Ricardo Luís d'Abreu Fernandes Carranca

Controlo de movimentos 3D com interpolação de eixos

3D motion control with axis interpolation

Ricardo Luís d'Abreu Fernandes Carranca

Controlo de movimentos 3D com interpolação de eixos

3D motion control with axis interpolation

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Mecânica, realizada sob orientação científica de José Paulo de Oliveira Santos, Professor Auxiliar do Departamento de Engenharia Mecânica da Universidade de Aveiro e de Francisco José Malheiro Queirós de Melo, Professor Associado do Departamento de Engenharia Mecânica da Universidade de Aveiro.

O júri / The jury

Presidente / President

Prof. Doutora Mónica S. A. de Oliveira Correia
Professora Auxiliar do Departamento de Engenharia Mecânica da Universidade de Aveiro

Vogais / Committee

Prof. Doutor José Paulo de Oliveira Santos
Professor Auxiliar do Departamento de Engenharia Mecânica da Universidade de Aveiro (orientador)

Prof. Doutor Francisco José Malheiro Queirós de Melo
Professor Associado do Departamento de Engenharia Mecânica da Universidade de Aveiro (co-orientador)

Prof. Doutor Mário Augusto Pires Vaz
Professor Associado da Faculdade de Engenharia da Universidade do Porto

Agradecimentos / Acknowledgements

Começo por agradecer à Universidade de Aveiro, o excelente ambiente e condições de estudo e de formação que me proporcionou, durante todo o período de aprendizagem.

Em seguida, agradeço ao meu Orientador, Professor Doutor José Paulo de Oliveira Santos e ao meu Co-orientador, Professor Doutor Francisco José Malheiro Queirós de Melo a possibilidade de poder realizar este trabalho que tanto me entusiasmou, e pelo apoio, ensinamentos e disponibilidade sempre prestados.

Agradeço, muito especialmente, à Professora Doutora Mónica S. A. de Oliveira Correia a amizade e o apoio muito especial que sempre me dedicou.

Agradeço, ainda, ao Prof. Manuel Raposo, o auxílio prestado na construção da unidade de controlo e todos os ensinamentos relativos ao tema. Graças à sua imensa disponibilidade foi-me possível concluir todos os pontos que me propunha nesse trabalho.

Agradeço, também, à firma italiana Sipro, que ao longo desta caminhada demonstrou sempre grande abertura e disponibilidade no esclarecimento de dúvidas.

Agradeço ao Engenheiro Alex Hitchcock, a disponibilidade em me receber na empresa StageTechnologies para discutir as principais preocupações a ter em conta aquando do desenvolvimento dum sistema desta natureza, bem como, mostrar o *software* e *hardware* que a empresa atualmente utiliza.

Finalmente, agradeço dum modo muito especial aos meus Amigos e colegas, toda a amizade e apoio que sempre me demonstraram, desde o meu primeiro dia nesta Faculdade e, muito em especial agora, durante o desenvolvimento desta Dissertação.

Por último, agradeço à minha Família, pelo amor e apoio constantes, ao longo de toda esta caminhada.

A Todos, um sincero MUITO OBRIGADO!

Palavras-chave Automação, Controlo, Sistema de vôo 3D.

Resumo

Há alguns anos que a automação e controlo entraram no mundo do entretenimento. A evidenciar este facto, tem-se assistido nos últimos anos a um aumento de soluções e inovações nesta área. Tal desafio resulta em grande parte, da exigência dos espetadores, com uma cada vez maior componente emocional, que deriva da transposição para a indústria do espetáculo, de façanhas extraordinárias, por exemplo com personagens importados do universo de heróis da banda desenhada.

Este campo de criatividade, com efeitos visuais, tem aplicações no teatro, cinema ou concertos. A par destes desenvolvimentos, tem surgido a necessidade de evolução de equipamentos para simulação de vôos/saltos. Por este motivo, é objetivo desta dissertação estudar uma solução a nível de sistemas de controlo - *software* e *hardware* - capaz de satisfazer as necessidades dum sistema deste tipo.

Para o sucesso de um projecto desta natureza é fundamental o controlo perfeito de movimentos e trajectórias complexas. O trabalho aqui desenvolvido centrou-se nestas temáticas, em especial, no desenvolvimento de um sistema de controlo capaz de simular cenas de vôo a três dimensões, com aplicabilidade na indústria do entretenimento.

Keywords Automation, Control, 3D flying system

Abstract

Since a few years ago, automation and control became an important tool in the simulation of visual effects as part of the entertainment industry. In fact, in the last few years the number of solutions in this highly creative field has increased. A challenging vector for this arises from the spectators' exigency, expecting to meet a quite high emotional experience rank when assisting for example to the performance of actors recreating the heroes of comics stories or even the incarnation of cartoons performers. Among these developments there is the need of upgrading the flying simulators.
So, the aim of this dissertation is to study a control solution - software and hardware - capable of meeting the needs that a system like this requires. The very realistic materialization of complex trajectories with perfect motion control is a field of primary importance for the success of the project, therefore that thematic has been explored and developed. The work here developed focused on these issues, particularly in developing a control system capable of simulating flight scenes in three dimensions, with application in the entertainment industry.

