Universidade de Aveiro Departamento de Engenharia Mecnica
2019

João Adelino Gonçalves Pereira

Virtual Reality simulation of a Harvester machine

Desenvolvimento de um Simulador de uma Máquina Florestal em Realidade Virtual

Universidade de Aveiro Departamento de Engenharia Mecnica
2019

João Adelino Gonçalves Pereira

Virtual Reality simulation of a Harvester machine

Desenvolvimento de um Simulador de uma Máquina Florestal em Realidade Virtual

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Mecânica, realizada sob orientação científica do Professor Rui António da Silva Moreira, Professor Auxiliar no Departamento de Engenharia Mecânica da Universidade de Aveiro, e sob a supervisão na Universidade de Tampere pelo Professor Asko Ellman, Professor no Departamento de Engenharia Mecânica e Sistemas Industriais da Universidade de Tampere.

o júri / the jury

presidente / president

Prof. Doutor Joaquim Alexandre Mendes de Pinho da Cruz
Professor Auxiliar da Universidade de Aveiro

Prof. Doutor Fernando Jorge Fraile Fernández
Professor Colaborador Fixo da Universidade de León, Espanha

Prof. Doutor Rui António da Silva Moreira
Professor Auxiliar da Universidade de Aveiro (orientador)

agradecimentos / acknowledgements

This Master Thesis was elaborated during my academic exchange period in Tampere University, Finland. These past months have been an exciting adventure while developing this work and its accomplishment would have not been possible without the supervision and assistance of some people, to whom I would like to address my acknowledgements.

I would like to express my sincere gratitude to my supervisor from Tampere University, Professor Asko Ellman, for providing me the opportunity to develop this work. His guidance, advice and feedback were an inspiration to me, helping me to overcome many difficulties and refining my work.

I would like to acknowledge my supervisor from University of Aveiro, Professor Rui Moreira, for his prompt assistance and feedback. Despite the long-distance, some obstacles during the work were overcome with his help in a short-time.

I would also like to acknowledge, from Tampere University, University Instructor Ilari Laine, for providing the entire assistance with the CAD software and model used in the Thesis.

The technical support and suggestions provided by Algoryx Simulation AB, in particular from Doctor Claude Lacoursière, proved to be crucial to overcome challenges in the course of this work. I am very grateful for the prompt and clear assistance received.

This journey abroad was possible thanks to the Erasmus program and its coordinator of the Mechanical Engineering Department of the University of Aveiro, Professor Joaquim Cruz. To him, I am grateful for the help and guidance during all the process.

I would like to express my gratitude to my Family for their unconditional love and support, regardless the distance and time away from home. Also, I would like to thank my longstanding friendships and the new ones that I have created during this journey. Everyday, all of them are an inspiration to me.

keywords Virtual Reality, CAD, Virtual Prototyping, Unity, AGX Dynamics

abstract Computer-aided design (CAD) software is used in the product design and development to design complex and detailed prototypes. It provides good assistance and solid data generation to designers and engineers. In order to remain competitive, industry is always seeking for higher process efficiency and product quality enhancement in the shortest period of time. Continuous research keeps going to make it possible.

Virtual reality has been one of the research focus in the recent years. It is studied and applied to be used as an assistant tool in the product lifecyle management, particularly in facilitating the development phase. However, the implementation process from CAD to virtual reality remains a challenge due to time consumption and technology complexibility.

In this work a real-time virtual reality harvester simulator was developed. The start point was a 3D harvester CAD model. It was used the CAD simulator AGX Momentum, a game engine Unity and the physics engine AGX Dynamics to create dynamics simulation, to design a virtual forest environment and to enable physical controllers interact with the model.

With the capabilities of AGX Momentum, it was added dynamics motion directly in the CAD software, creating fast CAD simulations. A virtual scene was designed with Unity to simulate an environment and the immersion of the user on it with Oculus Rift device. The harvester model was imported to the Unity scene with AGX Dynamics.

In the end it was obtained a real size virtual prototype, with the possibility of interacting and control it using physical controllers. The user can visualise the scene in real-time through a head mounted display, providing him the experience of a real machine operator. Driving the harvester in a simulated forest, allowed to test the model in a hypothetical real scenario.

The process of implementing the CAD model in virtual reality used in this work, revealed to be efficient and intuitive. However, because it is a complex and large model, it was necessary to remove certain bodies (without dynamics effect) and reduce the number of contact points between components in order to balance the speed and performance of the simulator.

Following the same method used in this work, Other CAD models can be imported to virtual reality and be dynamically simulated.

palavras-chave Realidade Virtual, CAD, Virtual Prototipagem, Unity, AGX Dynamics

resumo

Software de desenho assistido por computador – do inglês, computer aided design (CAD) é usado no processo de desenvolvimento e design de produto para projetar protótipos complexos e detalhados. Este tipo de software fornece boa assistência e gera consistentes dados de modelos para designers e engenheiros. Com vista a manter-se competitiva, a indústria está sempre à procura de uma maior eficiência do processo e de uma melhoria da qualidade do produto a desenvolver no menor período de tempo possível. A investigacão, nos últimos anos e em curso, tem tentado tornar isto possível. A realidade virtual tem sido um dos focos dessa investigação. Tem sido estudada e aplicada para ser usada como ferramenta auxiliar na gestão do ciclo de vida do produto, em particular na facilitação da fase de desenvolvimento. Contudo, o processo de implementação do CAD em realidade virtual continua a ser um desafio devido quer à complexidade tecnológica envolvida, quer ao tempo requerido.

Neste trabalho foi desenvolvido um simulador de tempo real a partir de uma máquina florestal harvester em realidade virtual. O ponto de partida consistiu num modelo CAD da máquina. Para isso, foi utilizado o simulador CAD AGX Momentum, o motor de jogo Unity e o motor de física AGX Dynamics para adicionar dinâmica ao modelo, criar um ambiente virtual de uma floresta e permitir que controladores físicos interagissem com o protótipo.

Com os recursos de AGX Momentum, foi adicionado movimento dinâmico diretamente no software CAD, criando rápidas simulações. Através do Unity, foi projetado um cenário virtual para simular um ambiente virtual e promover a imersão do utilizador com o dispositivo Oculus Rift. O modelo da harvester foi, depois, importado para a "cena" com AGX Dynamics. No final, foi obtido um protótipo virtual em tamanho real, com a possibilidade de interagir e ser controlado usando comandos físicos. O utilizador pode visualizar a cena em tempo real através de um dispositivo de realidade virtual, proporcionando-lhe a experiência de um real operador da máquina. Ao conduzir a harvester numa floresta virtual, permitiu testar o modelo num cenário hipoteticamente real.

O processo de implementação do modelo CAD em realidade virtual utilizado neste trabalho revelou-se eficiente e intuitivo. No entanto, por se tratar de um modelo complexo e extenso, foi necessário remover determinados sólidos (sem interferência dinâmica) e reduzir o número de pontos de contacto entre componentes para equilibrar a velocidade e desempenho do simulador.

Seguindo o mesmo método utilizado neste trabalho, outros modelos CAD podem ser importados para a realidade virtual e ser dinamicamente simulados.

Contents

List of Tables

Intentionally blank page.

List of Figures

Part I

Introduction and Background

Chapter 1

Introduction

To create quality and safe products without errors in the most efficient way is the aim of the industry. Software tools and development methods have been assisting the creativity process in order to turn it effective and to decrease the time required.

Computer-aided design (CAD) software have been used in industry by engineers to develop 3D modelling and optimise products and mechanisms. It allows them to achieve very complex, large and detailed designs and obtain organised information about it. Typically, all the process is done and obtained through a restricted view of a computer monitor, being challenging to quick comprehend the model size and all the exact features. The design validation is mostly done by graphic data reading and technical draws from the developed sketches.

Virtual Reality (VR) systems have been rapidly developed and commercially available for everyone. It provides to the user walk-in capability in a computer simulated world and the possibility to interact with virtual 3D objects.

In the past years, VR technology has been studied and applied in the product design and development. With this external tool, several benefits were identified. It is now possible to create real size virtual prototypes to visual inspection and interaction with external physical devices [Kuusisto *et al.* 2012] [Kuusisto *et al.* 2011] [Ellman *et al.* 2007]. By doing this, conceptual designs can be tested and verified in earlier stages of development. Furthermore, VR applications were developed to train operators in different fields of industry work cell, machine driving and collaborative work possibility [Guo *et al.* 2018].

Despite the benefits, importing CAD models to VR still remains a challenge process to do. The complexity of this process and the high degree of technological dependence involved create enormous conflicts, making this process hard and high consuming time action [Kuusisto *et al.* 2012].

The game engine Unity is a popular tool to develop games for a wide range of operative platforms, including to VR devices. It has been used also to create environment simulators. The owner, Unity Technologies, provides a free version of the software with a very comprehensive online support. Algoryx Simulation AB developed the physics engine AGX Dynamics, dedicated to virtual objects simulation with integration on the CAD software SpaceClaim and in Unity.

1.1 Research questions

Motivated by the potential of VR systems in the engineering field and by the software available, in this work it is intended to overcame the challenges of conversion CAD models to VR. Therefore, the following research questions are proposed:

- RQ1. In what way a CAD model can be imported to Virtual Reality environment using Unity game engine and AGX Dynamics?
- RQ2. How can a large model be simulated in real-time and what facts are affecting it?

1.2 Goal of the thesis

This thesis work has the purpose of building a real-time simulator in VR environment from a harvester 3D CAD model, providing user capability to real scale visualisation and the possibility of interact with it. This work intends to demonstrate the different benefits of software and how they can be used to build and test a VR prototype dynamics.

1.3 Research methods

The *Literature review* was done to acknowledge the context of the work proposed and to obtain the necessary technological background to support the simulator implementation, giving an overview about the current situation of VR systems and their application. This project combine *modelling and simulation* and *action research* methods. The simulator was built and, as a result, an interactive implementation led to a practical, quick and efficient solution.

1.4 Thesis structure

In Chapter 2, it is presented an introduction to VR systems, followed by a theoretical background focusing on its application in the product lifecycle management (with its advantages and limitations). The technical functionality of the software and hardware used in the implementation is done in Chapter 3. The harvester CAD model is analysed in detail in Chapter 4, where the simulator goals for this model are detailed. The VR simulator construction is presented on the fifth Chapter. The results are discussed in Chapter 6 and the final Chapter presents the final conclusions and future work suggestions.

Chapter 2

Theoretical background

2.1 Virtual Reality

Virtual reality is a the result of an implementation of efforts to develop a virtual space, non-physically existing, to simulate the real world or hypothetical future situations with human involvement. Using both software and hardware, VR is a medium that generates to the user a sense of presence, participation and immersion in a virtual space with bi-directional sensory feedback [Kalawsky 1993] [Sherman and Craig 2003]. In other words, it is an application that gives a person the capability to navigate and interact with a virtual scenario. This experience can be defined by fundamental elements: virtual environment, sensory feedback, interactivity and immersion [Sherman and Craig 2003].

2.1.1 Virtual environment

Virtual environment (VE) or virtual world is a computer-generated system of a 3D digital space scene containing virtual elements, objects and human immersion. It provides to the user a synthetic sensory experience communication of physical and abstract components [Kalawsky 1993]. The aim of the VE is to take the participant from the physical existing world and insert him in an artificial world [Baus and Bauchard 2014]. Figure 2.1 shows an example of a VE composed by virtual objects displayed, where some of them are simulating the human behaviour.

Figure 2.1: Example of a virtual environment [Jeannie 2006].

2.1.2 Sensory feedback

A person is allowed to participate in the VE through physical devices. The VR system creates a bidirectional communication: the physical user position is communicated to the VE and it devolves a feedback of the user actions. These VR devices provides the VE visualisation in real-time and normally they support tracking and orientation position. This produces a virtual representation of the user in the virtual space. Extra VR devices can also be used to track other specific body parts, e.g. hands. With them, the user wins external capability to interact with the VE and the virtual objects displayed inside. The feedback provided by the VR has impact in the interaction and mental user immersion, contributing to the sense of presence in the VE.

2.1.3 Interactivity

The user actions have an impact on the VE. This can be done, either from his position in the real world (and its transposition to the virtual world) or from the possibility that VR system provides to the user of interacting with the virtual elements represented. This interaction tries to give the closest realistic experience, simulating the physical behaviour of the real world. Although the user can interact with the environment from a third person point of view, by controlling the representation of himself (known as Avatar), the first person perspective is the most common used in a VR application.

2.1.4 Immersion levels

Immersion is the user sensation of mentally being in a particular environment, in spite of not being physically there. Different VR devices have been developed over the years providing different immersion levels: non-immersive, semi-immersive and full-immersive.

Non-immersive VR system
Is the conventional visualisation of 3D models on a computer monitor or portable devices (smartphone, tablet) with interaction mostly done with keyboard, mouse or directly finger interaction in the screen. The Figure 2.2 shows an example of user interacting with a VE displayed in a computer monitor.

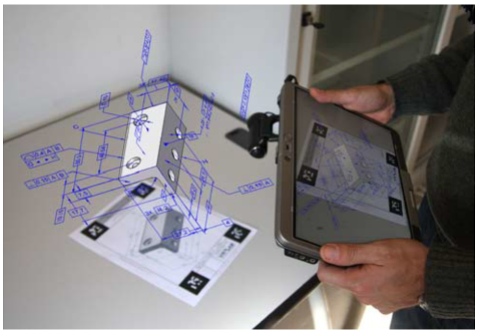
Figure 2.2: Display visualisation of a virtual environment [Wai 2014].

Semi-immersive VR system
Is a large display system visualisation that can include wearing stereo glasses. In this system, the user can use physical elements (e.g. cabin, cockpit) or enter in a Cave Automatic Virtual Environment (CAVE), a typical 3 to 6 projector walls in a cubic size room (figures 2.3a and 2.3b). The user can ear realistic and synchronised sounds alongside with the digital image. The user can also interact on VE with real physical tools as controllers, joysticks or a steering wheel [Tsyktor 2019].

Other concepts of VR can be included here. In recent years, the technologies of Augmented Reality (AR) and Mixed Reality (MR) were created. Their concept is related to the inclusion of virtual objects and their interaction with the real world. Through portable device, as smartphones, tables or glasses (e.g. Microsoft Hololens), users can observe the physical world as it is (using the merged camera) and also track their position. Then, depending on the application developed, computer generated objects are merged with the world, creating the sense of their real existence (Figure 2.3c).

(a) Car simulator [Tsyktor 2019].

(b) CAVE system [Campos 2018].



(c) Augmented Reality system [Fiorentino *et al.* 2009].

Figure 2.3: Examples of semi-immersive VR systems.

Although some of these technologies requires external device, the user is mentally aware of their physical real presence. Figure 2.3 presents examples of semi-immersive technologies of a car simulator using physical elements and monitors (2.3a) and using a CAVE (2.3b). The 2.3c shows an example of an AR system, where a portable device (tablet) uses its own camera to combine virtual objects with the real world.

Full-immersive VR system
Displays an image to a head-mounted device (HMD) that fills the visualisation and ear field of the user and tracks the head position in the VE. The use of data gloves or VR device controllers allows direct interaction with the virtual objects on VE. In opposite of

other levels, the full-immersive system total integrates the user in the VE, decreasing the sense of real world due to the full covered visualisation of the VE, as Figure 2.4 shows. Here it is also possible to observe, in an external monitor, what the user is visualising. The usability of physical movable components with VR systems also increases the user immersion [Kuusisto *et al.* 2011].

Figure 2.4: Full-immersive VR system with a head-mounted device [ESA 2017].

2.1.5 Current situation and general applications

The fast development of high-density displays, 3D graphics capabilities, sensor tracking, motion controllers and mobile technologies have been fundamental to the evolution of VR systems and to the reduction of their prices. Because of this, their application have increased. In 2017, the "virtual market" presented a revenue of 785 million dollars and it is expected to achieve 17 billion dollars by 2020 [SuperData 2017]. Currently, VR implementations are possible to be seen in education, communication, health care, entertainment, video games, training sessions and in engineering [Baus and Bauchard 2014].

