DEGREE PROJECT

Methodology for Real Time Simulations of Autonomous Utility Vehicles

Johan Borgström

Mechanical Engineering, master's level
2020

Luleå University of Technology
Department of Engineering Sciences and Mathematics

LULEÅ UNIVERSITY OF TECHNOLOGY

Preface

This master thesis was written by Johan Borgström during the first semester of 2020 as a final course in M.Sc. Mechanical Engineering, specialisation Machine Design, at Luleå University of Technology. The master thesis is a part of a research project where Luleå University of Technology collaborates with University of Oulu, SINTEF Narvik and Oulu University of Applied Sciences. The goal with the research project is to develop a Nordic platform for development of autonomous, environmental friendly and energy efficient heavy vehicles. Supervisors of the master thesis have been Magnus Karlberg, Håkan Lideskog and Mats Näsström, all three working at the Division of Product and Production Development at Luleå University of Technology.

I would like to thank my three supervisors for all advise and help concerning the project and the report, both at the weekly meetings and between them via email and discussions at their offices. I would also like to thank the PhD Student Mattias Lehto at the Division of Product and Production Development for feedback at the weekly meetings. At last I would like to send a great thanks to the engineers and software developers at Algoryx Simulation for their support and fast email responds regarding software licenses and answer to problems. Without their help the master thesis would not be close to finished yet.

Johan Borgström, 2020-05-26, Luleå

Abstract

This master thesis is a part of a research project where Luleå University of Technology (LTU) collaborates with University of Oulu, SINTEF Narvik and Oulu University of Applied Sciences. The goal with the research project is to develop a Nordic platform for development of autonomous, environmental friendly and energy efficient heavy vehicles in the forest, harbor and mining industry. The purpose with the master thesis is to assist LTU in their role in the research project.

The Nordic platform was positioned in the product development process, with the result that it could be useful in the fourth phase "Detail design" and in the fifth phase "Testing and refinement" in the Ulrich and Eppinger product development process.

A methodology has been developed, covering all necessary steps going from an assembly of a vehicle in an arbitrary CAD program to perform real time simulations (including HiL simulations) of the vehicle in Simulink.

The off-road research platform for forest- and agriculture applications developed by LTU was used as a case study in the master thesis. Applying the methodology on this platform showed that choosing correct simulation frequency is important and that graphics enabled in real time simulations requires large computational power.

Nomenclature

Abbreviation	**Description**
HiL	Hardware in the Loop
LTU	Luleå University of Technology
MiL	Model in the Loop
RCP	Rapid Control Prototyping
SiL	Software in the Loop
TFP	An off-road research platform for forest- and agriculture applications developed by LTU
TRL	Technology Readiness Level

File format	**Description**
.agx	A binary serialisation of an AGX Dynamics simulation
.mdef	An exported simulation from Siemens NX
.prt	A part file generated in Siemens NX
.scdoc	A file from Ansys Discovery SpaceClaim containing information (properties) from Algoryx Momentum
.sim	A simulation file generated in Siemens NX (in this report a simulation file from Siemens NX Motion)
STEP	A neutral format that enables part files to be imported in different CAD programs, independent of which CAD program the part file was generated in

Variable	**Description**	**Unit**
Δt	Largest allowed time step in explicit time integration	[s]
ρ	Density	[kg/m^3]
ω	Rotational speed	[rad/s]
C	Damper coefficient	[Ns/m]
E	Young's modulus	[Pa]
K	Spring coefficient	[N/m]
L_e	Element length	[m]
P	Power	[W]
T	Torque	[Nm]

Contents

1 Introduction

The development in the car industry and the heavy vehicle industry is in a fast and continuous change. Today the market demands competitive, environmentally friendly and energy efficient vehicles. In a project together with University of Oulu, SINTEF Narvik and Oulu University of Applied Sciences, Luleå University of Technology (hereafter called LTU) should develop a Nordic platform for development of autonomous, environmental friendly and energy efficient heavy vehicles in the forest, harbor and mining industry. This platform should be a middle step in the development process for such vehicles, where simulation environments are mixed with physical prototypes to enable the evaluation of the vehicle's subsystems before they are taken to expensive full scale tests in real environments. The Nordic platform will hereafter be called the mixed-VR system. By using the mixed-VR system manufacturers of heavy vehicles will, at an early stage, be able to test if their concepts will work and fulfil the demands that the real environment sets on the vehicles. This means that concepts that not reaches the demands can be discarded or changed/improved before to much work and resources are spent.

1.1 Background

Oulu University is today developing a so called Mobilab where heavy vehicles will be plugged into dynamometers. Together with physical environment simulations made by SINTEF Narvik, the vehicles responds to the challenges that Mobilab faces them with will be evaluated. SINTEF Narvik is developing a virtual environment that should describe where the vehicle is driving and which missions it has, whether the vehicle is a forklift in a harbor in Narvik or a forest machine in Kalix. The vehicles tested in the Mobilab will be able to connect four wheels and will not able to turn or be pitched or rolled in any direction in the Mobilab, thus turning and pitching and rolling of the vehicles will only be done in the virtual environment. To conclude, the heavy vehicles will be driven in their natural environments in the virtual environment and at the same time be subjected to a corresponding load by the Mobilab's dynamometers.

One responsibility for LTU in this research project is to develop the interface between the Mobilab and the virtual environment. This means to couple the virtual environment developed by SINTEF Narvik to the vehicle placed in the Mobilab and to Mobilab's dynamometers. This interface and coupling are hereafter called the VR system, and consists of both physical components and softwares. Figure 1.1 shows a schematic view of the mixed-VR system, and which institute that is responsible for which part in the mixed-VR system.

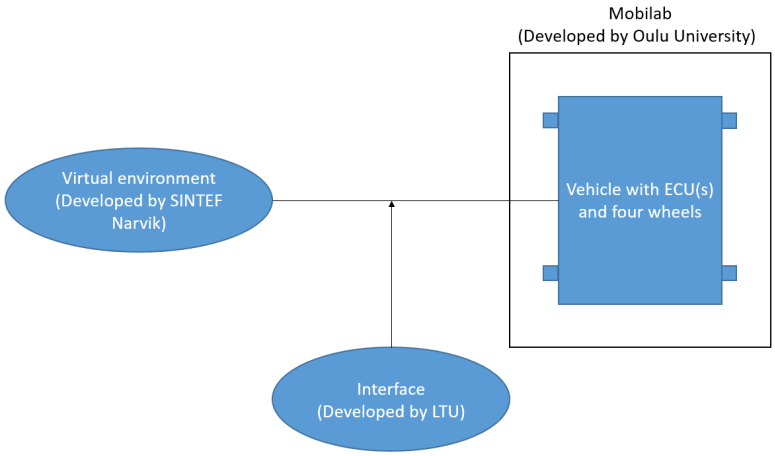

Figure 1.1: A schematic view of the mixed-VR system.

LTU has since a couple of years ago worked on developing an off-road research platform for forest- and agriculture applications (hereafter called the TFP), which in this Norwegian-Finnish-Swedish collaboration should be used to verify and validate the functionality in the mixed-VR system. The TFP is powered by a diesel engine that drives a hydraulic pump which delivers oil to four hydraulic motors, one motor per wheel. The TFP has got articulated steering and four pendulum arms which are manoeuvred by hydraulic cylinders. Figure 1.2 shows a rendered CAD image of the TFP developed by LTU.

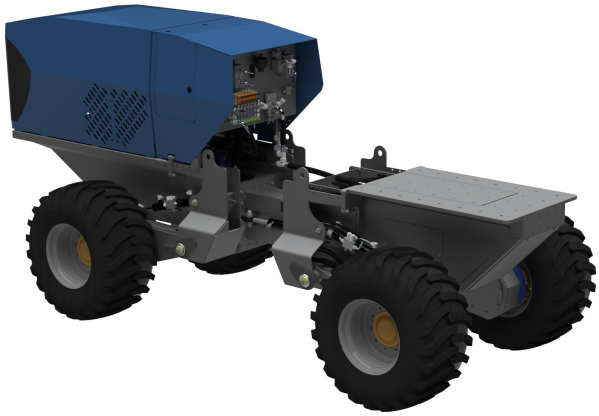

Figure 1.2: A rendered CAD image of the TFP developed by LTU.

More information about the history and development of the TFP can be read in [1].

1.2 Objectives and limitations

The objectives of this master thesis are:

- Describe how the mixed-VR system should be utilized in the product development process.
- Design the interface between the virtual environment and the physical components (vehicle and dynamometers) in the Mobilab.
- Test the VR system on the TFP.

The purpose with reaching the objectives above is to assist LTU in their role in the research project, that is to prepare the mixed environment (virtual environment and dynamometers) for future research about automation of heavy vehicles.

The limitations in this master thesis are:

- Assume that the vehicles that are tested in the mixed-VR system has got almost fully developed physical drivetrains including ECUs (controlling units) and almost fully developed controlling softwares.
- Assume that sensors in the vehicles that are tested in the mixed-VR system are known.
- All functions except driving (forward and backwards) and braking are performed only in the virtual environment. The driving and braking are also performed in the Mobilab.

The limitations above are important when positioning the mixed-VR system in the product development process and in the VR system design.

2 Theory

In this section the theory and terms used in this master thesis are presented and explained.

2.1 Product development process

The Ulrich and Eppinger product development process [2], is a stage gated process adapted for creative and innovative product development. The Ulrich and Eppinger product development process [2] covers six project phases, which are shown in Figure 2.1.

Figure 2.1: The six project phases in the Ulrich and Eppinger product development process [2].

The six project phases in the Ulrich and Eppinger product development process [2] are described in detail below:

0. Planning:

The planning activity is often referred to as phase zero because it precedes the project approval and launch of the actual product development process. The planning phase begins with opportunity identification guided by corporate strategy and includes assessment of technology developments and market objectives. The output of the planning phase is the project mission statement in which the target market for the product, business goals, key assumptions and constraints are specified.

1. Concept development:

In the concept development phase the needs of the target markets are identified, alternative product concepts are generated and evaluated and one or more concepts are selected for further development and testing. A concept is a description of the form, function and features of a product and is usually accompanied by a set of specifications, an analysis of competitive products and an economic justification of the project.

2. System-level design:

The system-level design phase includes the definition of the product architecture, decomposition of the product into subsystems and components and preliminary design of key components. Initial plans for the production system and final assembly are usually defined during the system-level design phase as well. The outputs of the system-level design phase are a geometric layout of the product, a functional specification of each of the product's subsystems and a preliminary process flow diagram for the product's final assembly process.

3. Detail design

The detail design phase includes the complete specification of the geometry, materials and tolerances of all of the unique parts in the product and the identification of all of the standard parts to be purchased from external suppliers. A process plan is established and tooling is designed for each part to be manufactured within the production system. The output of the detail design phase is the control documentation for the product, which includes the drawings or computer files describing the geometry of each part and its production tooling, the specifications of the purchased parts and the process plans for the fabrication and assembly of the product.

4. Testing and refinement

The testing and refinement phase involves the construction and evaluation of multiple preproduction versions (prototypes) of the product. The prototypes are built with production-intent parts and sometimes with parts supplied by the intended production processes. The prototypes are both evaluated internally and tested by customers in their own use environment. The goal for the prototypes is to determine whether the product will work as designed and whether it satisfies the key customer needs, and also to answer questions about performance and reliability in order to identify necessary engineering changes for the final product. The prototypes are also undergoing tests to obtain certification(s) of the final product.

5. Production ramp-up

In the production ramp-up phase the product is made using the intended production system. The purpose of the ramp-up is to optimise the workforce and to work out any remaining problems in the production processes. Products produced during production ramp-up are carefully evaluated to identify any remaining issues. The transition from production ramp-up to full scale production is usually gradual, and at some point in this transition the product is launched and becomes available for the market. A postlaunch project review may occur shortly after the product launch to identify ways to improve the development process for future projects.

2.2 TRL - Technology Readiness Level

TRL (Technology Readiness Level) is a scale often with nine levels originally developed by NASA [3], where TRL 1 is the lowest and TRL 9 is the highest level. The scale is used to measure the maturity level of a certain technology. NASA's technology projects are evaluated against the parameters in the TRL scale and are then assigned a TRL number based on the projects progress.

The TRL scale can also be used to describe the maturity levels for other technologies and projects than these related to the space industry. Figure 2.2 shows the nine levels in the TRL scale and the description of each level.

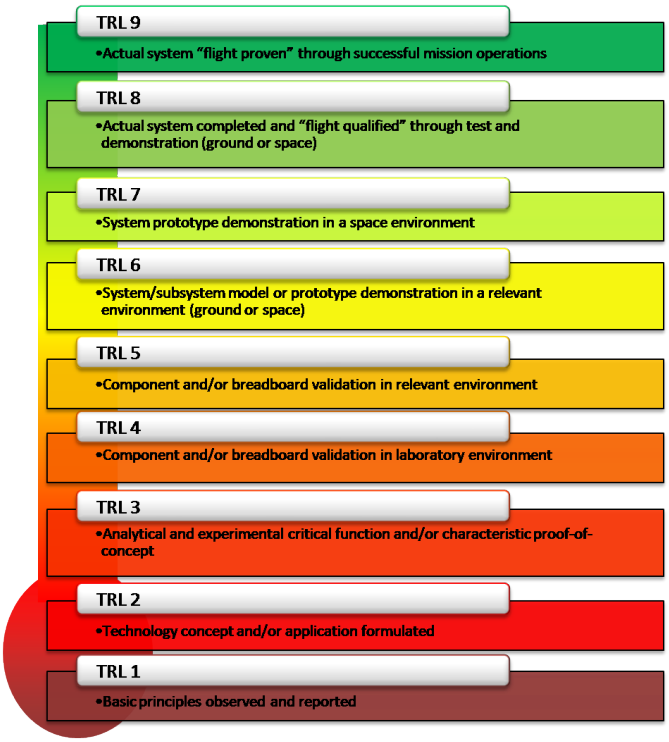

Figure 2.2: The nine levels in the TRL scale and the description of each level [3].

A TRL number is reached once the description for that number in Figure 2.2 has been achieved by the technology [3].

2.3 How dynamometers and a dyno work

A dynamometer is a device that absorbs and measures the power output of a prime mover (a large mechanical power-producing device), for example an internal combustion engine. Several methods of energy dissipation are used in various ranges of power, but the measurement techniques are governed by the same underlying principles. A cradled dynamometer measures mechanical power by measuring the rotational speed of a power-transmitting shaft, and the reaction torque required to prevent movement of the stationary part of the prime mover. A cradled dynamometer is supported in so called trunnion bearings, to allow the reaction torque to be transmitted to a torque or force-measuring device [4].

Equation (2.1) is used to calculate the power transmitted by the shaft connected to the dynamometer.

$$P = M \cdot \omega, \tag{2.1}$$

where M is the torque and ω is the rotational speed measured by the dynamometer [4].

One example on a cradled dynamometer is a waterbrake dynamometer, which uses fluid friction and momentum transport to create a means of energy dissipation. The load absorbed by waterbrake dynamometers can be adjusted using water level and flow rates in the brake, and they can be used for applications up to 7450 kW. Figure 2.3 shows two designs on waterbrake dynamometers, a viscous shear type and a momentum exchange (agitator) type [4].

