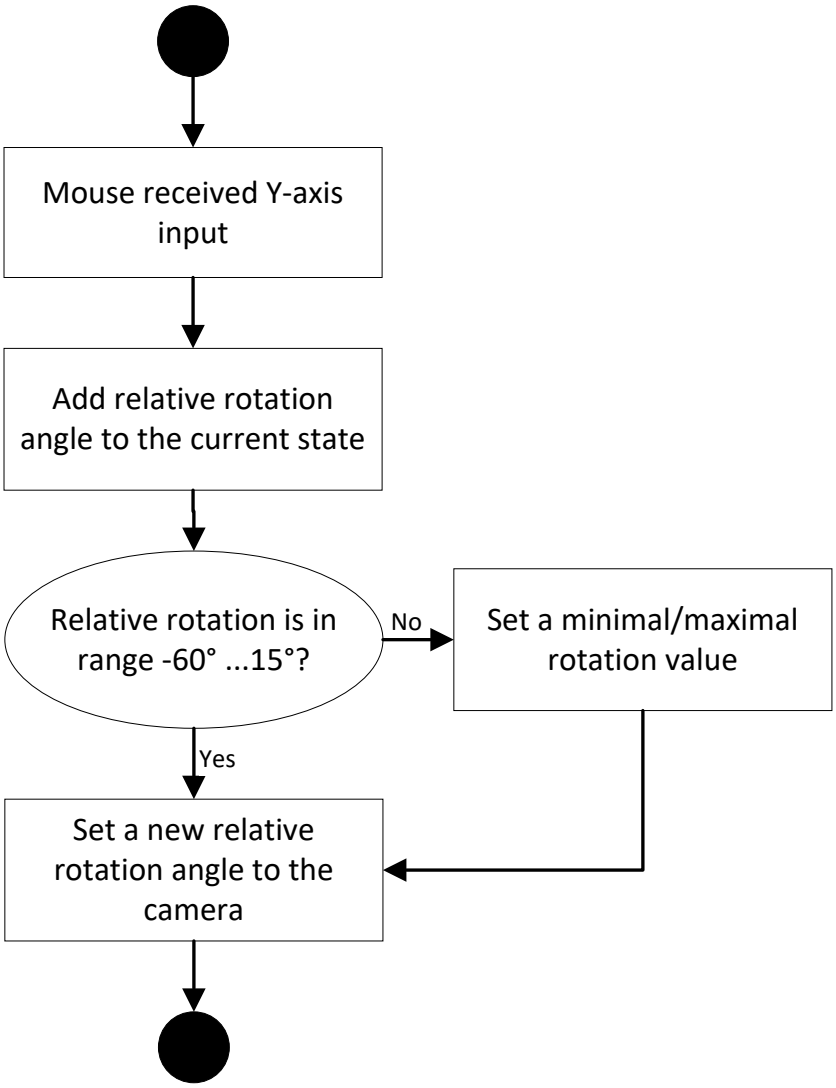

Figure 23. Flowchart of processing camera Y-axis rotation

Another important update was adding the lights, which are connected to the events such as stop-signal and reverse gear. The light object is a default tool in UE4 development environment and it has a numerous options that might be individually customised. Figure 24 shows a base options for the light.