The business market has started to adopt the virtual technology. Although the video game industry has currently the highest rate of investment (and the most profitable), other enterprises are strongly moving their efforts to implement and use VR in their business. The Figure 2.5 shows the different areas of implementation interest. The trend remains on training and simulation purposes, but other areas as product design, collaborative working, data analyses and marking are fields with expected VR applicability.

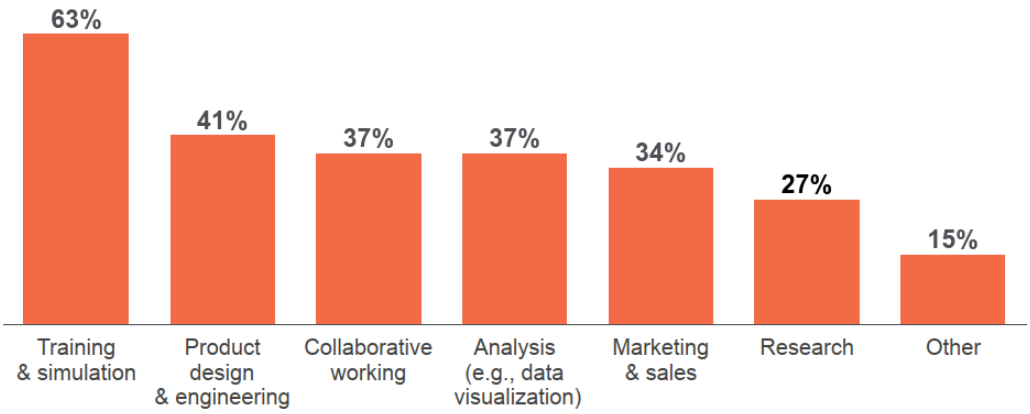


Figure 2.5: Enterprise use or plan to use virtual reality applications [SuperData 2017].

2.2 Virtual Reality and product lifecycle management

Product lifecycle management (PLM) is a process with multiple steps that goes from the very first idea generated until the product ends life. The necessity of creating new products generally came from customers needs, the implementation of new ideas or even from the development of new versions from existing ones. One fundamental stage of PLM is the product design and development, where the conception from the idea to the physical existing happens. This can be a long, interactive, detailed and expensive process. Also, all decisions made along the development phase have high impact in the final result. Increasing the efficiency and quality of the product as well as the conception process is a matter of constant development and improvement.

Creating new products has become increasingly complicated due to the required specification and complexibility of systems necessary to develop. Today, it is necessary not only to create good quality products that fulfil the customers requirements and needs, but also to develop them in a minimum time and costs possible. In order to make this possible, some trends have already been identified in the process of product design and development: current engineering, group activity, co-creation, usability of scenarios and applying game principles [Tideman *et al.* 2008]. The possibility and facilitation of integrating VR systems has been an important factor to use them.

In group activity, product development generally requires knowledge from different fields, leading to the necessity of joining together several different experts. Also, it is important to involve different stakeholders in the process. When there are different agents involved in the same project it is crucial to have proper communication. New platforms as social VR have the ability to bring people from different places around the world to the same virtual room, in real-time.

Currently, engineering teams aim to reduce the time of bringing a product to the market, giving the importance to perform several tasks at the same time in the conception process. The iterative development method replaces the traditional sequential tasks, where different departments are working simultaneously in the different stages of product's development. The availability of testing the design and the prototype in earlier stages and in hypothetical scenarios help to introduce faster changes and improvements.

The co-creation is the involvement of further end users, operators or/and customers in the product development stage. Involving them in the discussion of ideas and requirements and obtaining a Quality Function Deployment is indispensable to focus on product goal. But, having their further feedback on prototypes, discussing new design ideas or different approaches and improvements to make a better product is fundamental [Kuusisto *et al.* 2012] [Tideman *et al.* 2008]. Prototypes or mockups are a good way to improve the communication between designer and user/costumer, where both parts can analyse and discuss any details in particular. They can be obtained by using physical materials or 3D printing technologies, achieving a small scaled product with the most highlighted features. Other approach that have been used is the application of VR systems, creating virtual prototypes and simulators. They allow an evaluation in real size and interaction in a hypothetical real situation in a VE. This virtual approach can be an efficient way to promote co-creation, due to a higher focus of the user on the product development itself than on the physical prototypes [Tiainen *et al.* 2014]. Also, fast modifications can be made in the prototype to test it again, decreasing time consumption. Using VE can facilitate negations and remove barriers from different backgrounds or assumptions between designers and users [Ellman *et al.* 2007].

One example of VR system to co-creation is the VIP2M [Kuusisto *et al.* 2012], a walk-in VE prototyping for a mobile machine that allowed operators to test and to interact with it in a virtual scenario. In Figure 2.6 it is possible to see the VIP2M CAVE system with a movable chair that provides real-time feedback from the user actions in the scene.

Figure 2.6: Operator testing a virtual prototype in a CAVE system [Kuusisto *et al.* 2012].

Using scenarios to test a prototype gives a context and an environment that somehow reveals an hypothetical real situation. It is necessary to keep in mind that the product to develop will be used somewhere, to perform something. By creating a VE and displaying the prototype there is a good way to understand even better the design, to simulate behaviours, to find possible issues and challenges to solve [Tideman *et al.* 2008].

The game industry has been experiencing big developments over the time, creating even more video games that demands user decisions as in a real world situation. In every game, there is the need to find a solution or task to solve. Most of the times, user feelings and emotions are factors that are used in a game. Allowing virtual interaction of a prototype or virtual simulation of tasks in a specific context scenario, gives the user a near computer game experience. For example, it is necessary to test a functionality in a product, a detailed guidance can be given to the user to follow steps or solve tasks. Furthermore, developing a realistic game has become easier due to the fast development of games engines: very complete platforms that includes modelling possibility, rendering graphics, collision detection, audio performance and artificial intelligence capability. One good example of this is the game engine Unity, where it is allowed implementation of AR, VR and MR with prototypes in a VE. It has been used to assist designers in product development stage to build simulators [Zheng *et al.* 2017] and create future environment demonstrations in studies [Kaapu *et al.* 2013], enabling to provide good sense of immersion to the user.

2.2.1 Advantages

Due to the software and hardware technologies advances, designers and engineers have been using more and more VR technologies. Thereby, the product quality and work efficiency have been increasing [Blümel *et al.* 2004]. In Figure 2.7 it is possible to see the VR usability in each step of product development and design.

IDEA	• Rapid visualization/comparison of existing products and context
CONCEPT DESIGN	• Design reviews • Concept validations • Design comparison
DETAILED DESIGN	• Modelling • Assembly modelling
PROTOTYPING	• Design optimization • Shape optimization • Topology optimization
TESTING	• Simulators • Feasibility test • System integration • Ergonomics
PRODUCTION PROCESS PLANNING	• Fixture design • Job planning and training
MANUFACTURE	• Data observation
USE	• Teach and training • Data observation • Maintainability

Figure 2.7: General virtual reality capabilities in product lifecycle management.

To help in the process of brainstorming and idea generation, it is important to understand the current market and the actual products offer status. Having the possibility to rapidly examine existing products and scenarios where the product will be included, VR helps the designers to better and faster understand the direction to follow.

Computer-aided design (CAD) software is a popular tool to build and to model detailed designs. But the complexity of parts and assemblies requires detailed visualisation to understand the integrity and spacial location of each component designed. One good benefit that VR can bring is the walk-in the design. Herewith, it is possible to reach the prototype experience, allowing the model inspection and the possibility to better understand the design and system optimisation of the product.

When performing tests of the prototype through VE scenarios, a simulation of the product performance can be obtained in a hypothetical situation. At the same time, it can be easier to understand the ergonomics of the model due to the real sized prototype. VR technologies can assist in design and in maintainability optimisation, helping to simulate the entire process and to detect existing defects on the prototype, even before the actual production of the physical product [Guo *et al.* 2018].

When the product is mostly finished, it is necessary to make a production process planning. Through the capability of detailed visualisation over the layout design, VR can help with the optimisation layout and with the productivity, the assembly process and operator guidance performance [Blümel *et al.* 2004].

The VR technology can also assist in the business communication, helping with the marketing and the product demonstration, again even before start being produced. Fairs and exhibitions are common examples of places to do it. Whilst large real prototype dimensions can be physically hard to carry. HDMs have the advantage of high portability, enabling the fast immersion of users in VE to observe the product as it is [Kaapu *et al.* 2013]. Other interesting fact is that a VR presentation compared with video presentation allows participants to visualise the product from their own point of view, having a better understanding of the features and their complexity [Blümel *et al.* 2004].

In general, VR helps to accelerate the process of product development and design. Also, it facilitates the group activity, current engineering and co-creation. The usability of VE scenarios and game principles help on defect detection and can increase product quality.

2.2.2 Limitations

The CAD software is the standard tool to model and create a product with detailed design information. However, creating real size virtual prototypes remains a challenging process. Sharing and converting the 3D CAD models to VEs is a hard process and a very high time-consumption task since CAD software do not offer VR support yet. However, efforts and advances have been made in order to make it possible.

Game engines platforms have been used for developing VR systems. Using heavy CAD models data in these platforms demands conversion in specific lighter file formats, incompatible with CAD software. For that, it is necessary to use external software that can make detailed model information be lost. Other aspect is that interaction development between model and user may require external technical and programming capabilities, being a potential barrier to the fast implementation of VR systems with CAD models.

Chapter 3

Technological background

3.1 Physics engine

Physics engine is a computer software that allows representation of physical objects or systems such as rigid bodies or fluid dynamics. Consequently it simulates them, predicting and managing their behaviour and interactions by solving the dynamics system equations of motion and detecting collisions, evolving the simulation forward in time.

The generation of contacts and the involvement of controllers brings frequently variable and unpredictable systems during simulation. A physic engine uses a time step to detect collisions and then it generates a response by calculating velocity and position of the objects using integration of differential algebraic equations. This response relies on information properties such as mass, coefficient of restitution and collision points. The AGX Dynamics is the physics engine used in this work.

3.2 AGX Dynamics

AGX Dynamics, developed by Algoryx Simulation AB, is a multi-purpose physics engine to create simulators in different fields of mechanics, materials and industrial processes. It allows motion dynamics simulation of 3D elements such as terrains, vehicles, wires, hydraulics systems, materials elasticity, mechanical constrained systems, small particles, hydrodynamics systems and other complex mechanics systems. The results can be used to test virtual prototypes, operator training on a specific field of application or even to commercial purposes, with capability to demonstrate product features.

The software is architectured as a Software Development Kit (Figure 3.1) that provides representation and interaction between multibodies systems with nonsmooth dynamics. It uses the numerical based stepping method SPOOK [Lacoursiere 2007] as foundation for numerical time integration and high-performance parallel equation solvers. This results in robust and valid simulations. The background core consists in *C++* classes derived from discrete Lagrangian mechanics for constrained systems with dry frictional contacts.

Simulations can be done and managed directly through the *C++* API, using high level scripting with Python, C# or Lua. Furthermore, AGX Dynamics have the possibility to develop simulations through a graphical user interface (GUI) with integration in other two platforms: the CAD software SpaceClaim through a external plug-in, named

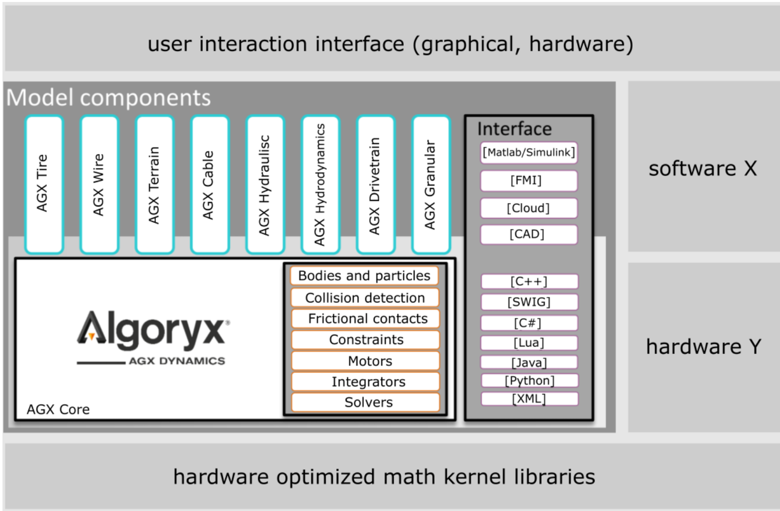


Figure 3.1: AGX simulation overview [Servin and Brandl 2018].

AGX Momentum, and the game engine Unity through AGXUnity. These inclusions will be discussed later on sections 3.3 and 3.4, respectively.

AGX Dynamics is divided into AGX classes that specifically calculates and integrates the simulation in a time step. The simulator requires a definition of *simulation frequency* to run the simulation in a specific time step, δt:

$$\delta t = \frac{1}{simulation frequency} \tag{3.1}$$

In each δt are calculated the rigid bodies behaviour, collision detection, contact points and solutions related with them. Simulations can be complex with necessary heavy calculations, requiring different δt. It is fundamental to have a right definition of *simulation frequency* to obtain proper results. Defining a low *simulation frequency* may lead to unreal collisions with large, or even missed, geometry penetrations. High *simulation frequency* can originate better results but can also slow down the simulation.

AGX Dynamics has a structural functionally, dividing different classes to caring with different aspects in simulation. In Figure 3.2 it is possible to see the simulation structure and its ramification. The *AGXCollide* class concerns about the simulation space in terms of contact collision. In each δt it is responsible for constantly geometric overlap tests: Broad Phase Test, where the geometries are testing volumes overlaps, and Near Phase Test (if Broad Phase Test detects bodies about to collide) provides detailed contact information of contact points, normal and depth penetration. The collision detection is related to geometry shape and material. The *DynamicsSystem* class solves in each step all constraints calculations pre-defined (rigid bodies and joints) and handle the collision detection constraints in the simulation. It can be configured to 3 different type of solvers: iterative, direct and split solver.

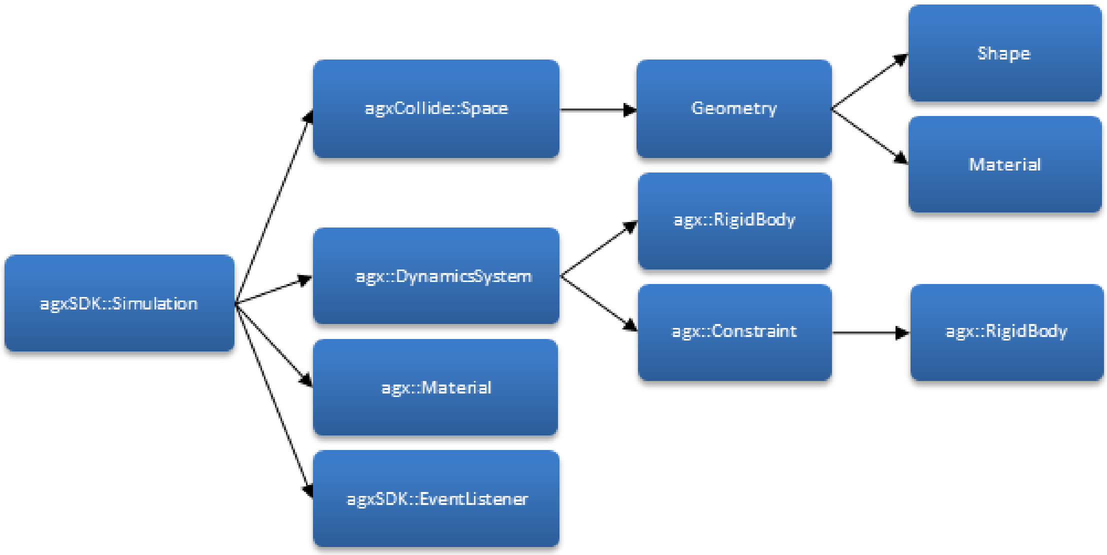


Figure 3.2: Structural description of AGX Dynamics simulation [Algoryx Simulation AB 2018a].