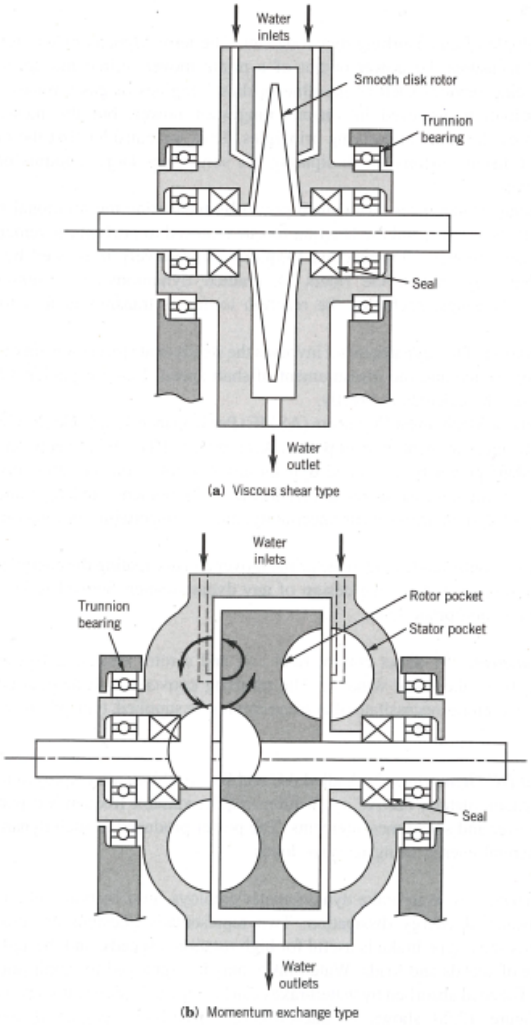

Figure 2.3: Two designs on waterbrake dynamometers, a viscous shear type (a) and a momentum exchange (agitator) type (b) [4].

Two other examples on cradled dynamometers are eddy current dynamometers, using the principle of induction to develop torque and dissipate power, and AC and DC generators, using cradled AC and DC machines as power-absorbing elements [4].

A dyno is an arrangement used to measure the performance of engines or vehicles' drivetrains with help of one or many dynamometers. The dynamometer(s) can be connected directly to an engine's crankshaft or to a vehicle's driveshaft(s), depending on what the purpose is with the measurements. If the purpose is to measure the performance of an engine, then the dynamometer is connected directly to the engine's crankshaft. If the purpose is the measure the performance of a whole drivetrain (including losses), then the dynamometer(s) is/are connected to one or many of the vehicle's driveshafts.

2.4 Explanation of terms

Different terms regarding simulations that are used in this master thesis are presented and explained below.

2.4.1 Batch simulations

Batch simulations is another name for ordinary simulations where the user chooses solver, sets up the solver settings and boundary conditions and so on, runs the simulation and waits for the computer (or cluster) to calculate the result(s). Solver settings and boundary conditions can not be changed while the simulations are running, such changes requires a new simulation to be ran. Two examples on batch simulations are crash test simulations of cars using the finite element method (FEM) and simulations of wind mill blades using computational fluid dynamics (CFD).

The smaller time step (higher sample rate) or the finer elements used in a batch simulation, the longer time is required to calculate the result(s) with the same computational power available. Below is an example on computing the time step to use in a batch simulation.

Equation (2.2) is used to calculate the largest allowed time step in a time-dependent one-dimensional analysis, using the finite element method and explicit time integration.

$$\Delta t \leq \frac{L_e}{\sqrt{\frac{E}{\rho}}}, \tag{2.2}$$

where L_e is the element length, E is Young's modulus and ρ is the density of the material. Equation (2.2) holds for the central difference method [5].

If the element length is 5 mm and the material is steel with a Young's modulus of 210 GPa and a density of 7800 kg/m^3, Equation (2.2) gives the time step 9,6 $\cdot 10^{-7}$ s, that corresponds a sample rate of approximately 1,04 MHz.

2.4.2 Real time simulations

Real time simulations are computer models of physical systems that are executed at the same rate as the time in the real world. In other words, the computer models run at the same rates as the actual physical systems. When running real time simulations a simulation frequency (sample rate) that is high enough to capture the behaviour of the system and to give enough accurate results must be used.

One difference between batch simulations and real time simulations is that in real time simulations the user is able to change the input parameters while the simulation is running, and directly gain a response. MiL, SiL and HiL simulations are three examples on real time simulations, and are described in the subsections below.

Simulators and video games are also examples on real time simulations, but with the main purposes to educate and train operators (simulators) and to entertain the players (video games) instead of deliver simulation results that can be used for product development. In simulators and video games the operator or player changes input parameters with controls or a hand controller and gets a response instantly on the screen.

The response is calculated by a physics engine where realistic motions are calculated using information about gravity and objects masses and moments of inertia [6].

The physics engine AGX Dynamics for example uses sample rates up to 60 Hz [7]. 60 Hz is more than 17000 times lower than 1,04 MHz (the sample rate computed in the example in subsection "2.4.1 Batch simulations" above), and is one of the explanations behind why real time simulations can be executed in a stable way.

2.4.3 MiL - Model in the Loop

A MiL (Model in the Loop) simulation is a simulation where a digital model represents the algorithm or function being developed, for example a control system. MiL simulations are used at an early stage in the development, to verify the algorithm or function and to ensure that it fulfils the requirements [8].

Figure 2.4 shows the architecture over a MiL simulation.

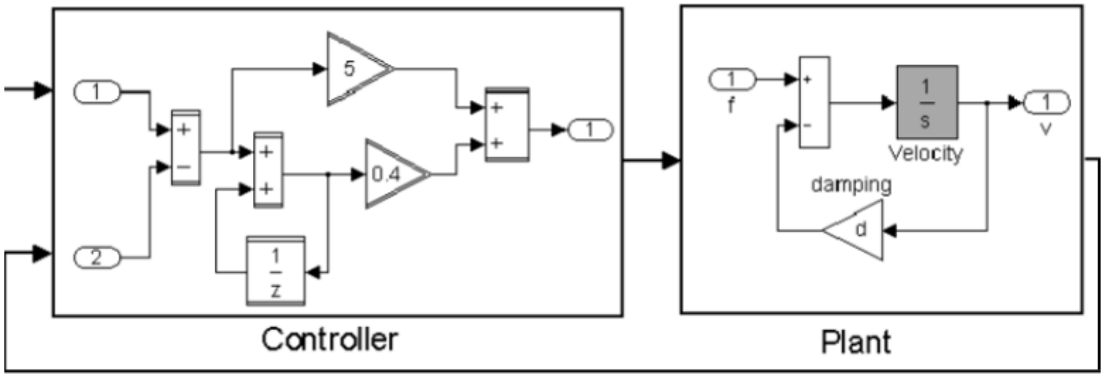

Figure 2.4: The architecture over a MiL simulation. Both the controller and the plant (the model of the system to control) are represented by digital models [8].

2.4.4 SiL - Software in the Loop

When code is automatically generated in a modelling environment to represent a model, it is not certain that the code and the model behave identical. This is because the modelling environment and the code might use different data types (floats and integers for example) which can lead to deviations. Because of this phenomena it is necessary to perform simulations using the generated code, so called SiL (Software in the Loop) simulations, when developing algorithms or functions. A SiL simulation of an algorithm or function is performed after the MiL simulation using the same plant model [9].

Figure 2.5 shows the architecture over a SiL simulation.

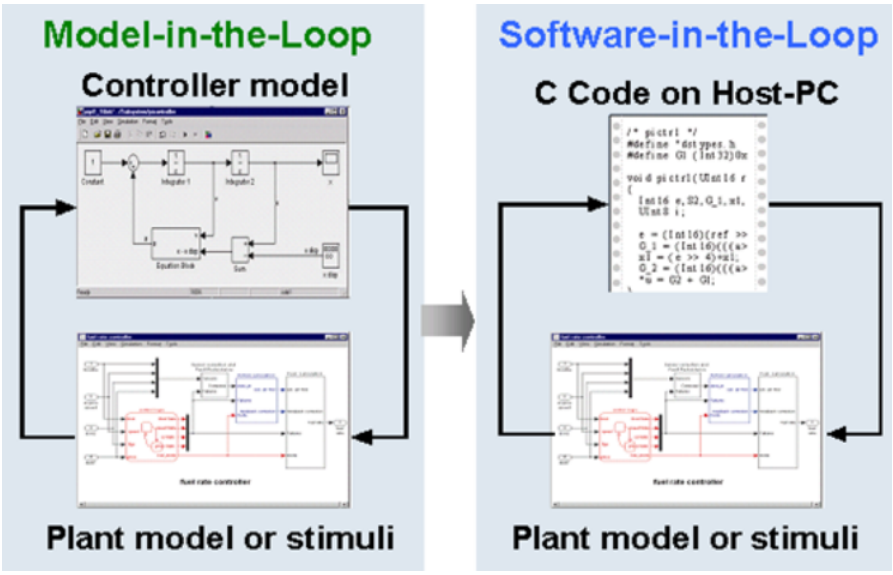

Figure 2.5: The architecture over a SiL simulation. Both the controller model and the plant model are digital models [9].

2.4.5 HiL - Hardware in the Loop

HiL (Hardware in the Loop) simulations is a method for validating control algorithms, running on physical controlling units, by using a digital model (a real time environment) to represent the physical system to control. The digital model also includes the sensors that the physical system is going to use. The physical controlling units are connected to the computer where the HiL simulation is running with matching I/O channels or ports [10].

With help of HiL simulations it is possible to test control algorithms without physical prototypes. HiL simulations are especially useful when testing the control algorithms on the real physical system is dangerous or too costly. HiL simulations are used for example in the automotive industry and in industrial automation [10].

To summarise, a HiL simulation uses a physical controlling unit with implemented controlling software and a digital representation of the system to control. Figure 2.6 shows the architecture over a HiL simulation using a real-time target machine from the manufacturer Speedgoat.

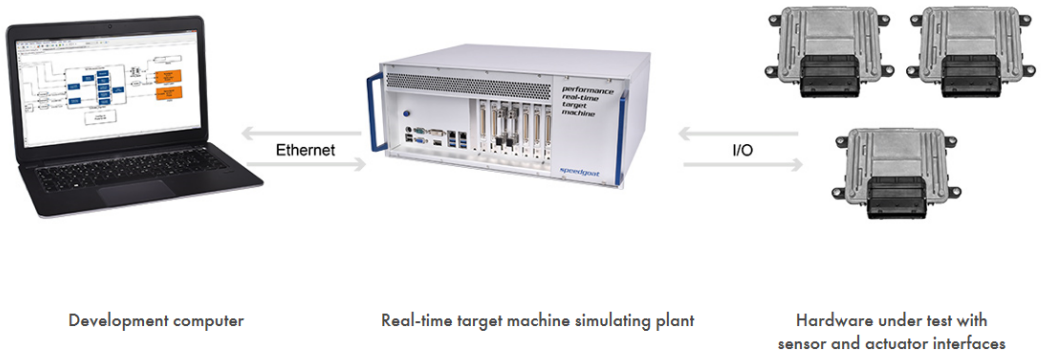

Figure 2.6: The architecture over a HiL simulation using a real-time target machine from the manufacturer Speedgoat [11].

2.4.6 RCP - Rapid Control Prototyping

RCP (Rapid Control Prototyping) is when a computer acts as a controlling unit to perform real time testing together with a system's hardware. RCP means that controller designs can be run, monitored and tested on the systems hardware, and also that testing and tuning of software and hardware designs can be done. Developers can make use of RCP when they want to start develop controller designs before a decision of physical controlling unit to use has been made [12]. Figure 2.7 shows the architecture over rapid control prototyping using a real-time target machine from the manufacturer Speedgoat.

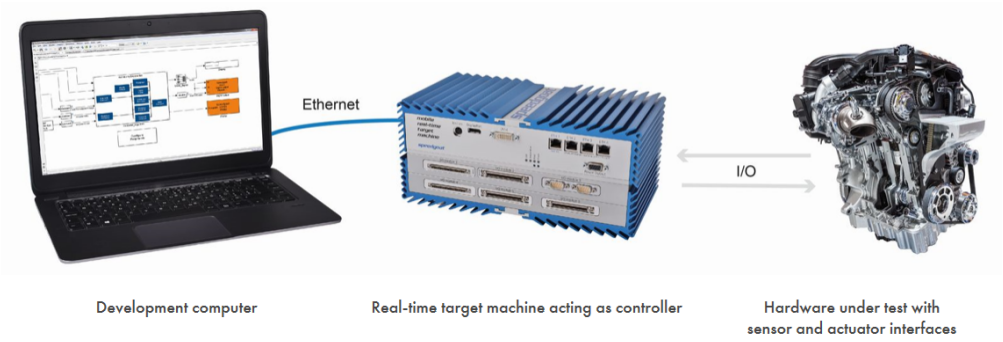

Figure 2.7: The architecture over rapid control prototyping using a real-time target machine from the manufacturer Speedgoat [12].

3 Method

This master thesis was performed using an adjusted version of the Ulrich and Eppinger product development process [2]. The product development process was adjusted since the master thesis was carried out by a single person and not a project group, and also since the master thesis is not a typical product development project. The adjustment led to that the sixth phase "Production ramp-up" in the Ulrich and Eppinger product development process [2] was removed, and that the other five phases were modified to fit the master thesis.

The first phase "Planning" consisted in making a time plan in the shape of a gantt-chart for the project, and setting up the objectives and limitations for the master thesis together with the supervisors.

The second phase "Concept development" consisted in identifying needs and creating a requirement specification for the VR system to be used at LTU, see section "3.2 VR system design" for detailed information. No concept generation was performed in this second phase which otherwise is a part in the Ulrich and Eppinger product development process [2]. Instead already existing solutions were compared and chosen between with help of the metric in the requirement specification.

The third phase "System-level design" consisted in establishing architectures for different user scenarios, see section "3.2 VR system design" for detailed information.

The forth phase "Detail design" and the fifth phase "Testing and refinement" consisted in positioning the VR system in the product development process and testing the VR system, see section "3.1 Positioning of the VR system in the product development process" and section "3.3 Test of the VR system" for detailed information.

3.1 Positioning of the mixed-VR system in the product development process

An investigation of in which of the six project phases (one or many), in the Ulrich and Eppinger product development process [2] the mixed-VR system could be useful was performed.

The mixed-VR system was positioned in the product development process by:

- Reading about which activities that are performed in the six project phases, and when prototypes are built in the Ulrich and Eppinger product development process [2].
- Reading technical/scientific articles about similar systems as the mixed-VR system, and how these systems are used to test and evaluate vehicles.
- Reading technical/scientific articles about the benefits of performing HiL simulations.
- Analysing which input/output information that is needed/generated in the mixed-VR system.

When the four steps above were performed the mixed-VR system could be positioned in the product development process by referring to examples.

An investigation of where in the TRL scale (which TRL levels that has been reached) the prototypes tested in the mixed-VR system should be positioned was also performed.

3.2 VR system design

The design of the VR system consisted of analysing which physical components that should be included and how these physical components should be connected to each other. The decision of which softwares that will be used in the VR system was taken before the start of the master thesis, and is thus nothing that this report will cover. However, the softwares used to test the VR system will be listed and described in section "3.3 Test of the VR system".

3.2.1 List different types of vehicles and their functions

To be able to develop the interface between the Mobilab and the virtual environment, the type of signals that flows between the vehicle placed in the Mobilab and the virtual environment must be investigated. This subsection describes the method used to perform the investigation.

To begin the most common types of vehicles that works in the forest, in the harbor and in the mining industry and their functions were listed. In this case, functions mean

things that has to do with the manual manoeuvring of the vehicles and performing their working operations (not things like using the headlights or adjust the temperature inside the cab). Some types of vehicles and their functions in the three different industries were known, but some types of vehicles and functions had to be analysed before they were listed. That was done by visiting websites of known vehicle manufacturers, and read brochures with specifications and features and look at pictures and videos of the vehicles. The functions of each type of vehicle where then put together in a net function list, to see how many and which type of functions the interface must be able to manage. Figure 3.1 shows the method used to create the net function list.

