UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

INTERACTION IN VIRTUAL ENVIRONMENTS WITH THE UPPER BODY

Mariana Barros Vital

MESTRADO EM ENGENHARIA INFORMÁTICA
Especialização em Engenharia de Software

2012

UNIVERSIDADE DE LISBOA
Faculdade de Ciências
Departamento de Informática

INTERACTION IN VIRTUAL ENVIRONMENTS WITH THE UPPER BODY

Mariana Barros Vital

PROJETO

Trabalho orientado pela Profª. Dr.ª Ana Paula Boler Cláudio e co-orientado pelo Prof. Dr. Francisco dos Santos Rebelo

MESTRADO EM ENGENHARIA INFORMÁTICA
Especialização em Engenharia de Software

2012

Acknowledgments

Este projeto marca o final de uma etapa gratificante. Pelo fantástico percurso académico que tive, quero tentar registar os meus sinceros agradecimentos às pessoas que o marcaram. Quero tentar, porque o que senti ao longo destes anos com essas pessoas é impossível descrever.

À Grande Família que tenho, que esteve sempre presente, apoio-me nos piores momentos e festejou os melhores. Mãe, Pai, Irmãs e Tia um grande obrigado. Foram vocês que me educaram e me fizerem ser quem sou. Meus Essenciais.

Ao fantástico namorado pela força que sempre me deu. A força da tua palavra foi sempre uma Luz no Escuro. Obrigado Nelson por estares sempre ao meu lado, és Essencial.

Aos meus amigos que me acompanham há anos e sempre demonstraram o seu interesse e preocupação pelo meu caminho, Filipa, BIPM, Su, entre muitos outros

Aos meus amigos Fculianos, obrigado pelas noites nos laboratórios do DI, cafés e "brainstormings" na ponte do C1, pausas no jardim, festas no mocho e principalmente à amizade... Tijolinho, Xico, André, Portimão, Reis, Tap, Rute, Milena, entre outros.

Aos meus padrinhos e afilhados que longe ou perto marcaram esta fase com momentos imensamente felizes.

Ao Pombal pela preciosa ajuda na execução deste relatório, um ‘*axu’* só para ti.

À minha querida orientadora, Professora Doutora Ana Paula Boler Cláudio, que esteve a meu lado profissional e pessoalmente em todo o meu percurso académico. Obrigada, é uma Grande Professora.

Ao meu orientador Prof. Dr. Francisco dos Santos Rebelo um obrigado pela excelente oportunidade de realizar este PEI consigo e obrigado por todo o apoio e conhecimento que me transmitiu.

Ao Luís Teixeira, um grandessíssimo obrigado pela prontidão e disponibilidade durante a realização deste projeto. A ajuda que me deste e os conhecimentos que me fizeste adquirir nas mais diversas áreas foram bastante importantes.

A toda a equipa do ErgoLab pelos conhecimentos que me transmitiu e por me terem recebido tão abertamente.

E por fim à "coisa mais linda da tia", que apenas com o seu sorriso majestoso, me conseguia dar forças em todas as alturas em que pensei desistir... À minha querida e Essencial Sobrinha Teresa!

É com saudades que deixo esta vida... Obrigado a vocês que tornaram MARAVILHOSO este percurso!

Aos meus Essenciais,

Resumo

Pode-se considerar que a Realidade Virtual é uma interface entre o utilizador e um sistema computacional, cujo objetivo principal é simular um Envolvimento Virtual realístico no qual é possível navegar e com o qual se pode interagir em tempo real. O objetivo é proporcionar ao utilizador uma forte sensação de imersão no Envolvimento Virtual ou seja, a sensação de presença física e efetiva nesse envolvimento. Para haver o máximo de imersividade nos Envolvimentos Virtuais são usados diversos dispositivos para que a navegação e interação sejam o mais credíveis possível, dispositivos tais como *Head-Mounted Displays*, luvas de dados, de rastreamento e dispositivos que geram sensações de tato e força (*feedback* háptico). Alguns sistemas dispõem de superfícies de representação de grandes dimensões como por exemplo a *CAVE*. Atualmente a Realidade Virtual é utilizada em diversas áreas porque é uma forma de simular uma experiência próxima da realidade reduzindo, em alguns casos, o perigo que existe no mundo real e permitindo de forma fácil a repetição de situações ou experiências.

A Realidade Virtual é aplicada em áreas tão diversas como a Medicina para treino cirúrgico em pacientes virtuais, o entretenimento com os jogos e filmes tridimensionais, a Psicologia no tratamento de fobias e traumas, entre outras.

Este projeto, intitulado "*Interaction in Virtual Environments with the Upper Body*", desenvolveu-se no âmbito da Realidade Virtual e enquadrou-se no projeto *"Future Safety Warnings: Virtual Reality in the study of technology-based warnings"* financiado pela Fundação para a Ciência e a Tecnologia (PTDC/PSI-PCO/100148/2008) que se contextualiza na área de Ergonomia. Este trabalho foi realizado no Laboratório de Ergonomia (ErgoLAB) da Faculdade de Motricidade Humana da Universidade Técnica de Lisboa, mais especificamente na unidade de Realidade Virtual chamada ErgoVR, e contou com o trabalho de uma equipa multidisciplinar composta por Ergonomistas, Psicólogos, Engenheiros Informáticos, Arquitetos, Designers entre outros. Para suportar todos os projetos realizados no ErgoVR existe um sistema de Realidade Virtual com o mesmo nome.

No projeto *"Future Safety Warnings: Virtual Reality in the study of technology-based warnings"* a Realidade Virtual é utilizada para avaliar a consonância

comportamental dos participantes perante avisos de segurança em situações de emergência no interior de edifícios.

O projeto descrito neste documento veio resolver dois aspetos que podem afetar a imersão: (1) o facto de o utilizador não ser representado por nenhum Humano Virtual; e (2) o facto de a interação com os objetos do Ambiente Virtual ser limitada e pouco natural.

Existe um sistema de interação que funciona da seguinte forma: é colocado um sensor na mão do participante, quando este se encontra num mínimo de uma distância pré-definida do botão, bastava esticar a mão para o sistema detetar que tinha havido um movimento e desencadear o evento associado à ação. No entanto este sistema tem algumas limitações visto que o único feedback visual que o participante tinha era um cursor bidimensional o que limita a perceção de distância do objeto que o participante quererá interagir. Para além desta limitação, o sistema apenas permitia a ação de pressionar.

Assim, o objetivo principal deste projeto é: (1) a criação de um Humano Virtual; (2) a possibilidade de reproduzir os movimentos do participante e refleti-los no Humano Virtual; e (3) permitir que o sistema suporte mais ações como por exemplo agarrar ou largar. Com a criação do Humano Virtual e a reflexão dos movimentos do participante, existirá uma maior perceção de distância no Envolvimento Virtual.

Para cumprir com este objetivo utilizaram-se sensores de movimento que captam a orientação 3D e dados cinemáticos relativos aos membros superiores. Estes sensores de movimento são colocados no braço, antebraço e mão, para capturar os movimentos e orientações reais do participante e passá-los para os membros do Humano Virtual, cujo modelo foi criado previamente. A este modelo e a todos os elementos do Envolvimento Virtual associaram-se características físicas, como por exemplo a massa, para dar realismo e credibilidade à simulação e fazer com que passasse a ser possível a interação do utilizador com determinados objetos presentes no ambiente.

O projeto descrito nesta tese envolveu quatro etapas.

A primeira etapa foi de familiarização com o sistema ErgoVR, com as ferramentas de desenvolvimento nele utilizadas e foi realizado um levantamento do estado da arte. Nesta etapa também se criou um Humano Virtual com as ferramentas de modelação 3D que permitiram criar um modelo com esqueleto, animações e texturas e que tornaram possível exportá-lo para o formato utilizado no sistema ErgoVR.

Na segunda etapa tratou-se de todas as questões relativas aos sensores, leitura de dados, transformações dos ângulos de Euler e transposição dos dados provenientes dos sensores no Humano Virtual.

A terceira etapa foi relativa à simulação das regras da física Newtoniana dentro do Envolvimento Virtual.

A quarta etapa foi relativa às formas de interação do utilizador com os objetos do Envolvimento Virtual, como agarrar, largar, puxar e pressionar.

Na etapa de familiarização decidiu-se que este projeto iria desenvolver-se sobre a plataforma Microsoft .NET (*DotNET*) visto o sistema ErgoVR ter sido desenvolvido sobre essa mesma plataforma. Deste modo garantia-se a integração do resultado deste projeto no sistema ErgoVR de uma forma mais simplificada. Ao mesmo tempo, as características da plataforma .NET enquadravam-se com as necessidades deste projeto.

Na fase inicial do projeto houve o envolvimento num trabalho de equipa que foi fundamental para a etapa inicial de familiarização e para compreender o funcionamento do sistema ErgoVR. Deste envolvimento surgiu a colaboração num artigo científico, elaborado pela equipa do laboratório, como co-autora, intitulado "*Using space exploration matrices to evaluate interaction with Virtual Environments*".

Neste artigo descreve-se um estudo realizado com o sistema ErgoVR que avalia as decisões dos utilizadores perante a influência da informação de segurança colocada nos ambientes quando são confrontados com situações de emergência. São recolhidos durante a simulação, entre outros, dados relativos à posição do participante, às distâncias percorridas, aos tempos do percurso. Com estes dados são geradas matrizes a partir das quais é possível identificar especificamente os fluxos e as zonas do ambiente que são mais visitadas pelos participantes, e deste modo avaliar o seu comportamento.

Na modelação do Humano Virtual, ainda na fase inicial do projeto, foram realizados testes exaustivos utilizando diversas ferramentas de modelação com o intuito de identificar as ferramentas que exportam o modelo para um formato aceite pelo sistema ErgoVR mantendo correta toda a informação necessária associada (malha, texturas, cabelo, animações corporais e faciais e roupas). Concluiu-se que a melhor abordagem seria trabalhar com o 3DS Max e com o Daz 3D.

Na segunda etapa foi implementada a biblioteca responsável por fazer a leitura dos dados dos sensores e transpô-los para os ossos do Humano Virtual. Esta biblioteca é composta por quatro classes, cada uma com funções diferentes. No final desta fase já

era possível colocar os sensores no utilizador e ver o Humano Virtual refletir os movimentos no ambiente.

Na terceira etapa, para realizar a biblioteca de física, foi necessário levar a cabo um levantamento sobre questões de física. Para não haver perda de imersão do utilizador no ambiente é necessário impedir situações como: (1) o Humano Virtual trespassar as paredes; ou (2) o Humano Virtual levar a mão a um objeto, e esta atravessá-lo. Para tal, todo o objeto tem de ter associadas massa, gravidade e forças para se deslocar no ambiente.

Na última etapa, para desenvolver a biblioteca de interação com objetos, foi necessário fazer um levantamento sobre quais as ações que um utilizador podia realizar no sistema. Conclui-se que seria necessário poder agarrar, largar, puxar e pressionar. Ao realizar esta biblioteca decidiu-se que alguns objetos virtuais tinham de ter associados a si informação extra sobre os possíveis modos de interação que o utilizador pode realizar sobre eles.

As bibliotecas desenvolvidas neste projeto constituem um módulo perfeitamente integrável no sistema ErgoVR e permitem a utilização de sensores nos membros superiores do utilizador cujos movimentos são refletidos no Humano Virtual correspondente. O utilizador pode, de modo natural, interagir com elementos do ambiente realizando gestos relativos às ações de agarrar, largar, puxar e pressionar.

Considerou-se que o módulo desenvolvido é uma mais-valia para o ErgoVR porque permite a aplicação deste sistema de Realidade Virtual a diferentes cenários em diversos âmbitos, sempre que a interação de um utilizador humano com objectos presentes no Envolvimento Virtual seja requerida.

Palavras-chave: Realidade Virtual, Envolvimentos Virtuais, Movimento, Interação Humana, Imersão.

Abstract

The Virtual Reality (VR) is an interface between the user and a system and is main goal is simulate a Virtual Environment (VE) next to the reality. The advantage of use VR is the possibility of simulates the dangerous that exist in real world or allowing the repetition of situations and experiences.

Nowadays VR is used in many areas, from Medicine, for surgical training in virtual patients, to the army in which the soldiers do virtual training.

The project "Interaction in Virtual Environments with the Upper Body" is developed in the context of Virtual Reality and is a part of the project *"Future Safety Warnings: Virtual Reality in the study of technology-based warnings"* funded by the Portuguese Science Foundation (PTDC/PSI-PCO/100148/2008) . This project was developed in the Ergonomics Laboratory (ErgoLAB) of the Faculty of Human Kinetics of the Technical University of Lisbon more specifically in the research unit ErgoVR.

In the project "Future Safety Warnings: Virtual Reality in the study of technology-based warnings", VR is used to evaluate the participant's behavior towards safety warnings in emergencies inside buildings. In this project the interaction was weak and due to the fact of a participant do not have virtual representation, the immersion was less. Therefore, the main goal of the described project is give to the participant the possibility of see is upper limbs' movements reflected in a Virtual Human (VH) inside the VE, and have the possibility to interact with virtual objects, give more sensation of immersion. To complete with this goal sensors were used and place in the arm, to capture the participant's movements. A VH was created to performed the participant´s movements in the VE. To this VH and to the VE, physics elements were added to give more credibility to the simulation and make possible the interaction with objects in the scene.

Keywords: Virtual Environment, Virtual Reality, Movement, Human Interaction, Immersion

Contents

List of Figures

List of Tables

List of Sample Code

Chapter 1

Introduction

1.1 Motivation

Virtual Reality (VR) tries to simulate reality by using tridimensional (3D) reproductions of objects and real environments. Computers are used to create those environments in which you can navigate and interact with devices (Gutierrez, Vexo, and Thalmann 2008).

The use of VR has increased and Virtual Environments (VEs) became a useful and effective tool in several areas, such as:

- In the automobile industry VEs are created to do impact tests that evaluate the resistance and the security of the automobile;
- In Medicine, VR is used for example in teaching anatomy and simulating operations;
- In military training, scenarios of virtual war are created to be able to evaluate the soldiers' reaction to certain combat situations or just to practice techniques;
- In the industry of entertainment there are games with interaction in real time with the help of *Head-Mounted Display*s (HMD) and motion trackers;
- In education, VR is applied in distance learning with remote meetings and virtual participation in events.

This work was developed in the context of a discipline of the second year of the Master's degree in Informatics Engineering at the Faculty of Sciences of University of Lisbon, called Project in Informatics Engineering *(PEI – Projeto de Engenharia Informática).*

This project, "Interaction in Virtual Environments with the Upper Body" fits in another one called *"*Future Safety Warnings: Virtual Reality in the study of technology-based warnings*"*, which is funded by the Portuguese Science Foundation (PTDC/PSI-PCO/100148/2008) and is developed in the Ergonomics Laboratory, ErgoLAB, of Faculty of Human Kinetics of the Technical University of Lisbon, more precisely in the research unit ErgoVR[1].

VR appears in the ErgoLAB with the goal to evaluate the participant's behavior in experimental VE. To be able to evaluate the behavior of the participant there must be immersion in the VE that is provided for navigation and interaction. To support this project has been created a VR system called ErgoVR.

This project focuses in the interaction that the participant can have with virtual objects, like grabbing or pushing. This kind of interaction are only in the virtual side because there is not available a haptic feedback system at the Ergonomics Laboratory. To accomplish the task, sensors are used, placed in the arms and hands of the participant in order to capture his movements and pass that information to the Virtual Human (VH). This VH reproduces in the VE the actions of the participant.

Providing ErgoVR system with interaction modes that are more natural will improve participant's sense of immersion and therefore he will behave naturally, as if he was experiencing the real situation.

1.2 Objectives

The VEs of the project *"Future Safety Warnings: Virtual Reality in the study of technology-based warnings"* consists in interior spaces of buildings consisted by rooms and corridors, where there are safety warnings and certain tasks to perform. In these VEs the behavior and attitudes of the participant will be evaluated in respect to the compliance and respect for the indications given by the warnings and the realization of the task.

The objective of this project is to allow the participant to interact in a more natural way with virtual objects in a virtual scenario in order to perform some required tasks (e.g., open a door, press a fire alarm button, or pick up a fire extinguisher).

[1] http://www.fmh.utl.pt/ergovr/index.htm

To accomplish this, the participant must have a corresponding VH model in the environment. Since the view of the participant will be from the point of view of the VH, when moving the upper limbs, the participant will see the same movements reflected in the VH.

1.3 Contributions

This project was developed in the Faculty of Human Kinetics of the Technical University of Lisbon, in the Ergonomics Laboratory's VR research unit called ErgoVR.

In this research unit, several different VE have been created to perform tests related with Ergonomics studies. However, a HV representing the participant was still missing. This project address this limitation and enables the possibility of representing the upper body movements of the participant on the HV, so that it would be possible for the participant to interact with objects in the VE, performing natural gestures such as grabbing, dropping, pushing and pressing.

To accomplish this, in this project, an ErgoVR (L. Teixeira et al. 2010) compatible module including several libraries, named ErgoInteract, was created. Besides, four VH models with different genders, types of hair and clothes, were modeled and animated, ready to inhabit any VE loaded by ErgoVR. These models are capable of reflecting the natural movements of the participant that are input to the system through sensors used in the upper limbs of the participant. Three of the VH models have facial and body animation to interact with the participant in a VE.

ErgoInteract is useful and can be important to several application areas. It can be used in the psychology area to help people that have phobias to interact with the VE, or in industrial design to visualize and interact with virtual prototypes of a product. In architecture it can be used to interact with architectural 3D models.

There was also a participation in a scientific paper, "*Using space exploration matrices to evaluate interaction with Virtual Environments*" (L. Teixeira et al., 2012). This collaboration was very useful in the familiarization phase, to understand the ErgoVR system and to ease the integration in the team.

1.4 Document Structure

This document is dividing into four chapters:

Chapter 1 - Presents the motivation, the objectives and the contributions of the project.

Chapter 2 - Presents VR concepts, explaining its origin and evolution. A state of the art in the area is also presented.

Chapter 3 - Presents the analysis and planning of the project. In the analysis it will be presented the context of the project, followed by the requirements and his use cases. In the planning the development method chosen will be presented and the project phases. Finally, the resources to fulfill the goals are enumerated.

Chapter 4 – Presents the developed work. The first section presents the work related with the VHs modeling and body/facial animation. The second section gives a general explanation of the developed system and, after that, it is given an explanation of each library in separate.

Chapter 5 – Presents conclusions and draws some reflections about future work.

Appendices – This document has five appendices. The first appendix is the Gantt chart that presents the preliminary project planning. The second appendix is the complete class diagrams of all libraries. The third appendix is a user manual that helps the creation of a compatible HV model with the ErgoVR system. The fourth appendix is a user manual that helps the creation/modification of a specific configuration file called ConfigFile.cfg. The last appendix is a complementary user manual to work with the MTx sensors.

Chapter 2

Context and State of the Art

This chapter presents the history and some fundamental concepts related to Virtual Reality and also a state of art covering some investigation projects in similar areas.

2.1 Fundamental Concepts of Virtual Reality

Virtual Reality can be defined as an advanced interface for computer applications that allows the participant to navigate and to interact using sensory channels (vision, hearing, touch and even smell and taste).

According to Gutierrez et al. (2008), a VR system simulates reality, using a computer to create 3D environments. The main goal is to give the participant the illusion of being inside a virtual environment that reacts and changes according to his interaction.

In 1950 a system was developed by the U.S. Air Force, that provided some immersion to a participant in a VE, which consisted in a flight simulator for testing (Tori, Kirner, and Siscoutto 2006). In 1962, Morton Heilig, considered the father of VR, created the Sensorama Simulator (M L Heilig 1962), which simulated a motorcycle ride in Manhattan. It consisted in multiple sensors and the participant were subjected to various physical sensations that created a sense of immersion: simulating the hole in the pavement, the vibration of the seat and body position corresponding to the inclination of the motorcycle, the wind in the face and in the hair, different smells in specific locations of the city (e.g. food smells near a restaurant) and the surrounding sounds.

A few years later, Heilig launched the *Experience Theater* (Morton L. Heilig 1969) which was a version of Sensorama Simulator but for large scale audiences.

However, the term VR only came to public knowledge in the beginning of the decade of 80 with the computer scientist, Jaron Lanier, which defined the term 'Virtual Reality'.

Virtual Reality involves three I´s aspects: Immersion, Imagination and Interaction (Burdea and Coiffet 2003) (Figure 1).

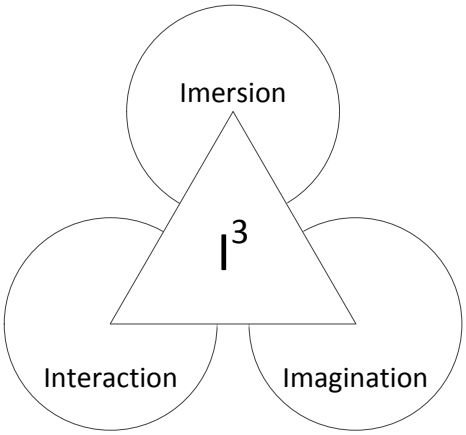


Figure 1 - The Three I´s of Virtual Reality[2]

The performance of the participant depends on these three I's. In an immersive VR system, the participant feels that he is completely inside the VE, using devices that stimulate human senses. Participant interaction is mandatory in VR systems. The participant must feel that he is really a part of the VE, and that he is able to change something inside it. Of course, participant's imagination also performs an important role in the sense of immersion.

The architecture of a VR system is usually composed by five components (Figure 2).