Iterative solver: progressively improves the solution with a finite number of iterations, delivering a solution with finite precision. This is a conceptually simple solver that is fast and efficient for large scale stacking problems with homogeneous type of bodies but less efficient for wires with large mass and force ratios.

Direct solver: it finds the exact solution in a limited number of operations. It can handle large multibody systems with high mass ratios and maintaining constraints accurately. It is slower when comparing with iterative solver but performs less errors and provides higher accuracy. It is the most stable solver when used to solve both constraints and contacts collisions.

Split solver: all constraints are solved in the iterative and the direct solver. First, the constraints are solved directly, and then the normal and frictional contacts are solved. This results in a fast solution for moderately large contact scenes with friction. Yet, for large systems, the pure iterative solver is better and more efficient. The split solver is slower when compared with the iterative solver but it results in less errors, producing better friction contacts. When using a large number of objects it can slow down the simulation.

The simulations are composed by geometry shapes and joints that provide fixed constraints in the simulation. These geometries are used as rigid bodies that have physical properties such as mass and inertia tensor and dynamic properties as position, orientation and velocity. A rigid body can have 1 of the 3 motion control schemes: *Dynamics*, behaving as a free body and struggling with forces; *Static*, not affected by any force and *Kinematics*, where the rigid body has infinite mass, enabling velocities and disabling forces interaction. Rigid bodies have a geometrical representation and a collision detecting mesh. Triangle meshes are used to create the outside shapes as boxes, cylinders or more complex and higher computer cost performance as planes. Here it is used rectangular grid divided into a lower left and an upper right triangle, allowing to be modified under runtime on collision events.

The AGX class *Material* is related to specific surface material properties of the geometry as Young's Modulus, density, roughness, viscosity (restitution), specification of stiffness of the rigid body contact and adhesion (definition of stickiness). These properties are used in solving contact collisions between rigid bodies and their behaviour on the simulation space. Contacts are detected and constantly updated during the simulation. Mechanically, they are a linear-elastic material constitution specified by the material properties. Contact surface, penetration and material volume are variables used to achieve a stable contact mechanics, originating an approach to the real contact behaviour pretended.

The physics engine allows definition of geometrical constraints to create relations between rigid bodies and the space. They can be placed in rigid bodies in a form of joints that restricts their degrees of freedom (DOF) in different ways. In Table 3.1 it is possible to verify the different type of joints available and understand what they represent, having a brief description and their individual DOF limitation associated.

Table 3.1: AGX Dynamic joints.

JOINT	**DESCRIPTION**	**DOF**	
		Translation	Rotation
Ball-joint	Ball and socket joint	0	3
Hinge	Rotation around 1 axis	0	1
Lock	Block all DOF	0	0
Prismatic	Translation along 1 axis	1	0
Cylindrical	Combination of hinge and prismatic joints	1	1
Spring	Linear spring	1	3

Besides restriction on DOF, the joints allow to modify the properties associated to them, changing their behaviour. Hinge and cylindrical joints can have *angular motor* enabled, forcing rigid bodies to rotate with a certain speed. Also, an *angle range limitation* with maximum torque associated can be defined in order to limit the rotation of the rigid body. On prismatic and cylindrical joints, the *linear motors* may force linear travelling and also have range limitation.

3.3 SpaceClaim and AGX Momentum

SpaceClaim is a 3D solid modelling software CAD owned by Ansys. It provides the functions to model 3D objects and create system assemblies. The software offers flexibility in importing files direct from other software format as CATIA, NX Siemens, Inventor and SolidWorks, making edition possible as an original SpaceClaim file. AGX Momentum is the integration of the physics engine AGX Dynamics in SpaceClaim, in form of an add-in, providing a GUI to create simulations from CAD models.

3.3.1 Interface

The Figure 3.3 shows generic SpaceClaim interface. In the window there is the working area, which is the place where the solid construction happens and the user visualisation occurs. On the *Menu task* it is possible to have access to all the tools concerning the modelling and assembly solids. Here, there is also the access to the AGX Momentum add-in, with the tools to build the simulation. The solids and the assemblies are disposed in the *Solid Structure tab*, where it is possible to create layers of components and to group solids into them. On the right side of the interface, there is the *Properties tab*. Here, it is possible to access the properties of the solids and of the simulation provided by AGX Momentum.

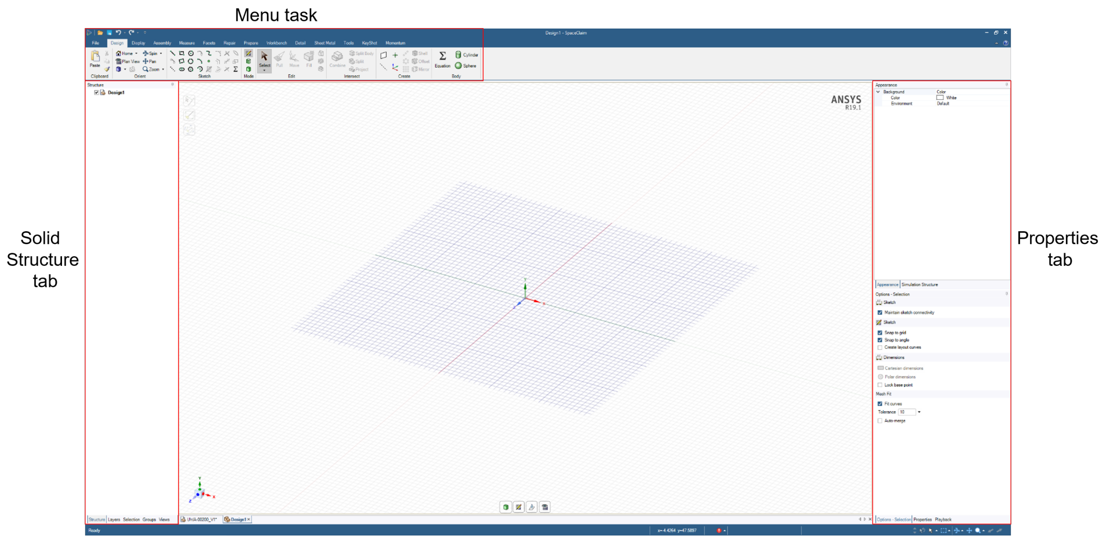


Figure 3.3: SpaceClaim interface.

The add-in AGX Momentum needs to be activated by the user in order to have the tools to start the simulation. After that, automatically options, tabs and a ribbon menu becomes available, as it is possible to see on the interface screen capture in Figure 3.4.

On the *ribbon menu* it is possible to define the frequency of the simulation, merge/split components in rigid bodies, define the motion control of each one of them, auto detection or manually place joints and joint modifiers, define material pairs and contact solving method, plot live data during the simulation, use Python scripting and have access to the information of the simulation. The *Options tab* displays various options for the current tool selected. When using *Detect Joints*, it is possible to define the thresholds of parameters to detect a joint (maximum diameter difference, maximum axial distance, minimum overlap, maximum angle difference). Also here, it is possible to specify the values when a new joint is created. The *Simulation structure* provides the structure of the dynamic simulation, containing all the rigid bodies, joints and solids. To change or set values of all properties concerning to solids, rigid bodies or joints it is used the *Properties tab*. Also here it is possible to enable/disable collisions of components/rigid

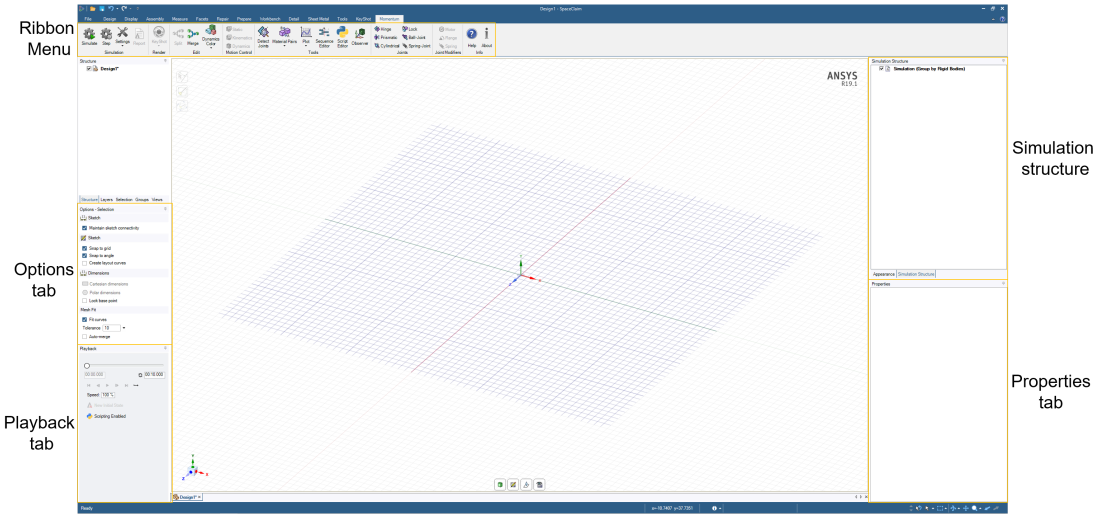


Figure 3.4: AGX Momentum interface in SpaceClaim.

bodies, motors and range of joints. After running the simulation, a playback can be done of the simulation using the *Playback tab*.

3.3.2 Workflow from CAD to simulation

To obtain a simulation of a CAD file, AGX Momentum requires defined steps [Algoryx Simulation AB 2018b]. The Figure 3.5 shows the AGX Momentum workflow.

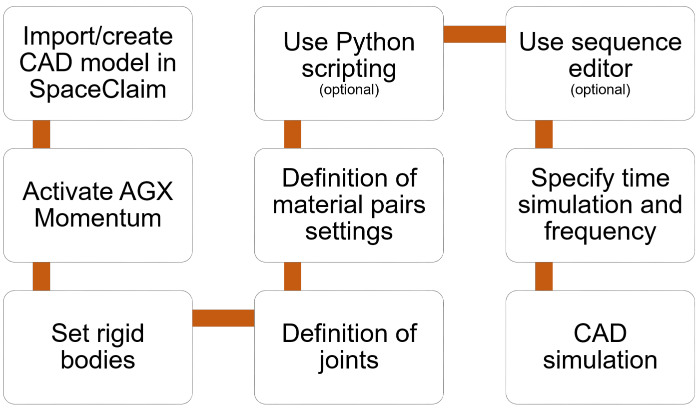


Figure 3.5: AGX Momentum simulation workflow [Algoryx Simulation AB 2018b].

The solids that are composing the CAD file in SpaceClaim needs to be associated to a component. After that, the plug-in AGX Momentum needs to be activated. By default the entire CAD model is one single rigid body. It is crucial to split all the components and then merge the desired ones to act as an individual rigid body. The software generates automatically data of mass properties (that can be manually changed) based on the solid dimensions and the material definition. Then, it is necessary to select the motion control of the rigid body: dynamics, static or kinematics. Also, by default a material is associated for each rigid body, having the possibility of choosing a different material or to create a new one, individually. When solids are in the same rigid body, collisions between them are disabled. It is also possible to disable collisions of a component manually, but the software will still provide their mass properties and perform their dynamics.

Having all desired bodies defined in the simulation, joints can be added. AGX Momentum can automatically recognise joints in touching or near rigid bodies. The user must define the most convenient joint type (Table 3.1). Joints can be manually set. As discussed before, joints can have motors associated, being possible to enable and to manipulate them in order to provide the desired behaviour of the joint. Here, the solving joint type can be defined. The Figure 3.6 shows on example of *Properties tab*, more specifically of the hinge constraint.

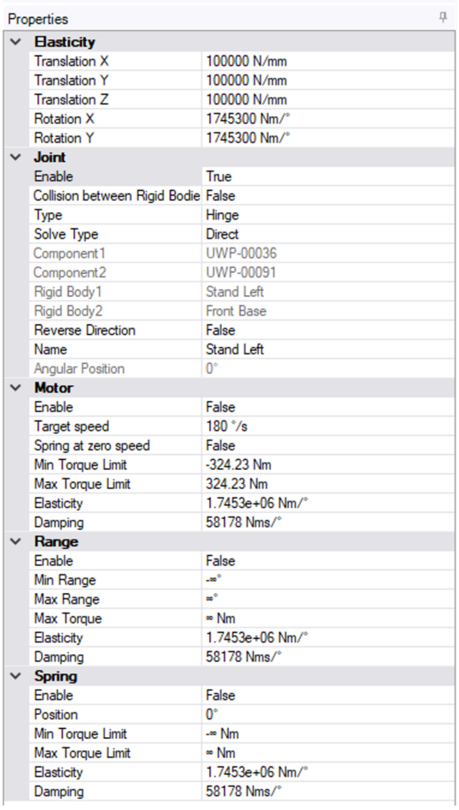


Figure 3.6: Hinge constraint properties on AGX Momentum.

Other important setting is the definition of material pairs: definition of interaction between the different materials created for each solid. A solid has one material associated with different properties. Then, different properties (restitution, friction, Young's Modulus, elastic domain and solving method) can be set to obtain a desired collision/interaction between different solids.

The software also provides a Python scripting environment to write scripts that will be executed during the simulation, being possible to write and read data. To add dependent actions into a simulation, AGX Momentum provides a sequence editor to create operations and to manipulate joint properties.

As discussed before, defining the simulation frequency is an important step to obtain the desired simulation. Moreover, defining the duration time of the simulation allows to change the time that is required to simulate the entire system. Then, it is possible to perform the simulation, having the possibility of visual analysis through the working area or through graphical area, depending on the parameters required to observe.

The simulation does not need to be done in the last step, it can be done along the process. This allows to a better understanding of the system and the definition of solids/rigid bodies/ joints. Also, to run the simulation, all constrains available from SpaceClaim (Tangent, Align and Orient components) defined on the CAD file will not be considered on the dynamics simulation.

The simulation parameters defined in the CAD simulation can be saved and exported as AGX Dynamics Binary file (.AGX), where it is kept all the data generated from the simulation.

3.4 Unity

Unity is a professional real-time game engine owned by Unity Technologies, frequently used to create and develop games. With this software, it is possible to develop virtual scenarios and add or create 2D and 3D assets into a scene and use light, audio, physics properties, animation, interactivity and game play logic. This software provides an integrated development environment, with a gathering of a GUI manipulation of objects on the scene, a code editor as Visual Studio and the game engine itself that allows to run the scene created with the objects. To step the virtual scene, it uses a frame rate, expressed in frames per second (FPS), creating a sense of motion to the human eye.

Unity provides possibility of scripting in C# and JavaScript to define logic, add behaviours and create reactions from user input. Unity supports development to a large number of platforms such as computer operation systems, mobile operation systems, web players, Smart TVs, video console games and AR, MR and VR devices.

The Unity project development is mostly done based on manipulation and interaction between user and game objects. Unity provides basic 3D objects as cube, spheres, cylinders, plans and terrains to add in the project scene. To use detailed and specific models, 3D objects developed in other software can be imported to Unity in formats of FilmBox (.FBX file) or .OBJ file. There is also the Unity Asset Store, where developers provides/sell their own models to use in the project. The Figure 3.7 shows a virtual scene with game objects developed for demonstration purposes in Unity.

Figure 3.7: Designed scene example in Unity [Körner 2015].

Due to several developing platforms availability, Unity allows to create and inspect models using a simple computer monitor or using AR and VR devices.

3.4.1 Interface

Unity provides an interface with several tabs with different functionalities. In Figure 3.8 it is possible to see the Unity interface.