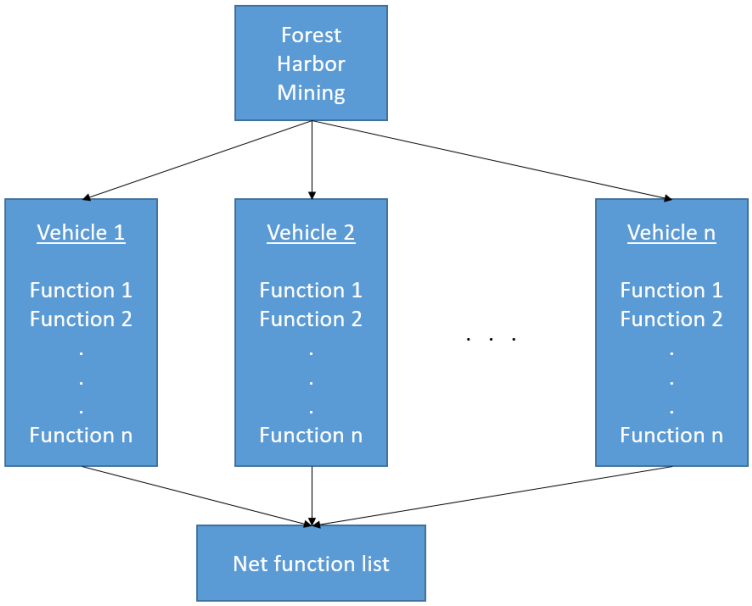

Figure 3.1: The method used to create the net function list.

The idea was then to translate the functions in the net function list to signals (type and magnitude) to and from the vehicles. This could however not be done because no information about signals were to be found on the manufacturers websites. A solution could have been to call or email manufacturers to gain such information, but that would have allocated too much resources for this master thesis. There is also a possibility that the manufacturers do not want to hand out such information due to confidentiality.

However, for the TFP developed by LTU which was used as a case in the project, the functions could be translated to signals (type and magnitude) to and from the TFP and be listed in a table. The functions and signals to and from the TFP were derived by talking to one of the supervisors, who is one of the people responsible for developing the TFP.

3.2.2 Identifying needs, creating a requirement specification and selecting a HiL computer

HiL simulations of the TFP will be performed at LTU to verify its control system, to conduct research and to test the VR system, and to perform these simulations a HiL computer will be needed. The procedure to select a HiL computer to be used at LTU started with identify needs and create a requirement specification for such a hardware.

The needs for a HiL computer were identified by talking to the supervisors and list their demands and desires on a HiL computer. The listed demands and desires were then translated to metrics according to the method described in [2], where the metrics is a part of a requirement specification for a HiL computer.

A benchmarking of four HiL computers from two different manufacturers (two HiL computers per manufacturer) were performed and resulted in two tables. One table with metric for two HiL computers from the first manufacturer, and one table with metric for two HiL computers from the second manufacturer. These two tables were then used to create a third table with allowed values and ideal values for a HiL computer from an arbitrary manufacturer. The allowed values and ideal values in the third table were derived with help of the values collected from the benchmarking. This third table is the requirement specification for a HiL computer to be used at LTU.

The allowed values and ideal values in the requirement specification were then used to choose a HiL computer to be used at LTU. All allowed values must be reached by the chosen HiL computer, and the more ideal values that is reached the better.

3.2.3 Establishing architectures for different user scenarios

Figures over architectures for different user scenarios were created by using the theory about dynamometers, dynos and HiL simulations described in section "2.3 How dynamometers and a dyno work" and "2.4 Explanation of terms", and by talking to the supervisors about the TFP and how it will work when in use. The figures were created using the program Microsoft PowerPoint 2016.

Figures over architectures for the following user scenarios were created:

- The architecture over a batch simulation of the TFP.
- The architecture over a hardware test of a vehicle's sub system.
- The architecture over when control code is compiled to the TFP.
- The architecture over a HiL simulation of the TFP.
- The architecture over the mixed-VR system.
- The architecture over when the TFP is in use.

Figures over architectures for the six user scenarios listed above were created to get an overview over the VR system and related areas. And also to gain a larger understanding of which physical components that should be included in the VR system and how these physical components should be connected to each other.

3.3 Test of the VR system

The equipment and methods used to test the VR system are presented and explained under the following subsections.

3.3.1 Hardware and software used to test the VR system

One of the hardwares used to test the VR system was a laptop.

The performance/specifications of the laptop used to test the VR system:

- Operational system: Windows 10
- Number of processor cores: 2
- Processor clock speed: 2,5 GHz
- Size of processor cache memory: 3 MB
- Size of RAM: 8 GB
- Size of disk memory: 256 GB SSD

When performing HiL simulations of the TFP, the UEISIM (the TFP's controlling unit) and a data acquisition device were used together with the laptop.

Figure 3.2 shows the UEISIM.

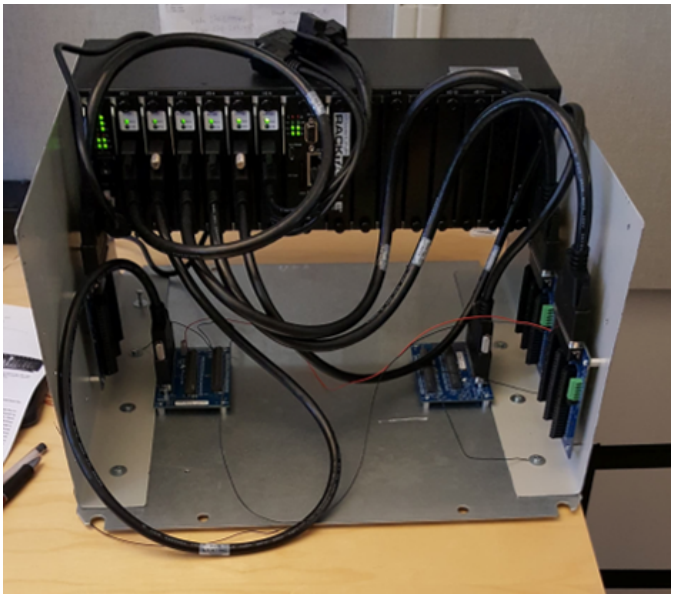

Figure 3.2: The UEISIM, the TFP's controlling unit.

Figure 3.3 shows the data acquisition device from National Instruments, model NI cDAQ-9174 with a NI 9205 inserted, used to measure output voltage from the UEISIM and acquire it to MATLAB/Simulink.

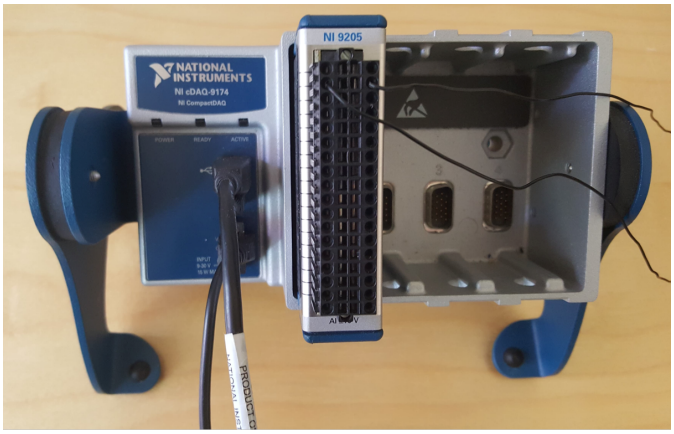

Figure 3.3: The data acquisition device from National Instruments.

Several softwares were used to test the VR system. Some of these softwares are developed and provided by the company Algoryx Simulation located in Umeå. The softwares from Algoryx Simulation were chosen because they are well suited for real time simulations, Algoryx Simulation could further provide evaluation licenses at a short notice. The softwares used to test the VR system and installation of them are presented below.

The following sofwares were installed on the laptop:

- Ansys Discovery SpaceClaim 2019 R3 (version 2019.3.0.07293)
- Algoryx Momentum (version 2.2.0)
- AGX Dynamics (version 2.28.1.0), including the module AGX Simulink
- CMake (version 3.16.4)
- Visual Studio Community 2019 (version 16.5.4), including the workload Desktop development with C++
- MATLAB R2019b, including Simulink (version 10.0), Simulink Desktop Real-Time (version 5.9), Data Acquisition Toolbox (version 4.0.1) and Data Acquisition Toolbox Support Package for National Instruments NI-DAQmx Devices (version 19.2.0)

Except the softwares listed above, Siemens NX 2020 (Version: 1899 and Build: 1700) and IDLE (Python 3.6 64-bit) was used on the computers in the project room and in the student computer labs at LTU, but not installed on the laptop.

To perform the tests of the VR system described under subsection "3.3.3 VR system test procedure" below, the softwares AGX Dynamics (including the module AGX Simulink), CMake, Visual Studio and MATLAB (including the four add-ons listed above) has to be installed on the same computer.

SpaceClaim is Ansys software for 3D modelling and can be used to create, edit and repair geometry. When installing SpaceClaim it is important to also install and choose the right settings for the CAD Configuration Manager, to make it possible to import CAD files generated in other CAD programs than SpaceClaim without using a neutral format.

Algoryx Momentum is an add-in to SpaceClaim and not a separate program. Once Algoryx Momentum has been installed it will appear as an extra tab within SpaceClaim. This means that SpaceClaim has to be installed on the computer before Algoryx Momentum can be used. Algoryx Momentum is used for motion dynamics for multi-body systems with joints, frictional contacts and analysis, driven by the physics engine (solver) AGX Dynamics. Both Algoryx Momentum and AGX Dynamics are developed and provided by Algoryx Simulation.

AGX Dynamics has to be installed in a folder on the computer where the user has full accessright to work as intended.

CMake and Visual Studio were used to configure AGX Dynamics according to the instructions in the user manual for AGX Dynamics.

MATLAB was used to configure the module AGX Simulink according to the instructions in the user manual for AGX Dynamics. The compiler used in MATLAB was Microsoft Visual C++ 2019.

Simulink was used to control and monitor the AGX simulation. When creating and running an AGX simulation in MATLAB/Simulink, MATLAB has to be started from the AGX Dynamics Command Line by referring to the folder where MATLAB is installed. If MATLAB is installed in the folder "C:\Program\MATLAB\R2019b" for example, then the command "C:\Program\MATLAB\R2019b\bin\matlab.exe" should be ran in the AGX Dynamics Command Line.

IDLE was used to make a Python script to couple the AGX simulation to Simulink.

3.3.2 Performing compatibility tests between Siemens NX and SpaceClaim

Ansys states that their software SpaceClaim is able to open files generated in the most other 3D modelling softwares on the market, without using a neutral format [13][14]. A couple of tests were therefore performed to test the compatibility between Siemens NX (one of the softwares that LTU uses and which the TFP is modelled in) and SpaceClaim.

To analyse the compatibility between Siemens NX and SpaceClaim the following four tests were performed:

- Generate a .prt-file in Siemens NX 2020 (Version: 1899 and Build: 1700), assign a material to it and try to import it in SpaceClaim.
- Generate a .prt-file in Siemens NX 2020 (Version: 1899 and Build: 1700), assign a material to it, export it to three STEP-files (STEP AP203, STEP AP214 and STEP AP242 format) and try to import them in SpaceClaim.
- Generate a .sim-file in Siemens NX 2020 (Version: 1899 and Build: 1700) and try to import it in SpaceClaim.
- Generate a .sim-file in Siemens NX 2020 (Version: 1899 and Build: 1700), export it to a .mdef-file and try to import it in SpaceClaim.

A .prt-file is a part file generated in Siemens NX.

A .sim-file is a simulation file generated in Siemens NX (in these tests a simulation file from Siemens NX Motion).

A .mdef-file is an exported simulation from Siemens NX.

STEP is a neutral format that enables part files to be imported in different CAD programs, independent of which CAD program the part file was generated in.

3.3.3 VR system test procedure

To test the VR system, the TFP was used as a case study. All necessary steps going from an assembly of the TFP in Siemens NX, to performing HiL simulations of the TFP with the UEISIM in Simulink was then carried out.

The assembly of the TFP used to test the VR system was a simplified version of the TFP without all small parts and details (to make the test of the VR system as simple as possible). The assembly of the TFP contained the chassis, the four pendulum arms, the four hydraulic motors, five hydraulic cylinders (one per pendulum arm and one for the articulated steering), the four wheels and a ground with an edge for the TFP to drive over. The assembly of the TFP did not contain the diesel engine, bolts and nuts, hoses, cables, electronic components and other hydraulic components than the hydraulic motors and cylinders. Figure 3.4 shows the assembly of the TFP used to test the VR system.

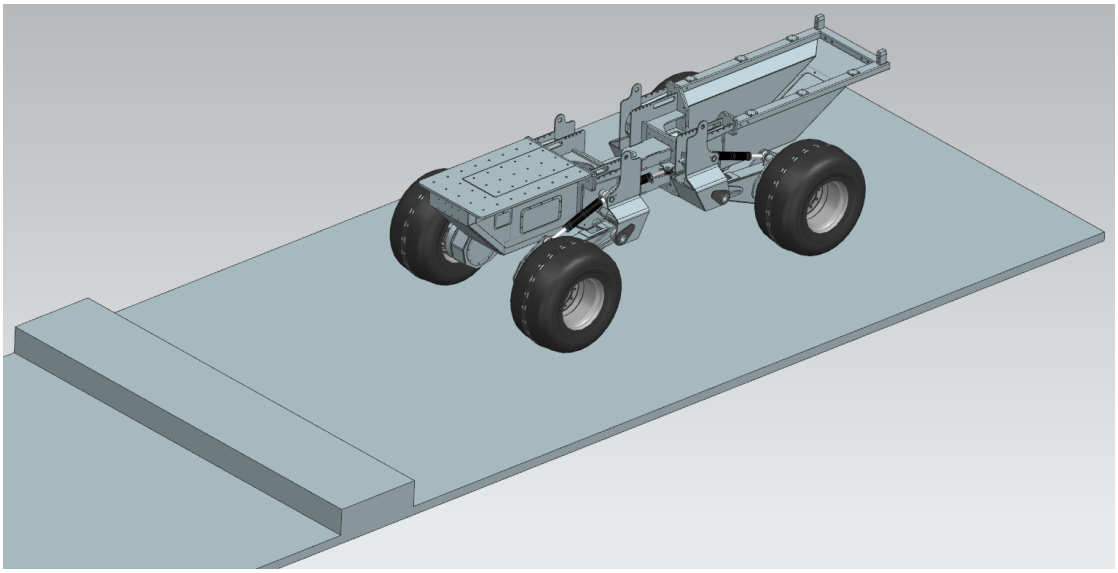

Figure 3.4: The assembly of the TFP used to test the VR system, taken in Siemens NX.