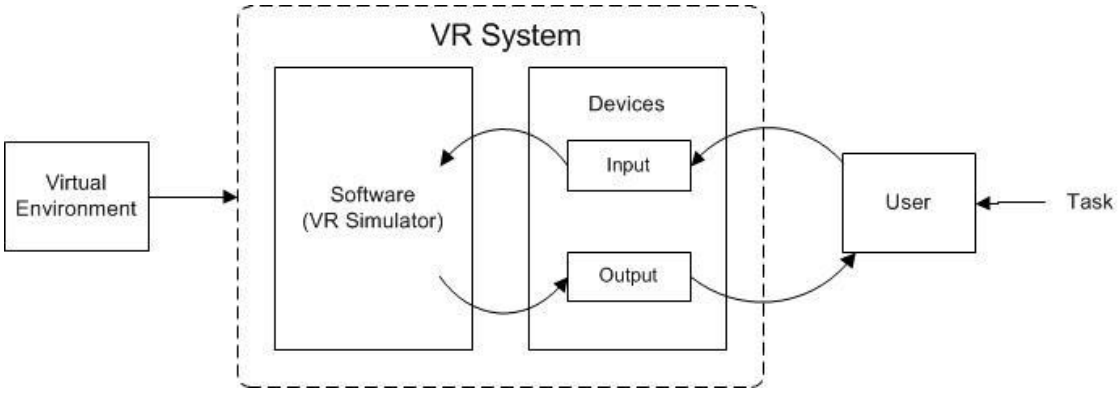


Figure 2 – General Architecture of Virtual Reality Systems[3]

[2] Images adapted from (Burdea and Coiffet 2003)

[3] Diagram adapted from (Burdea and Coiffet 2003)

The *User* performs some *Task* in the *Virtual Environment*.

The *Virtual Environment* is loaded by the *VR System*, which is composed by the *VR Simulator* and several *Devices*. The *VR Simulator* is the software, a set of applications that include software libraries, such as a graphics engine, Integrated Development Environment (IDE) and 3D modeling software. *Devices* are hardware components used to perform specific *Tasks* (input devices) and/or to provide an output feedback (output devices).

Input devices are classified as passive or active. Passive devices are usually sensors that capture the movement of a participant and can be supported by different technologies:

- Mechanical – consists of a kinematic structure composed of links that contain sensors (Figure 3);
- Electromagnetic – uses magnetic fields for the detection of the orientation and position of the sensor (Figure 4);
- Ultrasonic – using ultrasonic signals produced by a transmitter to determine the real-time position of a moving element (Figure 5);
- Optical – uses a system of cameras that capture the location of markers (Figure 6);
- Inertial – uses electromechanical instruments to detect movement by measuring the change in inclination, acceleration and gyroscopic forces (Figure 7);
- Hybrid – composed of two distinct systems, the most common is the combination of optics and inertia simultaneously.

Figure 3 - Mechanical Device[4]

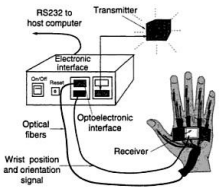


Figure 4 - Electromagnetic Device[5]

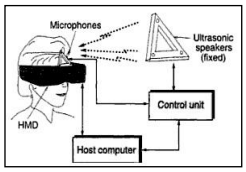


Figure 5 - Ultrasonic Device[5]

Figure 6 - Optical Device[6]

Figure 7 - Inertial Device[7]

Active input devices allow the interactive change based on decision of the participant, such as navigation, selection and manipulation of virtual objects through devices like a mouse, joysticks and others.

Output devices provide an active feeling to the participant such as sound, touch, odor and/or visual information that can be a response to changes in the VE.

Therefore, VR has its disadvantages:

- High costs that are associated with VR devices and production of their own VEs;
- Large limitation of force and tactile feedback devices;
- Some people, experience discomfort such as simulator sickness and disorientation using VR devices.

[4] Images taken from http://www.fakespacelabs.com/tools.html

[5] Images taken from (Burdea and Coiffet 2003)

[6] Images taken from http://digitalcortex.net/work/academic/virtual-reality/

[7] Images taken from http://www.intersense.com/categories/18/

2.2 State of the Art

Virtual Reality is applied in several areas, such as design, architecture, education, entertainment, industry and medicine. Within these areas, several research projects that use various devices are carried out.

In this chapter, some projects related with the scope of this project are presented. We divided this presentation in three parts. First, it is presented studies that involve real time participants interaction with objects in the VE, using various types of devices. The second part presents some projects that use the same sensors that are used in this project, *Xsens MTx*[8].

2.2.1 User interaction with objects in the VE

Two of the next presented works focus on the area of Augmented Reality (AR). Although the area of this work is not AR, is interesting to talk about the modes and interaction styles that are investigated and used in this study since they can also be applied in VR.

- *FingARtips – Gesture Based Direct Manipulation in Augmented Reality*

  FingARtips (Buchmann et al. 2004) is a project that studies a technique for interaction with virtual objects through gestures using fingers. Markers are used to track the user's gestures. This technique was applied to an interface of urban planning in order to make it possible for architects to build virtual cities modifying them as they wish. Users can interact with the application by grabbing, pointing and pressing virtual objects such as houses, and even navigating through the scene.

- *"GeFighters: and Experiment for Gesture-based Interaction Analysis in a Fighting Game"*

[8] http://www.Xsens.com/

Nowadays, the biggest focus in the gaming world is trying to give to the user the chance to perform commands without the need to press several buttons. This paradigm applies to the Nintendo Wii or to the PlayStation Move. The *GeFighters Game* (J. M. Teixeira et al. 2006) is a project that tries to give the user the possibility to interact with the game with gestures, using tracking sensors. This article talks about how a game can implement the use of gestures. There are several ways of capturing gestures such as, for example, with gloves or capturing images through cameras. To capture gestures this game uses *ARToolkit* on an AR library that uses fiducial markers. A marker is placed in each hand to capture the gesture and there is a set of figures that is previously given to the user that explains all the positions that correspond to game commands.

- *"Real Behavior in Virtual Environments: Psychology Experiments in a Simple Virtual-Reality Paradigm Using Video Games"*

  Virtual Reality has been widely used in Psychology because it is possible to create situations that are impossible or difficult to generate in the real life and is an easy way to treat difficult problem in some people. The paper *"Real Behavior in Virtual Environments: Psychology Experiments in a Simple Virtual-Reality Paradigm Using Video Games"* (Kozlov and Johansen 2010) talks about the usefulness of a simple video-game-based virtual environment for psychological research on real world behavior. Their main focus is Helping behavior. In this game, the participants are instructed to get to the exit of a building in a short time. When they are on their way out they are faced with virtual persons asking for help in the presence, or not, of virtual bystanders. The conclusion was that the bystanders and time-pressure reduce helping in a virtual game. The participant's behavior similar to previous real life experiments with human actors. The results of the study defend the usefulness of a VR game as a psychological experimentation.

2.2.2 **Tracking with the *MTx* sensors**

The studies referred in this section are focused on tracking lower or upper limbs, using MTx sensors, the same sensors that are used in this study.

- *Baseball pitchers analysis*(Brand 2008)

  An experiment was conducted (Brand 2008) with pitchers of a baseball team, in Chicago in August 2009. The purpose of this study was to make a real-time tracking of the movement of the scapula in pitchers with the objective to prove that after the launch and the stress suffered in the joints of the upper limbs, the scapula loses the ability to follow the movement of the humerus. For this study, four sensors *MTx* were used, in the thorax, scapula, humerus and forearm.

- *"Rowing with MVN"*

  Working in the areas of rehabilitation, ergonomics and sports training the project "Rowing with MVN"[9] focuses on competitive rowing. Their ultimate goal is to develop a tool that provides relevant and accurate information to the coach who led him to make decisions about training and activities that are beneficial to the competition. For this study was used the device MVN, that consists in a suit of lycra that has incorporated MTx Sensors.

- *"Outwalk: a novel protocol for clinical gait analysis based on inertial and magnetic sensors"*

  "Outwalk" (Cutti et al. 2010), is a protocol of walking that tries to measure the thorax-pelvis and lower-limb 3D kinematics in children with cerebral palsy or amputated during gait in free-living conditions. This tool helps in the

[9] http://www.Xsens.com/en/movement-science/sports-science/rowing-with-mvn

rehabilitation process. Walking can be expressed in a generalized way due to its simple kinematics. There are analysis of movement by stereo photogrammetry that provide useful and detailed data, the problem is that has high cost associated and the use of the stereo photogrammetry equipment is limited to the laboratory. That limitation restricts the movement of a few meters, leads to an exhaustive environment since that the patient has to walk back and forward to performed the meters and because the floor of the laboratory has different characteristics from the exterior floor (e.g., street, garden) making the data less realistic. To overcome those problems it was chosen to analyze the motion through inertial sensors. That way the analysis can be done in external environments being closer to the typical gait pattern. The MTx sensors are placed in the trunk, pelvis, hip, knee and ankle, and the data related to certain positions requested are recorded.

- *"Stair climbing: a comparison with an optical tracking device"*

  In studding the action of climbing stairs, *Stair climbing* (Bergmann, Mayagoitia, and Smith 2009), starts from the idea that the movement is the most important activity for health. The reports made by older people show the difficulty in climbing stairs and are useful to evaluate and define the functional state of these older people. The purpose of this study is to verify the anatomical angles using a portable system during stair climbing and compare them with the data acquired by the optical trackers. The portable system consists in MTx Sensors placed on the legs and active markers Codamotion[10] that are placed on the stairs.

[10] http://www.codamotion.com/

Chapter 3

Analysis and Planning

The analysis and planning of this project is presented in this chapter. The analysis consists in describing the context of use of this project, the functional and non-functional requirements and the use cases of the project. In the planning section, it is presented the development process, the planning of the project and the chosen resources.

3.1 Project Analysis

3.1.1 Context of Use

This project was developed in the Ergonomics Laboratory of the Faculty of Human Kinetics of the Technical University of Lisbon.

In the project "*Informação de Segurança. Avaliação da eficácia de sinais pictóricos de segurança"*(Duarte 2004) it is studied the comprehension of safety signs in different populations, through the completion of questionnaires. In this study, it was found, during the analysis of the results, that the context highly influences participants' response. With the need to provide contexts of use that are more realistic and dynamic and the possibility of evaluating the behavior and not only the safety materials arises the possibility of use VR. With VR, it is possible to introduce a user in a realistic and immersive world where he can perform a task which in real life could involve some risks. This led to the project *"*Using Virtual Reality to Evaluate the Safety Information Effectiveness" financed by FCT (PTDC/PSI/69462/2006), which has a main objective to use VR to evaluate the compliant behavioral against safety warnings in workplaces. Later, and in order to proceed with the work, arose the project *"*Future Safety Warnings: Virtual Reality in the study of technology-based warnings*"* also financed by FCT

(PTDC/PSI-PCO/100148/2008). This project focuses on the issue of technology based safety warnings and intends to study the ability of these warnings to counteract the effect of the environment affordances. *"Affordance refere-se às propriedades percebidas e efetivas de um objeto, essencialmente as propriedades fundamentais que determinam como os objetos podem ser usados"*(Norman 2002). To support these two projects has been created a VR system called ErgoVR. It is in this ErgoVR system that is inserted this project "Interaction in Virtual Environments with the Upper Body" for user interaction with the VE.

3.1.2 **Stakeholders**

The stakeholders of this project belong to the ErgoVR team. Observing the diagram in Figure 8 it is possible to distinguish the different stakeholders. The projects made in ErgoVR have the focus in Ergonomics and a responsible for the ErgoVR is an Ergonomist and a stakeholder of this project. VE are developed by Designers and Architects, which are another group of stakeholders. VH are necessary to inhabit these VE and to reflect the movements captured by sensors attached to the upper limbs of the user. The task of modeling and animating these VH as well as the development of a library called ErgoInteract, a new module in the ErgoVR system, were both performed by a Computer Engineer stakeholder in the context of the PEI. Another computer engineer is responsible for the whole ErgoVR system that supports the simulations. This is another stakeholder and he is the responsible for the non-functional requirements of the system.

Therefore, this project has four main groups of stakeholders, which are:

- Ergonomist;
- Designers and Architect;
- Computer Engineer PEI
- Computer Engineer - System Developer;

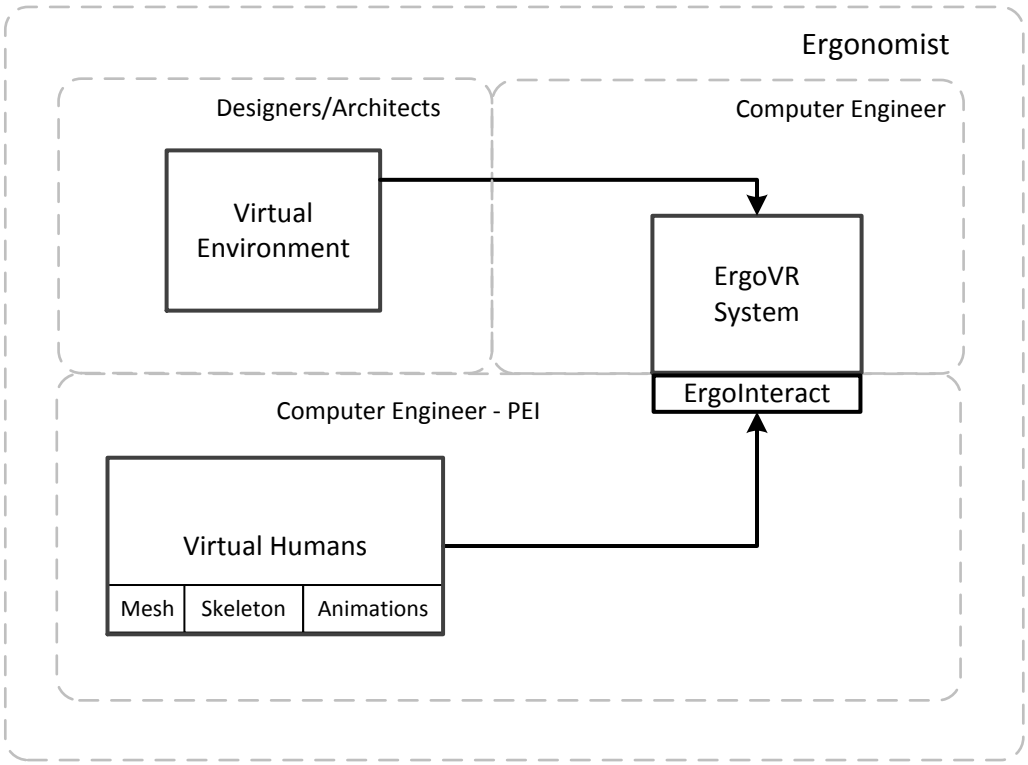


Figure 8 – Stakeholders' Areas

3.1.3 **Requirements**

Functional Requirements

There are several ways to interact with the VE and the features desired to this part of the system are going to be specified. Functional requirements are defined by two groups of stakeholders: the Ergonomist and Designers and Architects. The requirements are presented separately according to each group.

Requirements of the Ergonomist:

1. The user has a corresponding VH in the VE;
2. The VH reproduces the upper limbs' movements performed by the user;
3. The user can interact with specific objects in the VE using natural gestures like grabbing, dropping, pushing and pressing.

Requirements of the Designers and Architects team:

1. To be able to choose one VH from a set of predefined female and male models;
2. To be able to choose the number of sensors to work with;
3. To be able to choose a specific VE from a set of predefined ones;
4. To be able to make the previous choices just by editing a configuration file;

5. To be able to create new VH models with skin, skeleton, animations and textures, and make them available in the set of predefined models;
6. To be able to place properly the sensors in the upper limbs of the user.

Non-Functional Requirements

As previously mentioned, the computer engineer/system developer is responsible for non-functional requirements which are related with quality attributes. The system was developed as a module compatible with the ErgoVR system.

Comprehensibility and **Usability** were identified as requirements**.** To ensure these requirements the system has been developed having in mind that it is important for the team to know how the system works and how it can be modify or updated. To make this easier, there is an editable configuration file, all the developed code is documented and there are three user manual.

Reliability is another requirement. To ensure that, data provided by sensors are continually updated by the system and immediately reflected in the VH that exhibits fluid movements.

To ensure **Scalability,** ErgoInteract has been designed and implemented so that all modules can be as independent as possible from one another. This is important to reduce dependencies and problems with future changes in case it needs to be ported to other environment.

Modifiability is a requirement that depends on Comprehensibility, Usability and Scalability requirements.

3.1.4 Use Cases

This section describes the use cases for the functional requirements of the Ergonomist and the Designers/Architects.

Figure 9 presents the Use Cases diagram of the Ergonomist and Figure 10, presents the Use Cases diagram of the Designers and Architects both figures are followed by the corresponding specification.

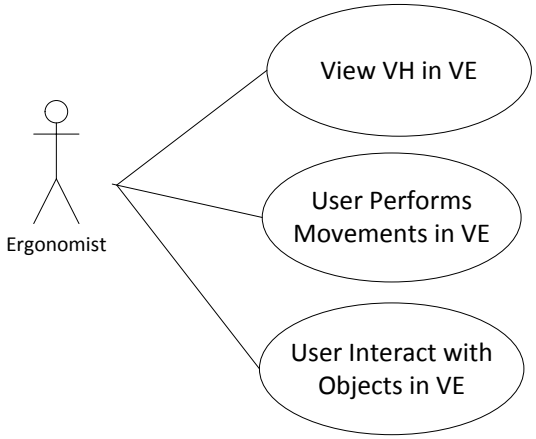


Figure 9 - Ergonomist Use Cases

Use Case 1

Use Case: View the VH

Actor: Ergonomist

Pre-Conditions:

- User must be informed about the application;

Body:

1. Run the application.
2. User moves the head and sees the body of the VH.

Post-Conditions:

- The user can see him represented in the VE.

Use Case 2

Use Case: User Performs Movements in VE

Actor: Ergonomist

Pre-Conditions:

- User must be informed about the application;
- The user needs to have the sensors connected.

Body:

1. Run the application.
2. User moves the upper limbs to perform the desired movements.

Post-Conditions:

- User can see their movements reflected in the VH in the VE.

Use Case 3

Use Case: User Interacts with Objects in VE

Actor: Ergonomist

Pre-Conditions:

- User must be informed about the application;
- The user needs to have the sensors connected.

Body:

1. Run the application.
2. User tries to interact with an object in the VE.

Post-Conditions:

- The user can interact with objects with the upper limbs in the VE.

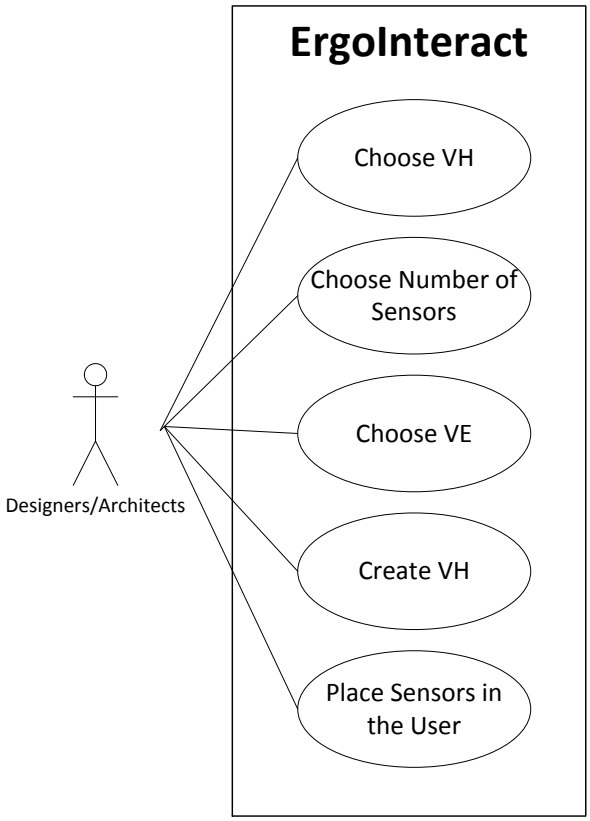


Figure 10 - Designers/Architects Use Cases

Use Case 1

Use Case: Choose VH

Actor: Designers/Architects

Pre-Conditions:

- The group needs to be informed about the use of the application.
- The group needs to have the User Manual “Configure FileConfig.cfg”

Body:

1. Open configuration file, ConfigFile.cfg.
2. In the “LoadHV” section, write in the “VHModel” key “man” or “woman”.
3. Save file.
4. Run the application.

Extensions:

1a. Error opening the configuration file.

1b. Create a new configuration file according with the User Manual Configure ConfigFile.cfg.

2a. If nothing is written in the field, a female VH is chosen.

3a. Error saving the file, create a new one, start in the step 1b.

4a. Error running the application, consult ogre.log.

Post-Conditions:

- User has the VH with the gender defined in the configuration file in the VE.

Use Case 2

Use Case: Choose the number of sensors.

Actor: Designers/Architects

Pre-Conditions:

- The group needs to be informed about the use of the application.

Body:

1. Open configuration file, ConfigFile.cfg.
2. In the “ConfigSensor” Section and “Arm” Key write what arms you want to work, in the ”NumSensores” Key put the number of sensors that are pretend to be used and in case of one sensor, in the “Member” Key place the bone to work with.
3. Save file.
4. Run the application

Extensions:

1a. Error opening the configuration file.

1b. Create a new configuration file according with the User Manual Configure ConfigFile.cfg.

3a. Error saving the file, create a new one, start in the step 1b.

4a. Error running the application, consult ogre.log.

Post-Conditions:

- User has the VH in the VE that will reflect movement with the chosen number of sensors.

Use Case 3

Use Case: Choose the VE.

Actor: Designers/Architects

Pre-Conditions:

- The group needs to be informed about the use of the application.

Body:

- Open configuration file, ConfigFile.cfg.
- In the "LoadScene" section, write in the "PathFile" the physics localization of the scene to charge.
- Save file.
- Run the application.

Extensions:

1a. Error opening the configuration file.

1b. Create a new configuration file according with the User Manual Configure ConfigFile.cfg.

3a. Error saving the file, create a new one, start in the step 1b.

4a. Error running the application, consult ogre.log.