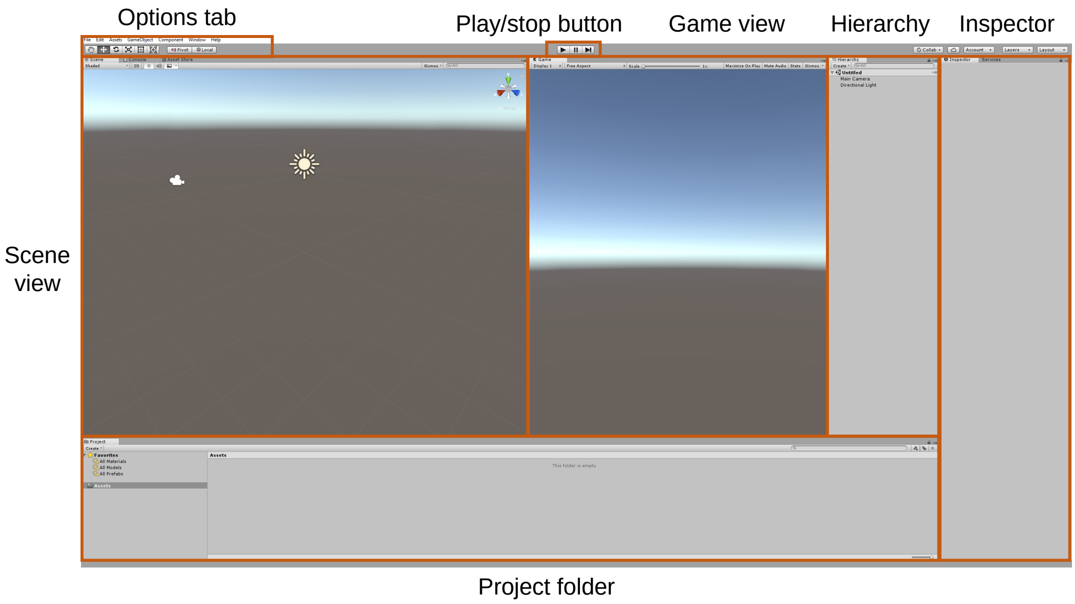


Figure 3.8: Unity interface.

The *Scene view* is where the graphical project part is built, allowing creation and direct interaction with objects on the scene. In the *Project folder*, all assets (3D models, textures, materials, audio clips, scripts) are disposed, available to use. The *Hierarchy tab* presents all objects that currently are being used in the scene. Here it is possible to create parent/child relations, where child objects will keep the same position relating to the parent object movements in the scene. On the *Inspector tab* there are all the details and properties through the components or scripts that each object or asset is associated with. Clicking on the *Play/Stop button* will activate/deactivate the *Game view* that allows to preview the built scene in the editor and making possible to test and interact with the player controllers. The *Options tab* provides access to the tools focus, move, scale and rotate the objects.

3.4.2 AGXUnity

Algoryx Simulation has integration and support of AGX Dynamics on Unity, providing conditions to create simulations in real-time, affording real physical properties to 3D objects and real collision solutions. Importing AGX assets already built for the Unity project, the software provides simulations using the tools of the AGX Dynamics physics engine. All the calculations for the simulation are performed by this software in real-time. The objects can be used from the AGXUnity library or attaching AGX scripts previously developed in Unity objects. The graphic quality and the virtual scene are provided by Unity.

AGXUnity allows to build simulations with the same workflow as AGX Momentum (discussed on Section 3.3), since both have the same background engine. The integration creates an external plug-in in Unity that provides access to AGX simulation components such as rigid bodies, constraints and all managed simulation settings as in AGX Momentum. Besides, it provides tools to create cables, wires, water and wind motion. AGX components can be added in form of scripts, being associated to objects and allowing to set simulation values and properties. Every AGX object created on the scene has a rigid body with mass properties and a body collision with type and properties material that can be accessed through the *Inspector tab*. Also, properties concerning constraints can be accessed in the same place with the same properties that AGX Momentum provides: motors, range controllers and forces. The Figure 3.9 shows the hinge joint properties on AGXUnity.

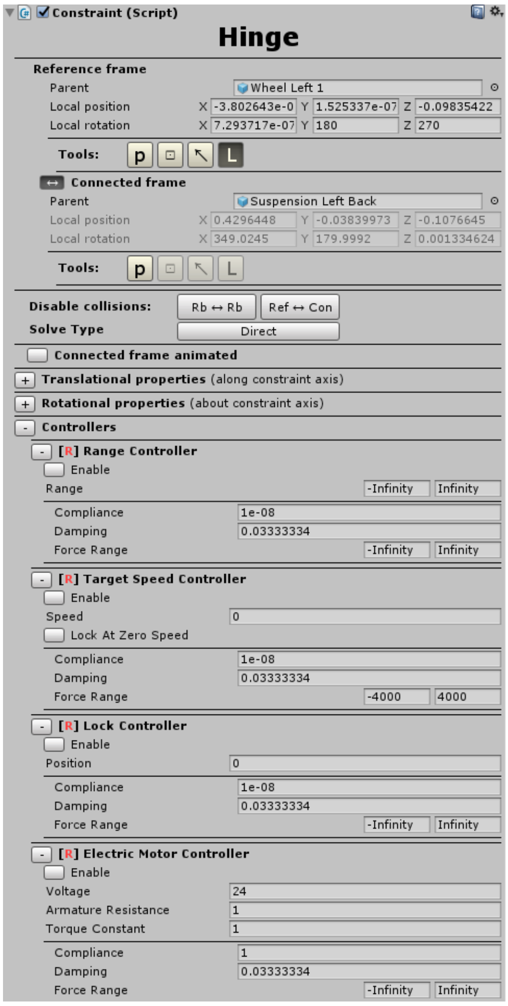


Figure 3.9: Hinge constraint properties on AGXUnity.

On Project folder, material pairs are created for the simulation. Through the *Inspector tab* the properties related to material collisions, Contact Solving Type, Friction mode and Contact Reduction Level can be accessed. The Friction Mode has three levels: *Box Friction*, *Scale Box Friction* and *Iterative Projected Friction*. The first one provides static model bounds on the solve stage. The second one uses the current normal force, requiring more computer performance compared with the first one but a more realistic dry friction. The third friction mode is the cheapest computer performance, where normal forces are first solved with a direct solve and then normal and tangential equations solved iteratively.

The contact reduction is a mode and level reduction of contact points to solve, with the aim of increasing the simulation performance. There are three modes available: *None*, where there is not contact reduction; *Geometry*, where the interactions in the narrow phase stage between geometries are reduce; and *All* mode, where the number of contacts are controlled and reduced in interaction between geometries and rigid bodies. The Contact Reduction Level (*Minimal*, *Moderate* and *Aggressive*) indicates the intended level of the Reduction Mode.

When running a project with AGXUnity, the *Game tab* provides statistical information concerning the simulation, as Figure 3.10 demonstrates. It updates the time (in milliseconds) used to collision detection and the simulation Dynamics solver in each time step, δt. Also, it disposes data information about the frequency in use, number of rigid bodies, shapes (components) and joints (constraints) present.

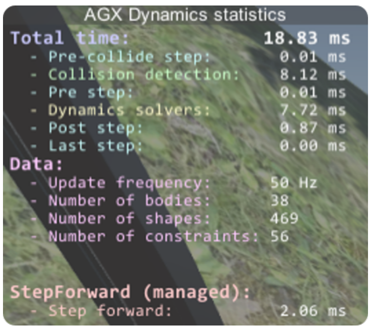


Figure 3.10: Real-time AGX Dynamics statistics in Unity.

With this integration is possible to import .AGX files and have 3D models, settings and properties from the simulation created in AGX Momentum.

3.5 Oculus Rift

Oculus Rift are a HMD VR device that provides full-immersion experience to the user in a virtual space. They have an AMOLED 5.7" display with a resolution of 960x1080 that gives a 100° field of view and headphones attached to provide sound. Apart of the headset, there are two portable touch controllers, ergonomically designed for each hand which allows the user to walk-in and interact with the VE scene, displayed in the AMOLED screen. The Figure 3.11 shows the Oculus Rift hardware with two touch controllers on the left, the headset and two tracking sensors on the right.

Figure 3.11: Oculus Rift devices: 2 touch controllers, headset and sensors.

The Oculus Rift devices (headset and touch controllers) have their positions tracked by two sensors positioned in front of the play space, replicating and providing the user head and hands movements in live time to the computer and in the VE. The touch controllers provide buttons, triggers and joysticks (Figure 3.12) to menu selection and hand interaction with virtual objects.

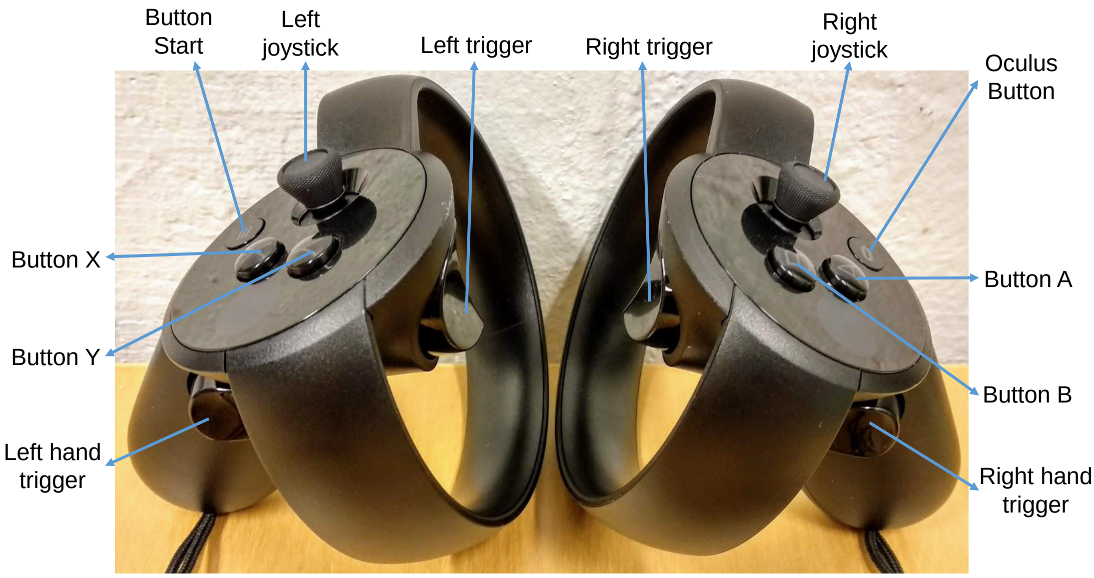


Figure 3.12: Oculus Rift controllers keys.

Oculus recommends to have a high hardware level to ensure a proper experience and performance in VR. The minimum computer specifications recommended are a CPU equal or superior of Intel i3-6100, 8GB of memory RAM, a graphic card Nvidia GTX 960 and Windows 10 as operative system.

3.5.1 Integration on Unity

Unity allows the project development to be used in Oculus Rift. For that, it is necessary to import an Oculus package developed by Oculus from the Unity Assets Store [Oculus 2019]. This package contains scripts and prefabs that when added to Unity scene, provide head and hands tracking, walk-in capability on the project and interaction with the virtual objects displayed. When the play button is pressed, the virtual camera view from the game environment is automatically presented on the headset. The user can freely move the head and have its own perspective. Also, the controllers may show their virtual perspective on the VE. The Figure 3.13 shows a screenshot of Oculus Rift running on the *Game tab*, where the user's perspective is shown.

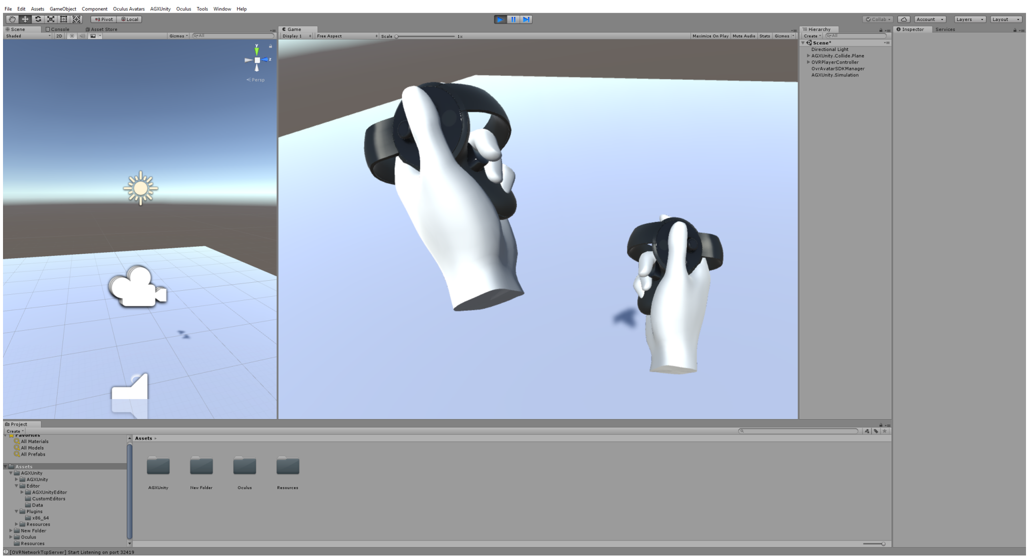


Figure 3.13: Oculus Rift running on Unity.

The Touch controllers buttons and joysticks can work on Unity as an user input that can be acceded and controlled by scripting. This allows to have action on other scene objects and in their own components. The integration of Oculus Rift and AGXUnity is possible without compromising the features of these two different platforms.

Intentionally blank page.

Part II

From Computer-Aided Design to Virtual Reality

Chapter 4

Harvester CAD model

In this work it was used and tested a real model scale of an harvester machine. The model is the Forest Master Turbo, developed by Usewood Forest Tec Oy, a Finnish company specialised in forestry machines. The model was provided in the SolidWorks CAD format representing all the complexibility and details of the harvester parts. This makes a good testing model, once there is low or non chance of modelling errors. Also, testing a large and complex model can provide better understanding the dynamics of the simulation tools and the factors that affect real-time simulation. In Figure 4.1 it is possible to compare the physical (4.1a) with virtual model of the harvester (4.1b).

(a) Physical harvester [Usewood Forest Tec Oy 2019].

(b) Virtual CAD model.

Figure 4.1: Physical and virtual harvester model.

Harvester machines are forest vehicles used to facilitate forest management. They can be used to cut and transport all types of vegetation by using a control crane and a specific tool attached in order to perform more demanding tasks. The design, number of wheels and dynamics provide stable driving in almost all the terrains with considerable inclination. This is necessary due to the fact that most of the work is being done in forest environment.

This model is a small dimensions harvester, used to deal with small/medium size vegetation. It has 8 driving and traction wheels, cabin, 2 support legs, 1 crane and 1 attachable cutter. The engine is positioned in the back part of the machine, providing a hydrostatic transmission to the operable parts.

In real use, an operator is necessary to drive the machine (as it can be seen inside the cabin in Figure 4.1a). Here, the operator has access to joysticks, pedals and buttons to drive and control the machine.

4.1 CAD Model details

The harvester has a length of 3,8 meters and a width of 1,5 meters. Each component of the virtual harvester was modulated and then assembled in a main assembly. In the model there are a total of 3298 solids in real scale, representing all the parts of the harvester. This includes detailed bodies inside the main frame, cabin components, harvester engine and wheels transmission system. Even more, there are sealing rings, bearings, and screws that contribute to the large number of solids. Other fact is that some components are composed by several solids. In Figure 4.2 it is possible to see the harvester back and front frames. The inside of the back one contains the engine components, and the front frame is equipped with wheels transmission components.

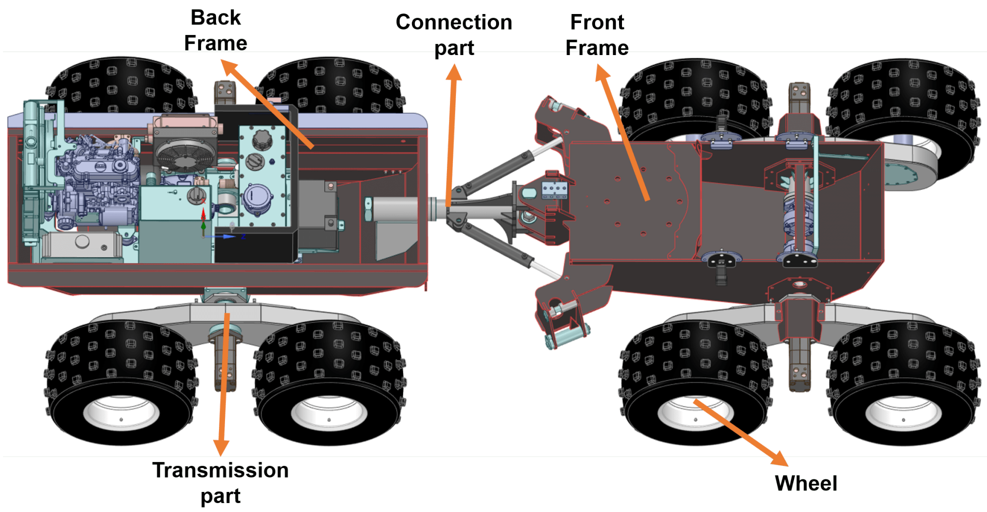


Figure 4.2: Inside back and front frame model.