The following steps were performed to test the VR system, listed in chronological order:

- Open the assembly of the TFP in Siemens NX 2020.
- Export the assembly to the neutral format STEP AP214.
- Import the assembly (in the format STEP AP214) in SpaceClaim.
- Activate Algoryx Momentum.
- In Algoryx Momentum, split the assembly into parts and merge the parts that belongs to the same link.
- Set the gravitational constant to use in the simulation.
- Set the coefficient of restitution, the friction coefficient and the hardness of contact to use between materials in the simulation.
- Apply and adjust joints to the assembly that describe the correct behaviour of the TFP. No degrees of freedom has to be counted when applying the joints, because Algoryx Momentum is able to handle over constrained systems.
- Apply motors to the joints that should be driven.
- Run the simulation in Algoryx Momentum with different simulation frequencies to discover suitable values (values for which the TFP behaves realistic and the computational power is enough) to use later in the HiL simulations.
- Save the simulation in Algoryx Momentum (it will become a .scdoc-file as default), and export it to an .agx-file.
- Make two Simulink scripts in which the AGX simulation is controlled and monitored with an AGX-block (a co simulation) in real time. One Simulink script without the UEISIM to start with and then one Simulink script with the UEISIM included (for performing HiL simulations).
- Make a Python script in IDLE to refer to the .agx-file (define the file path) and to define the input signals to and the output signals from the AGX-block in the two Simulink scripts.
- Compile control code to the UEISIM, or adjust the output from the UEISIM in real time from a PC (not the laptop used to test the VR system) via the LAN that the UEISIM is connected to (ask Håkan Lideskog for detailed instructions regarding the UEISIM).
- Connect output cables from the UEISIM to the data acquisition device.
- Connect the data acquisition device to the laptop with an USB cable, and adjust the settings for it in MATLAB/Simulink.
- HiL simulations of the TFP with the UEISIM in Simulink can now be performed.

Following the steps above (with some modifications depending on vehicle), is a methodology making it possible to go from an assembly of a vehicle in an arbitrary CAD program to performing real time simulations (including HiL simulations) of the vehicle in Simulink.

4 Results

In this section the obtained results are presented and explained. The results are presented in the same order and under the same subsections as the method.

4.1 The mixed-VR system in the product development process

In the second project phase "Concept development" in the Ulrich and Eppinger product development process [2], the generated concepts must be evaluated using some methods. This could be done using working full scale prototypes but is most probably done using simulations or downscaled experimental prototypes with few details. To develop all or at least several concepts to the point that they can be tested and evaluated in the mixed-VR system would be too expensive and time consuming.

Some sort of prototype of the vehicle is necessary to be tested in the mixed-VR system. In the Ulrich and Eppinger product development process [2] prototypes are built in the fifth phase "Testing and refinement". For complex systems (larger-scale products) like for example vehicles, the fifth phase "Testing and refinement" includes extensive testing and validation. The overall performance, reliability, and durability are tested and design changes are implemented. To perform these tests the mixed-VR system could be useful.

In the sixth and last phase in the Ulrich and Eppinger product development process [2] the products are evaluated to identify any remaining issues, but for a vehicle that evaluation is probably done in the real usage environment.

Scania has got a climate laboratory at their department in Södertälje, where their trucks and buses can be tested in various climates before they reach the market [15]. The climate laboratory can simulate climate over a wide temperature span, with wind and with rain and snow. The climate laboratory is a complement (and not a substitute) to tests in the trucks' and buses' real user environments outdoors. Examples on functions that are tested in the climate laboratory are components ability to handle heat and cold, and the climate inside the cab in different weather conditions. Performing these types of tests in such a laboratory fits well in the fifth phase "Testing and refinement" in the Ulrich and Eppinger product development process [2].

When HiL simulations of a system are performed, a decision of which physical controlling unit to use has been made [10]. This points toward that the project has reached the fourth project phase "Detail design".

However, by using HiL simulations software testing can begin without a complete physical prototype of the system (a vehicle for example), allowing for starting the tests earlier in the same project phase as before in the development process.

This is something the agricultural equipment manufacturer AGCO Fendt uses in their HiL test benches where a tractor's ECU is connected to a HiL computer containing sensors and a digital twin (a digital model of the tractor with drivetrain, power take off, hydraulics and more) [16].

One of the limitations in this master thesis is that the vehicles that are tested in the mixed VR-system has got almost fully developed physical drivetrains including ECUs (controlling units) and almost fully developed controlling softwares. Another of the limitations is that sensors in the vehicles that are tested in the mixed-VR system are known. These two limitations tells that the development of the vehicles has reached the fourth phase "Detail design" or the fifth phase "Testing and refinement". Between the VR environment and the vehicle in the Mobilab control signals and sensor signals will flow which points towards the same result.

To summarise, the mixed-VR system could be useful in the fourth phase "Detail design" and in the fifth phase "Testing and refinement" in the Ulrich and Eppinger product development process [2]. Using the mixed-VR system already in the second phase "Concept development" to evaluate and choose concept(s) to develop further would be too expensive and time consuming.

Figure 4.1 shows where the mixed-VR system could be useful in the Ulrich and Eppinger product development process [2].

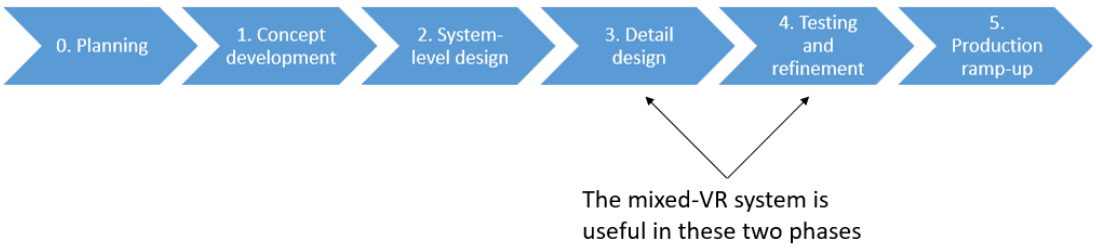

Figure 4.1: The two arrows show where the mixed-VR system could be useful in the Ulrich and Eppinger product development process [2].

Figure 4.2 shows the TRL level for a product under development as a function of time, and where the mixed-VR system comes in.

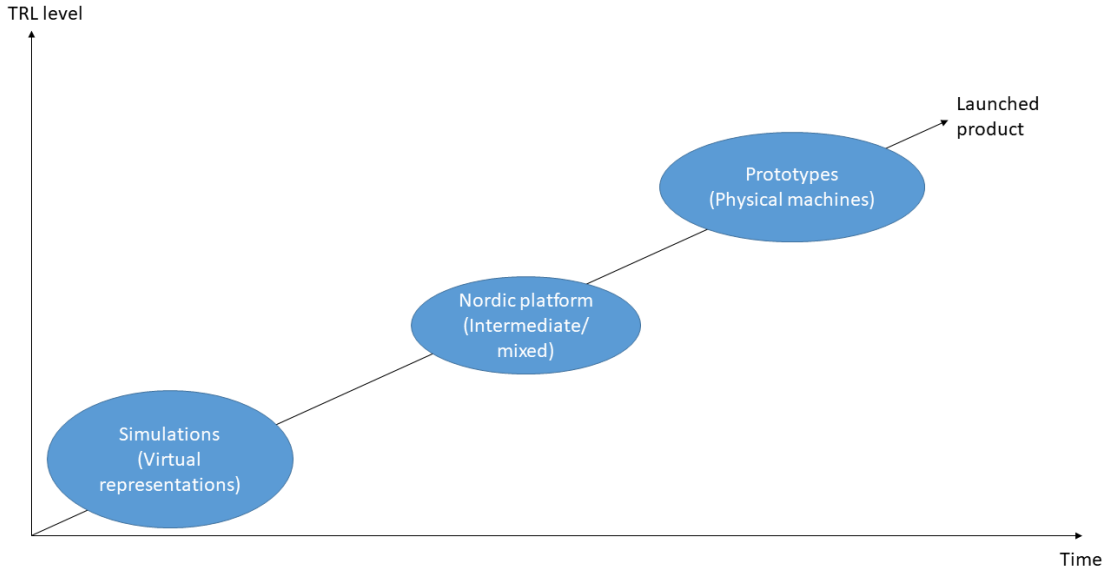

Figure 4.2: The TRL level for a product under development as a function of time. Notice that the TRL level might not increase linearly with the time in reality.

The prototypes tested in the mixed-VR system will be somewhere between level 4 and 6 in the TRL scale (has reached at least level 4), where the descriptions for level 4-6 are [3]:

- Level 4: Component and/or breadboard validation in laboratory environment.
- Level 5: Component and/or breadboard validation in relevant environment.
- Level 6: System/subsystem model or prototype demonstration in a relevant environment.

The relevant environment in the three descriptions above corresponds the mixed-VR system with both virtual environment and dynamometers. See Figure 2.2 for descriptions of all nine levels in the TRL scale.

4.2 VR system design

The results concerning the VR system design are presented and explained under the following subsections.

4.2.1 Different types of vehicles and their functions

The most common types of vehicles and their functions in the forest, in the harbor and in the mining industry are presented below in text and with figures.

The three most common types of vehicles in the forest are:

- Forwarder
- Harvester
- Scarifier (two types, disc trenchers and mounders)

Figure 4.3 shows a typical forwarder.

Figure 4.3: A forwarder from Komatsu, model 895 [17].

Figure 4.4 shows a typical harvester.

Figure 4.4: A harvester from Komatsu, model 951 [18].

Figure 4.5 shows a disc trencher, which is a type of scarifier, mounted on a forwarder.

Figure 4.5: A disc trencher from Bracke Forest, model T26.b, mounted on a forwarder [19].

Figure 4.6 shows a mounder, which is a type of scarifier, mounted on a forwarder.

Figure 4.6: A mounder from Bracke Forest, model M36.b, mounted on a forwarder [20].

See Appendix A for the functions of each type of vehicle in the forest.

The two most common types of vehicles in the harbor are:

- Forklift
- Reach stacker

Figure 4.7 shows a typical forklift.

Figure 4.7: A forklift from Toyota Material Handling, model Tonero counterbalanced forklift diesel 3,5 ton [21].

Figure 4.8 shows a typical reach stacker.

Figure 4.8: A reach stacker from Kalmar, model DRG100-120 [22].

See Appendix B for the functions of each type of vehicle in the harbor.

The four most common types of vehicles in the mining industry are:

- Excavator (two types, with tracks and with wheels)
- Hauler (two types, rigid haulers with four wheels and articulated haulers with six wheels)
- LHD loader (Load, Haul, Dump machine)
- Wheel loader

Figure 4.9 shows a typical excavator with tracks.

Figure 4.9: An excavator with tracks from Volvo CE, model EC250E [23].

Figure 4.10 shows a typical excavator with wheels.

Figure 4.10: An excavator with wheels from Volvo CE, model EW220E [24].

Figure 4.11 shows a typical rigid hauler with four wheels.

Figure 4.11: A rigid hauler with four wheels from Volvo CE, model R70D [25].

Figure 4.12 shows a typical articulated hauler with six wheels.

Figure 4.12: An articulated hauler with six wheels from Volvo CE, model A25G [26].

Figure 4.13 shows a typical LHD loader.

Figure 4.13: A LHD loader from Epiroc, model Scooptram ST3.5 [27].

Figure 4.14 shows a typical wheel loader.

Figure 4.14: A wheel loader from Volvo CE, model L60H [28].

See Appendix C for the functions of each type of vehicle in the mining industry. The net function list can be seen in Appendix D.

Table 4.1 and Table 4.2 shows the functions and the corresponding signals for the TFP, and Table 4.3 shows other functions and sensors and their corresponding signals for the TFP.

Table 4.1, Table 4.2 and Table 4.3 are configured with the prerequisites that autonomous control is active, and that the TFP has been loaded with a mission scheme (no external control is active and the TFP runs as "stand alone"). Moreover, the machine functions that are run physically on the TFP are motor drive (the four wheels are turning), entire drivetrain and sensors pertaining to those functions and the parking brake. Functions such as crane movements, movement of the pendulum arms and turning are deactivated to be virtually represented instead.

Target values are what the machine control has as wanted level in regards to the amount of flow that should go through the section. The actual values are what is measured from sensors when existing. Regarding the hydraulic sectional block: a value of 2,5 VDC means that no fluid is added or removed from the function. Any other value (between 0,5 VDC and 4,5 VDC) means that either the A side or the B side are opened to use the function and the other side (B or A) will open to relieve pressure to tank.

The idea for Mobilab to control the resistant forces from the ground to simulate that physically on the TFP is as follows: At timestep 1, a target value of required torque is measured in the VR system for all four wheels (depends on what the machine encounters). Same timestep, the torque in Mobilab is set to such target that the required torque will be met. Same timestep, the TFP measures actual wheel speeds and sends to the VR system which acknowledges it and calculates for example actual ground speeds and slip. Next time step, the VR system sends next target value to Mobilab. Speed control (target speed) is controlled only internally on the TFP. So if Mobilab increases torque the TFP needs to increase power to the wheels if maintaining speed is the current target.

Table 4.1: The first five (x2) of the ten (x2) functions and their corresponding signals for the TFP. The signals are numbered to be able to be detected in Figure 4.15 below. In this table VR stands for the VR system.

		A	**B**	**C**
	Functions	**Control signal from the UEISIM to machine function execution**	**Sensor signal to the UEISIM from VR (representing same sensors as existing on the machine)**	**Signals from the TFP to VR**
a	Drive forward and backwards (driven by wheels, four wheels)	1, analogue 0-5 VDC (”neutral” at 2,5 VDC)	None	11, same as aA
b	Turn right and left (articulated steering)	Deactive	3, analogue 0-5 VDC (”neutral” at 2,5 VDC). ”Actual value” of angle	12, 0-5 VDC (2,5 VDC = closed valves). ”Target value”
c	Raise and lower the pendulum arms (four pendulum arms’ positions)	Deactive	4, analogue 0-5 VDC. ”Actual value” of hydraulic piston position	13, 0-5 VDC (2,5 VDC = closed valves). ”Target value”
d	Use the parking brake	2, analogue 2,5 VDC or analogue 4 VDC	No feedback	14, same as dA
e	Rotate the crane clockwise and counter clockwise	Deactive	5, analogue current 0-24 mA (unknown angular span). ”Actual value”	15, 0-5 VDC (2,5 VDC = closed valves). ”Target value”

Table 4.2: The last five (x2) of the ten (x2) functions and their corresponding signals for the TFP. The signals are numbered to be able to be detected in Figure 4.15 below. In this table VR stands for the VR system.

		A	**B**	**C**
	Functions	**Control signal from the UEISIM to machine function execution**	**Sensor signal to the UEISIM from VR (representing same sensors as existing on the machine)**	**Signals from the TFP to VR**
f	Raise and lower the boom	Deactive	6, analogue 0-24 mA (4-20 mA = -135°-+135 ° linearly). "Actual value"	16, 0-5 VDC (2,5 VDC = closed valves). "Target value"
g	Raise and lower the arm	Deactive	7, analogue 0-24 mA (4-20 mA = -135°-+135° linearly). "Actual value"	17, 0-5 VDC (2,5 VDC = closed valves). "Target value"
h	Move the crane extension outwards and inwards	Deactive	8, analogue 0-5 VDC. "Actual value"	18, 0-5 VDC (2,5 VDC = closed valves). "Target value"
i	Rotate the grapple clockwise and counter clockwise	Deactive	9, analogue 0-24 mA (unknown angular span). "Actual value"	19, 0-5 VDC (2,5 VDC = closed valves). "Target value"
j	Open and close the grapple	Deactive	10, no physical sensor exists, but can be simulated in VR if needed: analogue 0-24 mA (4-20 mA = -135°-+135°). "Actual value"	20, 0-5 VDC (2,5 VDC = closed valves). "Target value"

Table 4.3: Other functions and sensors and their corresponding signals for the TFP. The signals are numbered to be able to be detected in Figure 4.15 below. In this table VR stands for the VR system.