Post-Conditions:

- User has the VH in the VE that will reflect movement with the chosen number of sensors.

Use Case 4

Use Case: Create VH

Actor: Designers/Architects

Pre-Conditions:

- The group needs to have the User Manual "Create an VH for the ErgoVR system"

Body:

1. Follow the steps detailed in the User Manual "Create an VH for the ErgoVR system".

Post-Conditions:

- User has the VH to load in the VE.

Use Case 5

Use Case: Place Sensors in the User

Actor: Designers/Architects

Pre-Conditions:

- The group needs to be informed about the use of the application;
- The group has to have the XM-B User Manual[11];
- The group has to have the Complementary User Manual to the Sensors;
- The ConfigFile.cfg must be updated in the "ConfigSensors" Section.

Body:

1. Open the Xsens box and connect the sensors.
2. Place the sensors in the user in the bones that is needed.
3. Turn on the sensors.
4. Run the application.
5. Move the upper limbs to test.

Extensions:

1a. Do not know how to connect the sensors, read the XM-B User Manual and the Complementary User Manual to the Sensors.

[11] It was not given permission by the Xsens to reference the Manual in this document.

2a. Do not know how to put the sensors in the user, read the XM-B User Manual and the Complementary User Manual to the Sensors.

4a. Error running the application, consult ogre.log to found the error and repeat step 4.

5a. Upper limbs do not move, verify command line to see if the sensors are being recognized, repeat step 3.

Post-Conditions:

- The sensors are working in the user´s upper limbs;
- The user sees the upper limbs movements reflected on the VH in the VE.

3.2 Project Planning

3.2.1 Development Process

The development strategy of this project focuses in an iterative and incremental model. The model used in this project is base in the Scrum model, as depicted in Figure 11 .

The choice of using Scrum was because this model is the most adequate for long projects and projects that are subject to changes of the requirements frequently. The justifications are still the same but there is another one: the planning meeting and the review of the accomplished work.

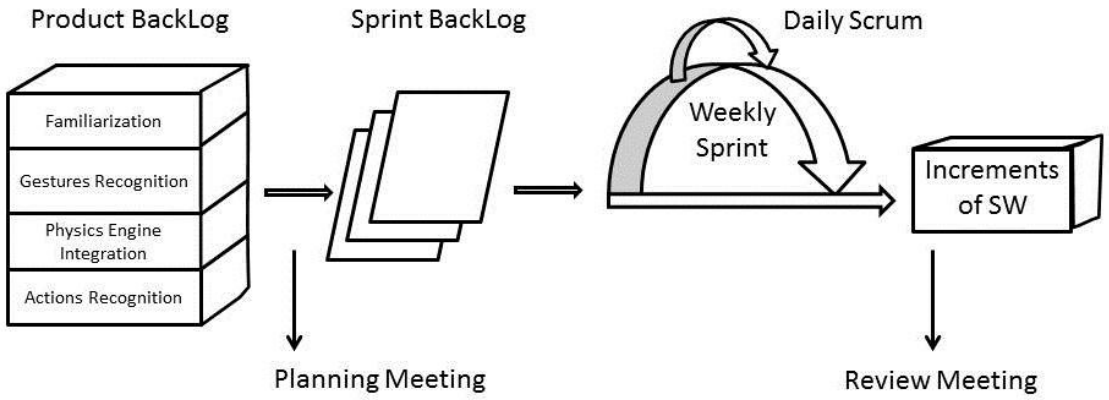


Figure 11 – Scrum

The Product Backlog is a set of all global requirements in the project and the Sprint Backlog are the selected requirements to do in a Sprint. A Sprint is a unit of

development process that can last between one week and one month and is done following the Plan-Do-Check-Act (PDCA) cycle that is going to be explained with more detail later on the document. A Daily Scrum is a short daily meeting, a Sprint Planning Meeting is Sprint Backlog for the next Sprint, and the Sprint Review is where the work done is reviewed.

In this project, the Product Backlog was divided in four groups, Familiarization, Gestures Recognition with motion sensors, Integration of the Physics Engine and Actions Recognition. Each of these general requirements is going to be divided in more specific requirements that can be made in Sprints of one week. At the beginning of each week the Sprint Planning Meeting takes place and at the end of the week there are the Sprint Reviews. Every day there is short brainstorming sessions. Normally these meetings are with different members according to the level of specificity of the topic. These brainstorming sessions are considered the Daily Scrum.

Three of the four groups referenced in the previous paragraph follow the PDCA cycle. The one that does not follow is the Familiarization in which is completed a theoretical study and some tutorials. In the other three groups, the first step is the definition of the goal and requirements in more detail (Plan), then it is followed by the implementation of the requirements defined (Do), test (Check) and in the end the validation and creation of the Demo (Act). This can be seen in Figure 12.

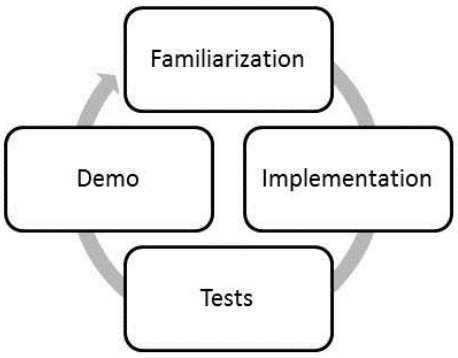


Figure 12 - PDCA Cycle

3.2.2 Project Planning

The initial planning had some changes and all the phases took more time than estimated to be completed mainly due to technical constraints. It have been removed the

last phase, Data Glove Integration, because it was a complementary goal to this project and now is considered as future work.

First Phase – Familiarization

This phase is composed by five steps, Theme Familiarization, Performed Tutorials, Modeling a Virtual Human, Develop Test Application and State of Art Study. The first three steps and last one were made within the expected times, but the fourth step, Develop Test Application, took more time than the estimated in the plan.

The task of defining a Virtual Human with mesh, skeleton, hair, clothes, body and facial animation in a compatible format with the ErgoVR system was more difficult than expected, consuming considerable more time that the scheduled. It was necessary to work with a considerable number of modeling software tools until the right one was found.

Second Phase – Gesture Recognizer

The main goal of this phase is to implement in the ErgoInteract system the possibility to reflect the real movements of the participants' upper limbs in the upper limbs of the VH. This phase was divided in three steps: Sensors Integration, Testing and Demo. The first step was the one with more duration, and all of the three steps were executed in the scheduled time interval.

Preliminary Report

The preliminary report has been written and delivered on schedule.

Third Phase – Physics Engine

This phase does not have changes in the main goal and steps that is composed. The main goal is to implement physics in the objects in the VE, to be possible to have physical behavior in the scene. The first step, Physics Engine Integration, took more time than the expected. There were some problems with the physics engine because it was used an open-source library that is still incomplete and does not have clear documentation, which makes it difficult the comprehensibility and further implementation in the system.

Fourth Phase – Actions Recognizer

The goal of this phase was to implement in the system the possibility for the participant to interact with smart objects.

Final Report

As predicted, this report has been made since the delivery of the preliminary report. It was updated every week with the performed work.

The completed Gantt Chart is in the Appendices.

3.2.3 Resources

The resources for this project have been defined in the in Phase 1 - Familiarization, and they are divided into material and technologic resources, Software (SW) and Hardware (HW) and human resources. As a result of this step the next decisions are taken:

Material and technological resources chosen by the laboratory team:

- *MOGRE*[12] (SW) – It is a *wrapper* to *OGRE* (*Object-Oriented Graphics Rendering Engine*), which is an open-source 3D graphics engine. As rendering system supports *OpenGL* and *DirectX* and runs on *Windows*, *Linux* and *Mac OS*. *MOGRE* (*Managed OGRE*) was developed for use OGRE across.NET languages.
- *Newton Game Dynamics*[13] (SW) – It is an open-source physics engine that simulates in real way interactions between rigid bodies in games and other applications in real time;
- *Xsens XBus Kit with 10 MTx sensors* (HW) – These are hybrid trackers and provide orientation and kinematic data (acceleration, earth's magnetic field and angular velocity) with three degrees of freedom (DOF) (Figure 13).
- *3DS Max (*SW*):* It is a commercial 3D modeling and animation tool.

[12] http://www.ogre3d.org/tikiwiki/MOGRE

[13] http://newtondynamics.com

Figure 13 - MTx Sensor

Material and technological resources chosen to be used in PEI:

- *Daz Studio Pro 4:* It is a free 3D modeling software that as posing and animation tools;
- *MakeHuman*[14] (SW) – It is an open-source software that generates 3D Virtual Humans with skeleton;
- *Microsoft Visual Studio* 2010[15] (SW) –It is a Microsoft IDE dedicated to the .NET Framework.
- *Microsoft Office Visio 2007* (SW)[16] – Application to create technical and professional diagrams of several types;
- *Ogre Command-Line Tools 1.7.2* (SW) – It is a *software* of OGRE that convert the exported files in xml to the native format of OGRE.

Human Resources

This work of PEI unfolds in collaboration with a multidisciplinary team[17] from the Ergonomics Laboratory that includes:

- One Ergonomist;
- One Architect;
- Five Designers;
- One Computer Engineer;
- One Professor of Numerical Methods;
- One Psychologist.

[14] http://www.makehuman.org/

[15] http://www.microsoft.com/visualstudio

[16] http://www.microsoft.com/visio

[17] http://www.fmh.utl.pt/ergovr/team.htm

This chapter presented the analysis and planning of the project. The analysis sections presented the context of the project, the requirements and the corresponding use cases. The planning sections described the development method chosen and the project phases. Finally, the material and human resources were enumerated.

Chapter 4

Design and Implementation

This chapter described the developed work. The first section explains how to obtain a complete VH model compatible with the ErgoVR system. Section two explains how the implemented module, ErgoInteract, works. The second section gives a general idea of how the developed system works followed by a separate detailed explanation of each library.

4.1 Modeling Virtual Human

One of the goals of this project is to find a standard method to create a VH model usable in ErgoVR system. As ErgoVR uses OGRE, the VH model must be OGRE compatible.

At the beginning, while still considering only the VH to reflect the participant's movements, upper and lower limbs with skeleton were enough to make possible the association of the necessary bones to the sensors.

However, along the project more VH models were needed to insert in the VE, for instance, as bystanders. As such, these VH should have:

- Body animation to give more realism to the model;
- Facial animation, for example, to simulate a dialogue or have natural facial expressions to give more realism;
- Realistic clothes and hair.

To model the VH many 3D modeling programs were tested. To have the model in the OGRE format with the desired characteristics, many importers, exporters and file formats were tested until a good conversion process was achieved. While performing

these tests, some problems have emerged due to the fact that several of these 3D modeling software are open-source and occasionally what is working in an older version stops working in the next version. Because of this, it was necessary to experiment many versions of the same software. Another reason is that the exporters and importers are also open-source, which sometimes makes them not complete tools. These exporters and importers were usually conceived for particular purposes and sometimes do not fulfill the general conversions. The exporters do not always export all the information and similar situations happen with the importers. Therefore, it is impossible to guarantee that all the information is correctly exported and imported. To find out which was the appropriated modeling/animation 3D tool to produce an VH model it was tested the following set of tools:

- Make Human 1.6 Alpha and Nightly Builds
- Poser PRO 2012;
- Daz 3 e 4 Pro;
- www.Evolver.com;
- 3DS Max 2009;
- Blender 2.49, 2.56 Beta and 2.57.

It was necessary that the last software in the conversion pipeline would be 3DS Max or Blender, because, from this list, they are the most complete and they have the plugins to export to the OGRE format.

Each software has their own importers/exporters supporting their specific import/export format (Table 1).

Export	Make Human		Poser		Daz		Evolver[18]	
Import	Blender	3DS[19]	Blender	3DS	Blender	3DS	Blender	3DS
File Formats								
Obj	**x**	**x**	**x**	**x**	**x**	**x**	**--**	**--**
Mhx	**x**	**--**	**--**	**--**	**--**	**--**	**--**	**--**
Dae	**x**	**x**	**--**	**x**	**x**	**x**	**--**	**--**
3ds	**--**	**--**	**x**	**x**	**--**	**--**	**--**	**--**
Fbx	**--**	**--**	**--**	**--**	**--**	**V**	**x**	**x**

Table 1 – V – Successful Conversion, **x** – Failed Conversion, -- – Nonexistent Conversion

[18] This software was used before being bought by Autodesk.

[19] 3DS Max

At the end of the test conversion, the choice was to work with the Daz Studio Pro 4 and 3DS Max 2009 tools.

The reasons for choosing Daz Studio Pro 4 were:

- There are pre-made basic VH models with skeleton;
- There is a library of textures, clothes and other objects to add to these pre-made models;
- There are skeleton and vertex animations already made, it is only required a drag-and-drop in the User Interface (UI) to apply it to the model;
- There is an exporter to the file format used by Autodesk's programs, the FBX format.

The reasons for choosing 3DS Max were:

- It is useful to work with this tool because is one of the most used by the ErgoVR team;
- Has support for the FBX file format, which supports all the characteristics that are needed for the VH (mesh, texture, skeleton, animations);
- There are several exporters to the OGRE format, but OgreMax was chosen because the team already used it.

The conventions to modeling a VH passed through the step of choosing an available model from the Daz Studio Pro 4's library, a texture for the skin, clothes and shoes, and then apply some animations. To apply facial animations the software has a pose controls menu that separate the face areas that can have deformation such as the mouth, eyes, eyebrows and nose, and each one of these areas has a slider to control the required deformation. For the body deformation there are pre-made animations such as walking, running, jumping, among others, and it is only needed to drag-and-drop them over the body. When the VH model is finished, it is exported to the FBX format and imported in 3DS Max. Before exporting to OGRE, it is required to verify the type of animation, vertex or skeleton, which will be detailed in the next sections, Facial and Body Animation. The VH is exported to the OGRE format represented by a .mesh and .skeleton files.

The next two sections are related with the creation of animations in the three VH model to insert in the VE.

4.1.1 **Body Animation**

The animations of the body are made through the skeleton. The skeleton is defined as a hierarchy of bones, that is, the shoulder is parent of the arm, the arm is parent of the forearm, and the forearm is parent of the hand, as it is possible to see in Figure 14.

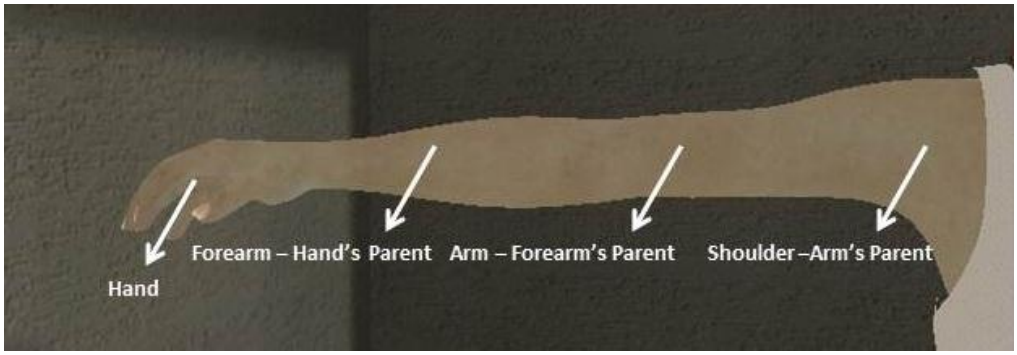


Figure 14 - Upper Limbs´ Hierarchy

The animation that uses the skeleton is called Skeleton Animation. This method attaches the vertices of the mesh to the bones of the skeleton, and this process is called skinning. Each vertex has up to four influences of independent bones, to each of that independence is assigned a weight together with the bone, so when the bone moves, the deformation of the other vertices can be weighted by that amount.

This technique is useful to determine the appropriate positioning of a certain bone when the bone's parent moves. This kind of influences of a parent to its child is called Inverse Kinematics (IK). IK is a branch of physics that describes body movements, widely used in games and 3D animations. IK is available in 3D modeling programs such as Daz Studio Pro 4 and 3DS Max.

OGRE does not have IK, only Direct Kinematics (DK). DK is for example, when the arm is moved it is only possible to see the movement on their children, in this case the forearm and hand. This is not a problem since OGRE plays pre-recorded animations.

As previously explained, to export the animations to OGRE it is necessary to indicate manually to the exporter, which are the keyframes of the animations and its

type. The skin option comes from skinning, the animation technique described earlier. It is required to select a bone as the root bone of the skeleton, otherwise the software does not export the skeleton, and since the animations need the skeleton, they are not exported.

The information of the animations is inside a .skeleton file that is connected to a .mesh file (both specific of OGRE).

4.1.2 **Facial Animation**

The facial animation is created through deformations of the mesh without using the skeleton. This happens because the deformations on the face are not deformation are caused by movements of the muscles. This is named vertex animation and has two animations modes: morph or pose.

Pose animation is a technique that stores the offsets to the original vertex data. In each keyframe in the track of the timeline, it is possible to blend one or more poses. Each animation track refers to a single set of geometry. When this animation is played, it is necessary to call these tracks one by one.

Morph animation is a technique that stores the absolute position of the vertices in a certain keyframe and stores another different absolute position in another keyframe and when playing the animation, in runtime, calculates the intermediate positions using interpolation.

As mentioned previously to do the facial animation, Daz Studio Pro 4 has a pose controls menu, so it is only necessary to select a time in the timeline and choose the deformation value in the slider of the selected area.

When that animation is imported in 3DS Max, it is possible to run and see the animation. In addition, it is convenient to run the animation to verify that it was correctly imported from Daz Studio Pro 4. After that, the model is exported to OGRE and it is required to choose if the animation should be exported as *n* poses or morph. As mentioned before, pose animations will have to run one by one and morph animations are called one time for each animation created.

All information related to the vertex animation is in the .mesh file.

To help with the process of modeling and animating a user manual has been produced to, guide step by step the creation of a compatible VH for the ErgoVR system.

This user manual is the Appendices **Erro! A origem da referência não foi ncontrada.**.

4.2 Implemented Module

The ErgoInteract module was designed and developed based on two factors:

- The system has to have the less dependencies as possible with external libraries (**Low Coupling**);
- The responsibilities are well divided between the classes (**High Cohesion**).

These two factors enable the possibility of code reuse and easy comprehension of the same.

There are seven libraries in this system. One of them has been provided, the XSensWrapper, and the other six were developed in the scope of this project: BaseApplication, DemoApp, ErgoInteract, PhysicsInteract, ObjectInteract and SkeletonInformation.

The XSensWrapper, the provided library, is the responsible for the connection and data capture of the sensors.

BaseApplication is a library with the necessary information to run the graphic engine MOGRE, and it was developed based on the tutorials[20] present in the Wiki of OGRE. It deals with everything related with the graphic engine. It creates cameras, viewports, lights, materials, and reads several OGRE configuration files.

SensorInteract receives all the data coming from the library of the sensors and handles that data to be able to associate it to the appropriate bone.

PhysicsInteract is the library that deals with the physics part in the scene.

The ObjectInteract library deals with the interaction of the VH with the objects in the scene using the concept of Smart Objects. As explained in (Magnenat-Thalmann and Thalmann 2004), the idea behind the concept of smart object is that each interactive object contains a complete description of its functionality and interaction capabilities.

In Figure 15, it is possible to see the Module Diagram of the system.

[20] http://www.ogre3d.org/tikiwiki/tiki-index.php?page=Mogre+Tutorials

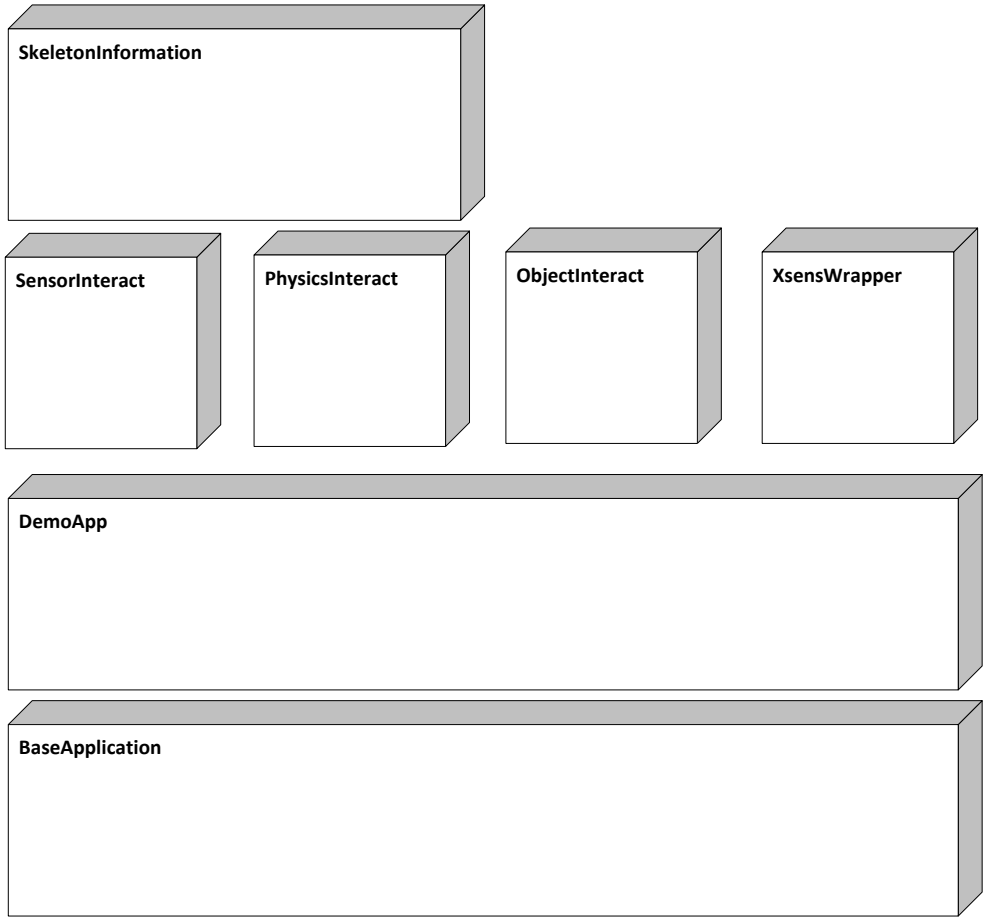


Figure 15 – System Modules

DemoApp is a derived type from the BaseApplication in order to the DemoApp override some of his methods. In addition, this class also creates an instance of the XSensWrapper, SensorInteract, PhysicsInteract and ObjectInteract. The SkeletonInformation is a library used by the SensorInteract and the PhysicsInteract.