In the same Figure 4.2 it is also possible to understand the mechanics design and used to steering turn the harvester. The Connection part has 2 hydraulic cylinders connected which are used to rotate the frames direction. This allows the harvester steering to be turned. In the machine, there are 8 wheels that are dynamically disposed in pairs (Figure

4.3). Each pair is connected by a Transmission part that is connected to the frame. This allows the wheels to be rotated individually and also to rotate the Transmission part. This Transmission part has a limitation provoked by the Connection and Frame parts.

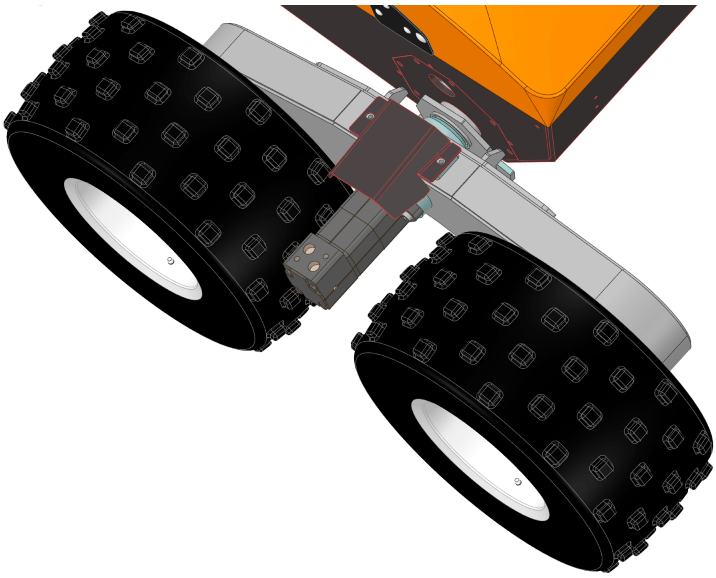


Figure 4.3: One pair of harvester wheels.

The cabin (Figure 4.4), where the operator is placed to operate the entire machine, has several bodies, such as seat, joysticks, pedal, buttons and a front panel control. Around the cabin there are transparent solids representing the windows.

Figure 4.4: Cabin model environment.

The harvester crane is possible to be seen in Figure 4.5. It is placed in a part that allows vertical rotation and uses 2 hydraulic cylinders to lift (Hydraulic cylinder 1) and bend (Hydraulic cylinder 2) the crane arm. On the top there is the Cutter tool attached. The mechanics of the crane allows to achieve a wide angle of action.

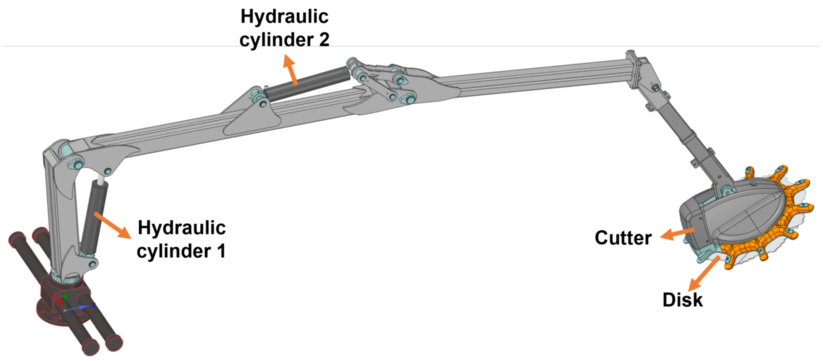
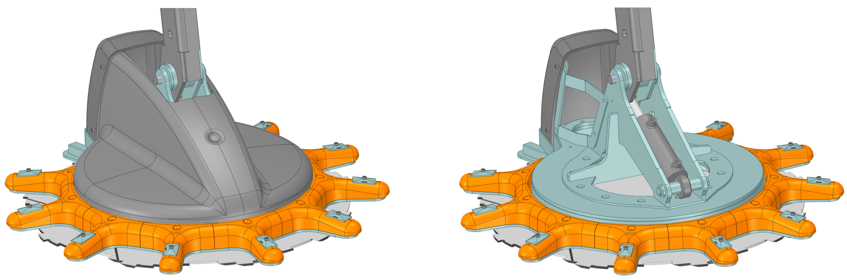


Figure 4.5: Crane model and tool attached.

The tool attached is the brushwood cutter model UW40, also commercialised by Usewood. It uses a rotational Disk that gets in contact with the vegetation to cut it. The tool is attached to the crane allowing rotation with a hydraulic cylinder, making the angle adjustment of the Cutter possible. This details can be observed in Figure 4.6.



(a) Outside view. (b) Inside view.

Figure 4.6: Harvester tool model.

The harvester is also provided with 2 Support legs to create dynamic stability when operating the crane. They are operable also with a hydraulic cylinder that controls the rotation position in relation to the Front frame. When activated in the real machine, the hydraulic presses the Support against the floor. This decreases the impact and force that the Crane dynamics creates in the machine. Figure 4.7 shows the left Support leg.

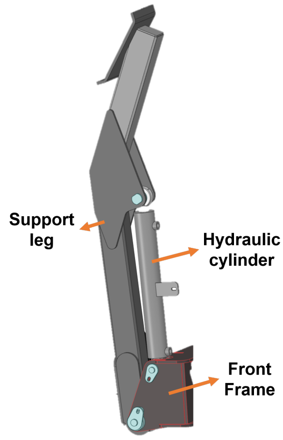


Figure 4.7: Left Support leg of harvester machine.

4.2 Considerations to simulator

The model in study is very detailed and specific with a large number of bodies. It is important to understand their roles and what is intended to be considered in the further simulator. The goal is to inspect the real size CAD model, interact and manipulate it as a real operator in a hypothetical scenario test. For that, it should be provided to the user driving conditions with frontwards, backwards and steering turn movements with dynamic control of Support legs, Crane and Cutter.

The real harvester uses hydraulic transmissions to bring power to hydraulic cylinders to operate the principal components. However, the entire process of hydraulic transmission is not possible to implement on the model due to the extreme complexity of it.

Other important fact to take in consideration is that the CAD model has components constituted by several bodies. It is not intended to simulate all bodies as individual ones, but to simulate bodies with realistic dynamics. This means that for the simulator, bodies should be merged to obtain geometries with the same features and properties as the real ones.

When driving the machine, contact of the tyre with the ground should occur. Also, Support legs and Disk cutter should interact with the remaining virtual objects. However, internal harvester components (e.g engine) are inside the machine components without any operator control and visual value to provide in this case. Despite this, they are important to provide mass properties that allow the remaining dynamics work properly.

Intentionally blank page.

Chapter 5

Simulator construction

5.1 Implementation process

Based on software features and capability discussed on Chapter 3, an implementation map was done to make it possible to bring the CAD model harvester to a VR simulator. The Figure 5.1 shows the overview of the process and the steps required in the different software.

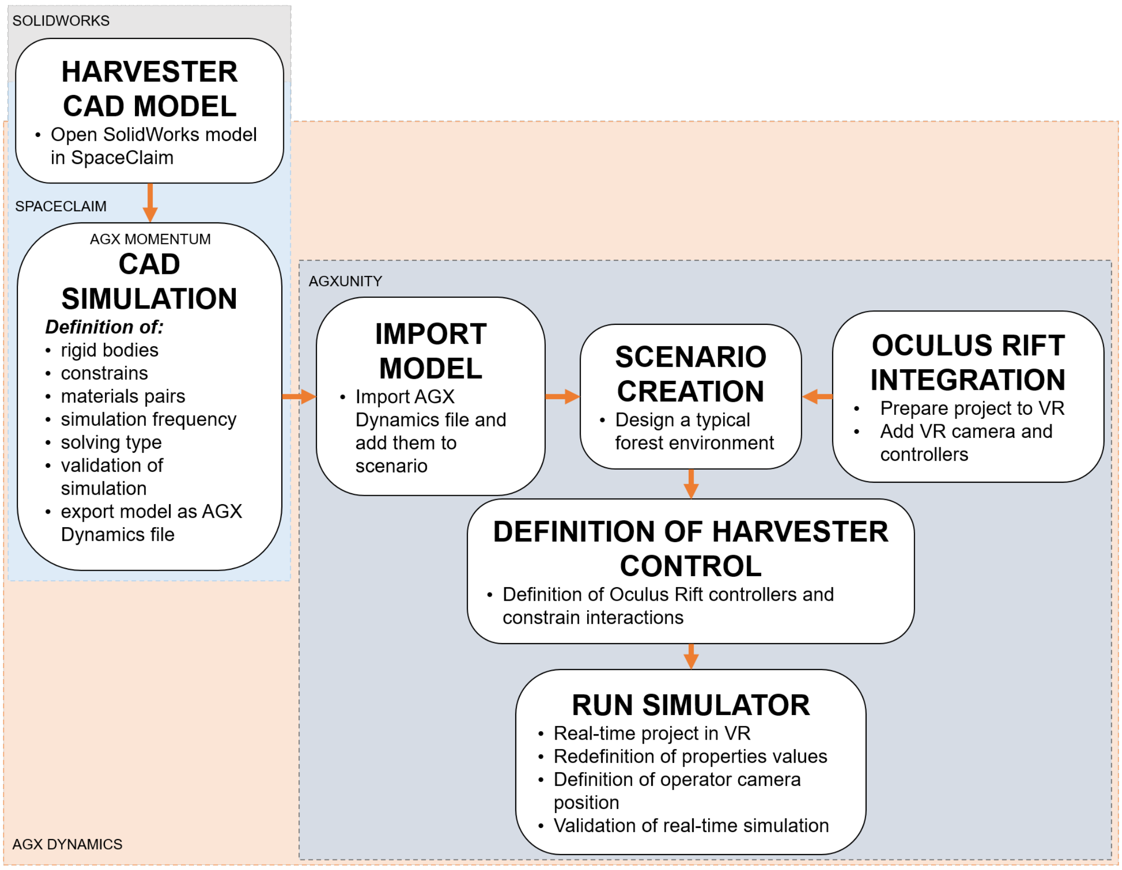
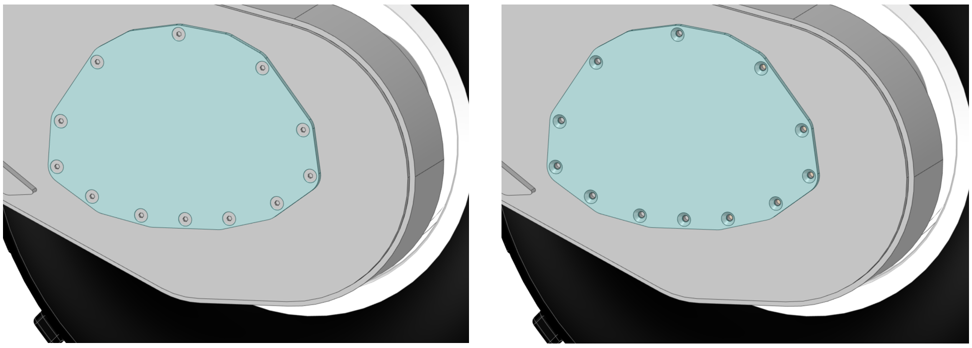


Figure 5.1: Overview of simulator implementation.

Import and preparation model to simulation

The SolidWorks model was imported to SpaceClaim. After an initial inspection and due to the large amount of solids (causing slow workflow), small solids of the model without direct dynamics interference and all interior components were not considered to the simulation. This decision was made to reduce the high number of components that do not need to be individually simulated. The focus was the main dynamics of the harvester and the operator use and not specifically the individual systems analyses. Also, reducing the number of parts can be fundamental in the simulation time and quality of the remaining components. This change reduced the model from 3298 to 527 bodies. The Figure 5.2 shows one example of twelve screws being removed without losing the assembly relations.



(a) Initial state. (b) After removing screws.

Figure 5.2: Screw removal on the model.

In this step also, the Crane angle of the CAD model was changed to provide a more stable initial position of the harvester. Despite this, the assembly relation were maintained. Colours were added to the model to better simulate the harvester. A rectangular solid was also added under the machine to simulate the ground and the dynamics of the harvester. The Figure 5.3 shows the model ready for simulation.

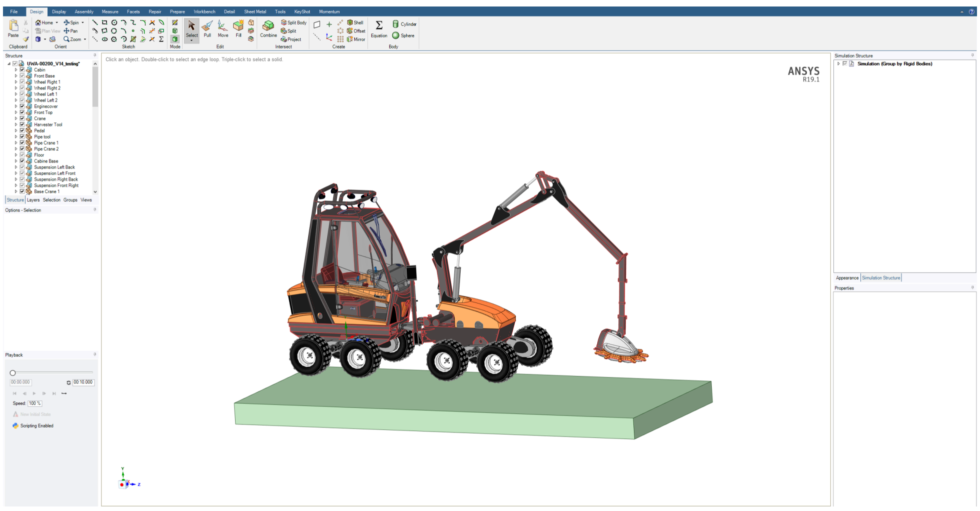


Figure 5.3: CAD model ready for simulation.

Start CAD simulation: definition of rigid bodies

AGX Momentum was activated to process the CAD simulation. The initial rigid body generated contains all the components of the model, where the tool *Split* was used to separate them. The Figure 5.4 shows all rigid bodies, where each different colour represents a different rigid body to the simulation.

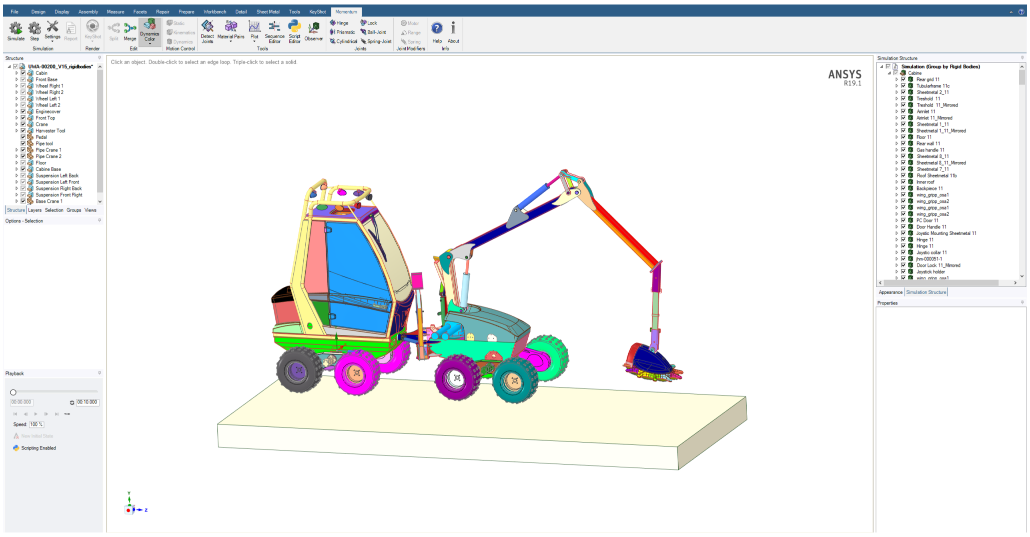


Figure 5.4: Rigid bodies separated by colours.