Other functions or sensors	**From Mobilab to the TFP**	**From VR to Mobilab**	**From the UEISIM to VR**	**From VR to the UEISIM**
Target torque on the motors and the motors' rotational speeds, four motors		21, torques in Nm. "Target value", sent to Mobilab to set torque values on motors	22, rotational speeds in °/s. "Actual value"	
Mobilab's counter-torque, four different	23, applied torques in Nm. Physical interface			
IMU, machine angles relative to G				CANopen interface. Two axis ±90°
Other proprioceptive sensors not mentioned above, such as hydraulic pressures and temperatures			Send data if needed	
Exteroceptive sensors				3D point clouds, RGB and more

Figure 4.15 shows the signals that goes to and from the TFP when the TFP is plugged in in the Mobilab.

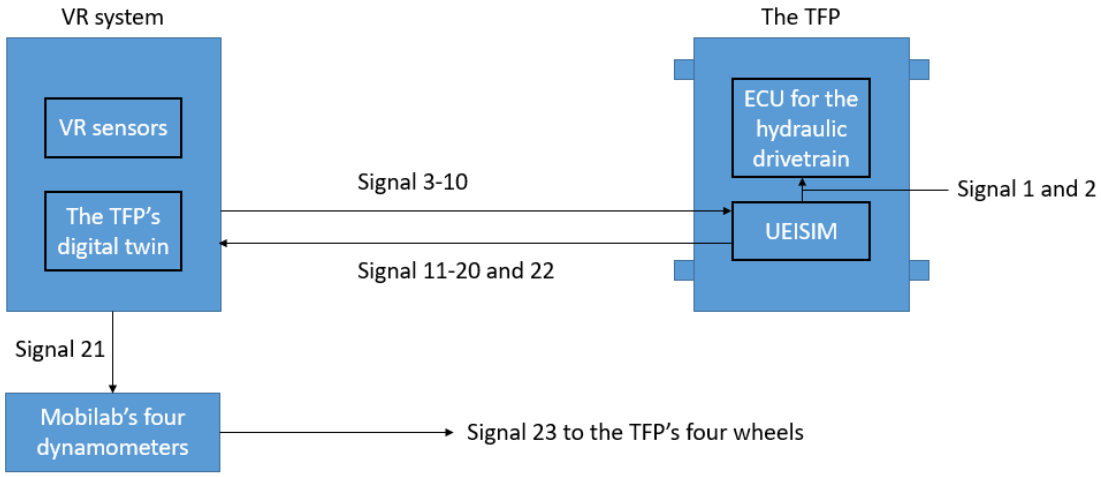

Figure 4.15: The signals that goes to and from the TFP when the TFP is plugged in in the Mobilab. The numbered signals are described in Table 4.1, Table 4.2 and Table 4.3 above.

4.2.2 Needs, requirement specification and choice of HiL computer

The need finding resulted in the following demands and desires on a HiL computer to be used at LTU:

Demands on a HiL computer to be used at LTU

1. Be able to run MATLAB and Simulink scripts in real time on the HiL computer.
2. Be able to run AGX Dynamics on the HiL computer.
3. Be able to run Mevea on the HiL computer.
4. Be able to run a program for environment simulations on the HiL computer.
5. Have enough computational power to perform HiL simulations of the TFP.
6. Be able to connect the HiL computer to the UEISIM (the TFP's controlling unit).
7. Be able to connect a PC to the HiL computer (with an Ethernet cable or wireless via Wi-Fi), or be able to connect a keyboard, a monitor and a mouse to the HiL computer.
8. Be able to use a normal wall outlet (230 VAC) as power supply.
9. The HiL computer must tolerate usage in an office environment (indoors in quite clean conditions).
10. Have a maximum price of about 50 000 SEK.

Desires on a HiL computer to be used at LTU

No desires, only the ten demands listed above.

See Appendix E for metric for two HiL computers from the manufacturer Speedgoat, and Appendix F for metric for two HiL computers from the manufacturer dSPACE.

Table 4.4 shows metric with allowed values and ideal values for a HiL computer from an arbitrary manufacturer. The allowed values and ideal values in Table 4.4 are derived with help of the values in Appendix E and Appendix F.

Table 4.4: Metric with allowed values and ideal values for a HiL computer from an arbitrary manufacturer. * = The computer's size and weight are well suited for use in an office environment, and the computer's enclosure are able to handle the temperature and the humidity in the office.

Metric number	**Need numbers**	**Metric**	**Units**	**Allowed values**	**Ideal values**
1	1	Compatibility with MATLAB and Simulink	[-]	Yes	Yes
2	2, 3, 4	Ability to install external programs	[-]	Yes	Yes
3	1, 2, 3, 4, 5	Number of processor cores	[-]	≥2	≥4
4	1, 2, 3, 4, 5	Processor clock speed	[GHz]	≥2,5	≥3,8
5	1, 2, 3, 4, 5	Size of processor cache memory	[MB]	≥8	≥20
6	1, 2, 3, 4, 5	Size of RAM	[GB]	≥4	≥16
7	1, 2, 3, 4, 5	Size of disk memory	[GB]	≥64 SSD	≥500 SSD
8	6, 7	Number of Ethernet ports	[-]	≥1	>2
9	7	Equipped with a wireless network card	[-]	No	Yes
10	7	Number of USB ports	[-]	≥0	>2
11	7	Number of HDMI, DVI or VGA ports	[-]	≥0	>1
12	8	Power supply with cable	[VAC]	230	230
13	9	Enclosure adapted for office environment*	[-]	Yes	Yes
14	10	Price	[SEK]	≤50 000	<50 000

By taking the allowed values and ideal values in Table 4.4 into account, a decision was made to use a PC as a HiL computer at LTU to begin with. If the PC turns out to fail in performing the HiL simulations of the TFP a new decision about HiL computer has to be done.

One reason behind the decision is that many of the regular PCs on the market meet all the allowed values and many of the ideal values. The PC used to test the VR system for example meets all the allowed values except one and eight of the ideal values, see section "3.3 Test of the VR system" for detailed information about that PC's performance.

Another reason behind the decision is that the Division of Product and Production Development already has got a PC available for performing HiL simulations, and therefore no new computer has to be bought and money can be saved. The third reason behind the decision is that HiL computers from the two manufacturers Speedgoat and dSPACE do not have ability to install external software (see Appendix E and Appendix F), which is one of the ten demands on the HiL computer to be used at LTU.

4.2.3 Architectures for different user scenarios

Figure 4.16 shows the architecture over a batch simulation of the TFP using one or many softwares. This procedure is done when developing the TFP.

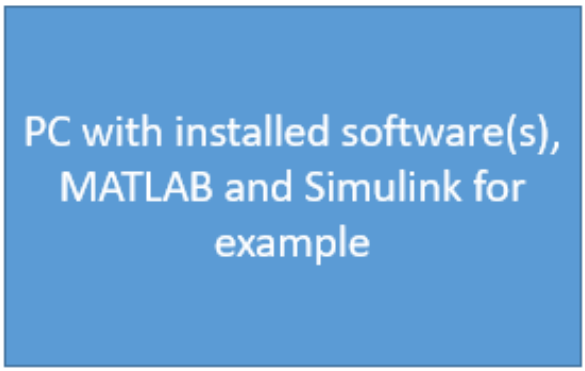

Figure 4.16: The architecture over a batch simulation of the TFP using one or many softwares.

Figure 4.17 shows the architecture over a hardware test of a vehicle's sub system. In this case a vehicle with four wheels which drivetrain is tested in a dyno. Note that the size of the blocks in the figure is not in proportion to the size of the components in reality. Processing of measurement data and monitoring is done in the PC.

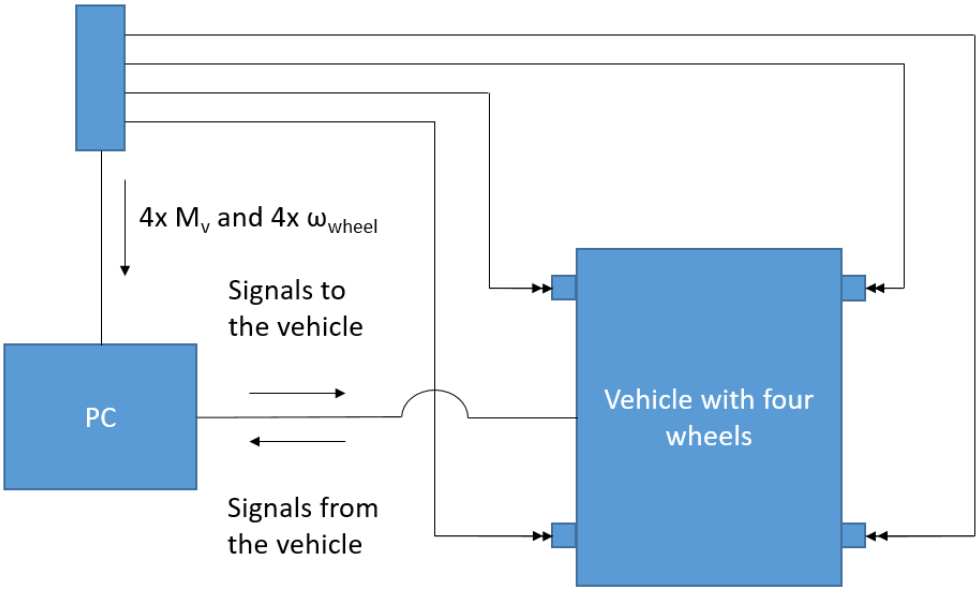

Figure 4.17: The architecture over a hardware test of a vehicle's sub system.

Figure 4.18 shows the architecture over when control code is compiled to the TFP. This procedure is done when developing the TFP. Note that the size of the blocks in the figure is not in proportion to the size of the components in reality. Programming of control code is done in the PC and the code is then compiled to the UEISIM. When this report was written a router was used between the PC and the UEISIM for setting up a LAN, making it possible for several users to connect to the UEISIM simultaneously. Other solutions than the router might be used in the future.

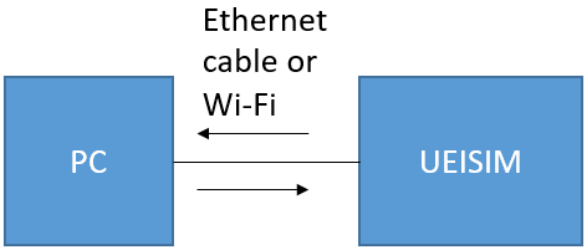

Figure 4.18: The architecture over when control code is compiled to the TFP.

Figure 4.19 shows the architecture over a HiL simulation of the TFP. Note that the size of the blocks in the figure is not in proportion to the size of the components in reality. Depending on choice of solution the PC and the HiL computer can be the same physical computer. HiL simulations of the TFP has been performed in this master thesis, see section "3.3 Test of the VR system" and section "4.3 Test of the VR system".

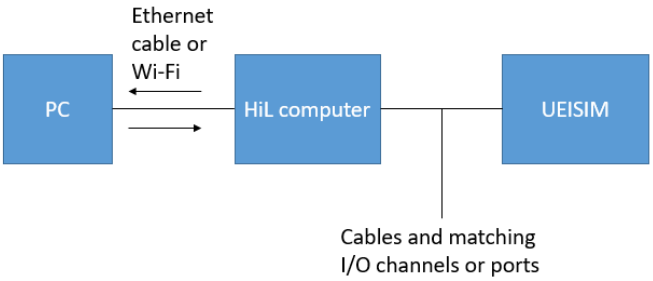

Figure 4.19: The architecture over a HiL simulation of the TFP.

Figure 4.20 shows the architecture over the mixed-VR system. Note that the size of the blocks and the ovals in the figure is not in proportion to the size of the components in reality. Processing of measurement data and monitoring will be done in the PC. Depending on choice of solution the PC and the HiL computer can be the same physical computer. Note that the mixed-VR system is under development and not finished when this report was written, but Figure 4.20 shows how the architecture will be.

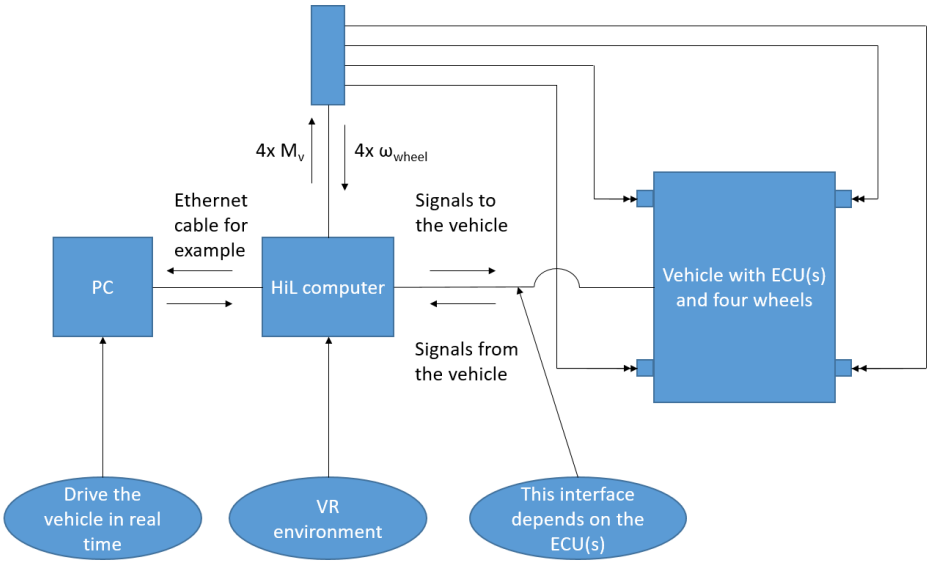

Figure 4.20: The architecture over the mixed-VR system.

Figure 4.21 shows the architecture over when the TFP is in use. Note that the size of the blocks in the figure is not in proportion to the size of the components in reality. All cables in the figure are Ethernet cables. The TFP will use several sensors, both external and internal. The precise number of sensors is not known yet, thereby the block "Sensor n". Examples on external sensors are Lidar and stereo camera, and examples on internal sensors are current sensors and voltage sensors. Note that the TFP is under development and not in use yet when this report was written.

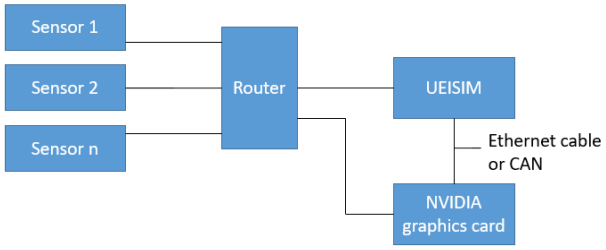

Figure 4.21: The architecture over when the TFP is in use.

4.3 Test of the VR system

The results concerning the test of the VR system are presented and explained under the following subsections.

4.3.1 Compatibility tests between Siemens NX and SpaceClaim

Table 4.5 shows the results of the four tests performed to analyse the compatibility between Siemens NX and SpaceClaim.

Table 4.5: The four tests performed to analyse the compatibility between Siemens NX and SpaceClaim and their results.