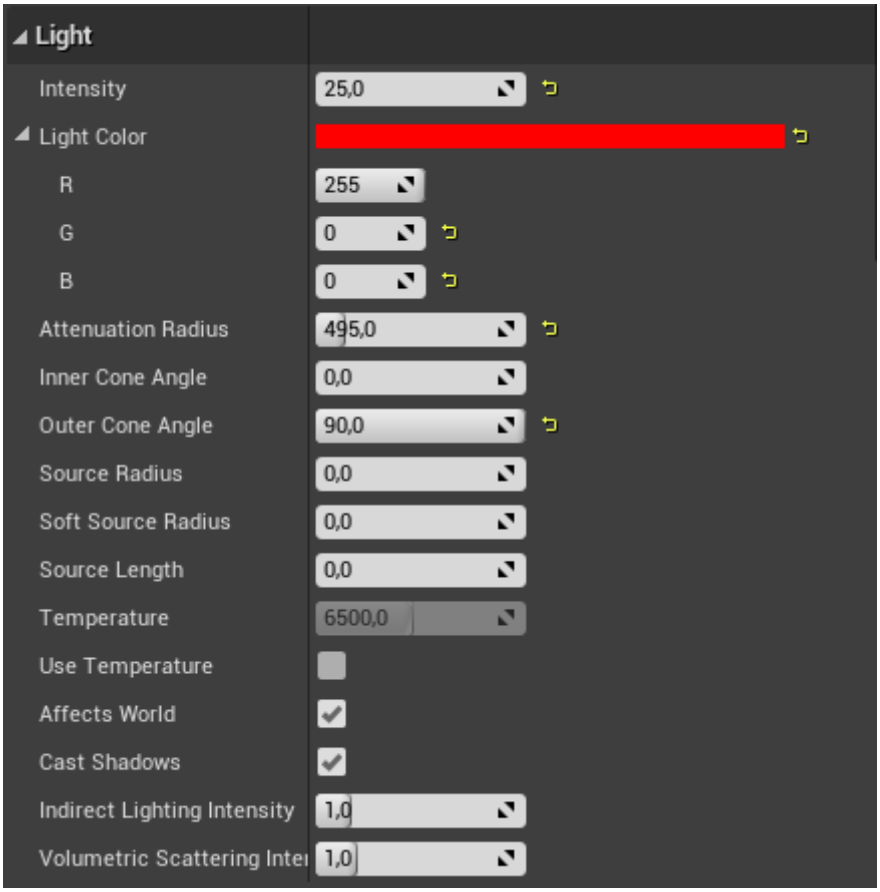

Figure 24. Light setup parameters in UE4

One of the latest features in the project were speedometer and tachometer (Fig. 25). This part is a crucial for every car system and in order to make digital environment more interactive to the user, it will provide a real information about the current speed and engine velocity.

Figure 25. Speedometer and tachometer of the car

3.3.9. VR support

The latest development of the project was Virtual Reality. The first problem that developer might notice is the fact, where user cannot use controllers and steering wheel simultaneously. Hence, it was decided to exclude using VR controllers from the project. Furthermore, VR cameras are not playing any role in the application, because user will always sit in a driver's seat. However, Head Mounted Display (HMD) in combination with Logitech Driver's seat is able to increase immersiveness to the next level. The only restriction for the VR camera is the position, which is locked to the car's seat. In contrast to the desktop display camera, user can rotate freely in every direction (Fig. 26).

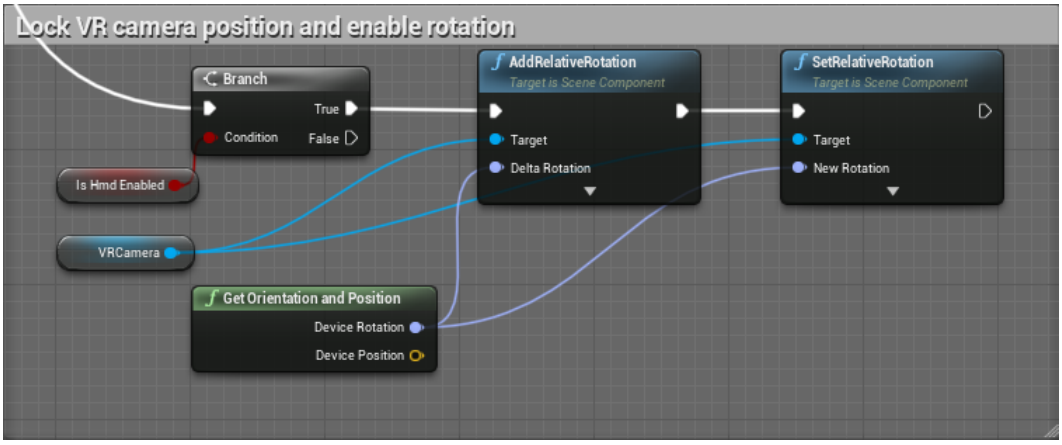

Figure 26. VR camera behaviour

3.3.10. Second step results

To sum up the second step of the project, the following features were implemented:

- Plots that include the data about acceleration/brake input from pedals, ABS object's wheel/car velocity in RPM, passed distance by the "car", slip coefficient, ABS object's brake/speed output and UE4 car's speed. Furthermore, the application supports displaying of two plots simultaneously;
- Testground with two types of road surfaces. For the current project version they would not affect on INTECO ABS object;
- Two additional game level scenes: main menu and ABS simulator level;
- Interactive user interface in main menu with the ability to save data and graphs selection menu in a run time;
- Fully functional car in two different control modes: using connected ABS Simulink model and default car control;
- Full support of Logitech G29 pedals and steering wheel with realistic rotation behaviour;
- Car lights, speedometer, tachometer and different car cameras;
- Support of VR HMD that simulates driver's position field of view.