Then, components were added into rigid bodies with the tool *Merge*. On the back frame of the harvester, with the exception of the Wheels and Transmission part, the components were merged into one rigid body, acting as a Cabin. On the front frame, the Crane base components were merged. The Crane rigid bodies were also created in the same way as it is done in a real one. The cutter was merged in a way where the Disk is an individual body in order to further enabling its rotation. To connect both frames, the Connection part was merged. Then, each Tyre component was merged with the Rim components and the Transmission rigid body part was established. The rigid bodies of the Support legs were also created. All the pistons of the hydraulic cylinders were kept as individual rigid bodies of the external cylinder. The Figure 5.5 shows the simulation rigid bodies (by colours) defined on this step.

In the end of this step, the simulation model was composed by 8 Wheels, 6 hydraulic cylinders and individual pistons, 4 Transmission parts, Cabin, Connection part, 2 Support legs, Front base, the Crane with 5 different parts, the Cutter with separated Disk and the Ground that was set to be dynamically static. In total the CAD model was divided in 38 rigid bodies.

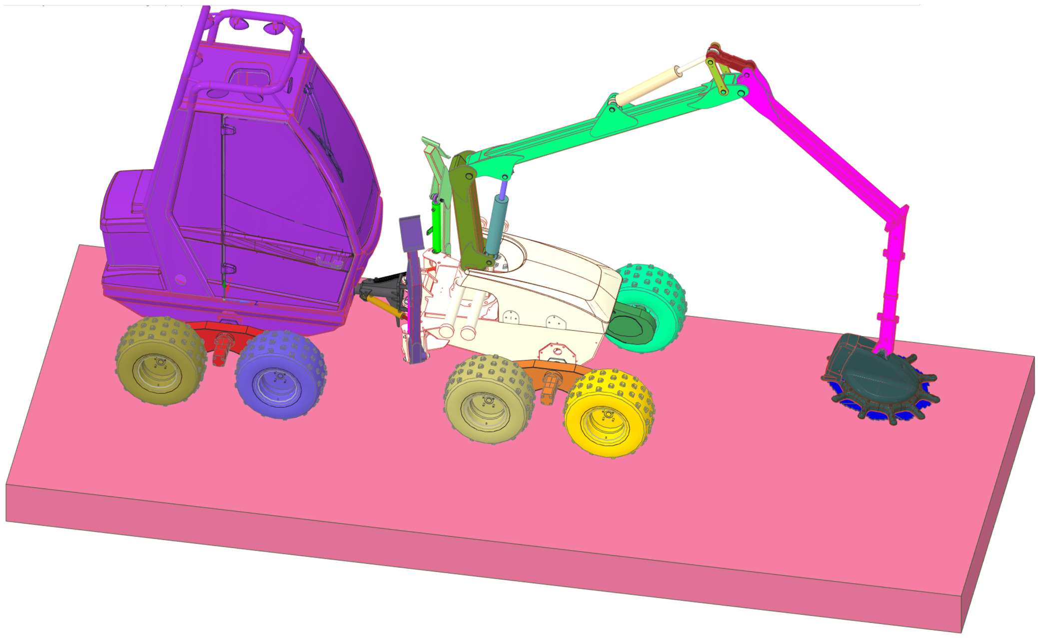


Figure 5.5: Rigid bodies defined for CAD simulation.

Joints definition

After defining the simulation bodies, it was necessary to create relations between them, adding joints. It were used prismatic joints between the piston and the cylinder in the hydraulic cylinders, creating one translational degree of freedom between them. Active motors makes a joint classified as an active joint. Since the harvester dynamics controlling is provided with by active ones, their individual motors were set to *target speed* zero and a high *force limit* to keep the components in static position.

To provide rotation of the Wheels, hinge joints were used between the Rim and the Transmission parts. Also, their motors were kept in the same conditions as the prismatic joints. The same was made to the relation front frame/Crane, Crane/Cutter and Cutter/Disk.

In the remaining rigid bodies relations, hinge joints were added to allow free rotation without any enabled motor. The Figure 5.6 shows all constraints added in the model. The straight arrows represent the prismatic joints and the curved arrows represents the hinge joints. The active joints are coloured in orange and the non active joints are coloured in grey.

To calculate the constraints, all the joints were set to have direct solver, as this method theoretically provides higher accuracy and less errors.

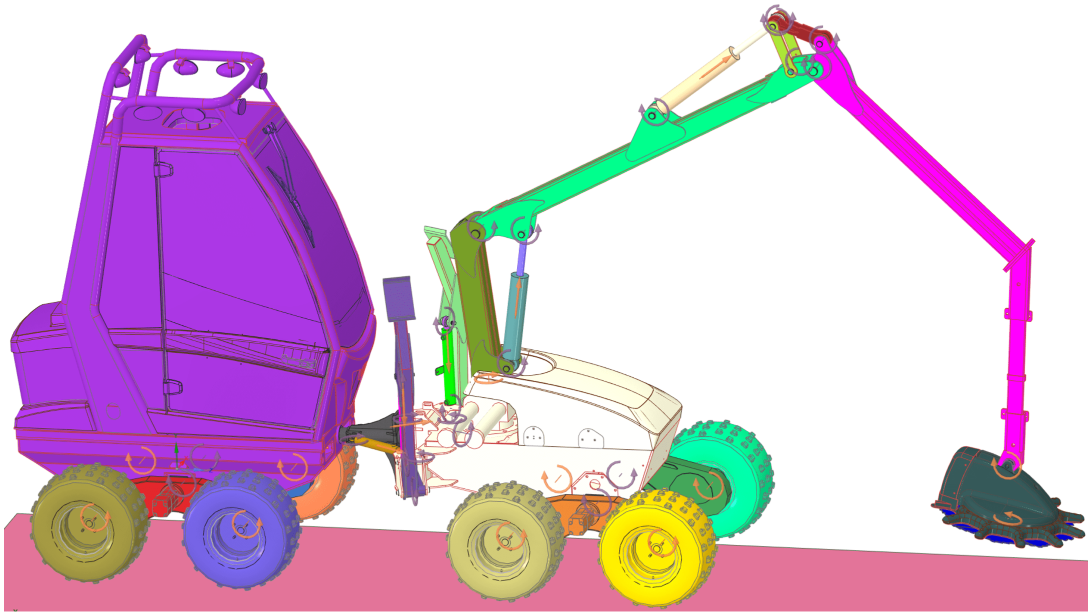


Figure 5.6: Joints defined on model.

Materials pair

By default all components have the same "unknown material". It was created the *Tire material*, *Floor material*, *Disk material* and *Support leg material* to be added to the components Tires, Floor, Disk and Support leg floor, respectively. The remaining components were kept with the default material. The software automatically creates pairs between all the different materials. The Figure 5.7 shows the different materials (defined by colours) added to the components.

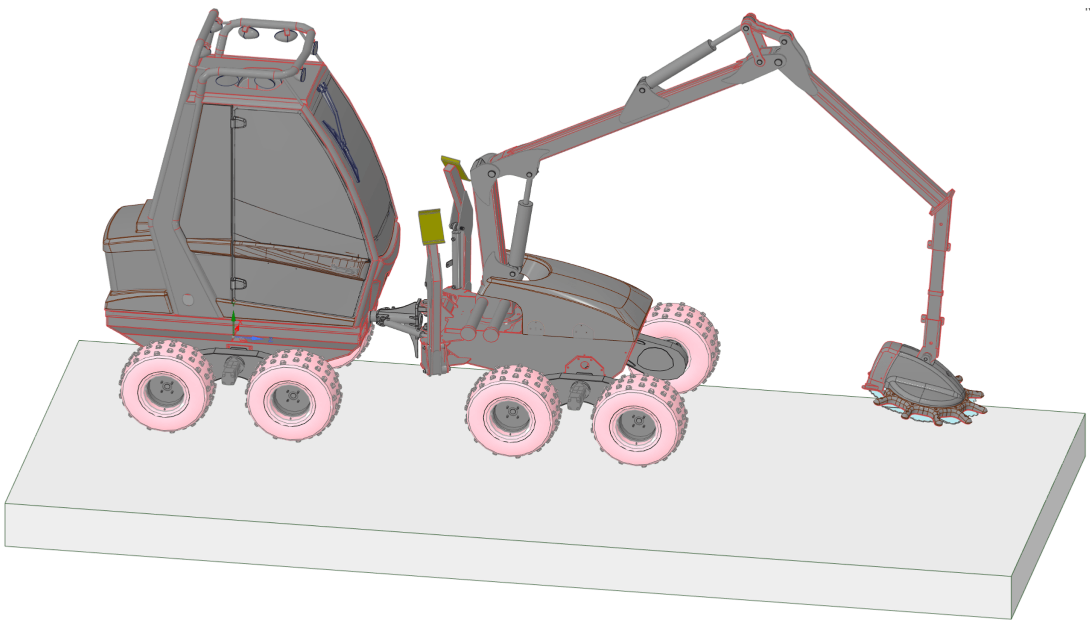


Figure 5.7: Different materials added in the model.

Simulation frequency
The simulation frequency was set based on Unity application, which is optimised to 3D visualisation as 50 Hz. Thereby, the same value was used in the CAD simulation.

Test simulation model and export
The CAD model was simulated to check and validate the rigid bodies, joints and material pairs previously defined. Different tests were made: simple stay on the floor, moving frontwards, harvester turning and Crane, Support legs, Cutter and Disk moving. These motions were provided by changing the speed value of the intended joints: to rotate the Wheels, the *target speed* of the motor in the hinge Wheel joints was set to non zero, making them rolling against the floor and providing frontwards movement to the harvester; the same happened in the Crane rotation the Crane, the Cutter and the Disk, where the hinge joint speed was changed; to move the crane arm, the support legs or steering turn the harvester, the speed of the prismatic joints of the hydraulic cylinders was changed. The other inactive motor constraints were acting freely according to the active ones.

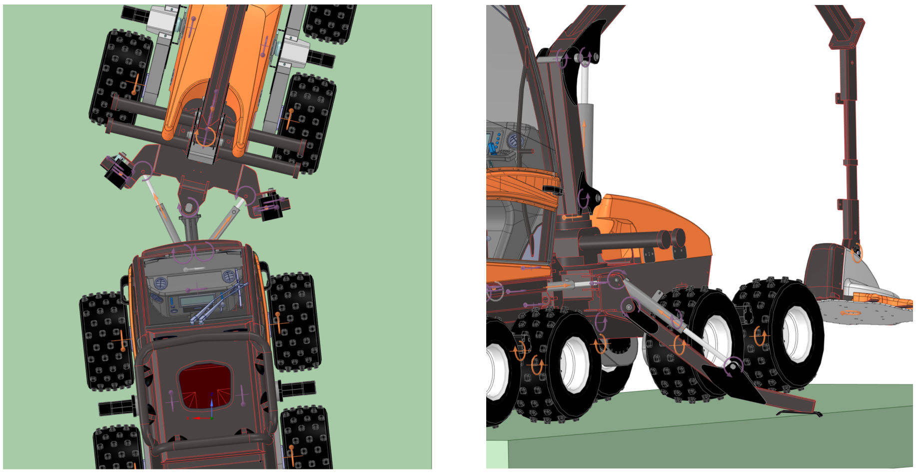
In Figure 5.8 it is possible to see the hydraulic cylinders test with initial (5.8a) and final position (5.8b). The main prismatic joint was set to a positive target speed, lifting the arm. The secondary prismatic joint was set with a negative value (contrary of the prismatic way), opening the arm.

(a) Initial Crane position.

(b) Final Crane simulation position.

Figure 5.8: Simulation of crane dynamics.

To test the steering turn, target speed for the Turn prismatic joints was set to symmetrical values, making the Back and the Front Frame rotating in opposite ways (Figure 5.9a). The prismatic joint of the Support Legs was tested too (Figure 5.9b).



(a) Harvester steering turn right. (b) Lower harvester Support leg right.

Figure 5.9: Simulation of harvester CAD dynamics.

The contact points generated in the simulation with all active joints in static mode, were observed only between the components Tire and Floor. It is possible to see this contact represented by the yellow dots in Figure 5.10.

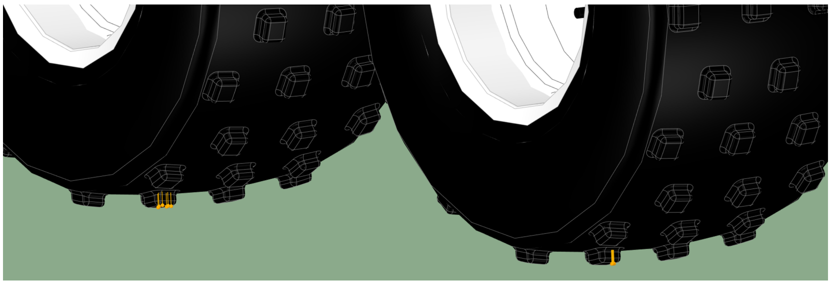


Figure 5.10: Contacts between Tires and Floor.

After validating all the joints and the rigid bodies, the simulation was exported as AGX Dynamics Binary file.

Scenario creation and model import

The virtual scenario was created in the Unity platform. Using the game object *Terrain*, it was designed the ground, creating different height in several points with the same shape as mountains and road paths. Then, it was painted with a grass texture and sand texture to provide a more natural aspect. After that, the AGX script *Height Field* was added to *Terrain* inspector. Moreover, 3D model mesh Trees (in different sizes and styles) from a Unity Asset Store package [Rakshi Games 2016] were added to the scene.

The harvester was imported in the format AGX Dynamics file as prefab, creating the model automatically in the project assets, with the same definitions and properties as in AGX Momentum. Also, a folder was generated containing all simulation information.

After the importation, the model was added to the scene and the Floor material (carried in the imported simulation files) was added into the script *Height Field* of the *Terrain* created. In Figure 5.11 it is possible to see the designed forest with the different textures and the 3D model imported in the scene.

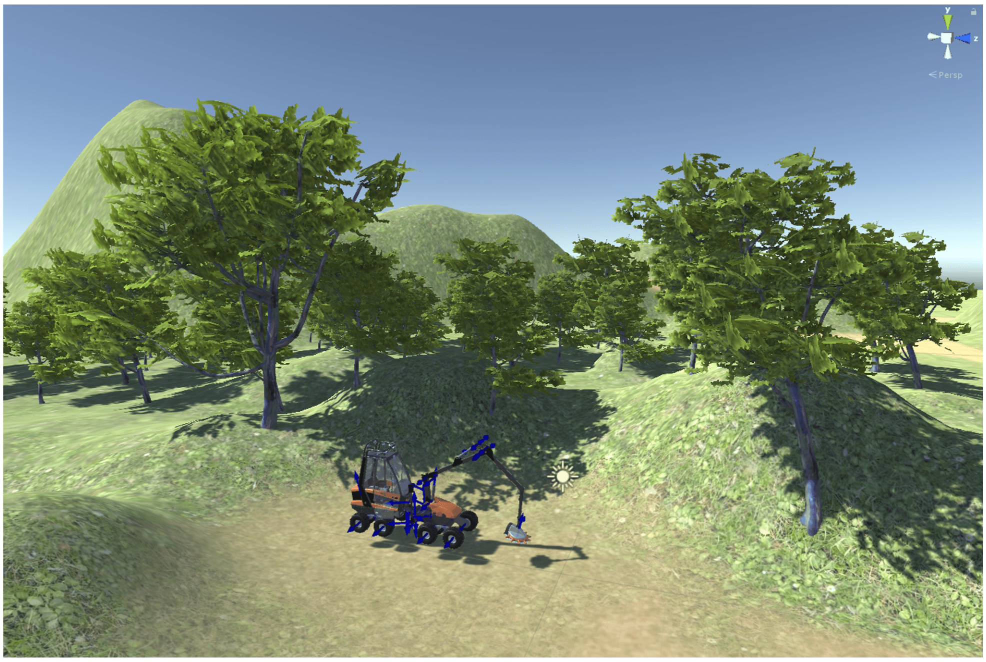


Figure 5.11: CAD Model and the virtual forest on Unity.