Test	**Result**
Generate a .prt-file in Siemens NX 2020 (Version: 1899 and Build: 1700), assign a material to it and try to import it in SpaceClaim	.prt-files generated in Siemens NX 2020 (Version: 1899 and Build: 1700) are possible to import (both single parts and assemblies), and material assigned to .prt-files can be detected by SpaceClaim
Generate a .prt-file in Siemens NX 2020 (Version: 1899 and Build: 1700), assign a material to it, export it to three STEP-files (AP203, AP214 and AP242) and try to import the three STEP-files in SpaceClaim	All three STEP-files could be imported, but the material assigned to the .prt-file could not be detected by SpaceClaim for any of the three STEP-files
Generate a .sim-file in Siemens NX 2020 (Version: 1899 and Build: 1700) and try to import it in SpaceClaim	.sim-files generated in Siemens NX 2020 (Version: 1899 and Build: 1700) are not possible to import
Generate a .sim-file in Siemens NX 2020 (Version: 1899 and Build: 1700), export it to a .mdef-file and try to import it in SpaceClaim	.mdef-files generated in Siemens NX 2020 (Version: 1899 and Build: 1700) are not possible to import

The results in Table 4.5 tells that .prt-files, both single parts and assemblies, generated in Siemens NX (Version: 1899 and Build: 1700 and older) do not have to be exported to STEP-files before imported in SpaceClaim.

When performing the two tests on the two last rows in Table 4.5, the error message "Unknown file type." appeared in SpaceClaim. This means that SpaceClaim is not able to open .sim-files and .mdef-files, independent of which version of Siemens NX they were generated in. This also means that there are no meaning of creating a simulation in Siemens NX Motion before working in Algoryx Momentum, because SpaceClaim is not able to open .sim-files and .mdef-files.

4.3.2 VR system test procedure

The assembly of the TFP used to test the VR system consisted of 152 parts, which were divided into 18 links in Algoryx Momentum. Table 4.6 shows the name of the link, a description of the link and number of links for each of the links used to simulate the TFP in Algoryx Momentum.

Table 4.6: Name of the link, description of the link and number of links for each of the links used to simulate the TFP in Algoryx Momentum.

Name of the link	**Description of the link**	**Number of links**
Chassis	Rear and front chassis plus hydraulic cylinder for the articulated steering	1
Pendulum_arm	Pendulum arm	4
Cylinder	Hydraulic cylinder for pendulum arm, "cylinder side"	4
Piston	Hydraulic cylinder for pendulum arm, "piston side"	4
Wheel	Wheel (tire and rim)	4
Flat_ground_with_edge	The ground with an edge for the TFP to drive over	1

Algoryx Momentum automatically assigns all parts the material steel with a density of 7800 kg/m^3. The material selection for the parts was not changed, because the purpose with the test is not to perform a rigid body simulation of the TFP with realistic properties.

The following general settings regarding materials and contacts were used in Algoryx Momentum:

- A gravitational constant of 9,82 m/s^2 (standard value in Algoryx Momentum).
- A coefficient of restitution of 0,2 (standard value in Algoryx Momentum) between steel and steel.
- A friction coefficient of 0,5 between steel and steel.
- The hardness of a contact of $2 \cdot 10^{11}$ Pa (standard value in Algoryx Momentum) between steel and steel.

Figure 4.22 shows the TFP divided into links in Algoryx Momentum.

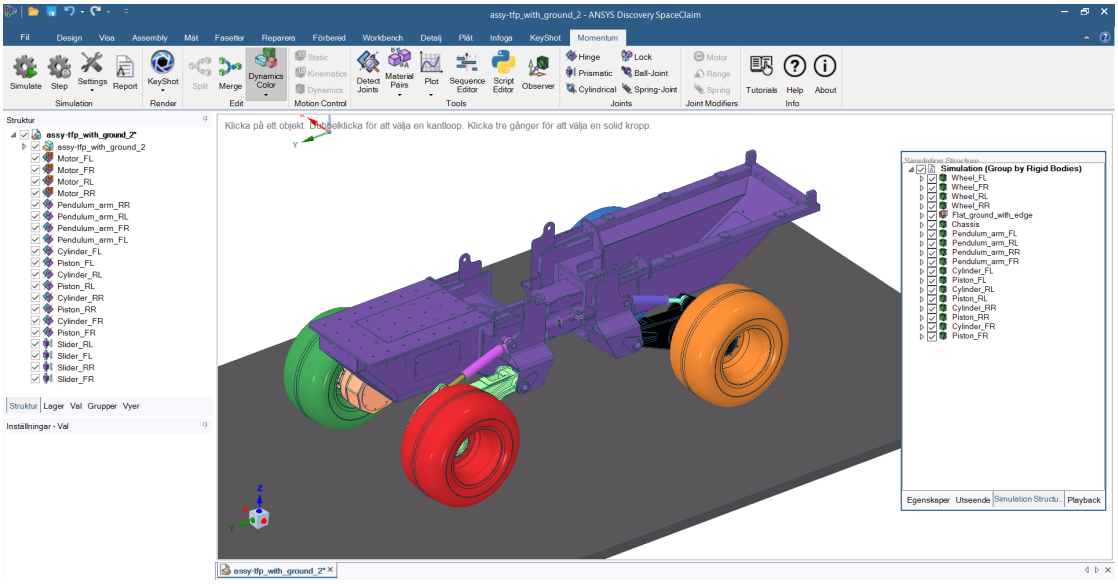

Figure 4.22: The TFP divided into links in Algoryx Momentum. The links are listed to the right and different links has different colours in the figure.

Table 4.7 shows the type of joint, placement of the joint and number of joints for each of the joints used to simulate the TFP in Algoryx Momentum.

Table 4.7: Type of joint, placement of the joint and number of joints for each of the joints used to simulate the TFP in Algoryx Momentum.

Type of joint	**Placement of the joint**	**Number of joints**
Hinge	Between the wheels and the hydraulic motors	4
Hinge	To connect the four pendulum arms with the chassis	4
Hinge	To connect the four hydraulic cylinders for the pendulum arms with the chassis and the pendulum arms	8
Prismatic	The pistons movement inside the hydraulic cylinders for the pendulum arms	4

For all joints in Table 4.7, except for the four prismatic joints on the last row, the pre settings were used. The four prismatic joints, describing the pistons movement inside the hydraulic cylinders for the pendulum arms, were given a range of ± 150 mm and spring and damper properties with spring coefficient, K=1000 kN/m, and damper coefficient,C=50 kNs/m. The four prismatic joints were given these spring and damper properties to prevent the TFP's four pendulum arms from getting in their lowest position and stay there when the TFP drove over the edge in the assembly.

Four motors, one motor per hinge joint between the wheels and the hydraulic motors, were applied. The speed of the four motors was set to $50°/s$, and the torque that the motors could deliver was set to infinitely large. The link "Flat_ground_with_edge" was set to static (which means that all six degrees of freedom are locked).

Figure 4.23 shows the TFP with joints applied in Algoryx Momentum.

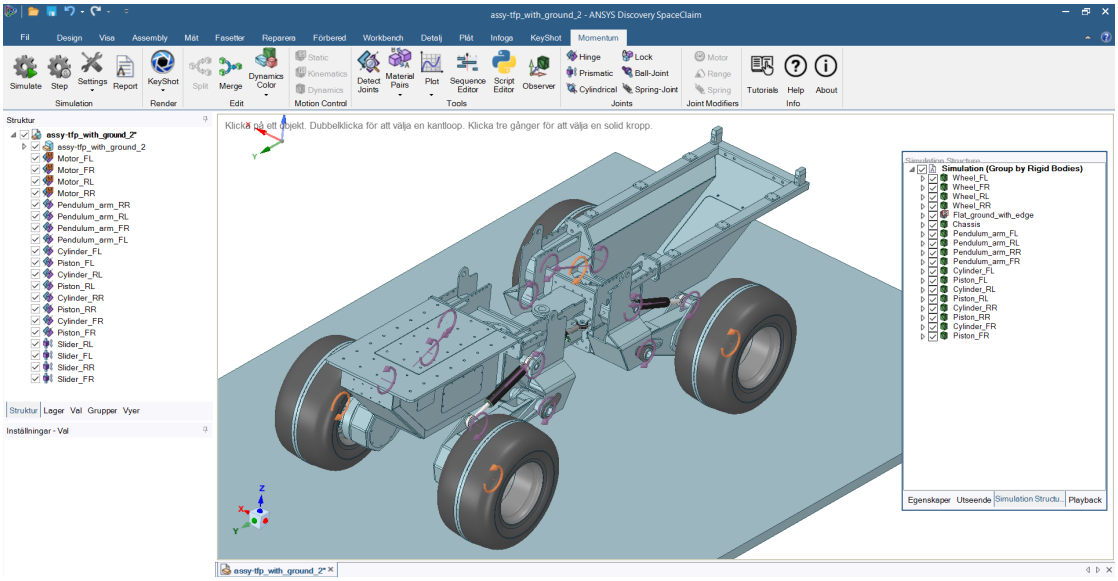

Figure 4.23: The TFP with joints applied in Algoryx Momentum. The joints are listed to the left and each joint is represented by an arrow in the figure. The four joints represented by orange arrows have motors applied.

Running the simulations in Algoryx Momentum showed that a simulation frequency higher than 30 Hz (the higher the better accuracy, as long as the computational power is enough) and a simulation length of 20 s was suitable to use. For simulation frequencies lower than 30 Hz the TFP behaved unrealistic and fell apart, probably due to numerical errors. Figure 4.24 shows what happened with the TFP when using a simulation frequency lower than 30 Hz (in this case 15 Hz) in Algoryx Momentum.

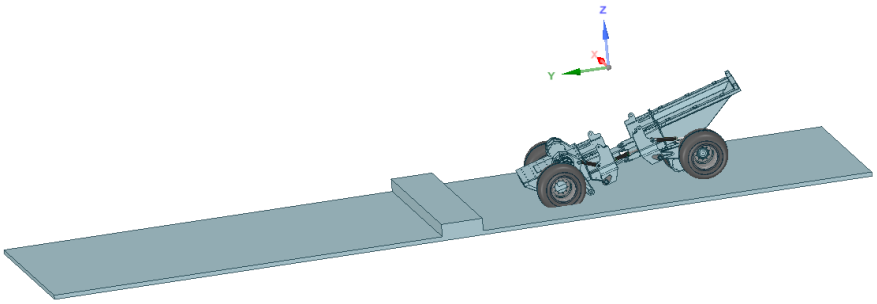

Figure 4.24: The TFP behaves unrealistic and falls apart when using a simulation frequency lower than 30 Hz (in this case 15 Hz) in Algoryx Momentum, probably due to numerical errors.

Figure 4.25 shows the Simulink script without the UEISIM. The model settings used in this Simulink script was a fixed step solver with automatic solver selection, and the used simulation mode was "Normal".

The input signals to the AGX-block are the rotational speeds of the TFP's four hydraulic motors, which are controlled by the knob in the upper left corner (the four motors get the same rotational speed). The output signals from the AGX-block are the forces acting on the joint "Slider_RR" in the x-, y- and z-direction which are plotted. The purpose with the output signals is to verify that AGX Dynamics is performing calculations while the simulation is running, and the choice of joint is arbitrary. The block "Real-Time Sync" ensures that the simulation is running in real time, and not as fast as possible which is the normal case for Simulink scripts. The number of allowed missed ticks for the simulation can be specified (if the number is reached the simulation will be cancelled) and the number of missed ticks are plotted.

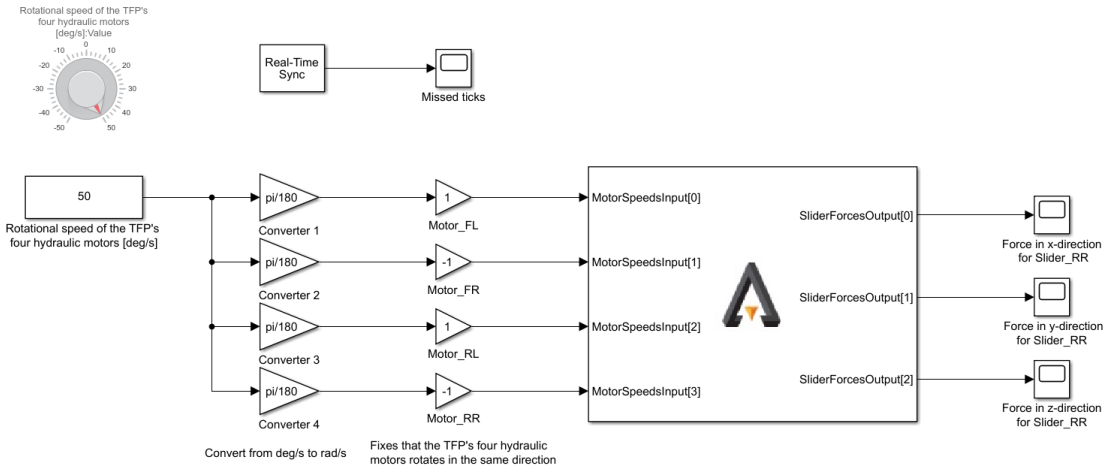

Figure 4.25: The Simulink script without the UEISIM.

When using a simulation frequency of 30 Hz in Simulink with graphics enabled, the 20 s long simulation took about 120 s to complete. The number of allowed missed ticks for that simulation was set to infinity. However, when using a simulation frequency of 30 Hz in Simulink without graphics enabled, the 20 s long simulation was able to run in real time with less than ten missed ticks.

The large difference in time required to complete the simulation between graphics enabled or disabled, is probably due to the fact that the laptop used to test the VR system does not have an external graphics card.

Figure 4.26 shows the Simulink script with the UEISIM included. The model settings used in this Simulink script was a fixed step solver with automatic solver selection, and the used simulation mode was "Normal".

The block furthest to the left represents the data acquisition device from National Instruments and measures the output voltage from the UEISIM, which controls the rotational speeds of the TFP's four hydraulic motors (the four motors get the same rotational speed) according to the criterion. The output signals from the AGX-block are the forces acting on the joint "Slider_RR" in the x-, y- and z-direction which are plotted. The purpose with the output signals is to verify that AGX Dynamics is performing calculations while the simulation is running, and the choice of joint is arbitrary. The block "Real-Time Sync" ensures that the simulation is running in real time, and not as fast as possible which is the normal case for Simulink scripts. The number of allowed missed ticks for the simulation can be specified (if the number is reached the simulation will be cancelled) and the number of missed ticks are plotted.

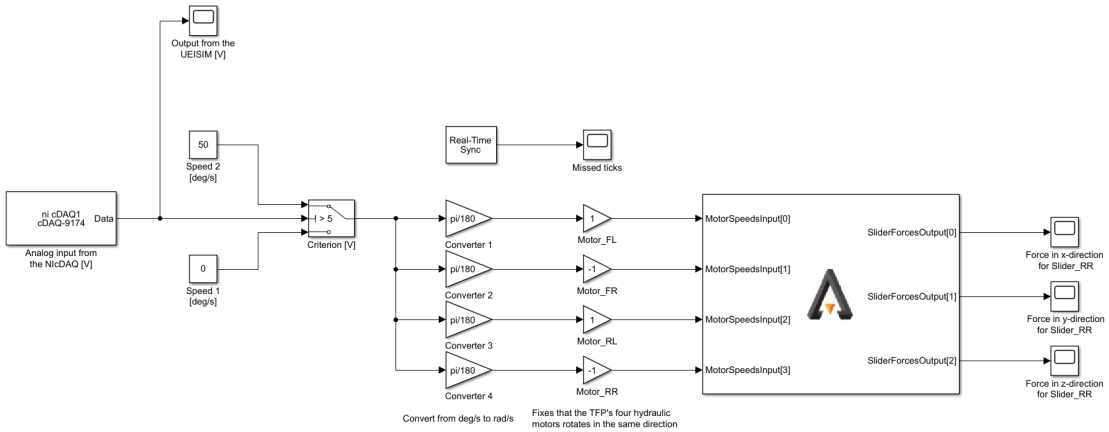

Figure 4.26: The Simulink script with the UEISIM included.

Running the co simulation in real time in Simulink with the UEISIM included also showed a large difference in time required to complete the simulation between graphics enabled or disabled.