The system was developed trying to ensure that the libraries dependencies were injected instead of created inside the libraries. This guarantees low coupling within the system and there is a centralized location on the system responsible to create the dependencies and pass them to the libraries. That is done in the DemoApp initialization code.

4.2.1 **BaseApplication and DemoApp**

These two libraries are responsible for run the OGRE graphics engine and presenting the OGRE graphic interface. As mentioned before, the DemoApp overrides some BaseApplication methods. The decision to create two different libraries for similar responsibilities was taken because the BaseApplication has the minimum and necessary

information to run any MOGRE application outside the ErgoVR system. ErgoVR has already a kind of BaseApplication, but the DemoApp is necessary because there are elements that are only necessary for this system, for example, the VH.

Five classes, Base, Base.Input, DebugOverlay, CameraMan and ZSightHandler compose the BaseApplication (Figure 16) (the complete class diagram is in the Appendix B.1).

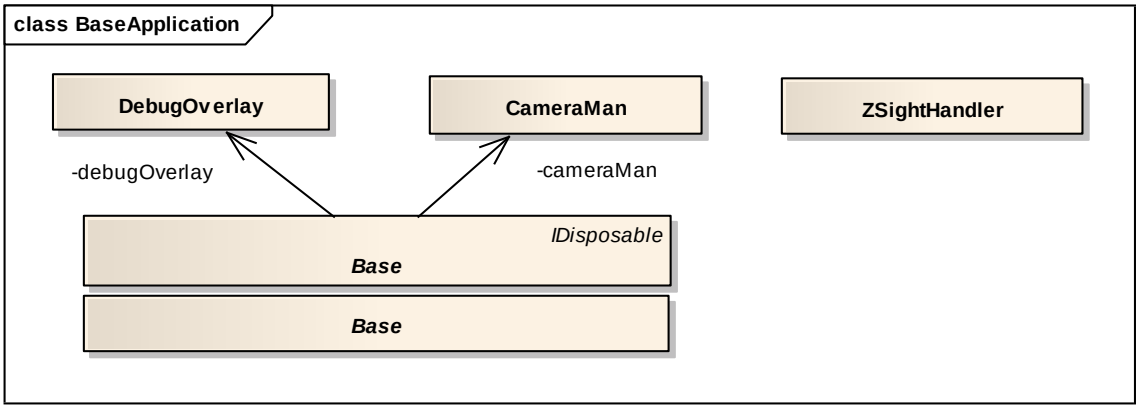


Figure 16 – BaseApplication Class Model

Base and Base.Input are partial classes and they split the information about input and output issues.

The Base partial class is responsible for load the OGRE configuration files that has information about which plugins are loaded, where the resources and media are located and creates the necessary things to present the OGRE graphic interface (camera, viewports, and ambient lights).

The Base.Input partial class deals with the interaction with VE using the keyboard, the mouse and the joystick.

The DebugOverlay is the class that specified the information presented in the graphic interface but is not part of the scene. It gives us information such as, for example, frames per second.

The CameraMan is the responsible for moving the created camera in the VE through the keyboard or mouse.

The ZSightHandler is the class that allows a simpler interaction with the motion tracker that is part of the Sensics ZSight HMD. Team members had previously implemented this class and it was adapted to the ErgoInteract. This sensor allows the participant to move the head giving a more natural point of view of the scene, giving a higher degree of realism to the simulation.

Five classes compose the DemoApp: RunApplication, InitMogre, ObjectInformation, InitLibraries, InitTrackers and PhysicsWorld (Figure 17) (the complete class diagram is in the Appendix B.2).

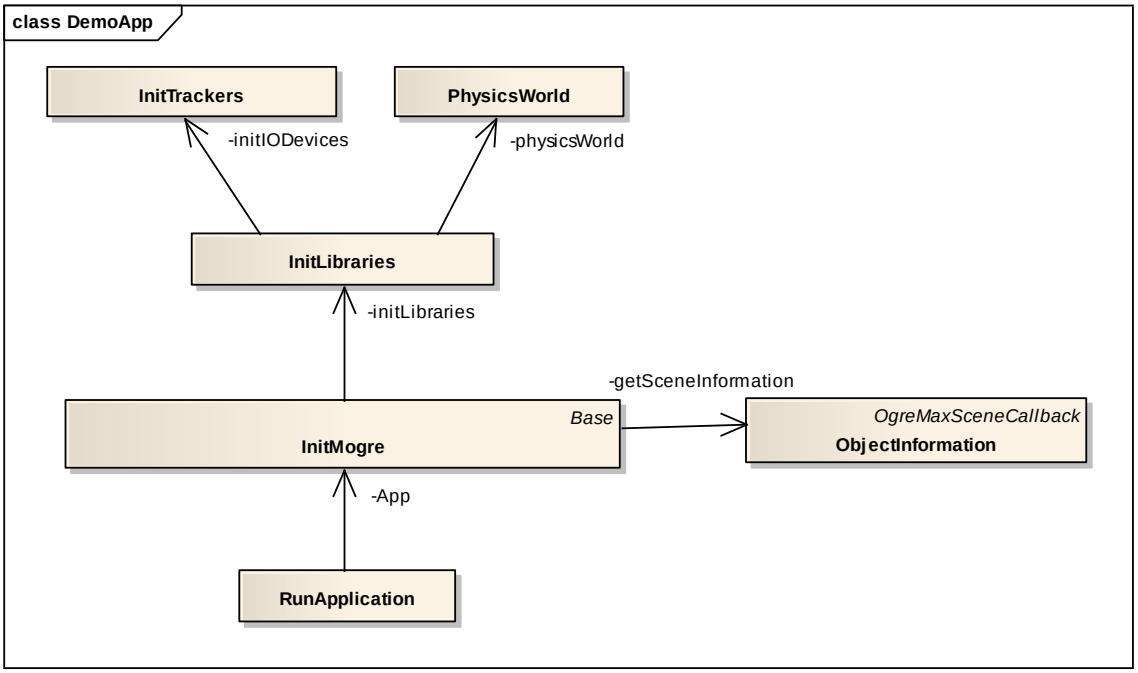


Figure 17 – DemoApp Class Model

RunApplication initializes the InitMogre object. InitMogre is the class that overrides some BaseApplication methods in order to load the configuration file ConfigFile.cfg, to add the VH, create another viewport and create new keyboard shortcuts to move the VH. The VH moves in the VE through keys that make him move forward, backward and rotate. Each key pressed causes the application of a force in a specific direction to the VH, allowing the physics engine to move the VH based in physical properties.

The introduction of physical forces to induce VH movements allows in the future the abstraction of the navigational interfaces used in the ErgoVR system (e.g., joysticks, Nintendo Wii Balance Board, motion trackers, among others).

The InitLibraries is responsible for creating an instance of the libraries of the system, SensorInteract, PhysicsInteract, ObjectInteract and XSensWrapper and get all the trackers used in the simulation (glasses and sensors) through the class InitTrackers.

The ObjectInformation is responsible for reading the information in the .scene file about the Smart Objects, explained in the 4.2.4 .

The test scene loaded by these two libraries is an existing VE provided by the team composed by two adjacent rooms. In the first room, (Figure 18), there are a table

with a lamp, a fire extinguisher, a pillar and the VH (Figure 19**).** Every object can be integrated on the .scene file or can be loaded in the system separately. The VH is loaded separately. The entire objects have their functions, the table is an object to interact by pushing, the lamp and the fire extinguisher are to grab and the pillar is to test the collision situations.

Figure 18 - Scene

Figure 19 - Virtual Human

A small viewport is created and displayed in the lower left corner of the screen. The biggest viewport displays the first person perspective, and the smaller displays the VH.

4.2.2 Sensors Integration - SensorInteract

The initial position of the VH, necessary to calibrate the system, is a T-pose (Figure 19).

The developed module is prepared to receive a maximum of six sensors, three for each arm. It is possible, in the configuration file ConfigFile.cfg, to select the desired arm, choose the number of sensors to activate and in case of using only one sensor it is possible to select which bone it will affect. This last possibility is valuable for testing purposes.

As such, ErgoInteract, accepts the use of two or three sensor in each arm. If is chosen two sensors it moves the arm and the hand, in case of three sensors it moves the three bones, arm, forearm and hand. If both arms are chosen, there is the possibility to choose four or six sensors. The four sensors are placed in the arms and hands, and if it six are chosen, they are for the three segments from both sides. Table 2 helps to know the possibilities to place the sensors.

One Arm	
Two Sensors	Three Sensors
Hand/Shoulder	Arm/Forearm/Shoulder

Table 2 – Possible Number of Sensors in One Arm

Both Arms	
Four Sensors	Six Sensors
Hand/Shoulder	Arm/Forearm/ Shoulder

Table 3 - Possible Number of Sensors in Both Arms

After the number of sensors is chosen, the sensors can be attached to the participant's body. To help placing the sensors there is the XM-B User Manual and a Complementary User Manual for the Xsens Motion Sensors that is presented in the Appendices.

The library SensorInteract is the responsible for the sensors and has three classes, SensorInteraction, UpdatePositionBones and SkeletonConfigFile (Figure 20) (the complete class diagram is in the Appendix B.3).

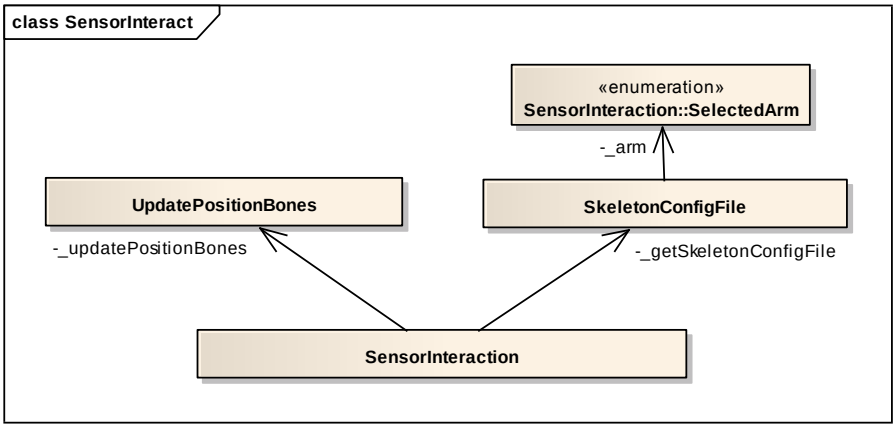


Figure 20 - SensorInteract Class Model

The class SkeletonConfigFile is the class that read the section of the sensors in the configuration file named ConfigFile.cfg and saves the number of sensors used, the chosen arm and the selected arm.

The orientations of the VH skeleton are given through the articulations of the shoulder, elbow and wrist so it is necessary to place the sensor near these zones. The sensor gives us the 3D orientation, the 3D acceleration, 3D rate of turn and 3D earth-magnetic field data. The orientation is the only information used.

The SensorInteract class is the class in charge of reading the configuration file ConfigFile.cfg, creating the class UpdatePositionBones with the right attributes and send the data from the sensors to that class.

The best way to work with the upper limbs movements is with yaw, pitch and roll. The sensors[21] use the Cartesian System[22]. The roll is the rotation about *X* axis, the pitch is the rotation about the *Y* axis and the yaw is the rotation about the Z axis.

Sensors and VH's skeleton use right-handed coordinate systems. However, sensors have the *Z* axis pointing up (Figure 21) while the skeleton has the *Y* axis pointing up (Figure 22).

As we can see from the two mentioned figures (Figure 21 and Figure 22), the coordinate system is different in the sensor and the bone, what causes different movements.

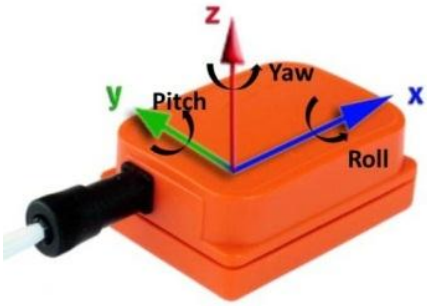


Figure 21 - Sensor Coordinate System

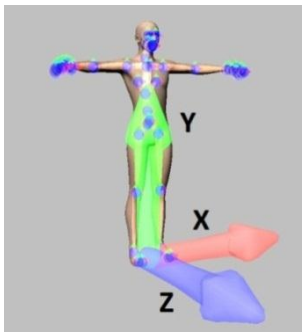


Figure 22 - Virtual Human Coordinate System

Note that the sensors are placed along the arm of the participant in different positions, which alters the orientation of the coordinate system. Figure 23 shows the correct placement of the sensors; Appendicesx contains a complementary user manual for XSens with more details about this question.

[21] http://eris.liralab.it/viki/images/8/82/XsensMtx.pdf

[22]http://en.wikipedia.org/wiki/Euler_angles

Figure 23 - Sensors Placed in the Upper Limbs

Therefore, due to the differences in the orientations of the coordinate systems involved, data from sensors are subjected to referential transformations before being applied to the corresponding bones. Figure 24, Figure 25 and Figure 26 show the original Euler angles on the VH bones and the Euler angles in the sensors, and doing a comparison it is possible to note the referential transformation. However, what is most important after that comparison is to see if the Euler Angles are equal in both elements (bones and sensors) despite the coordinates system being different.

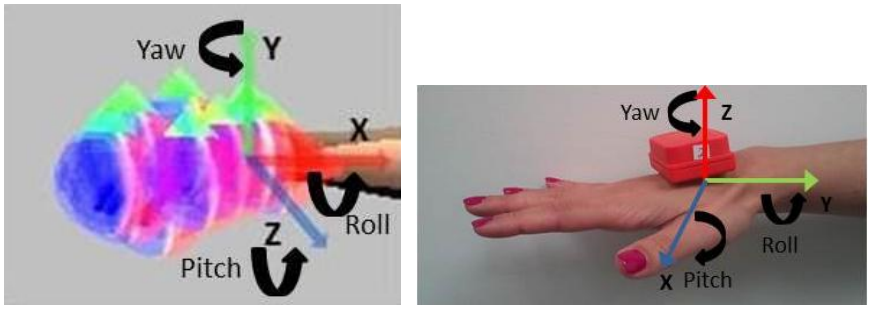


Figure 24 - VH and Sensors Euler Angles at Hands

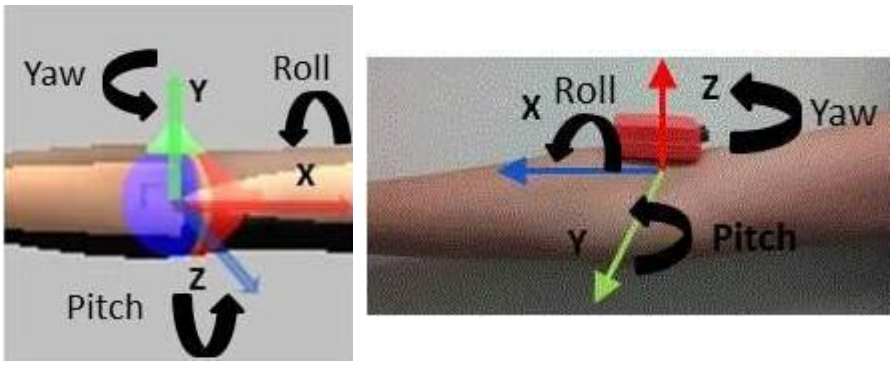


Figure 25 - VH and Sensors Euler Angles at Forearm

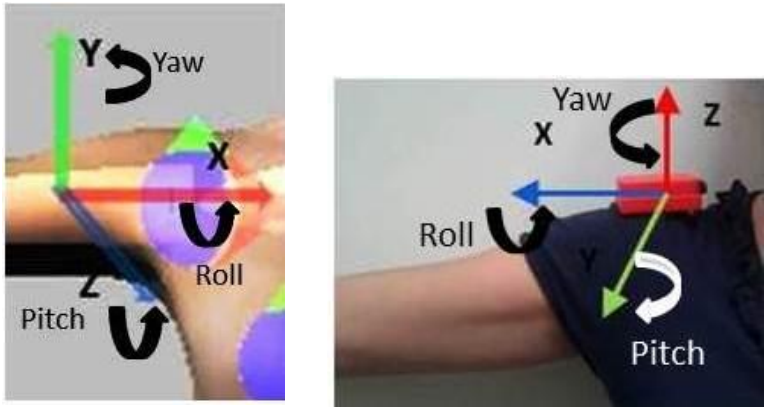


Figure 26 - VH and Sensors Euler Angles at Shoulder

After comparing, it is possible to conclude that in the hand, all axes are changed, the *Y* is transformed in to the *Z*, the *X* in to the *Y* and the *Z* in to the *X* (Figure 24). However, it is possible to see that the unique Euler angles that change are the pitch and roll. The yaw is made in different axis but in the same direction. In the elbow and shoulder, we can see that the axis *Y* and *Z* are changed in the sensor (Figure 25 and Figure 26), but the Euler angles are at the right place. Observe that, the Euler angle roll in the *X* axis is positive in the VH, but negative in the sensor.

Besides, not every bone has the three movements and there are some restrictions in some degrees. The hands have the three movements, roll, pitch and yaw. In the elbow the pitch and yaw are the same movement because the internal part of the elbow needs to rotate to do both movements. Considering that the calibration position is with the internal part of the elbow turned front, it is only considered the yaw. Therefore, the roll and pitch in the elbow are not passed to the bone in order to not reproduce that movement. The shoulder does not have roll.

The UpdatePositionBones is the class that makes this conversion and places the right data in the right bone (Code Sample 1- Example of Euler Angles Conversion).

```
1  Matrix3 matrix = mogreQuatSensor.ToRotationMatrix();
2  Radian yawSensor, pitchSensor, rollSensor;
3  matrix.ToEulerAnglesZYX(out yawSensor, out pitchSensor, out rollSensor);
4
5  #region 3 Sensors
6              if (numberSensors == 3)
7              {
8                 if (indexSensor == 0) //UP ARM
9                 {
10                     yawUpArm = yawSensor;
11                    pitchUpArm = pitchSensor;
12                     rollUpArm = - rollSensor;
13
14                     matrix.FromEulerAnglesYXZ(yawSensor, pitchSensor, 0));
15                     mogreQuatSensor.FromRotationMatrix(matrix);
16                     return mogreQuatSensor;
17                }
18                if (indexSensor == 1) //LO ARM
19                {
20                    yawLoArm = yawSensor;
21                    pitchLoArm = rollSensor;
22
23                   yawSensor = yawSensor - yawUpArm;
24                    pitchSensor = pitchSensor - pitchUpArm;
25
26                    matrix.FromEulerAnglesYXZ(yawSensor, 0, -pitchSensor);
27
28                    mogreQuatSensor.FromRotationMatrix(matrix);
29                    return mogreQuatSensor;
30                }
31                if (indexSensor == 2) //HAND
32                {
33                    yawSensor = yawSensor - yawLoArm;
34                    pitchSensor = pitchSensor - rollLoArm;
35                    rollSensor = boneHandLOrientation - pitchLoArmL;
36
37                    matrix.FromEulerAnglesYXZ(yawSensor,-rollSensor , pitchSensor);
38
39                    mogreQuatSensor.FromRotationMatrix(matrix);
40                    return mogreQuatSensor;
41                }
42             }
43 #endregion
```

Code Sample 1- Example of Euler Angles Conversion

The data comes from the sensor as a quaternion. To work with the Euler angles it is necessary to transform that quaternion into a rotation matrix (line 1). From that rotation matrix, it is possible to take the Euler angles (line 3).

After that it is necessary to verify from which sensor is the data.

The first data comes from the sensor ID 0, it is for deposit in the shoulder. As we talk previously in the shoulder, we can see that the axis *Y* and *Z* are changed in the sensor (Figure 26), but the Euler angles are at the right place so it is not necessary to change the coordinates (line 14). The roll is 0 because as mentioned that movement does not exist.

The second data comes from the sensor ID 1, that information it is for the elbow . As with the shoulder, the elbow has Euler angles at the right place so it is not necessary to change the coordinates (Figure 25) (line 26). The pitch is 0 because as mentioned that movement does not exist.

The third data comes from the sensor ID 2, that information it is for the hand.

In the hand, the Euler angles that change are pitch and roll (Figure 24). That way it is necessary to change the values when the matrix is being created (line 37), but in this case all the values are passed because the hand has all the movements.

There is another question to have in mind. When it is passed, for example, a roll movement to the shoulder it needed to be considered that the elbow is going to suffer also that roll movement. Therefore, to correct that problem is necessary to subtract the value of the shoulders on the value of the elbow and in the values of the elbow is necessary to subtract from the hand. As is it possible to see in the Code Sample 1- Example of Euler Angles Conversion in each if statement is saved the values (line 10, 11 and 12) and those respective values are subtract in the next if statement (line 23 and 24).

4.2.3 **Physics Engine - PhysicsInteract**

In order to have more realism in the VE it is necessary to have a credible physical behavior in the VH and all object presented in the scene.

If an object trespasses through walls, tables and other objects in the scene, for example, it is obvious that the sense of realism (and immersion) is lost.