Oculus Rift integration

From the Unity Asset Store it was imported the Oculus Rift integration library [Oculus 2019]. The prefab *OVRCameraRig* (VR camera object) was added into the project scene. It was placed inside the Cabin in the same position as the real operator eyes would be. Then, the prefab was moved in the Hierarchy scene to inside the Cabin rigid body, becoming a child of it. This allowed the VR camera object to follow the same cabin position all the time.

Definition of model control

In the Unity Inspector, it was set manually the *Range limit* of the active hinge joints Crane rotation, Cutter and all prismatic joints. The values were based on the desired minimum and maximum range/angle of the joints. Then, it was created a program with Visual Studio to control the harvester dynamics. In this code, conditions have been created to access values through the Unity inspector tab, so that they can be manually changed. These values have influence in the target speed of the active joints. Also in this program, through class *OVRInput*, the Rift controllers were set as input keys. If the input from user (using Rift controllers) is detected, the values of the target speed are the values manually previously defined. If there is no input detected, the target speed values remain zero.

The full controller program developed was uploaded to a github repository (accessible from the URL link `https://bit.ly/2XszLYN`). In Program 5.1 there is a short code demonstration from the program, where we can see how the Crane rotation is done with the Rift Controllers.

Program 5.1: Crane rotation and Rift controller code

```
1 public class Controller : MonoBehaviour {
2 //Speed manually definition
3 public float SpeedRotationCrane = 5.0f;
4 // Crane joint definition
5 public Constraint jointCrane = null;
6 //Defining Joystick axis value
7 private float JoystickAxis;
8
9 void Start() {
10 //Initial state of joint motor (active and zero speed)
11 jointCrane.GetController<TargetSpeedController>().Enable = true
      ;
12 jointCrane.GetController<TargetSpeedController>().Speed = 0.0f;
13 }
14 void Update() {
15 //Get the Joystick axis value
16 JoystickAxis= OVRInput.Get(OVRInput.Axis2D.SecondaryThumbstick)
      [0];
17 //Update target speed value in the joint
18 RotateCrane.GetController<TargetSpeedController>().Speed =
      RightJoystickAxis* SpeedRotationCrane;
19 }}
```

In line 3, the hinge joint associated to the rotation crane is called. In the method *Start*, the *target speed* is enabled and changed to zero. Then, in the method *Update*, the value of the axis left/right from the joystick constantly called. This value is provided inside the interval of [-1,1], representing physical joystick movement. Negative values are provided when joystick is on the left side and positive values when it is on the right side. When no changes are detected, the value given is zero. Multiplying this with the initial speed, rotation and make it equal to the *target speed* allows to have a real-time changeable value.

The Table 5.1 shows which Rift controllers keys affects each joints and their direction speed in the program developed.

Table 5.1: Target speed joints affected by Rift controllers keys

	Rift Touch Controllers	Joint affected	Target Speed joint direction
Left Rift controller	Button X	Left Support leg	positive
	Button Y		negative
	Joystick up/down	Secondary hydraulic Crane	negative (up)/ positive (down)
	Joystick left/right	Cutter rotation	positive (left)/ negative (right)
	Joystick Button	Disk	On/off
	Hand trigger	Steering Left/Right prismatic joints	negative on Left prismatic joint; positive on Right prismatic joint
	Trigger	All Wheel joints	negative
Right Rift controller	Button A	left Support leg	positive
	Button B		negative
	Joystick up/down	Primary hydraulic Crane	positive (up)/ negative (down)
	Joystick left/right	Rotation Crane	negative (left)/ positive (right)
	Hand trigger	Steering Left/Right prismatic joints	positive on Left prismatic joint; negative on Right prismatic joint
	Trigger	All Wheel joints	positive

Simulator test

The simulator was built with a desktop computer with an Intel processor i7 2.67Hz, 6 GB of memory RAM and a dedicated NVIDIA GTX 970 graphic card.

The project was tested to better understand the simulator performance, user experience and AGX Dynamics statistics. The initial tests were done without the VR device, having the visualisation been done on computer monitor. This allowed to focus on the inspection and error detecting in the harvester dynamics and the simulation in general.

In this initial tests, the AGX Dynamics statistics were showing a total calculation time higher of 20 milliseconds. This led to a slow simulation, consequently, a poor user experience. It was observed that, when driving the machine, the number of contact points generated between each Wheel/Terrain (Unity) where higher when compared with Wheel/Solid Floor (AGX Momentum). Because of that, the solver type of these contact points was changed from Direct mode to Iterative mode and then to Split mode. The objective was to discover a faster solution enhanced the simulator, once they theoretically require less computer performance. However, no significant improvements were observed on the time calculation. Moreover, high interceptions were noticed, which affected physical realistic behaviour.

In SpaceClaim, to reduce the contact points, it was added to each Wheel rigid body a hidden flat cylindrical solid with the same center and approximated outside diameter as the Tyre treads solid. Then, the collision from the previews were disabled. Thus, the designed cylindrical solid acted as Tyre surface. The Figure 5.12 shows one pair of wheels in contact with the floor solid, both with the cylinder solid designed associated (being hidden in the right wheel).

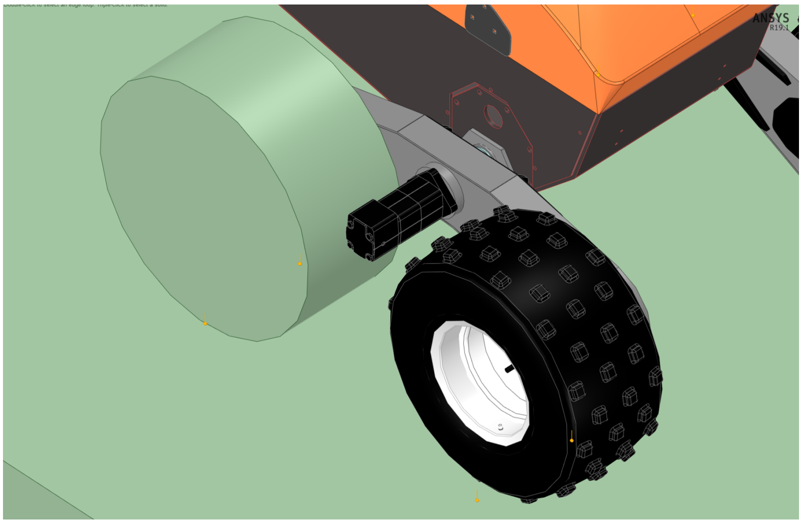


Figure 5.12: Wheels contact reduction using hidden cylinders solids.

Since the Tread tyre is visible and it rotates with the same speed as cylindrical solid, it gives an appearance as of a real one touching the ground and providing the harvester with front and reverse movements. This change provides less contact points with the Terrain, decreasing the amount of collisions calculations necessary to solve.

Also, all the rigid bodies and components in the model without expected contacts in the forest scene (no contribution for direct or manually provoked collisions) were changed to contact disabled. With this external step, their geometric overlap tests (broad phase and near phase) were not performed. In the end, the collisions were only enabled to Tyre, Support legs and Disk Cutter.

After the harvester and the simulation validation, the project was tested with the VR headset on the operator view.

Intentionally blank page.

Part III

Results, Discussion and Conclusions

Chapter 6

Results and discussion

6.1 Simulator

The simulator was built to be used in front of a desktop computer. The virtual camera was placed inside the cabin which always follows the same position of the Cabin. Also, it was placed to coincide with the user seat in a physical desk chair inside the play area. The physical element, however, is not a virtual limitation since the user has the ability to stand up and move. The virtual space visualisation matches with the headset freedom movements without any virtual boundary limitation from the virtual rigid bodies geometries. The Figure 6.1 shows the setup of the desktop computer and monitor, the sensors oriented to the play area and the user seated in a chair with the HMD and Rift controllers, performing interaction with the simulator. On the screen it is possible to have access to what the user is actually observing in the headset.

Figure 6.1: Setup and play area of simulator.

The headset provides the virtual site from the perspective of the user head orientation in real world. The virtual camera starts inside the Cabin as shown in Figure 6.2a. The windows components are transparent, allowing to view the outside machine bodies and the scene environment designed. The manipulation in the harvester and the effects that provided by the VE are observable to the user. The shape, colours and constraint relations are possible to be seen with all the details and sizes as in the CAD model. Running the simulator in Unity project allows to quickly change the camera position, for example to outside of the cabin (6.2b), maintaining all the simulation properties.

(a) Inside cabin view.

(b) Outside machine view.

Figure 6.2: Headset user view.

The Rift controllers give to the user the capability to interact with the harvester. The user is able to drive the machine and manipulate the Crane and Support legs with them. The Table 6.1 describes the function of each button in the simulator. The ergonomic design, specific for each hand, establishes an easy association between them and the cabin's virtual joysticks, that also operate with individually hand manipulation.

The controllers interact directly with the *target speed* of the active joints inside of *range limits* previously defined. When no buttons are in use, the joints remain standing and no dynamic movements are detected. This actuation mode helps to experience a similar behaviour to the real hydraulic transmission, used in the real harvester. The speed of the bodies movements defined provides a realistic behaviour to the user, with high sense control of the machine.

Table 6.1: Rift controllers interaction with the harvester.

	Rift Touch Controllers	**Machine effect**
Left Rift controller	Button X	Lift left Support leg
	Button Y	Lower left Support leg
	Joystick up/down	Open/close Crane arm
	Joystick left/right	Rotate Cutter
	Joystick button	On/off Disk rotation
	Hand trigger	Steering turn left
	Trigger	Move backwards
Right Rift controller	Button A	Lift right Support leg
	Button B	Lower right Support Leg
	Joystick up/down	Lift Crane
	Joystick left/right	Rotate left/right Crane
	Hand trigger	Steering turn right
	Trigger	Move frontwards

In the Unity project, the script developed to operate the machine allows the active joints speed (m/s) to be manually modified in the Inspector tab, as Figure 6.3 shows. Here, it is also where the association between the model joints and the public Constraints defined in the script.

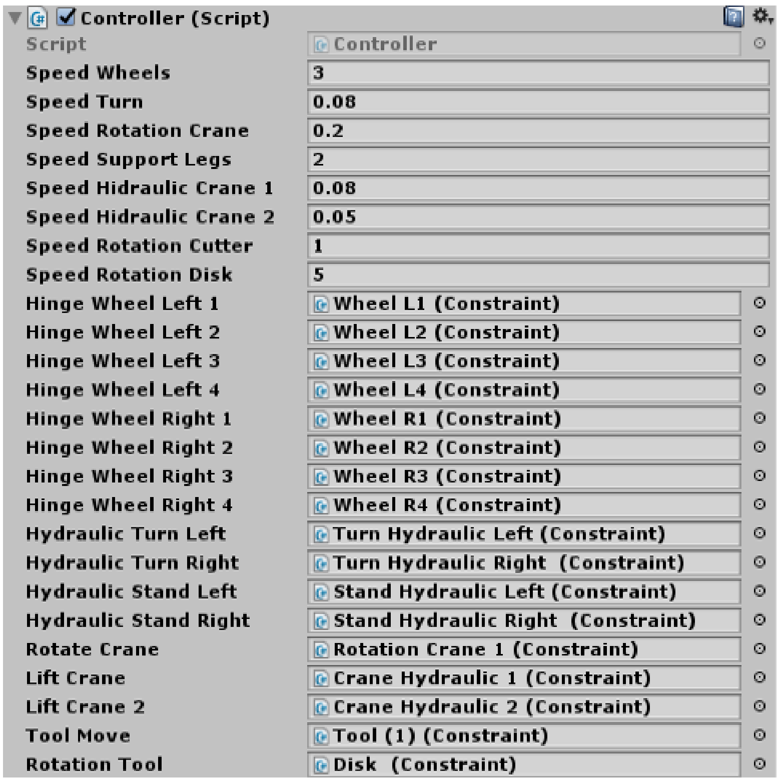


Figure 6.3: Unity script public values and harvester joints association.

The user can drive the machine in the virtual designed scene which has different terrain heights and is composed by a zone of Trees. The directional Light object simulates the daylight, affecting the objects in the scene by creating shadows in real-time, contributing for the environment simulation quality. The different sizes of the Trees added to the scene also create a more realistic scene simulation.

Operating the harvester results in machine interactions with the Floor. The contact that can be provoked by the Cutter and the Support legs, directly affects the free joints in the machine. The user actions give realistic movements and precise feedback to the Cabin, making it possible to understand the effects of those actions.

Collisions were only enabled to Tyre, Support legs and Disk Cutter. The remaining components do not provide any contact interaction with the virtual scene. One example of this is the Crane arm that overlaps all virtual scene bodies. These overlaps happen because of a decision that it was made previously, which consisted of disabling contact collision in specific components in order to enhance simulator speed.

6.2 CAD model in Virtual Reality

In the simulator, the 3D machine model used was provided by a CAD file, imported to Unity with the same information previously defined in AGX Momentum. In the virtual scene, all the solids were visible and maintained with the same dimensions and geometry, assembly relations and visual features. Figure 6.4a shows the model before the exporting process from SpaceClaim and after have been imported to the virtual scene in Figure 6.4b.

The implementation process consisted in transferring the harvester model from the CAD software to a VE. Initially, the model was composed by a large number of solids that were already assembled as desired (6.4a). Using the AGX Momentum capabilities inside SpaceClaim, it was created the CAD simulation environment with rigid bodies, joints, materials and solving methods. The result is a detailed mesh of each solid in the model, keeping individual geometry and visual information with additional physical properties. By using tools and material properties, it generates relations and behaviours between them and the gravity. All this simulation information developed and defined here is the information incorporated in an AGX Dynamics Binary File. When imported to Unity, the model is organised and provided with the exactly same information as previously defined in AGX Momentum. Figure 6.4b shows the model in the AGXUnity scene, after the conversion.

(a) Model in SpaceClaim.

(b) Model in Unity scene.

Figure 6.4: Comparison of the model in different environments.

The necessity of export/import the model is related to the different capabilities that the different software provides. In the building process, AGX Momentum provided the environment to create the simulation information and test directly the CAD model. To create an environment scene with the model and to test it in real-time and in VR was possible thanks to the use of the Unity and AGXUnity libraries.

The testing scene was designed using the Unity tools. Adding the object scripts from the AGX package to the scene objects turned automatically possible to have all the contacts with the model solved by the physics engine. As an example of this, the material properties of the Ground solid in AGX Momentum defined was then added in the Terrain object.

The Oculus Rift camera in the scene creates the user perspective in human scale size, independently of the remaining objects and VE. The user is able to be immersed in the scene not only visually, but also with the sense of being physically there. When the camera is positioned inside or close to the machine, and because all the CAD information was imported correctly, the harvester is observed having the real size which was modelled for. The virtual camera added in the scene is on the left side of Figure 6.5. In the Game tab (right side of Figure 6.5) is the user perspective. These two perspectives compare the relation that the VR device creates in the display, where it relates to the human eye and the head position with the harvester.