Figure 4.27 shows the content in the AGX-block in the two Simulink scripts in Figure 4.25 and Figure 4.26 above. The path to the Python script is specified and no initialisation variables are used. The time step used in this simulation is 1/30 s and graphics showing the simulation is enabled.

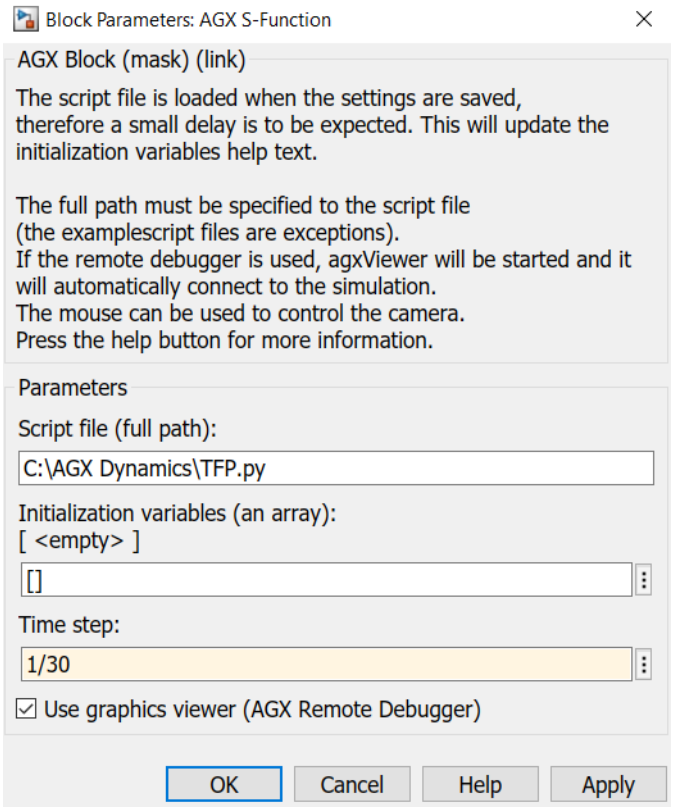

Figure 4.27: The content in the AGX-block in the two Simulink scripts in Figure 4.25 and Figure 4.26 above.

See Appendix G for the Python script used to refer to the .agx-file and to define the input signals to and the output signals from the AGX-block in the two Simulink scripts in Figure 4.25 and Figure 4.26 above.

5 Discussion and Conclusions

In this section the presented results are discussed. Conclusions that can be drawn are stated and future work to do are also mentioned.

The compatibility between two softwares might be limited to certain versions of the softwares. Thus, if one software is updated, the other software might also need to be updated to retain the compatibility. This fact is important to bear in mind, especially if the updates are done by a company's IT department and not by the user itself.

A software might need one or several other softwares to be used. To check if other softwares are needed to use the "main" software is important, because even if the "main" software is for free the other softwares might require licenses. Licenses that the user might or might not have access to.

It is also important to keep in mind which softwares that really needs to be installed on the same computer. If for example two softwares, A and B, are needed they can still be installed on two different computers. Software A is used to generate files on one computer, and then the files are saved or exported and used on another computer, where software B is installed. In this example only one license of each software (A and B) is needed.

Performing real time simulations might require large computational power, and it is important to have enough computational power to be able to run the simulation with a simulation frequency that is high enough to give accurate results.

It might be difficult to know before running the simulations how large computational power that will be needed. Therefore it is recommended to perform a background research and read about similar simulations (with a similar setup of softwares and hardwares) and how large computational power that was needed in these. Another solution is to contact software suppliers and ask what hardware solutions their customers use.

When buying a HiL hardware a modular solution is therefore to prefer. A modular solution allows the available computational power to be increased over time if needed, without having to buy an entire new hardware solution.

5.1 Conclusions

The three objectives of this master thesis has been reached. How the mixed-VR system should be utilized in the product development process has been described, the interface between the VR environment and the physical components (vehicle and dynamometers) in the Mobilab has been designed and the VR system has been tested on the TFP. By reaching the three objectives some conclusions can be drawn.

The mixed-VR system could be useful in the fourth phase "Detail design" and in the fifth phase "Testing and refinement" in the Ulrich and Eppinger product development process [2]. Using the mixed-VR system already in the second phase "Concept development" to evaluate and choose concept(s) to develop further would probably be too expensive and time consuming.

A regular PC can be used as a HiL computer, as long as the I/O channels or ports matches the physical controlling unit that will be used in the HiL simulations. An advantage with using a regular PC as a HiL computer is that a regular PC has the ability to install external software, which might be a demand on the HiL computer used in some HiL simulations. Having the ability to install external software is not possible for many specified HiL computers.

Choosing a suitable simulation frequency is important when performing real time simulations. The higher simulation frequency the better, as long as the computational power is enough. Choosing a too low simulation frequency will lower the accuracy of the results and might lead to unrealistic behaviour of the simulated system, probably due to numerical errors.

When using a simulations frequency of 30 Hz the simulation of the TFP was possible to run in real time in Algoryx Momentum (using graphics). When running the simulation of the TFP as a co simulation in real time between Simulink and the solver AGX Dynamics, a simulation frequency of 30 Hz was not even close to be able to run in real time with graphics enabled. The large difference in capacity between these two examples indicates that the coupling between Simulink and the solver AGX Dynamics slows down the simulation significantly.

The performed real time simulations in Simulink showed a large difference in time required to complete the simulations between having graphics enabled or disabled. Performing real time simulations with graphics enabled in Simulink will (probably) require the used HiL computer to have an external graphics card.

The purpose with the master thesis has also been fulfilled. The purpose with the master thesis was to assist LTU in their role in the research project, that is to prepare the mixed environment (virtual environment and dynamometers) for future research about automation of heavy vehicles. The results in the master thesis has given LTU staff more knowledge about the problems that can arise when performing real time simulations, and ways to solve them.

5.2 Future work

Before the mixed-VR system can be opened for manufacturers of heavy vehicles, the interface between the Mobilab and the virtual environment must be adapted to fit other vehicles than just the TFP. One option is that the manufacturers hand out information about which signals (type and magnitude) that goes to and from their vehicles a time before the vehicles are tested in the mixed-VR system. The interface can then be adapted to fit the vehicles to be tested.

The interface between the Mobilab and the virtual environment contains a HiL computer, that has to be bought. The choosen solution of HiL computer must not only have enough computational power to perform the real time simulations in the mixed-VR system, but must also fit to the physical components (vehicle and dynamometers) in the Mobilab.

Before the TFP is tested in the mixed-VR system the software Mevea must be implemented, which is the software that will be used in the mixed-VR system. An assembly of the TFP needs to be inserted/loaded into the program and there set up its dynamic and material properties. In the tests of the VR system in this master thesis a simplified version of the TFP without all small parts and details was used. When the TFP is tested in the mixed-VR system a more complete assembly needs to be used, including for example the crane and allowing the TFP to articulate. Sensors, tire and ground models must also be implemented in Mevea, and a suitable simulation frequency to use must be decided in some way.

At last the development of the TFP (including hardware and control code) must be finished so it can be tested in the mixed-VR system.

6 Acknowledgement

This master thesis is a part of the research project "Nordic Platform for Development of Autonomous Utility Vehicles" (the NUVE project), a collaboration between Luleå University of Technology, University of Oulu, SINTEF Narvik and Oulu University of Applied Sciences. The NUVE project is funded by Interreg Nord, a European Regional Development Fund.

References

[1] Rånman H. Utveckling av kontaktorgan till terrängdrönare [degree project]. Luleå: Luleå University of Technology; 2015.

[2] Ulrich KT, Eppinger SD. Product Design and Development. Fifth Edition, International Edition. New York: McGraw-Hill Education; 2012.

[3] Thuy M. Technology Readiness Level [Internet]. NASA; 2012 [updated 2017-08-07; cited 2020-03-12]. Available from: https://www.nasa.gov/directorates/heo/scan/engineering/technology/txt_accordion1.html.

[4] Figliola RS, Beasley DE. Theory and Design for Mechanical Measurements. Sixth Edition. Hoboken: Wiley; 2015.

[5] Logan DL. A First Course in the Finite Element Method. Fifth Edition, SI. Stamford: Cengage Learning; 2012.

[6] Hillar GC. 3D Game Development with Microsoft Silverlight 3 [Internet]. Birmingham: Packt Publishing; 2009. [cited 2020-03-16]. Available from: http://eds.a.ebscohost.com/eds/ebookviewer/ebook/ZTAwMHh3d19fMzMzNDU3X19BTg2?sid=0f45e1b5-a7bf-4d0a-9678-4a5a64601ce4@sdc-v-sessmgr01&vid=4&format=EB&rid=1.

[7] Algoryx. Heavy equipment [Internet]; [cited 2020-03-16]. Available from: https://www.algoryx.se/by-industry/heavy-equipment/.

[8] Fengco. Model-In-the-Loop Simulation [Internet]; [cited 2020-04-30]. Available from: https://www.fengco.se/en/model-based-development/verification/#2.

[9] Fengco. Software-In-the-Loop Simulation [Internet]; [cited 2020-04-30]. Available from: https://www.fengco.se/en/model-based-development/verification/#3.

[10] MathWorks. Hardware-in-the-Loop (HIL) Simulation [Internet]; [cited 2020-05-01]. Available from: https://se.mathworks.com/discovery/hardware-in-the-loop-hil.html?s_tid=srchtitle.

[11] Speedgoat. Hardware-in-the-Loop Testing (HIL) for Real-Time Plant Simulation [Internet]; [cited 2020-05-01]. Available from: https://www.speedgoat.com/applications-industries/applications/plant-simulation-hil.

[12] Speedgoat. Rapid Control Prototyping (RCP) [Internet]; [cited 2020-03-12]. Available from: https://www.speedgoat.com/applications-industries/applications/controller-prototyping.

[13] Ansys. Design and Concept Modeling [Internet]; [cited 2020-04-29]. Available from: https://www.ansys.com/products/3d-design/ansys-spaceclaim/design-and-concept-modeling.

[14] Ansys. Model Prep for Manufacturing [Internet]; [cited 2020-04-29]. Available from: https://www.ansys.com/products/3d-design/ansys-spaceclaim/model-prep-for-manufacturing.

[15] Ny Teknik. Så får Scania till ovädret [Internet]; [cited 2020-05-12]. Available from: https://www.nyteknik.se/fordon/sa-far-scania-till-ovadret-6402053.

[16] Speedgoat. AGCO Fendt: Automated testing of tractor controllers using Hardware-in-the-Loop test benches [Internet]; [cited 2020-05-12]. Available from: https://www.speedgoat.com/user-stories/speedgoat-user-stories/agco-fendt.

[17] Komatsu Forest. Komatsu 895 [Internet]; [cited 2020-02-14]. Available from: https://www.komatsuforest.co.uk/forest-machines/our-forwarders/895-2020.

[18] Komatsu Forest. Komatsu 951 [Internet]; [cited 2020-02-14]. Available from: https://www.komatsuforest.co.uk/forest-machines/our-harvesters/951-2020.

[19] Bracke Forest. Bracke T26.b [Internet]; [cited 2020-02-14]. Available from: https://www.brackeforest.com/products/disc-trenchers/160-bracke-t26-b-disc-trencher.

[20] Bracke Forest. Bracke M36.b [Internet]; [cited 2020-02-14]. Available from: https://www.brackeforest.com/products/mounders/163-bracke-m36-b-three-row-mounder.

[21] Toyota Material Handling. Toyota Tonero Diesel Forklift 3.5t [Internet]; [cited 2020-02-14]. Available from: https://toyota-forklifts.co.uk/our-products/ic-counterbalanced-trucks/high-tonnage/toyota-tonero-diesel-forklift-35t/.

[22] Kalmar. Reachstacker for empty and semi-laden container handling DRG100-120 [Internet]; [cited 2020-02-14]. Available from: https://www.kalmarglobal.com/equipment/reachstackers/Reachstacker-for-empty-and-semi-laden-container-handling-DRG100-120/.

[23] Volvo Construction Equipment. Large crawler excavators, Volvo EC250E; [cited 2020-02-14]. Available from: https://www.volvoce.com/europe/en/products/excavators/ec250e/.

[24] Volvo Construction Equipment. Medium wheeled excavators, Volvo EW220E; [cited 2020-02-14]. Available from: https://www.volvoce.com/europe/en/products/excavators/ew220e/.

[25] Volvo Construction Equipment. Rigid dump trucks, Volvo R70D; [cited 2020-02-14]. Available from: https://www.volvoce.com/europe/en/products/rigid-haulers/r70d/.

[26] Volvo Construction Equipment. Articulated hauler, Volvo A25G; [cited 2020-02-14]. Available from: https://www.volvoce.com/europe/en/products/articulated-haulers/a25g/.

[27] Epiroc. Scooptram ST3.5; [cited 2020-02-14]. Available from: https://www.epiroc.com/en-uk/products/loaders-and-trucks/diesel-loaders/scooptram-st3-5.

[28] Volvo Construction Equipment. Large wheel loaders, Volvo L60H; [cited 2020-02-14]. Available from: https://www.volvoce.com/europe/en/products/wheel-loaders/l60h/.

[29] Bracke Forest. Disc trenchers; [cited 2020-02-14]. Available from: https://www.brackeforest.com/products/disc-trenchers.

[30] Bracke Forest. Mounders; [cited 2020-02-14]. Available from: https://www.brackeforest.com/products/mounders.

[31] Volvo Construction Equipment. Medium wheeled excavators, Volvo EW240E MH; [cited 2020-02-14]. Available from: https://www.volvoce.com/europe/en/products/excavators/ew240e-mh/.

[32] Epiroc. Loaders and trucks (LHD's); [cited 2020-02-14]. Available from: https://www.epiroc.com/en-uk/products/loaders-and-trucks.

[33] Speedgoat. Performance real-time target machine; [cited 2020-02-26]. Available from: https://www.speedgoat.com/products-services/real-time-target-machines/performance.

[34] Speedgoat. Mobile real-time target machine; [cited 2020-02-26]. Available from: https://www.speedgoat.com/products-services/real-time-target-machines/mobile.

[35] dSPACE. SCALEXIO Processing Unit, Product lines for high core performance and high parallel performance; [cited 2020-02-26]. Available from: https://www.dspace.com/en/pub/home/products/hw/simulator_hardware/scalexio/scalexio_processing_unit.cfm.

[36] dSPACE. DS6001 Processor Board, High-performance processor board; [cited 2020-02-26]. Available from: https://www.dspace.com/en/pub/home/products/hw/simulator_hardware/scalexio/ds6001.cfm.

A Different types of vehicles and their functions in the forest

Below the three different types of vehicles that are used in the forest and their functions are listed.