It deals with gravity, mass, forces, among others. As referred in section 3.2.3 , the physics engine that was used is Newton Game Dynamics[23]. It is an open-source library, which works in OGRE and has a wrapper to MOGRE. This physics engine has collision

[23] http://newtondynamics.com/forum/newton.php

detection and dynamics behaviors. The PhysicsInteract library is divided in eight classes, (Figure 27) (the complete class diagram is in the Appendix B.4). The classes are ActionEventArgs, Actor, InteractionActor, MogreNewtHandler, NormalCamera, VRCamera, WorldCallback and TriggerCallback.

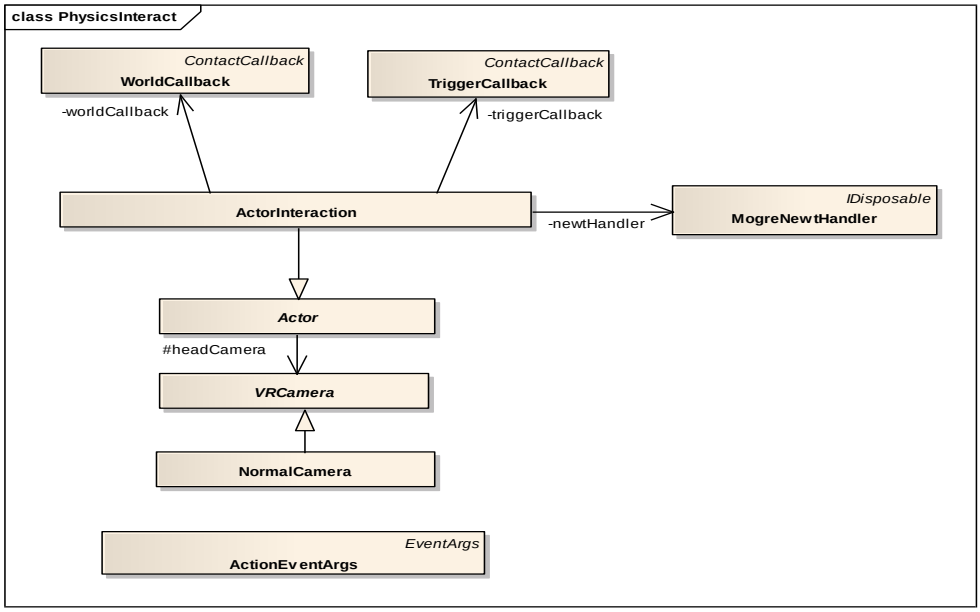


Figure 27 - PhysicsInteract Class Model

The InteractionActor is the main class that creates the physics bodies and places them with the right size in the right position using the skeleton file to access to that information. It is not necessary a very detailed physic definition in the VHs body, only it is necessary in the arms. That way, the VHs body has one physics shape and for the arms exist two for each arm in order to have more realism in the physics behavior (Figure 28).

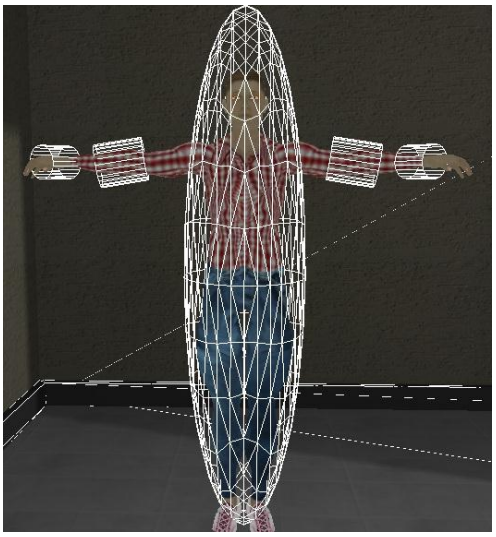


Figure 28 - Body's Physics

It is not necessary to place physic bodies on the shoulder, because, as it is possible to infer from this figure, it is not likely to touch only with the shoulder in an object.

The MogreNewtHandler is the class that initializes the MogreNewt library. In order to have physics behavior in the ActorInteraction it is necessary to create a physics world that is also created in this class with a physics visual debugger. The debugger of MogreNewt shows in the VE the physics bodies (Figure 29).

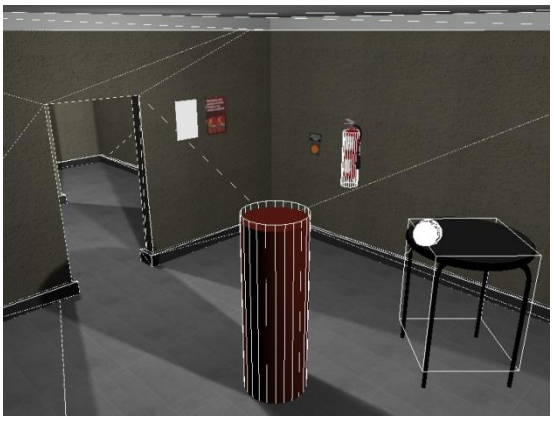


Figure 29 - Physics Objects in the Scene

The class Callback is divided in two parts: WorldCallback and TriggerCallback. WorldCallback treats the collision between physics bodies that do not have physical effects when they collide; TriggerCallback treats the collision between the hand and all the smart objects.

The class ActionEventArgs defines the information that should be sent to the receiver of the event notification when one of the callbacks is activated.

The class Actor is responsible to call the class InteractionActor to give the initial position of the scene node that has the VH.

NormalCamera and VRCamera are two classes of the ErgoVR system and they are used for preparing the system to use an HMD (Head Mounted Display) or a projection in a screen.

4.2.4 Interaction with Objects - ObjectInteract

The ObjectInteract library has three classes: ObjectInteraction, PlayAction and Trigger (Figure 30)(the complete class diagram is in the Appendix B.5).

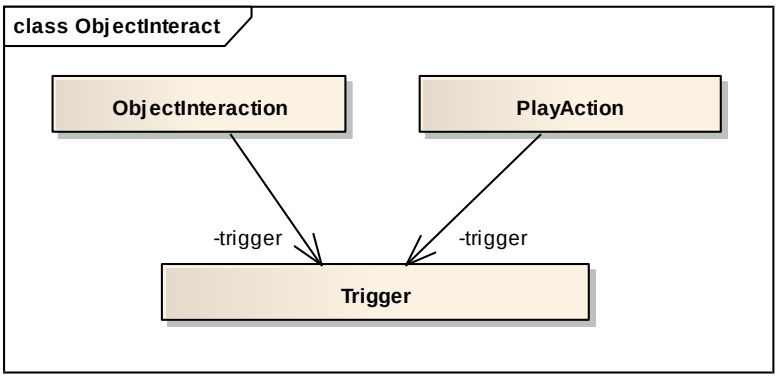


Figure 30 - ObjectInteract Class Model

The interaction with the objects in the VE can be accomplished with one or both hands depending on the type of the object and the action to be performed.

All the objects in the VE are inside the .scene file. Inside this xml file, exists a node named Interaction that contains a field called userData, where it is possible to add customized information about that object, changing it into a Smart Object. An example of the userData field:

```
<node name="Interaction">
        <userData>
                <![CDATA[
                name: lamp, action: grab, NumNecessaryHands: 1, drop: dropAreaLamp
                name: button, action: press, ID: 1;2;3, NumNecessaryHands: 1
                name: fireExtinguisher, action: grab, NumNecessaryHands: 1, drop:
      dropAreaExtinguisher
                ]]>
      </userData>
      <position x="0" y="0" z="0" />
      <scale x="1" y="1" z="1" />
      <rotation qx="0" qy="0" qz="0" qw="1" />
</node>
```

Code Sample 2 - UserData Example

That information has a field *name* that is the name of the object, the field *action* that contains the type of action that is possible to do with it (grab, drop, press or push), the field *NumNecessaryHands* that has the number of hands that is necessary to that interaction. Regarding the action of pressing, in the information there is also an array of elements named *ID* that enumerates the type of behavior of the pressed object. For the object that has the action grab a field *drop* is defined to give the name of the area were is to bem drop the object. *For* example, if we press a button we can change the material to become brighter and change the position of the button to give the sensation that is pressed.

This information presented in the .scene file is accessed and treated in the class ObjectInteraction and, afterwards, it is saved in an object named Trigger. All the smart objects that are read from the scene are saved in a list of triggers.

As explained in section 4.2.3 there is an event named Callback that sends a message when two physics objects have contact with each other. One of the callbacks defined is to be executed when there is interaction between one hand and one object in the scene. When that event happens, one condition verifies if the object touched by the hand is a smart object in the trigger list and, if it is, then the action related to that object is performed.

If the action is grabbing, when the two bodies have physical contact the object position is changed to the hands position.

An object that was grabbed, has to be dropped in some specific area at some point, depending on the task. For example, if the participant wants to grab a lamp, he has a specific place to drop it. To be possible to drop the object it was created an invisible box that limits a zone that specifies the drop location. When the object passes in that zone and stays longer than a specific time threshold (e.g., two seconds), the position of the object is no longer attached to the hand and stays in that zone.

If the action is to push or pull, for example, a door, is necessary to change the position of the door or run an animation to give a dynamic sensation.

4.2.5 **Skeleton Information**

The information about the skeleton is necessary on two of the developed libraries: SensorInteract and PhysicsInteract. The SkeletonInformation library works with the .skeleton file of the mesh, which specifies the bones positions and their hierarchy.

This information is necessary to reproduce the movements of the sensors on the VH's bones, to place the physics bodies at the right position, and to calculate the exactly size of each bone.

Only one class composes this library, InformationSkeleton (Figure 31) (the complete class diagram is in the Appendix B.6).

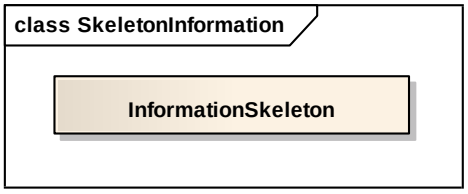


Figure 31- SkeletonInformation Class Diagram

The first section of this chapter described issues related with modeling and animation of VH models compatible to the ErgoVR. The second section was dedicated to the module ErgoInteract developed in this project giving a comprehensive explanation of each library.

Chapter 5

Conclusion and Future Work

Virtual Reality is used in many areas nowadays. For instance, it can be used for treating phobias and to construct virtual realities that demand higher cost or that are dangerous in real life. ErgoVR supports VE that can simulate dangerous situations. The main purpose of this project was to enable the possibility of representing the upper body movements of the participant on the HV, so that it would be possible for the participant to interact with objects in the VE, performing natural gestures using their own hands, such as grabbing, dropping, pushing and pressing.

Four VH models were modeled and animated to inhabit any VE loaded by ErgoVR. These models are capable of reflecting the natural movements of the participant using sensors attached to the upper limbs of the participant.

It was necessary to develop a library that implement physics attributes in the VE to give more realism to the simulation.

It was been created a gesture library to implement smart objects in the VE. This library and the previous one supports the interaction activities of the participant with objects in the VE.

The last and complementary goal was to add support in the ErgoInteract module for the Data Glove. It was not possible to reach this goal during project duration but it is considered as future work.

To create a VH, it was necessary to work with two tools, one for modeling and the other to export to the OGRE format. However, during the export process, many tools lost fundamental model information. Therefore, it was necessary to test different modeling tools with distinct file formats. This work stage consumed a considerable amount of time but at the end it was a good investment because now it is possible to have a VH to represent the participant and three more VH models to include in any VE.

The library of the sensors took less time than the expected. However, some difficulties appeared related with Euler angles transformations and anatomy questions about the movements of the hand, elbow and shoulder. The last difficulty was easily overcome due to the presence of an Ergonomist Professor in the team.

The physics library took more time than expected. The physics engine used is an incomplete wrapper for the latest version of the Newton Game Dynamics, which caused some difficulties in understanding the complete features of the wrapper (there was no documentation) and which features of the physics engine are truly implemented. Despite these problems, the wrapper used seemed the most complete to use with MOGRE. The positive side is that with these problems it was possible to learn more about the library and the limitations of the wrapper.

The gesture library took the predicted time. At this moment, it is possible to grab, drop, push and press objects in the scene.

The participation in the scientific paper, "Using space exploration matrices to evaluate interaction with Virtual Environments" (L. Teixeira et al., 2012) was very useful. This collaboration brings more knowledge in several areas. Also, it was important to get acquainted with the ErgoVR system and the team.

In this project, the high cost associated to the materials, means that not everyone can buy the motion trackers used (XSens XBus Kit). In addition, there are some possible side effects with the use of VR, mainly simulator sickness.

The advantage to the research´s area is that this work use VR and, as referred above, is very useful to use in many areas.

The advantage to me was the fact of that with this project knowledge of some areas such as physics, ergonomics, anatomy, and informatics were acquired and remembered. Another advantage was the fact of knowing different people of different areas that brings to me more information and culture.

Despite all the problems, like in every project, there is a functional prototype. This prototype can be improved in the future with the implementation of support for a Data Glove and a gesture recognition library. The Data Glove because it can capture the finger flexion allowing more definition in the hand movements, and the gesture recognition library because it offers more possibilities for gestures. For example, to

associate the movement of the hand from the left to the right in a straight line to a slide action or, another example, to associate the action of staying more than *x* seconds with the hand stopped to select something like an option in a menu.

Another future work that can be implemented in the ErgoVR system are reactive intelligent agents. In order to introduce human behavior in the simulation and get more natural behavior by the participant giving more immersion also. Additionally, to be possible to have more kind of studies and simulations in the ErgoVR.

At the end, this project is a positive point to the ErgoLAB. Now the simulations are richer in terms of interactivity because it is possible, besides navigating to grabbing, dropping, pushing and pressing in the VE. This allows for more complex environments and different kind of studies to be performed with more focus on interaction.

Acronyms

Terms	Definition
VE	Virtual Environment
CIDA	Chaotic Interaction Device Abstraction
DOF	Degree of freedom – 6 degrees in total, 3 of rotation and 3 of translation
FCT	Fundação para a Ciência e Tecnologia
GNSS	Global Navigation Satellite Systems
GPS	Global Positioning System
HDM	Head-Mounted Display
VH	Virtual Human
MEMS	Micro-Electro-Mechanical Systems
MOGRE	Managed Object-oriented Graphics Rendering Engine
PEI	Projeto Engenharia Informática
AR	Augmented Reality
VR	Virtual Reality
VS	Microsoft Visual Studio

References

Bergmann, Jeroen HM, Ruth E Mayagoitia, and Ian CH Smith. 2009. "A portable system for collecting anatomical joint angles during stair ascent: a comparison with an optical tracking device." *Dynamic medicine DM* 8: 3. http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2684094&tool=pmcentrez&rendertype=abstract.

Brand, Richard A. 2008. "Elbow and Shoulder Lesions of Baseball Players." *Clinical Orthopaedics and Related Research* 466(1): 62-73.

Buchmann, Volkert et al. 2004. "FingARtips: gesture based direct manipulation in Augmented Reality." In *Virtual Reality*, ACM, p. 212-221. http://portal.acm.org/citation.cfm?id=988871.

Burdea, Grigore, and Philippe Coiffet. 2003. *Virtual reality technology, Volume 1*. Wiley-IEEE.

Cutti, Andrea Giovanni et al. 2010. "'Outwalk': a protocol for clinical gait analysis based on inertial and magnetic sensors." *Medical & Biological Engineering & Computing* 48(1): 17-25. http://www.ncbi.nlm.nih.gov/pubmed/19911214.

Duarte, Emília. 2004. "Informação de segurança. Avaliação da eficácia de sinais pictóricos de segurança." Faculdade de Motricidade Humana. Universidade Técnica de Lisboa.

Gutierrez, Mario, F Vexo, and Daniel Thalmann. 2008. *Stepping into Virtual Reality*. 1st ed. Santa Clara, CA, USA: Springer-Verlag TELOS.

Heilig, M L. 1962. "Sensorama simulator." *US Patent 3050870*.

Heilig, Morton L. 1969. "Experience Theater." *US Patent 3469837*.

Kozlov, Michail D., and Mark K. Johansen. 2010. "Real Behavior in Virtual Environments: Psychology Experiments in a Simple Virtual-Reality Paradigm Using Video Games."

Magnenat-Thalmann, N, and Daniel Thalmann. 2004. *Handbook of Virtual Humans*. Wiley.

Norman, Donald A. 2002. 1-3. New York Basic Books Chapters 257 *The Design of Everyday Things*. Basic Books. http://www.amazon.com/dp/0465067107.

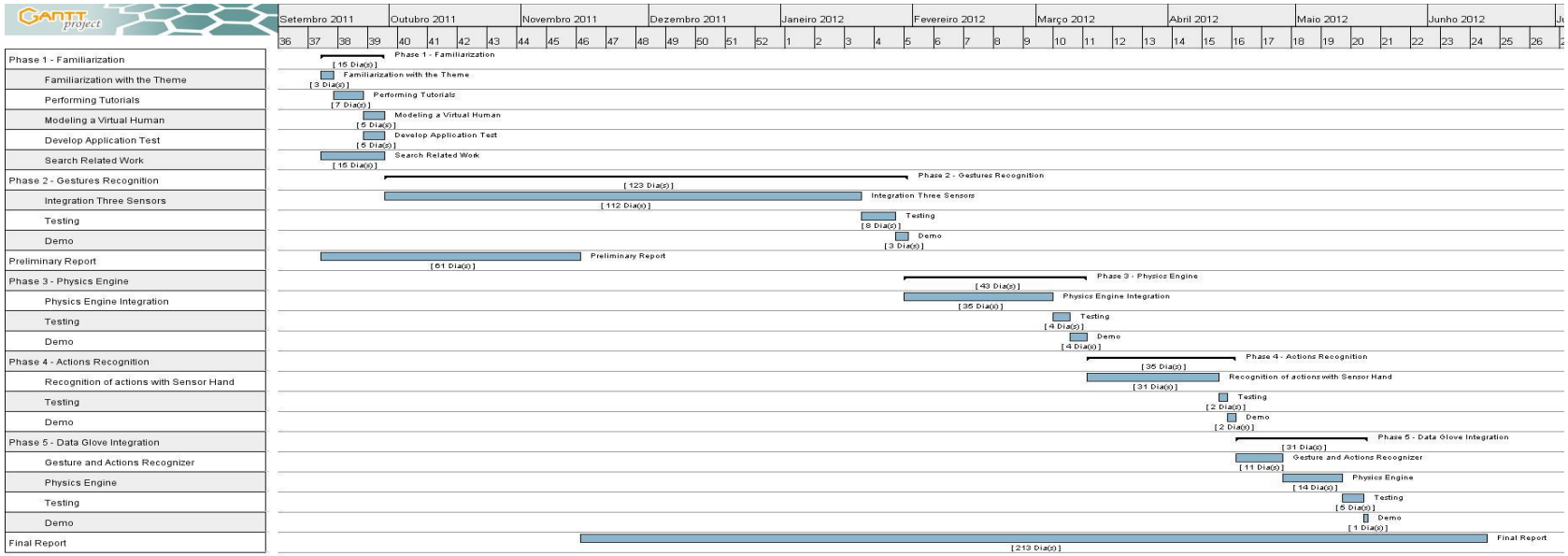
Teixeira, João Marcelo et al. 2006. "GeFighters: an Experiment for Gesture-based Interaction Analysis in a Fighting Game."

Teixeira, Luís et al. 2010. International Symposium on Occupational Safety and Hygene SHO 2010 505-509 *ErgoVR - An approach for automatic data collection for Ergonomics in Design studies*. eds. P Arezes et al. Portuguese Soc Occupational Safety & Hygiene. http://www.sposho.pt/sho2010/proceedings2010.pdf.

Tori, Romero, Claudio Kirner, and Robson Siscoutto. 2006. Virtual Reality *Fundamentos e Tecnologia de Realidade Virtual e Aumentada*. ed. Romero Tori. SBC. http://romerotori.org/Sumario-Livro-RV2006.pdf.