4. Conclusion

To conclude the finished work, it has a potential to become a base for further researches. Also, some changes might be done in a setup including ABS object that should be improved in order to support a control of four-wheeled car. Unfortunately, INTECO ABS does not support modern operating systems and Matlab versions, so it is less practical to use in manufacturing. Nevertheless, it is still provides an ability to create own slippage control systems that would work in a modern systems. As an alternative to the using real ABS object, INTECO has a simulation model for Simulink, but it would not be a real Digital Twin.

In addition, there are still several missing points in UE4 application. First of all, Virtual Reality HMD has a problem with displaying a 2-dimensional (2D) widgets. Therefore, it will require to replace 2D elements with 3-dimensional (3D). Another problem related to both UE4 application and ABS object side is the lack of real slippage behaviour. It means the user cannot observe realistic skid of the car. The reason is ABS object's movement constraint, because it cannot make a side turn. Fortunately, UE4 has an option to demonstrate such behaviour, but with a current ABS object it is hard to predict skid trajectory.

Also, more features to the car might be added such as tire tracks during braking. Then, testground should be updated in order to provide a more interactive environment, where the user can test the brakes with and without enabled ABS. Unfortunately, the application has a lack of several UI elements, so it creates difficulties for those, who is not familiar with a key controls. Lastly, displaying plots could be reworked since currently they are not movable in a run time.

Finally, ABS is an extremely important topic and the cost of testing such system is too high. Therefore, creating of a Digital Twin is an unique project for each object. However, new researches and published projects will definitely help developers, who is interested or going to work on a creating ABS simulation environment. Therefore, this project is able to help in a choosing of tools for connection establishment, system modelling and debugging.

References

[1] Z. Wei and G. Xuexun, "An ABS control strategy for commercial vehicle," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 1, pp. 384–392, feb 2015.

[2] J. Sinay and Z. Kotianová, "Automotive industry in the context of industry 4.0 strategy," *TRANSACTIONS of the VŠB – Technical University of Ostrava, Safety Engineering Series*, vol. 13, no. 2, pp. 61–65, sep 2018.

[3] J. Radianti, T. A. Majchrzak, J. Fromm, and I. Wohlgenannt, "A systematic review of immersive virtual reality applications for higher education: Design elements, lessons learned, and research agenda," *Computers & Education*, vol. 147, p. 103778, apr 2020.

[4] F. Tao, H. Zhang, A. Liu, and A. Y. C. Nee, "Digital twin in industry: State-of-the-art," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2405–2415, apr 2019.

[5] A. Rasheed, O. San, and T. Kvamsdal, "Digital twin: Values, challenges and enablers from a modeling perspective," *IEEE Access*, vol. 8, pp. 21 980–22 012, 2020.

[6] R. Minerva, G. M. Lee, and N. Crespi, "Digital twin in the IoT context: A survey on technical features, scenarios, and architectural models," *Proceedings of the IEEE*, vol. 108, no. 10, pp. 1785–1824, oct 2020.

[7] V. Liagkou, D. Salmas, and C. Stylios, "Realizing virtual reality learning environment for industry 4.0," *Procedia CIRP*, vol. 79, pp. 712–717, 2019.

[8] F. Bellalouna, "New approach for industrial training using virtual reality technology," *Procedia CIRP*, vol. 93, pp. 262–267, 2020.

[9] S. Yao, J. Zhang, Z. Hu, Y. Wang, and X. Zhou, "Autonomous-driving vehicle test technology based on virtual reality," *The Journal of Engineering*, vol. 2018, no. 16, pp. 1768–1771, nov 2018.

[10] S. Oprea, P. Martinez-Gonzalez, A. Garcia-Garcia, J. A. Castro-Vargas, S. Orts-Escolano, and J. Garcia-Rodriguez, "A visually realistic grasping system for object manipulation and interaction in virtual reality environments," *Computers & Graphics*, vol. 83, pp. 77–86, oct 2019.