Forwarder, totally 14 (x2) different functions

- Drive forward and backwards (driven by wheels, four, six or eight wheels)
- Turn right and left (articulated steering)
- Raise and lower one of the shafts in the bogie at the time (if the forwarder has got a bogie)
- Use the driving brakes
- Use the parking brake
- Rotate the crane clockwise and counter clockwise
- Raise and lower the boom
- Raise and lower the arm
- Move the crane extension outwards and inwards
- Rotate the grapple clockwise and counter clockwise
- Open and close the grapple
- Move the gate forward and backwards
- Raise and lower the stakes
- Raise and lower the front blade

Harvester, totally 14 (x2) different functions

- Drive forward and backwards (driven by wheels, four, six or eight wheels)
- Turn right and left (articulated steering)
- Raise and lower the pendulum arms (one per wheel, four, six or eight wheels)
- Use the driving brakes
- Use the parking brake
- Rotate the crane (and also the cab) clockwise and counter clockwise
- Raise and lower the boom
- Raise and lower the arm
- Move the crane extension outwards and inwards

- Rotate the harvester head clockwise and counter clockwise
- Tilt the harvester head (depending on operation)
- Hold the tree during the operations
- Drive the chain saw during the operations
- Rotate the rollers on the harvester head (feed the tree through the harvester head)

Scarifier, totally nine (x2) or ten (x2) different functions

- Drive forward and backwards (driven by wheels, four, six or eight wheels)
- Turn right and left (articulated steering)
- Raise and lower one of the shafts in the bogie at the time (if the scarifier has got a bogie)
- Use the driving brakes
- Use the parking brake
- Raise and lower a disc trencher's arm (maximum three arms)
- Move a disc trencher's arm to the right and to the left (maximum three arms)
- Rotate the discs clockwise and counter clockwise (maximum three discs, one per arm)
- Rotate the discs' engine housings clockwise and counter clockwise around their vertical shafts (maximum three discs, one per arm)
- Raise and lower a mounder's arm (maximum three arms)
- Move a mounder's arm to the right and to the left (maximum three arms)
- Rotate the mattock wheels "forward" (maximum three mattock wheels, one per arm)
- Raise and lower the front blade

Information about scarifiers is collected from [29] and [30].

B Different types of vehicles and their functions in the harbor

Below the two different types of vehicles that are used in the harbor and their functions are listed.

Forklift, totally seven (x2) different functions

- Drive forward and backwards (driven by wheels, four wheels)
- Turn right and left (steering at the rear wheels)
- Use the driving brakes
- Use the parking brake
- Raise and lower the fork carriage (that runs in the mast)
- Tilt the mast or the fork carriage forward and back to vertical position
- Make the distance between the two forks wider and narrower

Reach stacker, totally nine (x2) different functions

- Drive forward and backwards (driven by wheels, four wheels)
- Turn right and left (steering at the rear wheels)
- Use the driving brakes
- Use the parking brake
- Raise and lower the boom
- Move the crane extension outwards and inwards
- Rotate the lifting unit clockwise and counter clockwise
- Make the lifting unit wider and narrower (move the edges outwards and inwards)
- Open and close the four locking mechanisms (that connects the container)

Information about reach stackers is collected from [22].

C Different types of vehicles and their functions in the mining industry

Below the four different types of vehicles that are used in the mining industry and their functions are listed.

Excavator, totally 13 (x2) different functions

- Drive forward and backwards (driven by tracks, two tracks, or driven by wheels, four wheels)
- Turn right and left (the tracks are rotating at different directions or with different velocities, or steering at one pair of wheels)
- Use the driving brakes (when driven by wheels)
- Use the parking brake (when driven by wheels)
- Rotate the upper structure (where the cab and the crane are mounted) clockwise and counter clockwise
- Raise and lower the boom
- Raise and lower the arm
- Tilt the bucket (or other equipment) in the working direction (up and down)
- Rotate the bucket (or other equipment) clockwise and counter clockwise
- Tilt the bucket (or other equipment) sideways
- Open and close the locking mechanism for the bucket (or other equipment)
- Raise and lower the blade
- Raise and lower the ground supports (one or two pairs)

Information about wheeled excavators is collected from [24] and [31].

Hauler, totally five (x2) different functions

- Drive forward and backwards (driven by wheels, four or six wheels)
- Turn right and left (articulated steering or steering at the front wheels)
- Use the driving brakes
- Use the parking brake
- Raise and lower the dump body (and thereby automatically open and close a possible tailboard)

Information about haulers is collected from [25] and [26].

LHD loader (Load, Haul, Dump machine), totally eight (x2) different functions

- Drive forward and backwards (driven by wheels, four wheels)
- Turn right and left (articulated steering)
- Use the driving brakes
- Use the parking brake
- Raise and lower the two booms
- Tilt the bucket up and down
- Tilt the bucket sideways
- Open and close the locking mechanism for the bucket

Information about LHD loaders is collected from [32].

Wheel loader, totally seven (x2) different functions

- Drive forward and backwards (driven by wheels, four wheels)
- Turn right and left (articulated steering)
- Use the driving brakes
- Use the parking brake
- Raise and lower the two booms
- Tilt the bucket (or other equipment) up and down
- Open and close the locking mechanism for the bucket (or other equipment)

D Net function list

Below the resulting net function list can be seen.

Net function list, totally 42 (x2) different functions

Driving:

- Drive forward and backwards with wheels (four, six or eight wheels)
- Drive forward and backwards with tracks (two tracks)
- Turn right and left with articulated steering
- Turn right and left with steering on one pair of wheels
- Raise and lower one of the shafts in a bogie at the time (if the vehicle has got a bogie)
- Raise and lower pendulum arms (one per wheel, four, six or eight wheels)
- Use driving brakes
- Use parking brake

Operate a front loader:

- Raise and lower the two booms
- Tilt a bucket (or other equipment) up and down
- Tilt a bucket (or other equipment) sideways
- Open and close the locking mechanism for a bucket (or other equipment)

Operate a crane:

- Rotate the crane (and sometimes also a cab) clockwise and counter clockwise
- Raise and lower the boom
- Raise and lower the arm
- Move the crane extension outwards and inwards

Operate a crane's equipment:

- Rotate an equipment clockwise and counter clockwise
- Tilt an equipment in the working direction (up and down)
- Tilt an equipment sideways
- Open and close a grapple (forwarder)
- Hold a tree during the operations (harvester)

- Drive a chain saw during the operations (harvester)
- Rotate the rollers on a harvester head (feed the tree through a harvester head)
- Rotate a reach stacker's lifting unit clockwise and counter clockwise
- Make a reach stacker's lifting unit wider and narrower (move the edges outwards and inwards)
- Open and close a reach stacker's four locking mechanisms (that connects the container)
- Open and close the locking mechanism for a bucket (or other equipment)

Operate a mast:

- Raise and lower the fork carriage (that runs in the mast)
- Tilt the mast or the fork carriage forward and back to vertical position
- Make the distance between the two forks wider and narrower

Operate a scarifier's unit:

- Raise and lower a disc trencher's arm (maximum three arms)
- Move a disc trencher's arm to the right and to the left (maximum three arms)
- Rotate discs clockwise and counter clockwise (maximum three discs, one per arm)
- Rotate discs' engine housings clockwise and counter clockwise around their vertical shafts (maximum three discs, one per arm)
- Raise and lower a mounder's arm (maximum three arms)
- Move a mounder's arm to the right and to the left (maximum three arms)
- Rotate mattock wheels "forward" (maximum three mattock wheels, one per arm)

Other functions:

- Move a gate forward and backwards (forwarder)
- Raise and lower stakes (forwarder)
- Raise and lower a blade
- Raise and lower a dump body (and thereby automatically open and close a possible tailboard)
- Raise and lower one or two pair of ground supports (wheeled excavator)

E Metric for two HiL computers from Speedgoat

Metric for two HiL computers from the manufacturer Speedgoat. Metric for the Performance real-time target machine is collected from [33] (data for the third model from left) and (Email B Miesch 2020-02-26), and metric for the Mobile real-time target machine is collected from [34] (data for the model to the left) and (Email B Miesch 2020-02-26). * = The computer's size and weight are well suited for use in an office environment, and the computer's enclosure are able to handle the temperature and the humidity in the office.

Metric number	**Need numbers**	**Metric**	**Units**	**Performance real-time target machine**	**Mobile real-time target machine**
1	1	Compatibility with MATLAB and Simulink	[-]	Yes	Yes
2	2, 3, 4	Ability to install external programs	[-]	No	No
3	1, 2, 3, 4, 5	Number of processor cores	[-]	2/4	2
4	1, 2, 3, 4, 5	Processor clock speed	[GHz]	3,7/4,0	1,4/2,5
5	1, 2, 3, 4, 5	Size of processor cache memory	[MB]	No information on web page	No information on web page
6	1, 2, 3, 4, 5	Size of RAM	[GB]	16/32/64 DDR 4	4/12 DDR 3
7	1, 2, 3, 4, 5	Size of disk memory	[GB]	120/256/500 SSD or 1000/2000 SDD	64/128/256 SSD
8	6, 7	Number of Ethernet ports	[-]	3	4
9	7	Equipped with a wireless network card	[-]	No	No
10	7	Number of USB ports	[-]	2x USB 3.0	4x USB 3.0 2x USB 2.0
11	7	Number of HDMI, DVI or VGA ports	[-]	2x DVI-D	1x DVI, 1x VGA
12	8	Power supply with cable	[VAC]	230	230
13	9	Enclosure adapted for office environment*	[-]	Yes	Yes
14	10	Price	[SEK]	No information on web page	No information on web page

F Metric for two HiL computers from dSPACE

Metric for two HiL computers from the manufacturer dSPACE. Metric for the SCALEXIO Processing Unit is collected from [35] and (Email M Härberg 2020-03-02), and metric for the DS6001 Processor Board is collected from [36] and (Email M Härberg 2020-03-02). * = The computer's size and weight are well suited for use in an office environment, and the computer's enclosure are able to handle the temperature and the humidity in the office.

Metric number	**Need numbers**	**Metric**	**Units**	**SCALEXIO Processing Unit**	**DS6001 Processor Board**
1	1	Compatibility with MATLAB and Simulink	[-]	Yes	Yes
2	2, 3, 4	Ability to install external programs	[-]	Perhaps, depends on which program	Perhaps, depends on which program
3	1, 2, 3, 4, 5	Number of processor cores	[-]	4/8	4
4	1, 2, 3, 4, 5	Processor clock speed	[GHz]	3,8/2,6	2,8
5	1, 2, 3, 4, 5	Size of processor cache memory	[MB]	8/20	8
6	1, 2, 3, 4, 5	Size of RAM	[GB]	16/8	4 DDR 4
7	1, 2, 3, 4, 5	Size of disk memory	[GB]	480 SSD	8 flash
8	6, 7	Number of Ethernet ports	[-]	$\geq$1	2
9	7	Equipped with a wireless network card	[-]	No	No
10	7	Number of USB ports	[-]	No information on web page	1
11	7	Number of HDMI, DVI or VGA ports	[-]	No information on web page	0
12	8	Power supply with cable	[VAC]	230	24 VDC (Is built for use in a slot)
13	9	Enclosure adapted for office environment*	[-]	Yes	No (Is built for use in a slot)
14	10	Price	[SEK]	No information on web page	No information on web page

G Python script used to couple the AGX simulation to Simulink

The Python script used to refer to the .agx-file and to define the input signals to and the output signals from the AGX-block in the two Simulink scripts is shown on the following three pages.

```
# This file is used for defining input and output signals to/from the AGX-block in Simulink,
# and to load the .agx-file containing the TFP

# This file will only work if AGX has been built with agxMex

# Johan Borgström, 2020-05-06


# Import libraries, not sure that all libraries below are needed
import agx
import agxSDK
import agxOSG
import agxCollide
import agxPython
import agxIO
import agxUtil
import agxMex
import math
import sys
import numpy
import random
```

```python
# Define a class that will be responsible for input/output
# from/to Matlab/Simulink
# You can have one class for input and another for output if needed,
# or just implement the input/output methods to handle both
# Just make sure to specify the correct number of input/output variables
class InputOutputControl(agxMex.PythonControlArgument):
    def __init__(self, numInput, numOutput):
        # Call the constructor of PythonControlArgument
        # Specify the number of input and output variables
        super().__init__(numInput, numOutput)

    # This method is called at the startup of the simulation
    # It will get the intialization data from Matlab/Simulink
    def input(self, sim, data):
        assert(len(data) == 4) # 4 input ports to the AGX-block in Simulink

        rotVel_1 = data[0]  # The first input is the rotational speed of the TFP's front left motor
        constraint_1 = sim.getHinge("Motor_FL") # The joint's name is set in Algoryx Momentum
        assert constraint_1
        motor_FL = constraint_1.getMotor1D()
        motor_FL.setSpeed(rotVel_1)

        rotVel_2 = data[1] # The second input is the rotational speed of the TFP's front right motor
        constraint_2 = sim.getHinge("Motor_FR") # The joint's name is set in Algoryx Momentum
        assert constraint_2
        motor_FR = constraint_2.getMotor1D()
        motor_FR.setSpeed(rotVel_2)

        rotVel_3 = data[2] # The third input is the rotational speed of the TFP's rear left motor
        constraint_3 = sim.getHinge("Motor_RL") # The joint's name is set in Algoryx Momentum
        assert constraint_3
        motor_RL = constraint_3.getMotor1D()
        motor_RL.setSpeed(rotVel_3)

        rotVel_4 = data[3] # The fourth input is the rotational speed of the TFP's rear right motor
        constraint_4 = sim.getHinge("Motor_RR") # The joint's name is set in Algoryx Momentum
        assert constraint_4
        motor_RR = constraint_4.getMotor1D()
        motor_RR.setSpeed(rotVel_4)

        return True

    # This method will be called every time step
    # Reading data from AGX and feeding it back to Matlab/Simulink
    def output(self, sim, data):
        assert(len(data) == 3) # 3 output ports from the AGX-block in Simulink

        constraint = sim.getPrismatic("Slider_RR") # The joint's name is set in Algoryx Momentum
        assert constraint

        # read output data from simulation
        lastForce = agx.Vec3()
        lastTorque = agx.Vec3()
        constraint.getLastForce(0, lastForce, lastTorque)
        data[0] = lastForce.x() # The first output is the force in x-direction acting on the TFP's rear right prismatic joint
        data[1] = lastForce.y() # The second output is the force in y-direction acting on the TFP's rear right prismatic joint
        data[2] = lastForce.z() # The third output is the force in z-direction acting on the TFP's rear right prismatic joint

        return True
```

```
def setupInputOutputToMatlab():
    # Specify number of input/output signals to/from the AGX-block in Simulink
    numInput = 4
    numOutput = 3
    argA = InputOutputControl(numInput, numOutput)

    # Register the Input port (to AGX) to the controller
    agxSDK.SimulationControl.instance().addInputArgument("MotorSpeedsInput", argA) # The text inside "" will be seen in the AGX-block

    # Register the Output (from AGX) port to the controller
    agxSDK.SimulationControl.instance().addOutputArgument("SliderForcesOutput", argA) # The text inside "" will be seen in the AGX-block


# Now call the function (from global scope) which initializes the input/output
setupInputOutputToMatlab()


# Construct the simulation scene, load the .axg-file containing the TFP
# and enable control of the rotational speeds of the TFP's four motors
def buildScene():

    app = agxPython.getContext().environment.getApplication()
    sim = agxPython.getContext().environment.getSimulation()
    root = agxPython.getContext().environment.getSceneRoot()

    path = "C:/AGX Dynamics/assy-tfp_with_ground_2.agx" # The path to the .agx-file containing the TFP
    agxOSG.readFile(path, sim, root)

    initCamera(app)


# Defines the position of the camera,
# however, the view of the scene can be dragged, rotated and zoomed in Simulink
def initCamera(app):
    if not app:
        return

    cameraData = app.getCameraData()
    cameraData.eye = agx.Vec3(5.6719499826431286E-001, -1.2931400299072267E+001, 7.8934302330017090E+000)
    cameraData.center = agx.Vec3(5.6017297506332397E-001, -1.1988800048828125E+001, 7.5596299171447754E+000)
    cameraData.up = agx.Vec3(-2.4866606752084130E-003, 3.3379652952352334E-001, 9.4264186910869718E-001)
    app.applyCameraData(cameraData)
```