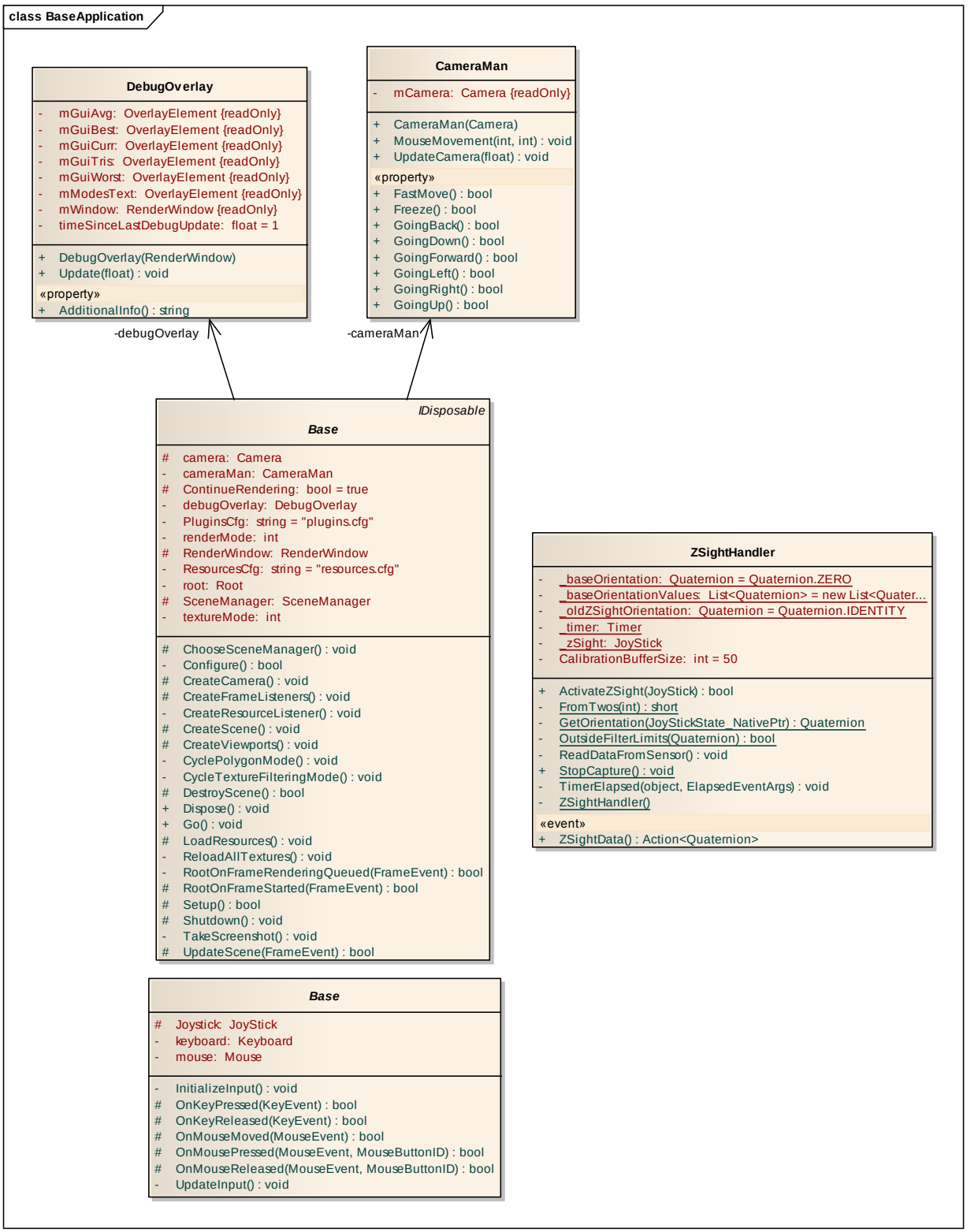
Appendices

A. Gantt Chart

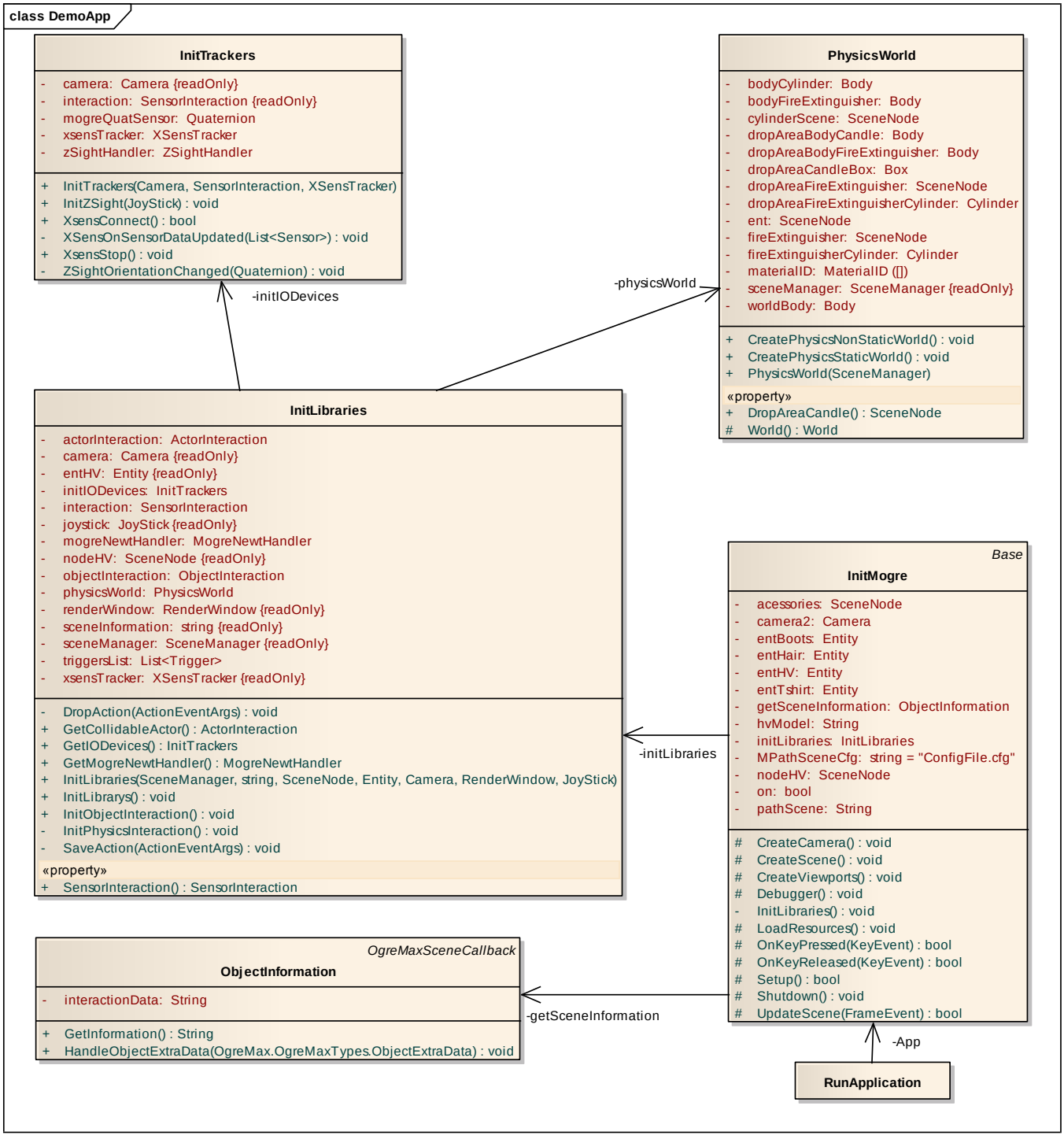


B. **Class Diagram**

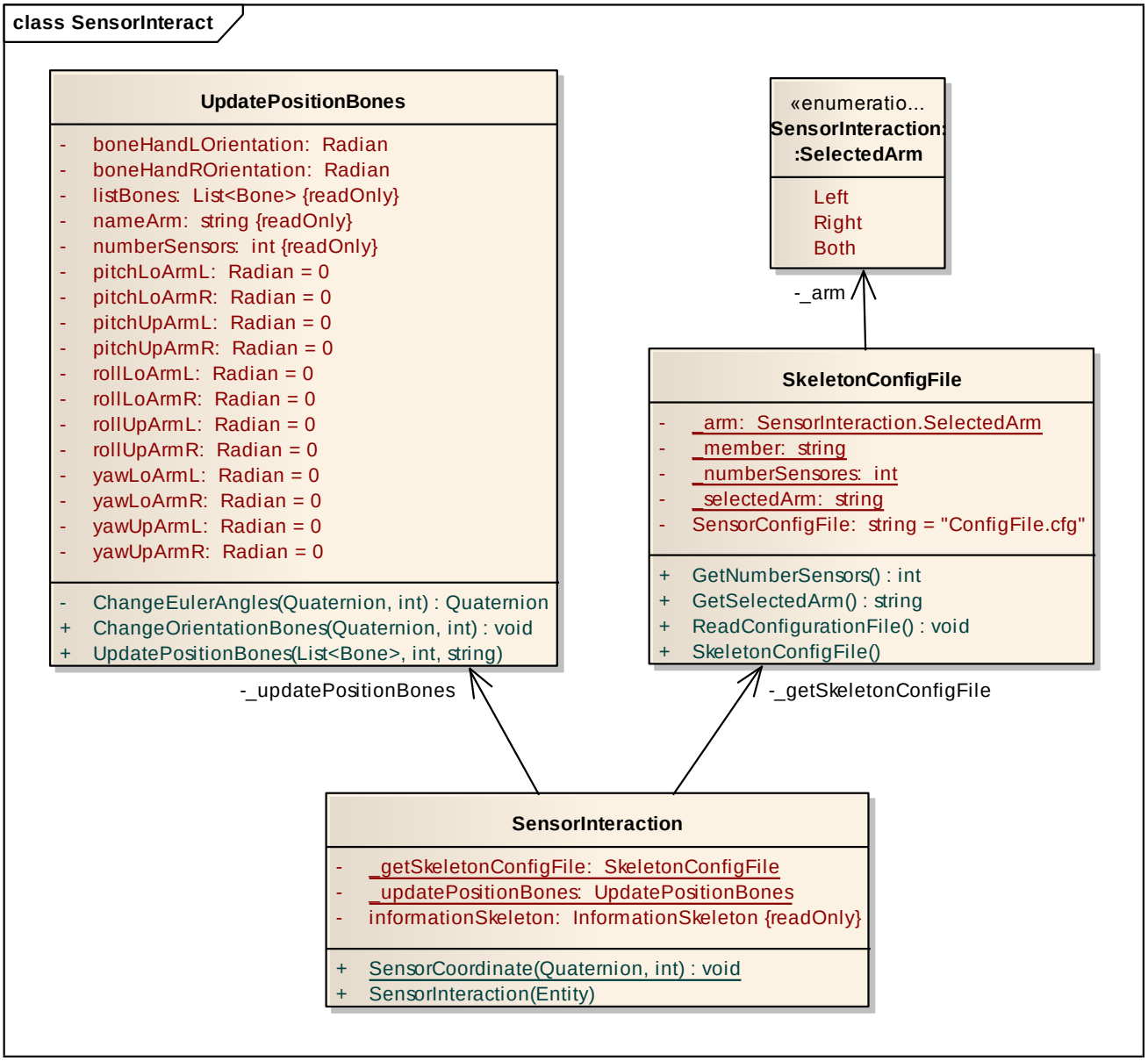
B.1 BaseApplication



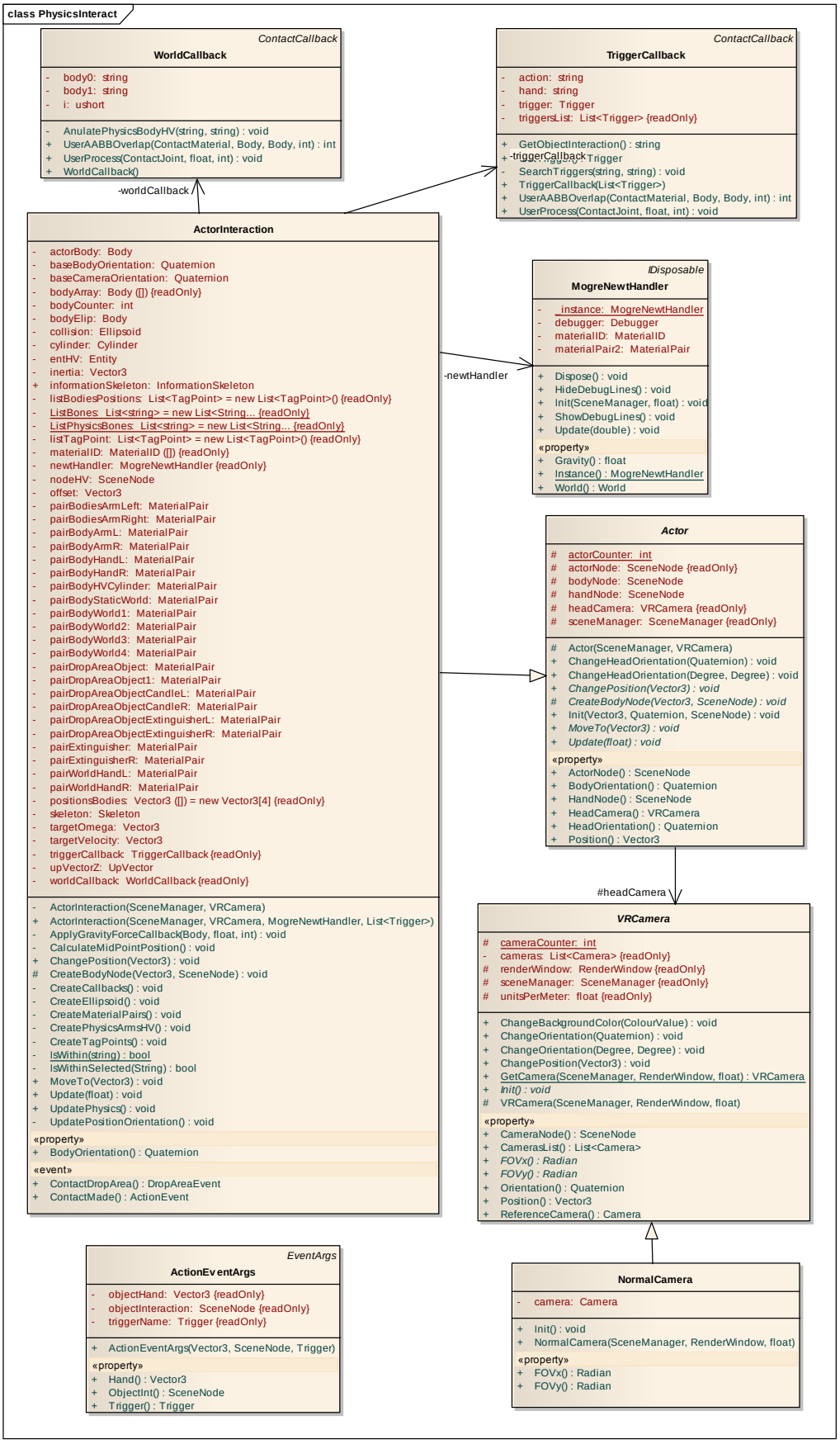
B.2 DemoApp


class DemoApp
InitTrackers
- camera: Camera {readOnly}
- interaction: SensorInteraction {readOnly}
- mogreQuatSensor: Quaternion
- xsensTracker: XSensTracker
- zSightHandler: ZSightHandler
+ InitTrackers(Camera, SensorInteraction, XSensTracker)
+ InitZSight(JoyStick) : void
+ XsensConnect() : bool
- XSensOnSensorDataUpdated(List<Sensor>) : void
+ XsensStop() : void
- ZSightOrientationChanged(Quaternion) : void
-initIODevices
PhysicsWorld
- bodyCylinder: Body
- bodyFireExtinguisher: Body
- cylinderScene: SceneNode
- dropAreaBodyCandle: Body
- dropAreaBodyFireExtinguisher: Body
- dropAreaCandleBox: Box
- dropAreaFireExtinguisher: SceneNode
- dropAreaFireExtinguisherCylinder: Cylinder
- ent: SceneNode
- fireExtinguisher: SceneNode
- fireExtinguisherCylinder: Cylinder
- materialID: MaterialID ([])
- sceneManager: SceneManager {readOnly}
- worldBody: Body
+ CreatePhysicsNonStaticWorld() : void
+ CreatePhysicsStaticWorld() : void
+ PhysicsWorld(SceneManager)
«property»
+ DropAreaCandle() : SceneNode
# World() : World
-physicsWorld
InitLibraries
- actorInteraction: ActorInteraction
- camera: Camera {readOnly}
- entHV: Entity {readOnly}
- initIODevices: InitTrackers
- interaction: SensorInteraction
- joystick: JoyStick {readOnly}
- mogreNewtHandler: MogreNewtHandler
- nodeHV: SceneNode {readOnly}
- objectInteraction: ObjectInteraction
- physicsWorld: PhysicsWorld
- renderWindow: RenderWindow {readOnly}
- sceneInformation: string {readOnly}
- sceneManager: SceneManager {readOnly}
- triggersList: List<Trigger>
- xsensTracker: XSensTracker {readOnly}
- DropAction(ActionEventArgs) : void
+ GetCollidableActor() : ActorInteraction
+ GetIODevices() : InitTrackers
+ GetMogreNewtHandler() : MogreNewtHandler
+ InitLibraries(SceneManager, string, SceneNode, Entity, Camera, RenderWindow, JoyStick)
+ InitLibrarys() : void
+ InitObjectInteraction() : void
- InitPhysicsInteraction() : void
- SaveAction(ActionEventArgs) : void
«property»
+ SensorInteraction() : SensorInteraction
-initLibraries
Base
InitMogre
- acessories: SceneNode
- camera2: Camera
- entBoots: Entity
- entHair: Entity
- entHV: Entity
- entTshirt: Entity
- getSceneInformation: ObjectInformation
- hvModel: String
- initLibraries: InitLibraries
- MPathSceneCfg: string = "ConfigFile.cfg"
- nodeHV: SceneNode
- on: bool
- pathScene: String
CreateCamera() : void
CreateScene() : void
CreateViewports() : void
Debugger() : void
- InitLibraries() : void
LoadResources() : void
OnKeyPressed(KeyEvent) : bool
OnKeyReleased(KeyEvent) : bool
Setup() : bool
Shutdown() : void
UpdateScene(FrameEvent) : bool
OgreMaxSceneCallback
ObjectInformation
- interactionData: String
+ GetInformation() : String
+ HandleObjectExtraData(OgreMax.OgreMaxTypes.ObjectExtraData) : void
-getSceneInformation
-App
RunApplication

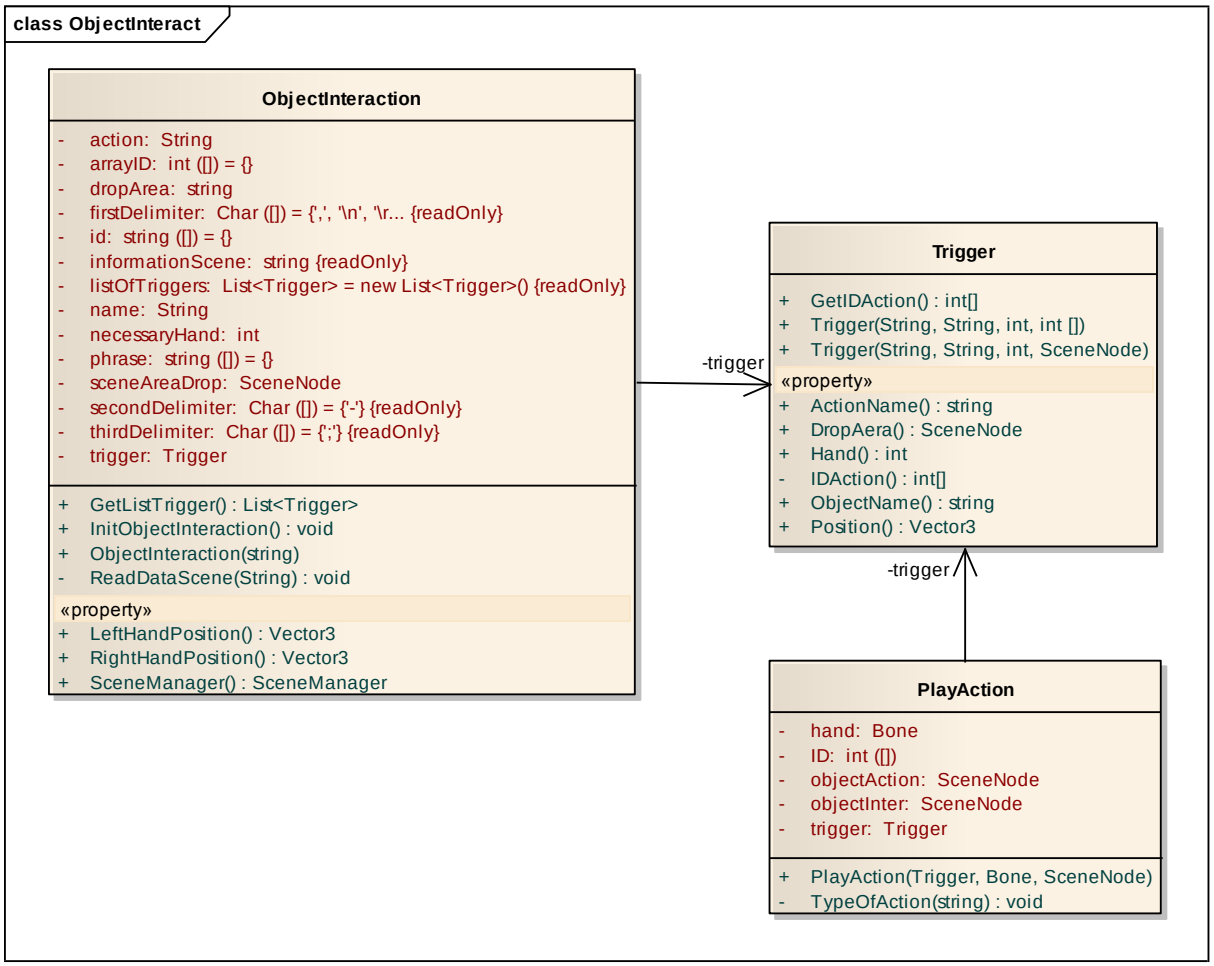
B.3 SensorInteract



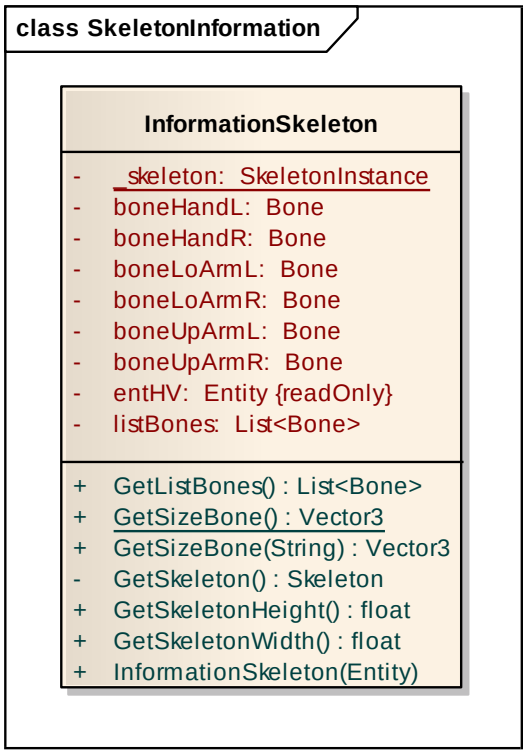
B.4 PhysicsInteract



B.5 ObjectInteract



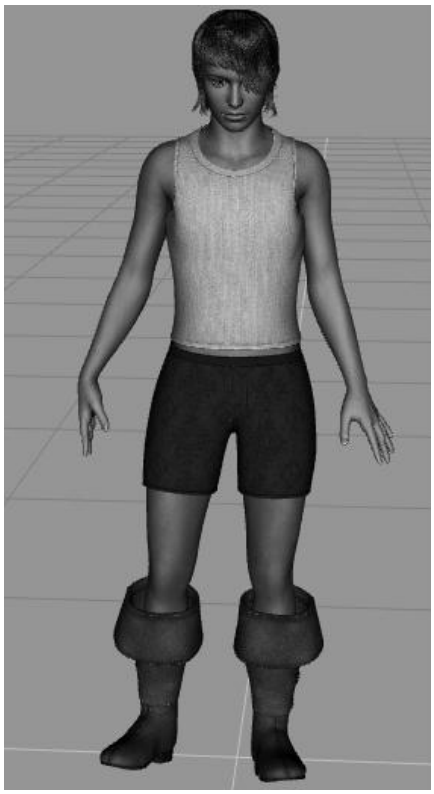
B.6 SkeletonInformation



C. User Manual to Create VH

User Manual

Create a VH for the ErgoVR system



Contents

List of Figures

1. Introduction

This manual has the goal to help modeling a virtual model (an VH) to use in the ErgoVR system. This VH has skeleton, clothes, hair and facial and body animations.