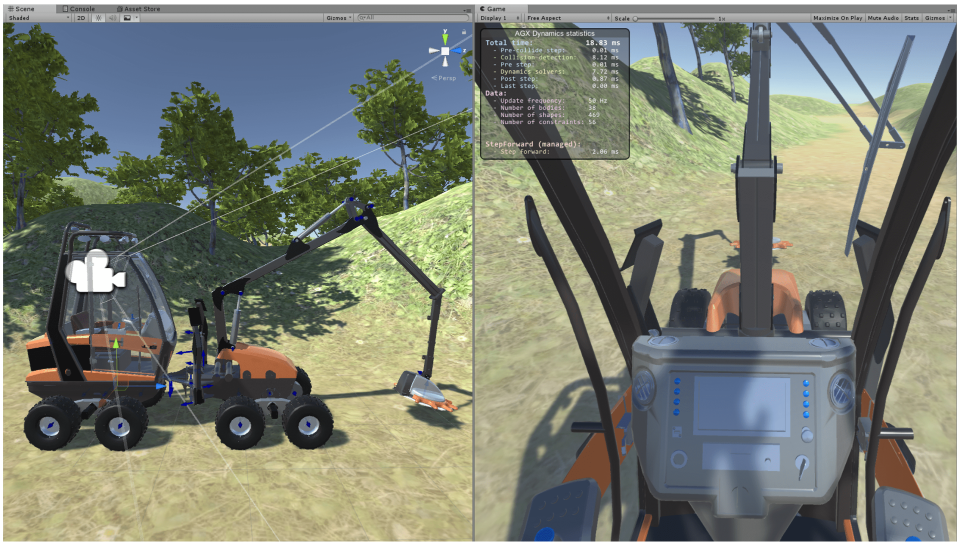


Figure 6.5: Camera position in Scene and user perspective.

Interacting with the model was possible due to the joints tools by manipulating their motor and range limits. Although they allow to simulate the CAD model, they required to be manually added. Since the very initial model was very large, the workflow in AGX Momentum was slow. Knowing also the restriction that this could cause in the further real-time simulation, it was fundamental to disregard the components that would not interfere dynamically in the simulation, for example the screws. By not considering them in the model, it allowed to reduce the physics simulation information and improved the workflow quality of the model.

The assembly position was changed directly in the CAD model. This was made only to start the simulation in a more neutral way. Hydraulic pistons were moved to the central position and the bodies were rotated also with the hydraulic cylinders. This helped the limits setting of the active joints and provided an initial dynamic stabilisation of the harvester.

6.3 Simulation factors

The simulator construction was developed by taking in consideration the computation needs to provide the best VR experience possible. There were decisions that notoriously decreased the solving time in the physics engine. Namely not considering unnecessary solids in the CAD model dynamics simulation, reducing the Tyres and the Floor and disabling contact detection of several components. The critical part is still how to move the machine. The complexity of the Height Field shape used as Floor is efficient in bringing realistic behaviour to the Tyres but it generates several inconstant contact points. Plus, the direct method is the solver that better results provides but also highly contributes for a higher time consumption.

Manipulating and controlling the Crane, Cutter, Disk and Support legs do not cause notorious changes on dynamics calculations, performing smoothly movements. The same behaviour is observed when these components contact the Floor, not causing any constraint in the simulation. This can be related to their small contact area when collisions occur and the lower weight when compared with the Tyres, where all the harvester load is placed.

In addiction of the physics engine running, the simulator also uses at the same time the headset to transmit the virtual image, the Rift sensors to tracking their position, the Unity to display the virtual camera, to run the VE and for user interaction. These software and hardware use the maximum computer capabilities and this restricts its performance. When the physics solving takes a lot of time, besides slowing the simulation, it also decreases the image quality in the HMD device. Other evident interference case is the loss of accuracy in tracking the position of headset. Also, using the maximum computer performance makes that, despite the user being able to rotate the head and inspect the surrounding VE, the HMD tracking position quality from the play area is slow and with low accuracy, disabling the user to quickly approximate to objects or experience the natural head moving in VE. Finally, it made not possible to visualise and track the virtual controllers in the VE, which was a limitation for the creation of other ways for the user to interact with the objects in the scene.

6.4 Discussion of the research questions

6.4.1 RQ1: The way to import a CAD model to Virtual Reality environment using Unity game engine and AGX Dynamics

The implementation process of this work started with the CAD software SpaceClaim alongside with the CAD model. The intended dynamics simulation were then added by using AGX Momentum. The model was exported as a AGX Dynamics Binary File containing the entire simulation information defined and imported to the AGXUnity. The use of the Oculus Rift integrated in Unity provided the visualisation in VR.

Considering the work developed, a general path from CAD to VR can be established. Surely, the modelling phase is the first step in a CAD software. This may be executed with traditional tools, although it is highly recommend to use SpaceClaim, since it is here that the path to VR starts. However, the models can be imported from other software. Using the physics engine AGX Dynamics and the GUI AGX Momentum, it is possible to create rigid bodies by merging components and solids, adding joints to constraint relations and to generate material pairs. A dynamics CAD simulation is available to perform at this stage. The entire CAD and simulation information is then exported as an AGX Dynamics Binary File. The use of the AGX Dynamics GUI in Unity, named AGXUnity, generates a transferable file with the exactly same simulation information. The game engine provides the project development to VR, implementing free packages from Assets Store to complement physical VR device with a VE scene, also freely designed as intended. The HMD displays the VE with the same position and orientation of the user head.

The Figure 6.6 summarises the general procedure with the different software and hardware to bring a CAD model to VR. The physics engine is responsible for keeping all the data and to transfer it from different platforms. The VR devices provides the capability for a person to participate in the VE and to inspect the real size model.

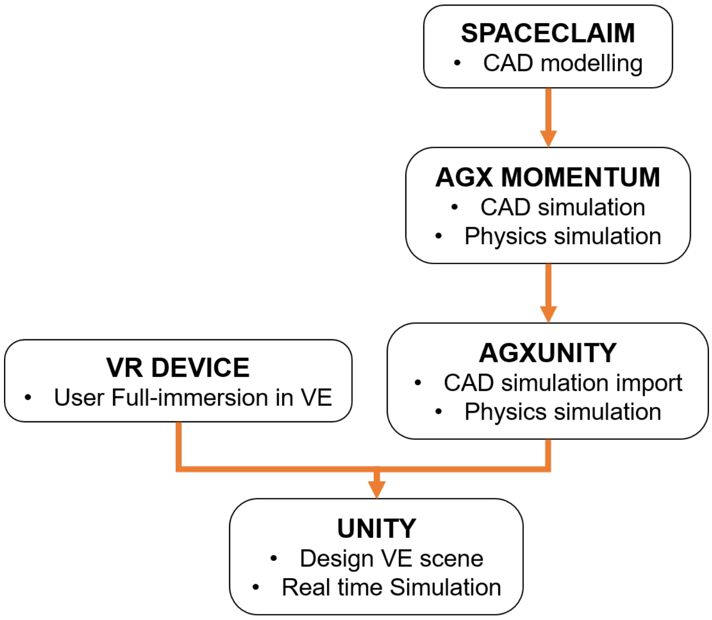


Figure 6.6: From CAD model to VR.

6.4.2 RQ2: Facts that affects a large model on real-time simulation

After developing the simulator based on a large CAD model, several factors interfered in the construction process decisions and in final simulator. First of all, it is important that The CAD model is assembled having in mind the further rigid bodies and their relations between them. Then, the number of solids directly interferes with the CAD simulation workflow and in the further real-time simulation. Relocating the necessary components and disregarding solids that dynamically do not have impact in the simulation, are techniques that can be applied.

In the RQ1 it is shown the process which defines the simulation of the CAD model and brings it to Unity, through AGXUnity. Using scripting, it is possible to manipulate the joints and interact with the model with physical controllers or even with a computer keyboard. By reducing considerably the number of contacts and by disabling the Contact collision detection in the majority of the solids, the physics solving time decreases considerably. In a real-time simulation, it is important to focus on what is exactly wanted, in order to do not slow the simulator capability.

Although Unity provides a game physics engine, for this work it was used the simulation calculations executed by the physics engine AGX Dynamics. This makes the process much simpler, since the initial CAD simulation is done in another platform, but still using the same background. Also, it uses motion equations to solve all dynamics, providing robust results.

The different solving methods that AGX Dynamics provides is a factor that affects the simulation. To material pairs, the split and iterative methods revealed to be a slightly less computationally heavy, comparing with the direct method. However, this last one was the most capable to simulate realistic behaviours.

Other important aspect to take into consideration is the computer itself used to simulate. The results in this work were visibly restricted by the facts mentioned above.

Although, they can provide different impact or even not being an embarrassment in a presence of a more powerful and advance computer. The processing and the graphics required by the physics engine to provide the VE scene are fundamental not only to obtain a good VR experience, but also to have more freedom on the simulation constraints and to achieve even better and faster results.

Intentionally blank page.

Chapter 7

Conclusions

Traditional CAD tools have been used in industry to create, design, model and improve products, mechanisms or other systems. These can be composed from very simple and small components to very high detailed, complex and large scale ones. Their development requires great work efficiency and rapid design validation.

Previews studies showed that VR has a great potential to help engineers and designers in the development phase. However, simulating all CAD dynamics and/or bringing them to VR was identified as a hard, very high time-consumption and non-efficient process. The solution developed and tested in this work explains an approach to a direct creation of a model dynamics simulation inside the CAD software and the process to merge a model in VR environment. Although it was used different software and hardware to make it possible, the process revealed to be efficient, intuitive and relatively fast.

The results of the implementation provided a real scale prototype of an harvester CAD model being tested in real-time in a VE scenario. A VR device provides the context for a user to have a full-immersion experience inside the VE, by visualising, interacting and manipulating the machine. The simulator provides a good sense of realism.

A few obstacles to the simulator were identified. Having a large and complex model can require that the developers focus on the CAD essential features for the simulation. For that, some examples of what can be done are as follows: not consider solids without dynamic relevance in the model, decrease the number of contact points and disable the collision detection of components without expected contacts. Also the VE turned the computationally very heavy due to running a physics engine and a VR device at the same time.

In the end, a general path has been achieved from CAD to VR. By using the path, virtual prototypes can be visualised in their real size and can interact with a person. This benefits the quality and speed of design and can improve the benefits of co-creation in the earliest development phase. The outcome allows to have faster and more efficient products. Along other stages of PLM, CAD in VR can also be used directly in demonstrations, in operator training and for marketing purpose.

7.1 Future work suggestions

For future directions of this work, it is proposed to explore other ways to faster implementation of interaction with the model in a VE. Additional virtual elements, as environment sound and vibration feedback, can be also combined in further development of machine simulators in VR.

Other research paths can be the reduction of different platforms used in this work by automatising even more the process of CAD to VR.

References

[Algoryx Simulation AB 2018a] Algoryx Simulation AB. "AGX Dynamics Documentation: Structural description of a Simulation", 2018.

[Algoryx Simulation AB 2018b] Algoryx Simulation AB. AGX Momentum Documentation, 2018.

[Baus and Bauchard 2014] Oliver Baus and Stéphane Bauchard. Moving from Virtual Reality Exposure-Based Therapy to Augmented Reality Exposure-Based Therapy: A Review. *Frontiers in Human Neuroscience*, Vol. 8, 2014.

[Blümel *et al.* 2004] E Blümel, S Straßburger, R Sturek and I Kimura. Pragmatic Approach to Apply Virtual Reality Technology in Accelerating a Product Life Cycle. pp. 199–207, 2004.

[Campos 2018] Guy Campos. "VR CAVE installed at university of technology". `https://bit.ly/2x36c1d`, 2018. visited on 2019-06-19.

[Ellman *et al.* 2007] A. Ellman, J. Laitinen and T. Tiainen. Combination of Virtual and Physical Objects in User-Centered Design of a Mobile Work Machine Cabin. *Proceedings of ASME International Mechanical Engineering Congress and Exposition*, pp. 143–148, 2007.

[ESA 2017] ESA. "European Space Agency: Space in Images". `https://bit.ly/2rEYZBo`, 2017. visited on 2019-06-19.

[Fiorentino *et al.* 2009] M. Fiorentino, G. Monno and A. E. Uva. Tangible digital master for product lifecycle management in augmented reality. *International Journal on Interactive Design and Manufacturing*, pp. 121–129, 2009.

[Guo *et al.* 2018] Ziyue Guo, Dong Zhou, Jiayu Chen, Jie Geng, Chuan Lv and Shengkui Zeng. Using virtual reality to support the product's maintainability design: Immersive maintainability verification and evaluation system. *Computers in Industry*, 101:41–50, 2018.

[Jeannie 2006] Jeannie. "Philips Design joins Second Life". `https://bit.ly/2IXNNsf`, 2006. visited on 2019-05-30.

[Kaapu *et al.* 2013] Taina Kaapu, Tarja Tiainen and Asko Ellman. User Interpretations of Virtual Prototypes: Physical Place Matters. *Scandinavian Journal of Information Systems*, pp. 83–104, 2013.

[Kalawsky 1993] Roy Kalawsky. The Science of Virtual Reality and Virtual Environments. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1993.

[Körner 2015] Erland Körner. "Working with Physically-Based Shading: a Practical Approach". https://bit.ly/2FkpfZs, 2015. visited on 2019-04-30.

[Kuusisto *et al.* 2011] Jukka Kuusisto, Asko Ellman, Taina Kaapu and Tarja Tiainen. Effect of the Immersion Level of a Virtual Loader Simulator on the Sense of Presence. *Proceedings of the ASME World Conference on Innovative Virtual Reality*, pp. 155–163, 2011.

[Kuusisto *et al.* 2012] J. Kuusisto, T. Kaapu, A. Ellman and T. Tiainen. Developing VIP2M: A virtual environment for prototyping mobile work machines. *Proceedings of International Design Conference- Design*, pp. 657–666, 2012.

[Lacoursiere 2007] Claude Lacoursiere. Ghosts and Machines: Regularized Variational Methods for Interactive Simulations of Multibodies with Dry Frictional Contacts. 2007.

[Oculus 2019] Oculus. Oculus Integration version 1.38, Unity Asset Store. Realised on Jul. 22, 2017. https://bit.ly/2JwHDPC, 2019. visited on 2019-02-03.

[Rakshi Games 2016] Rakshi Games. Realistic Tree 9 version 1, Unity Asset Store. Realised on Feb. 24, 2016. https://bit.ly/2JOGMqB, 2016. visited 2019-02-15.

[Servin and Brandl 2018] Martin Servin and Michael Brandl. Physics-based virtual environments for autonomous earthmoving and mining machinery. *in: CVT2018: 5th International Commercial Vehicle Technology Symposium Kaiserslautern*, 2018.

[Sherman and Craig 2003] William R. Sherman and Alan B. Craig. Understanding Virtual Reality: Interface, Application, and Design. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[SuperData 2017] SuperData. "Where are the real opportunities in the immersive tech market?". https://bit.ly/2WT7ztG, 2017. visited on 2019-05-19.

[Tiainen *et al.* 2014] Tarja Tiainen, Asko Ellman and Taina Kaapu. Virtual prototypes reveal more development ideas: comparison between customers' evaluation of virtual and physical prototypes. *Virtual and Physical Prototyping*, pp. 169–180, 2014.

[Tideman *et al.* 2008] Martijn Tideman, Mascha C. van der Voort and Fred J. A. M. van Houten. A new product design method based on virtual reality, gaming and scenarios. *International Journal on Interactive Design and Manufacturing*, pp. 195–205, 2008.

[Tsyktor 2019] Vasyl Tsyktor. "What Is Semi-Immersive Virtual Reality?". https://cyberpulse.info/what-is-semi-immersive-virtual-reality/, 2019. visited on 2019-04-12.

[Usewood Forest Tec Oy 2019] Usewood Forest Tec Oy. "Forest Master Turbo". https://bit.ly/2X3n2wa, 2019. visited on 2019-03-24.

[Wai 2014] David Man Wai. "Virtual-reality-based rehabilitation system helps people with cognitive deficits". https://bit.ly/2Kre7Ow, 2014. visited on 2019-04-12.

[Zheng *et al.* 2017] Yili Zheng, Bowen Cheng, Qingqing Huang and Jinhao Liu. Research on Virtual Driving System of a Forestry Logging Harvester. *Wireless Personal Communications*, pp. 667–682, 2017.