[11] D. Sportillo, A. Paljic, and L. Ojeda, "Get ready for automated driving using virtual reality," *Accident Analysis & Prevention*, vol. 118, pp. 102–113, sep 2018.

[12] M. Hlavaty, A. Kozakova, and D. Rosinova, "Efficient fuzzy control of a laboratory ABS," in *2018 Cybernetics & Informatics (K&I)*. IEEE, jan 2018.

[13] Z. Taixiong and Z. Yage, "Development of hardware-in-loop and virtual reality co-simulation platform for automotive anti-lock braking system," in *IET International Conference on Information Science and Control Engineering 2012 (ICISCE 2012)*. Institution of Engineering and Technology, 2012.

[14] A. Challa, K. Ramakrushnan, S. C. Subramanian, G. Vivekanandan, and S. Sivaram, "Analysis of thresholds in rule-based antilock braking control algorithms," *IFAC-PapersOnLine*, vol. 53, no. 1, pp. 404–409, 2020.

[15] A. Gohar and S. Lee, "A cost efficient multi remote driver selection for remote operated vehicles," *Computer Networks*, vol. 168, p. 107029, feb 2020.

[16] M. Martinez-Gardea, C. Acosta-Lua, I. Vazquez-Alvarez, and S. di Gennaro, "Event-triggered linear control design for an antilock braking system," in *2015 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC)*. IEEE, nov 2015.

[17] INTECO. (2021) Abs antilock braking system. [Last access: 04.05.2021]. [Online]. Available: http://www.inteco.com.pl/products/abs-antilock-braking-system/

[18] M. Goadrich. (2018) Oculus rift vr space setup. [Last access: 04.05.2021]. [Online]. Available: http://mgoadric.github.io/oculus/unity/2018/10/16/oculus-setup.html

[19] B. Nash, A. Walker, and T. Chambers, "A simulator based on virtual reality to dismantle a research reactor assembly using master-slave manipulators," *Annals of Nuclear Energy*, vol. 120, pp. 1–7, oct 2018.

[20] D. Michalik, O. Mihalik, M. Jirgl, and P. Fiedler, "Driver behaviour modeling with vehicle driving simulator," *IFAC-PapersOnLine*, vol. 52, no. 27, pp. 180–185, 2019.

[21] A. Martyanov, E. Solomin, and D. Korobatov, "Development of control algorithms in matlab/simulink," *Procedia Engineering*, vol. 129, pp. 922–926, 2015.

[22] A. Digrase and A. Wayse, "Model following control for anti-lock braking system with inertial delay observer," in *2017 International Conference on Intelligent Computing and Control Systems (ICICCS)*. IEEE, jun 2017.

[23] P. Bisták, M. Halás, and M. Huba, "Modern control systems via virtual and remote laboratory based on matlab," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 13 498–13 503, jul 2017.

[24] H. W. Alomari, V. Ramasamy, J. D. Kiper, and G. Potvin, "A user interface (UI) and user eXperience (UX) evaluation framework for cyberlearning environments in computer science and software engineering education," *Heliyon*, vol. 6, no. 5, p. e03917, may 2020.

[25] C. Angus. (2021) Kantan charts. [Last access: 04.05.2021]. [Online]. Available: https://github.com/kamrann/KantanCharts/tree/feature/per-point-colors

A. Non-exclusive licence

I Mark Chernyaev

1. Grant Tallinn University of Technology free licence (non-exclusive licence) for my thesis "Digital Twin for ABS control models" , supervised by Aleksei Tepljakov and Saleh Alsaleh

   (a) to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright;

   (b) to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.

2. I am aware that the author also retains the rights specified in clause 1 of the non-exclusive licence.

3. I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

B. Project repository and control keys

The latest version that was used during writing the document: `https://drive.google.com/file/d/1dkfW5Fr89JFjkt2BZHG1hCzs1p-F_mq5/view?usp=sharing`

Control keys in the application:

1 - Outside camera of the car (works only if VR HMD is off)

2 - Inside camera of the car (works only if VR HMD is off)

W - Gas input

S - Brake input

A - Turn left

D - Turn right

Spacebar - Handbrake

R - Reverse gear

P - Graph selection menu