2. Requirements

To be possible to complete this tutorial it will be necessary to have installed the following software:

- DAZ Studio 4.0 Pro;
- 3DS Max 2009;
- Autodesk FBX exporter v2011.3;
- OgreMax exporter v2.1.2

3. Overall workflow

To create an VH for the ErgoVR system it is required an VH with clothes, hair and animation to be the more credible as possible. To do that, Daz Studio 4.0 Pro is used which has pre-made models with skeleton, pre-made animations and clothes and hair to be used in the VH simply by drag-and-drop them into the model. However, Daz Studio 4.0 Pro cannot export to the OGRE format. So an intermediate step is needed, on which the model is exported to the FBX format and imported into 3ds Max 2009, because this tool has an exporter to the OGRE format. In 3ds Max 2009 it is only necessary to separate and save the animations that are going to be exported to the OGRE format.

4. Modeling the VH

VH

Step 1: Open DAZ Studio 4.0 Pro. In the main window appears immediately a human model with incorporated skeleton (Figure 1).

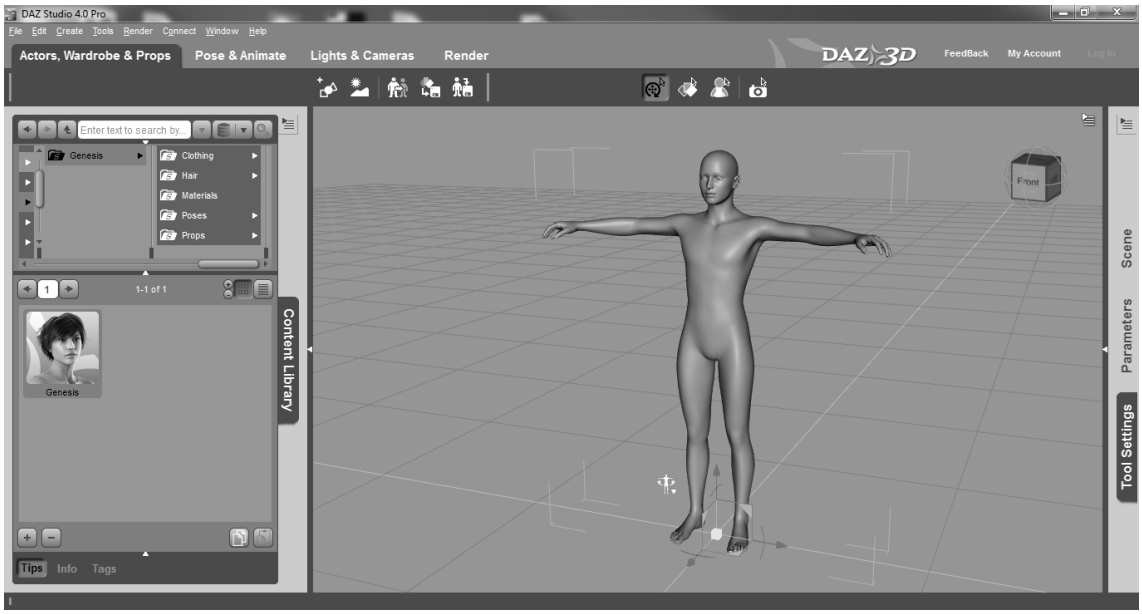


Figure 1 – Main Window

Step 2: To assign a skin texture select the label Actors, Wardrobe & Props, and next, inside the folder Materials choose the intended material (Figure 2).

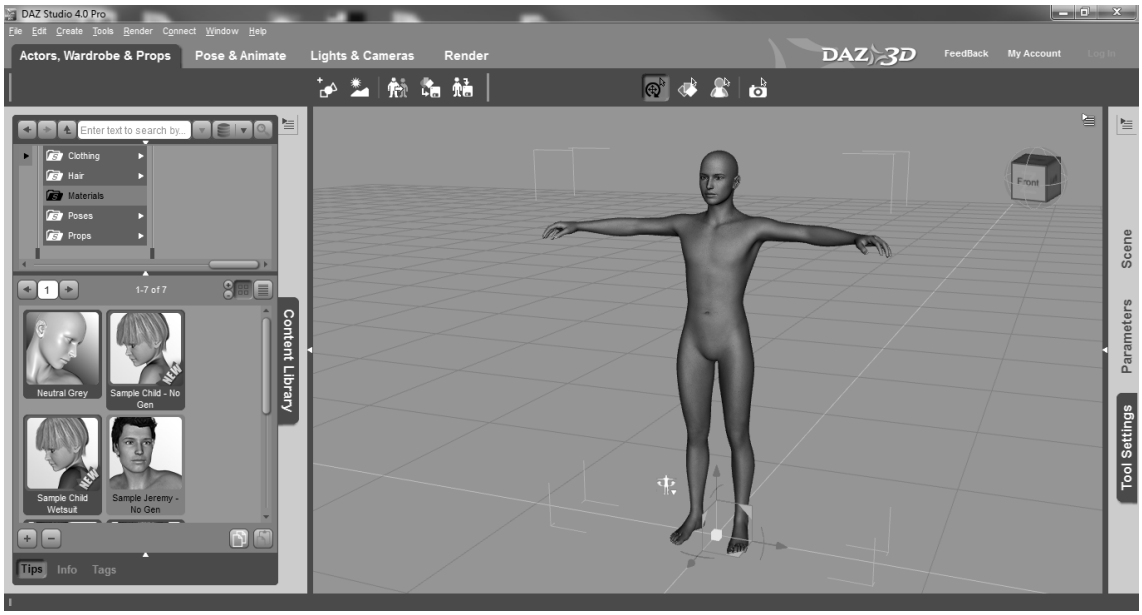


Figure 2 – Skin

Step 3: To add hair to the model go to the option Hair and select the desired hair (Figure 3)

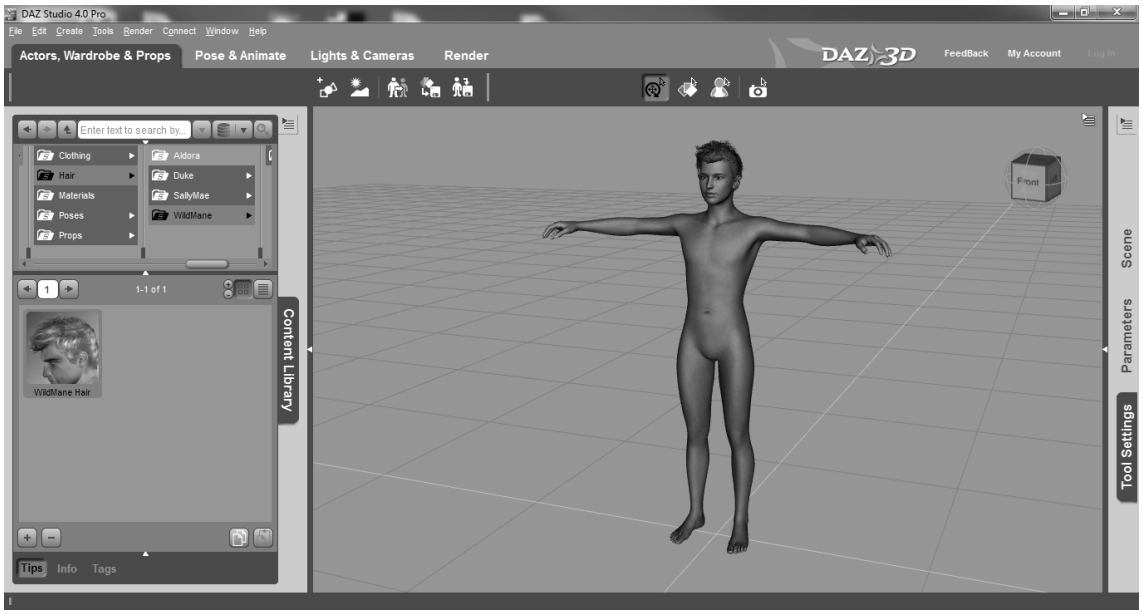


Figure 3 – Hair

Step 4: To add clothes to the model go to the option Clothing in Content and select the desired clothes from the ones available (Figure 4).

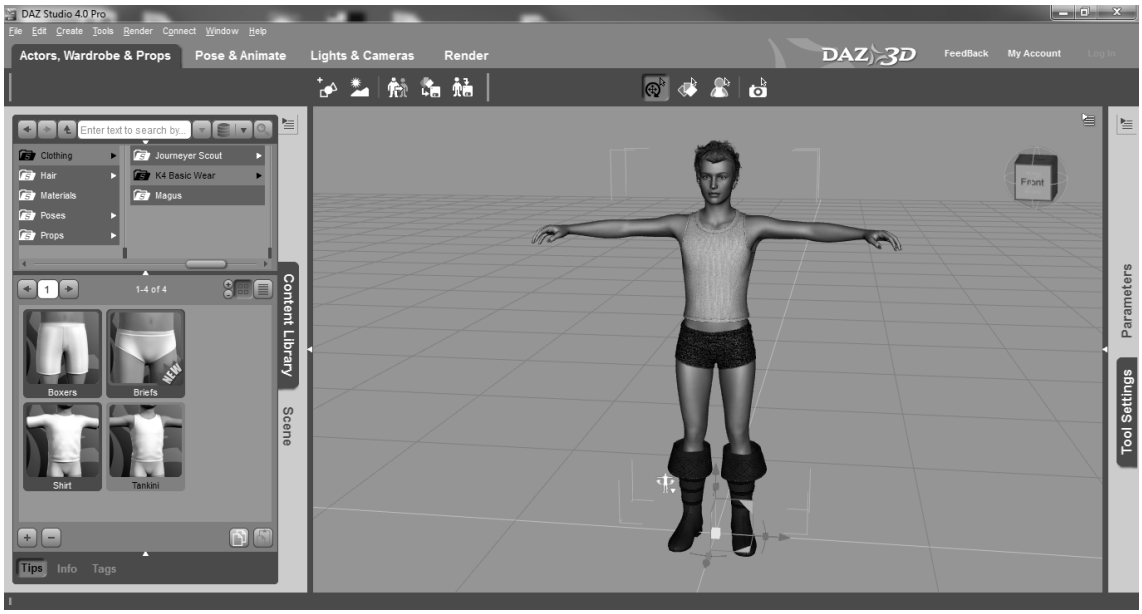


Figure 4 – Clothes

Step 5: In order that the animations affect all the elements in the scene it is necessary to attach the elements to the mesh. To do that select the desired objects and

with the right mouse button select the option Fit To (Figure 8). In the new window that appears select Genesis (name of the model) and press the Accept button (Figure 6).

Figure 5 – Fit To "Window"

Body Animation

To add body animations it is necessary to open the label Pose & Animate, open the menu at the bottom of the application, with the name aniMate2. AniMate2 is a timeline were the animations are placed (Figure 6).

Step 1: There are several pre-made animations and it is only necessary to drag-and-drop them into the timeline (Figure 7).

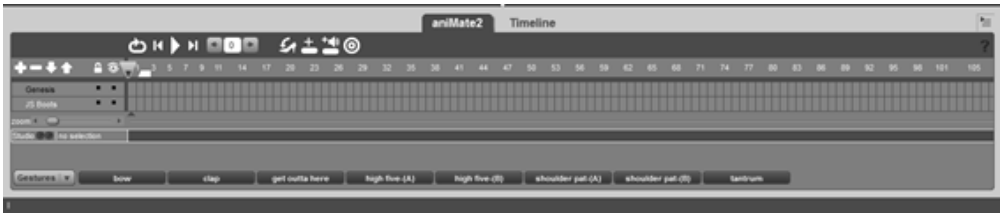


Figure 6 – Timeline of Animations

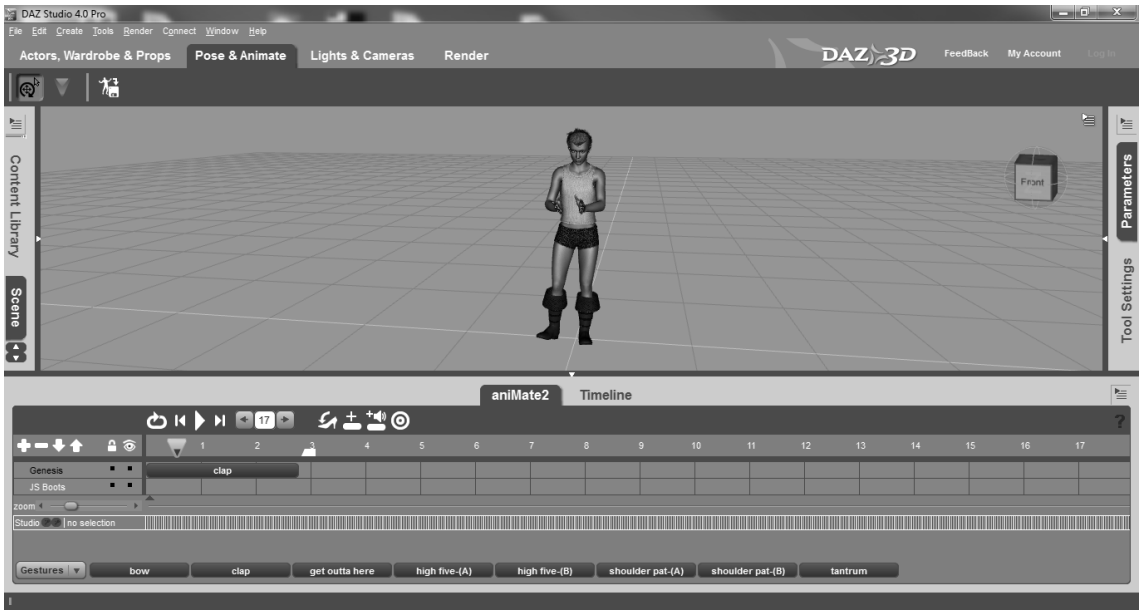


Figure 7 – Put Clap Animation

Attention 1: If you want to add facial animation, go to the section Facial Animation, if not, do Step 3.

Attention 2: It is advisable to save first in the .daz format before doing Step 2, because after doing this step it is not possible to change the animations.

Step 2: To export the model it is necessary to prepare the animations. This is done by clicking with the right mouse button on the play bar and selecting the option Bake to Studio Keyframes (Figure 8).

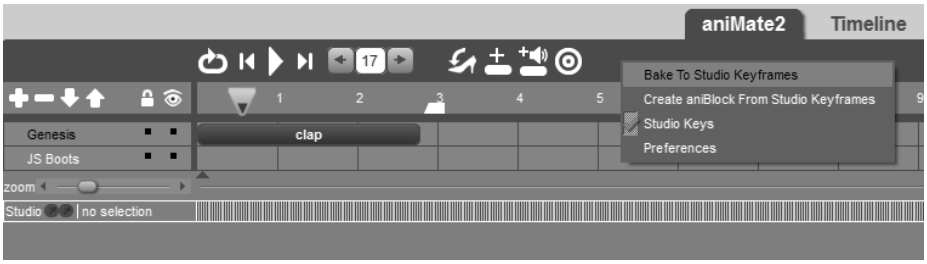


Figure 8 – Bake To Studio Keyframes

Facial Animation

If you already have done the body animations, you are familiar with the timeline. If not, you have to open the label Pose & Animate, open the menu at the bottom of the application, with the name aniMate2 (Figure 9). If you add body animations, you can do the facial animation in the same timeline that the body animations are but is better to do them separately in order to become easier to understand.

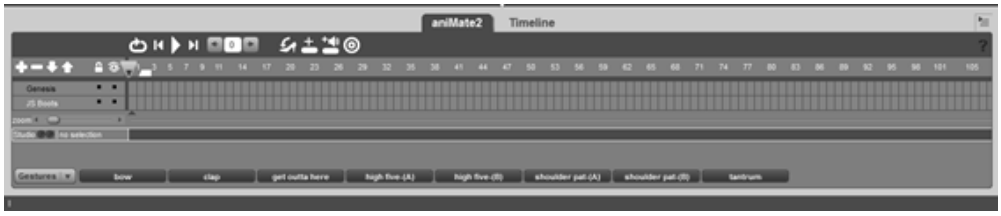


Figure 9 – Timeline Animations

Step 1: In the timeline click in the Plus ("+") to add a sub track to Genesis (Figure 10).

Figure 10 – Add Subtrack to Timeline

Step 2: In the sub track timeline, click the right button of the mouse, choose Add Empty Block (Figure 11). It will appear a Properties window and you choose Start with no properties, click Accept, and an empty block will appear in the sub track timeline. Click again with the right button up on the empty block and do rename (if you add more than one animation, is good rename all of them so you can distinguish), Figure 12.

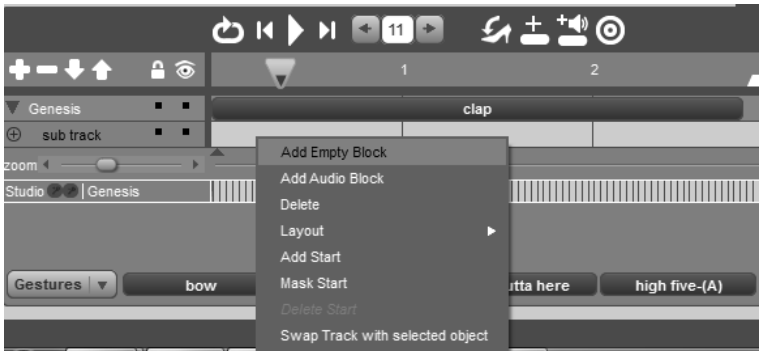


Figure 11 – Add Empty Block in Timeline

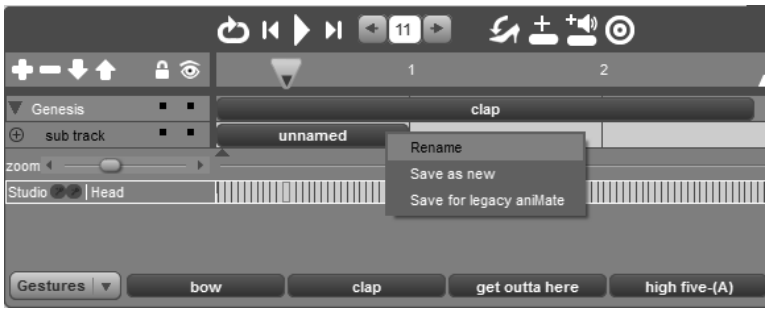


Figure 12 – Rename Empty Block

Step 3: In the label Pose & Animate, go to the Editor menu, Pose Controls menu and Head menu. Select the head of the model and then, in the Editor menu, select which part of the face you want to animate (Eyes, Mouth, Brow, etc.). Choose a position in the sub track timeline and move the slider of what deformation you want to do (Figure 13). To remove the animation select another position in the sub track timeline and move the slider until the initial position (Figure 14).

Warning: If you do not remove the expressions that you add, the model will always have that expression. Otherwise, if you want only a momentary animation you have to add and remove the animation as was made in Figure 13 and Figure 14.

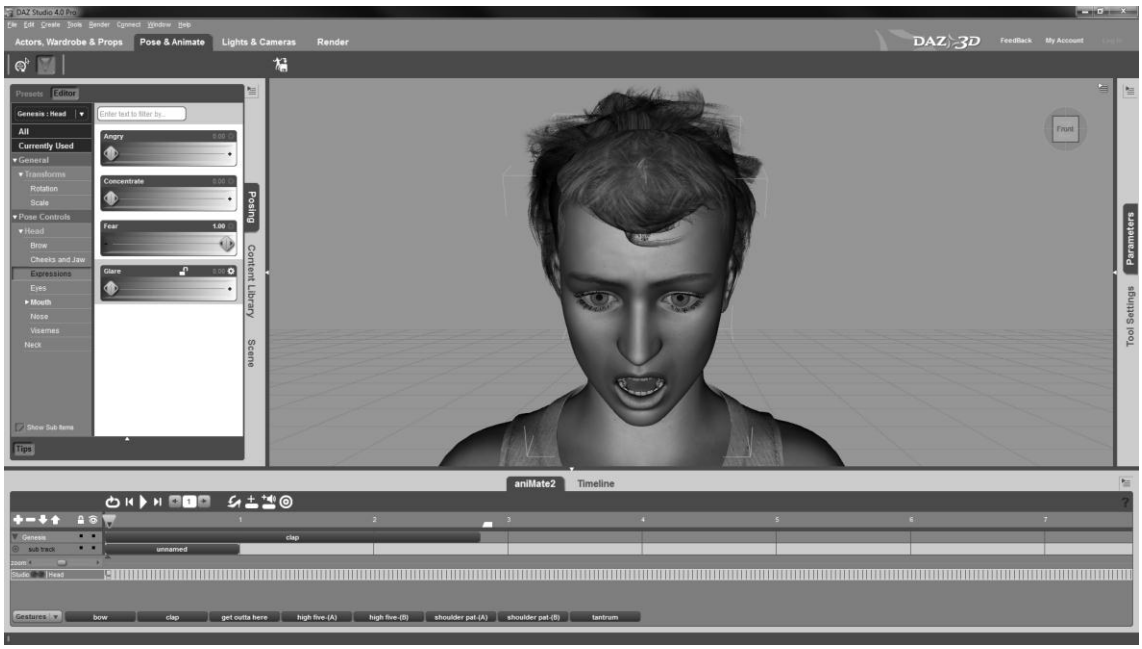


Figure 13 – Add Face Deformation

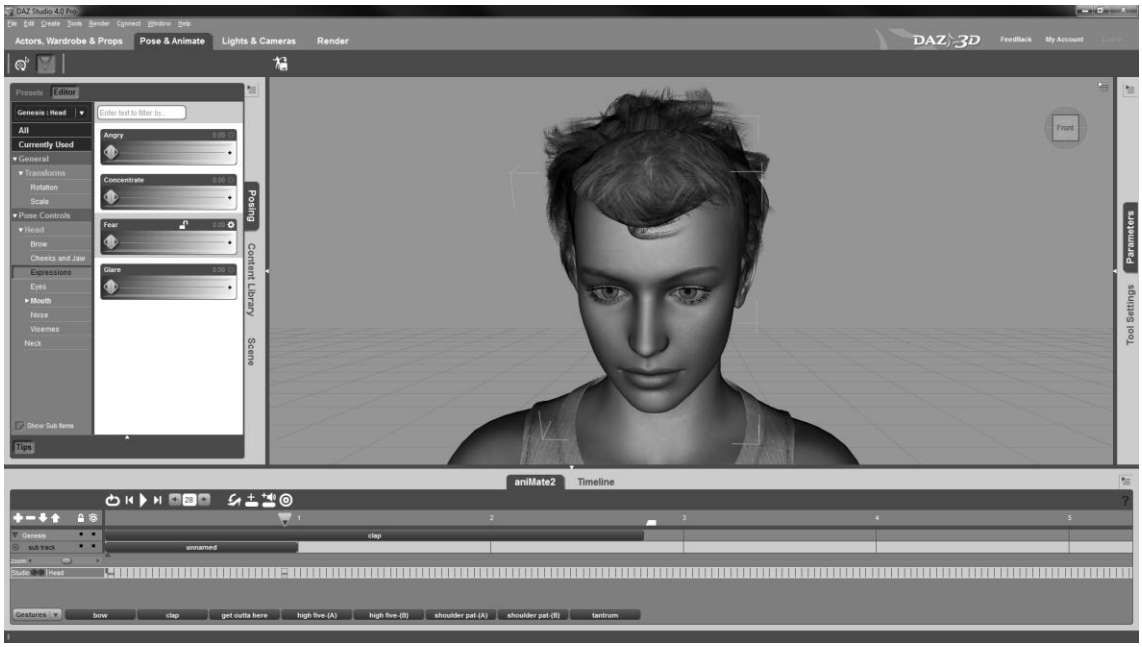


Figure 14 – Remove Face Deformation

Attention: It is advisable to save first in the .daz format before doing Step 4, because after doing this step it is not possible to change the animations.

Step 4: To export the model it is necessary to prepare the animations. This is done by clicking with the right mouse button on the play bar and selecting the option Bake to Studio Keyframes (Figure 15).

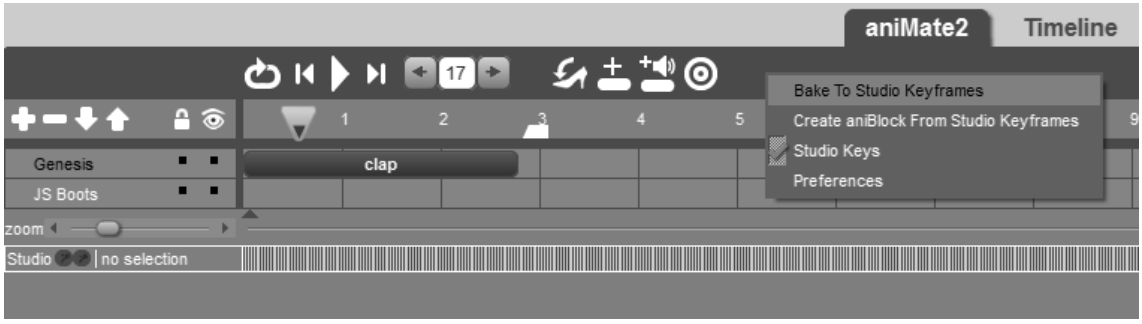


Figure 15 – Bake to Studio Keyframes

5. Imports and Exports

Export from Daz Studio 4.0 Pro

Step 1: To export the model to the FBX format, go to the File menu and click on the Export option, choose the Autodesk FBX (*.fbx) format. The FBX options window will appear. On that window leave the default options and click Accept (Figure 16).

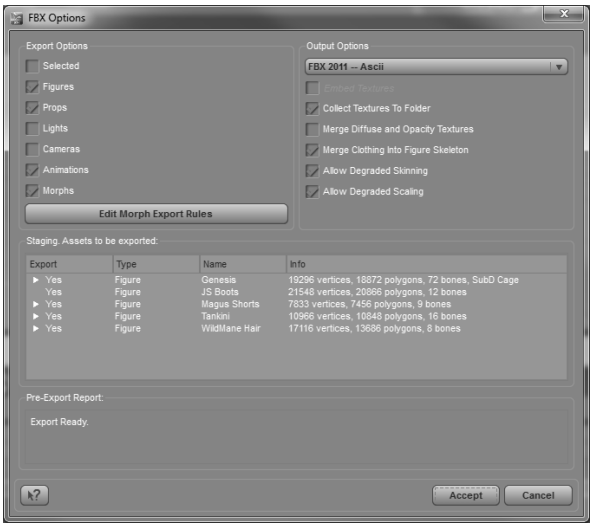


Figure 16 – Export FBX Options

Import in 3DS Max

Open 3DS Max and do the following steps:

Step 1: Go to the File menu, choose the Import option and choose the .fbx file exported from DAZ Studio 4.0 Pro. A window will appear, and in the Presets section select the option Autodesk Media & Entertainment and click OK (Figure 17).

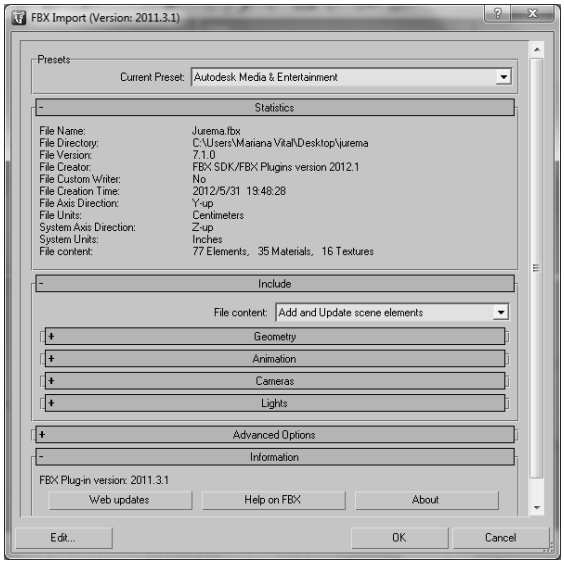


Figure 17 – Import FBX in 3DS Max

Step 2: Run the animation to ensure that has been well imported.

Step 3: Select the model's body and go to the Group menu (Figure 18), create a new group and click OK (Figure 19). Next, choose an object (clothes and hair) one at a time and go to the Group menu and click Attach (Figure 20). (The animations are made with the skeleton. The mesh has skeleton but the hair and clothes do not, so, they do not have animations. If those objects were attach to the main mesh, they become part of the mesh and they will have animations).

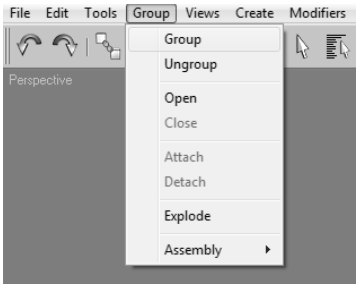


Figure 18 – Group Menu

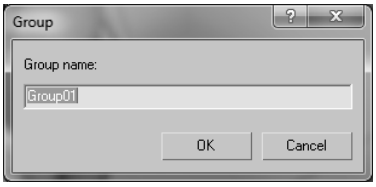


Figure 19 – Named Group

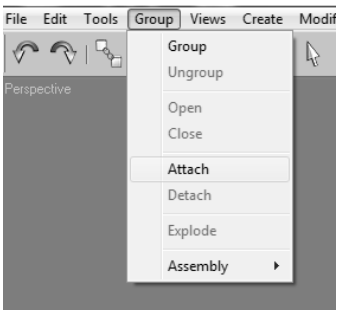


Figure 20 – Attach Object to Group

Export to OGRE format

To export to the OGRE format it is necessary to install the OgreMax plugin in 3DS Max. After that, you can complete the following steps.

Step 1: Select the model, go to the OgreMax menu and choose the option Object Settings (Figure 21). In the new window, select the label Mesh Animations put a check in the option Always Export Skeleton in the field Skeleton Settings (Figure 22).

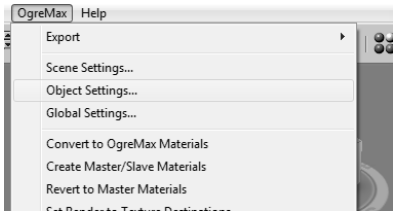


Figure 21 – OgreMax Menu

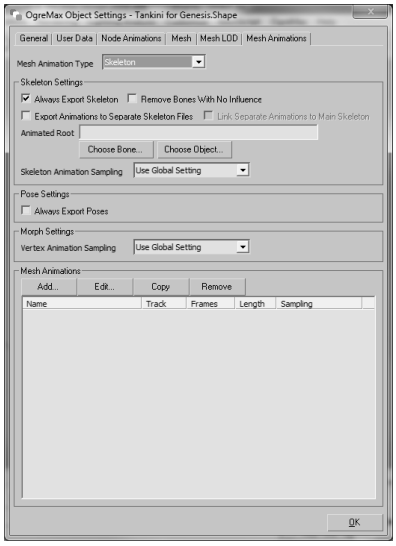


Figure 22 – OgreMax Object Settings

Step 2: In the same window (Figure 22), in the Mesh Animations section click in the button Add, to add an animation to the mesh (it is necessary to add the animations manually to the exporter since it cannot distinguish correctly between Morph and Skin animations). You will see another window in which you will assign a name to the animation, and choose the track as Morph or Skin, according to the type of animation, facial or body respectively.

Step 2.1: If the track is of type Skin (body animation).

Step 2.1.1: In the Skeleton Animation field, you put a check in the Copy First Animation Key to Last option, (Figure 23).

Step 2.1.2: In the Start/End Frames you put the time of the specific animation that you are saving.

Step 2.1.3: Click in the Add Bone option, and choose the bone Hip. There, select the Include and Apply to Bone and Descendants options,

click OK, (Figure 24), OK in the Add Mesh Animation window, (Figure 23), and final again OK in the Object Settings (Figure 22).

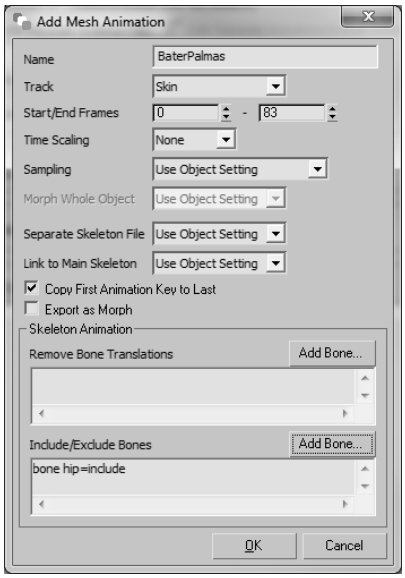


Figure 23 – Add Mesh Animations

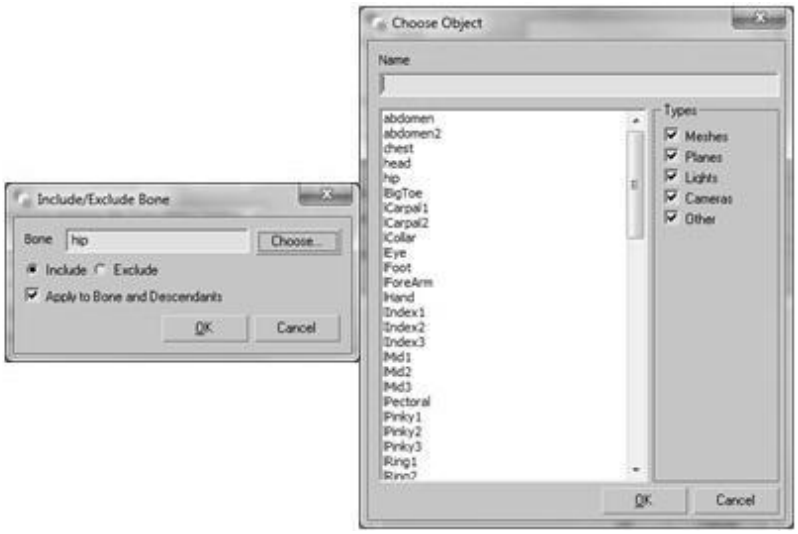


Figure 24 – Add Root Bone

Step 2.2: If the track is of type Morph (facial animation).

Step 2.2.1: Put a check in Copy First Animation Key to Last and Export as Morph options.

Step 2.2.2: In the Start/End Frames you put the time of the specific animation that you are saving. Click OK, (Figure 25), and again OK in the Object Settings (Figure 22).

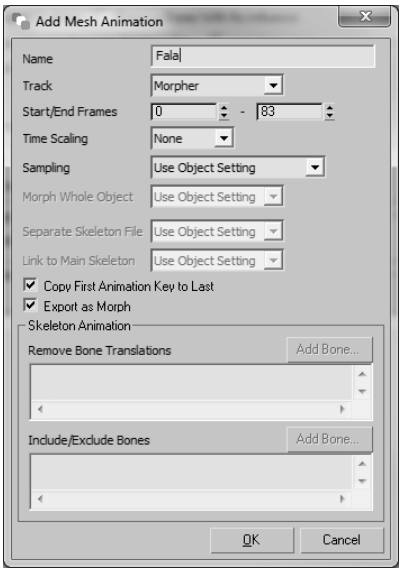


Figure 25 – Add Mesh Animation

Step 3: In the same menu of the window in Figure 21, go to the Export option, Export Scene (Figure 26) and give a name to the file.

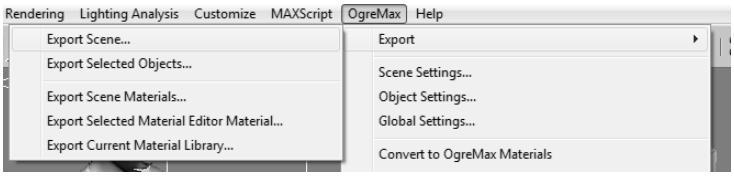


Figure 26 – OgreMax Export Scene

Now the file is ready to be loaded in the ErgoVR system.

6. Glossary

Morph: Morph is a vertex animation. It uses information about the movement of vertices directly to animate the mesh. This technique stores the absolute position of the vertices in a certain keyframe and stores another different absolute position in another keyframe and when playing the animation, at runtime, interpolates the intermediate positions.

Skin: Skin comes from skinning, is a skeleton animation. This animation technique consists in a mesh that represents the skin attached to the bones, and when a bone moves the skin follows the movement of the bone that is attached.

D. User Manual to Configure FileConfig.cfg

User Manual

Change the Configurations of the ErgoInteract

Contents

1. Introduction

This manual has the goal to instruct how to modify the ErgoInteract system configuration via a configuration file. The use of a configuration file allows changing some properties of the system without the need to recompile the complete system. This configuration file is placed in the folder where the system is going to be used, near the executables of the application.

2. Overall Workflow

The configuration file is divided in sections. Each section has keys that have the information to be read by the system. The file is divided in three sections named LoadScene, LoadHV and ConfigSensor. The LoadScene has the information about the scene, the LoadHV has information regarding the VH (i.e., gender) and the ConfigSensor section has information regarding the sensors and the arms to be used.

3. Template of Configuration File

The configuration file has the name ConfigFile.cfg and looks like this:

```
#Comment
 [Section]
“Key”=”Value”
```

4. Section LoadScene

In the section LoadScene there is a property called “PathFile” that represents the location of the .scene file on the hard drive to be loaded in the system. For example:

```
#Path of the scene to load
[LoadScene]
PathFile=C:\Users\Mariana Vital\Desktop\Corredores\Corredor Sinais com Node Interaction\teste.scene
```

5. Section LoadHV

In the section LoadHV there is a property with the name “VHModel”, which allows changing the gender of the VH to be loaded by the system. If it is intended to

load a man's VH, the value should be "man" and if it is intended a woman's VH the value should be "woman". For example:

```
#Sex of the model you want to load
[LoadHV]
VHModel=woman
```

6. Section ConfigSensor

In the section ConfigSensor there are three properties, Arm and NumSensores. In the Arm property, it is possible to write "left", "right" or "both" depending on the arm(s) that is intended to have control over with the motion sensors. In the NumSensors property it is possible to define the number of sensors to use. This property is dependent on what was defined on the Arm property:

- Left or Right: You can only have 2 or 3 sensors;
- Both: You can chose 4 or 6 sensors;

```
#Configure the number of sensor you want to load
[ConfigSensor]
Arm = Both
NumSensores = 6
```

Complementary User Manual for the Xsens Motion Sensors

How to connect the sensors and place them on a participant

ERGOLAB RESEARCH UNIT
FMH // TECHNICAL UNIVERSITY OF LISBON

Contents

List of Figures

1. Introduction

This manual has the goal to help the connection of the sensors and place them in the participant.

A XM-B User Manual exists and it shows how to install the software, how the software works, what information it gives us, how the sensors work and how to connect them through the cables with the wireless receiver. The problem is that it does not show the order in which the sensors should be connected and the correct placement on the participant.

2. Requirements

To be possible to complete this tutorial it will be necessary to have the following documentation and material:

- XM-B User Manual;
- XM-B Technical Documentation;
- Xbus Master (Figure 1)
- MVN Mounting Straps (Figure 2 - MVN Mounting Straps)
- MVN Mounting Straps User Manual

Figure 1 - Xbus Master

Figure 2 - MVN Mounting Straps

3. Connecting the Sensors

All the sensors have their ID written, as it is possible to see in Figure 3. It is necessary to connect them by ascending order. In total, it is possible to work with six sensors, three in each arm (Hand, Forearm and Shoulder). If it is intended to work with the six sensors it is required to connect the Sensor ID 0, Sensor ID 1 and Sensor ID 2, in one side of the Xbus connector of the wireless receiver and Sensor ID 3, Sensor ID 4 and Sensor ID 5 in the other side, in order to have three sensors separated for each arm. It is possible to see in Figure 4.

Figure 3 - Sensors ID

Figure 4 - Connecting the 6 Sensors

If you want to work with fewer sensors, you just need to remove the sensors with the higher ID. Example: If you want to work with four sensors, you need to remove the Sensor ID 5 and Sensor ID 4. In the side that you remove the both Sensor you need to place the Sensor ID 2.You need to change the Sensor ID 2 to the secondary connection in order to have two sensors on each side.

Attention: It is necessary to work with the same number of Sensor on each side.

4. Place the Sensors in the Participant's Body

Figure 5 - Straps with Sensor, shows how to place the sensors in the Mounting Straps.

Figure 5 - Straps with Sensor

There are six straps, each strap has a label and in that label it has written the member and an arrow to indicate the direction in which the sensor should be inserted. Two straps have written "L-Leg", another two "F-Arm". The straps "L-Leg" are for the shoulder because they are bigger, the "F-Arm" are for the forearm, and for the hand there are two gloves.

The sensors need to have the cables in the direction indicated in the label by the arrow, and placed in the user with the arrow for the upper side.

The series of Sensor ID 0, 1 and 2 goes in order to shoulder, forearm and glove, as well as the series 3, 4, and 5. It is possible to see in Figure 6 - Connected Sensors with the Straps.

Figure 6 - Connected Sensors with the Straps

After that, you just need to place them in the body's user participant as shown in Figure 7 - Place Sensors (Front View) and Figure 8 - Place Sensors (Side View).

Figure 7 - Place Sensors (Front View)

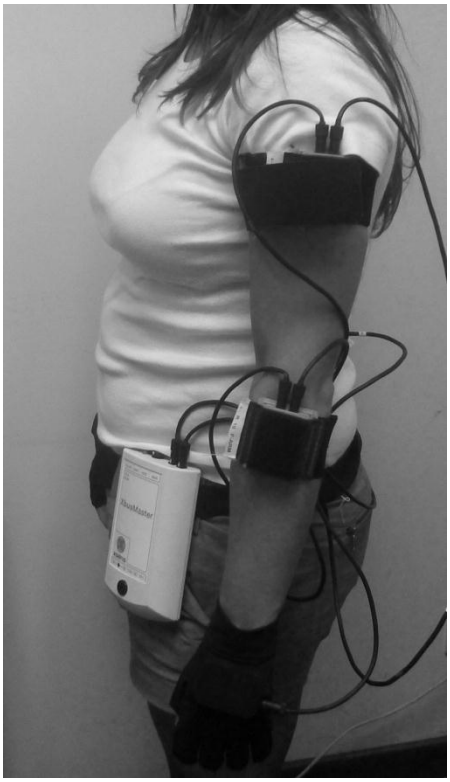


Figure 8 - Place Sensors (Side View)