Contents

List of Tables

List of Figures

Symbols and Acronyms

AC	Aternating Current
bit.s^{-1}	Bit per second
CAD	Computer Aided Design
cm	Centimeter
DC	Direct Current
ºC	Degree Celsius
e.g.	For example
GIMP	GNU Image Manipulation Program
GTK+	GIMP Toolkit
Hz	Hertz
I/O	Input/Output
ms	Millisecond
cm.s^{-1}	Centimeter per second
MTV	Music Television
NC	Numerical Control
N.m	Newton-meter
PC	Personal Computer
PID	Proportional Integral Derivative
PLC	Programmable Logic Controller
PRG	Production Resource Group
PWM	Pulse Width Modulation
®	Registered Mark
s	second
SCS	Stage Command System
t	Tonne
TM	Trade Mark
TV	Television
UK	United Kingdom
USA	United States of America
VE	Virtual environment
$	US dollar
%	Percent
2D	Two-dimensional
3D	Three-dimensional

Part I

Guidelines

Chapter 1

Introduction

1.1 Context

The objective of this dissertation arises from the emerging necessity of creating new technological solutions for the entertainment industry, since nowadays everyone is constantly merged on a sea of technological developments without even noticing. For example, rock music concerts have existed for many years, but nowadays a live concert of a super star is a prime form of live entertainment using high technological solutions. Hollywood, Broadway, Las Vegas live shows or even live shows in theme parks are other ways of telling a story by means of entertainment technology, contributing for the welfare of the society and for the economical growth of the country [24].

Technology in showbiz is an emerging market, expressed in the grosses involved. For example on Broadway - between 2000 and 2010 - an amount of $17.78 billion was generated, which means a growth of 63% in that period for the New York based productions and 49% for the tours, as shown in the chart of the Figure 1.1.

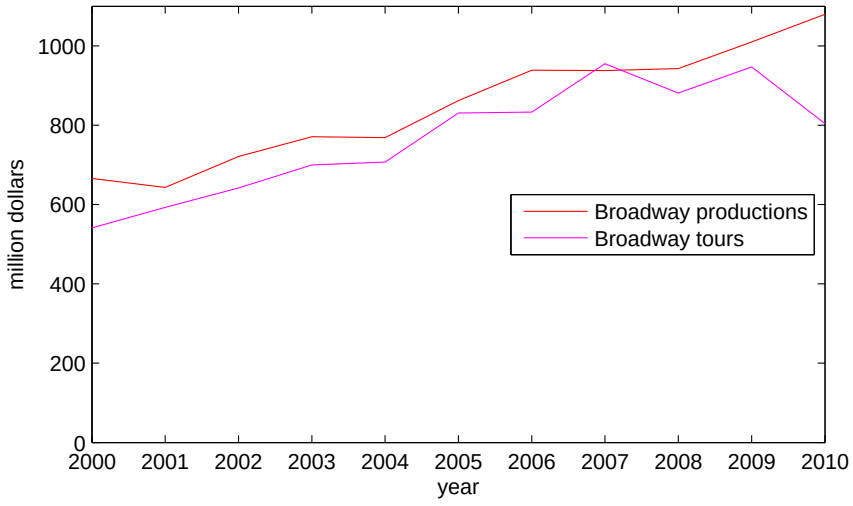


Figure 1.1: Grosses of Broadway from 2000 to 2010 [6].

For example, the most expensive production ever on Broadway spent about $80 million to create the show in 2009/2010 where almost 50% were for technological developments including an expensive 3D flying system which is the last trend in this area. In about three years, this show earned $110 million only from ticket selling [6].

On the other hand, the film industry is the greatest private sector in USA with a massive contribution of big productions such as James Cameron's Avatar which had a budget of about $237 million, where $14 million were to develop the nowadays largely used 3D technology. However, James Cameron, and particularly Avatar are synonyms of unusual mega success, once it grossed $2,78 billion only in box office all over the world.

These numbers are too far away from what is usual in Portugal, where culture and technology still live apart from each other and the research in this area is almost non-existent.

1.2 Historical Review

Along with the effort of countries like USA, UK, Japan or even India, the entertainment industry is constantly changing and concepts like virtual reality, and motion control have helped this growth in the last decades. Therefore, with audiences expecting always more and more, the demand for surprising increases, as well as the role of automation and special effects [28].

The automation started to be used in the entertainment to replace the human effort helping to 'change fantasy to reality' (Bob Crownley, set designer, 2007). The effects in theatre, for instance, vary from a simple pneumatic cylinder closing a door to a complex system of winches moving pieces of a chandelier above the audience to create an assembling scene during the *Phantom of the Opera in Las Vegas*.

Usually, for each show or entertainment production, the machinery is customized and designed specifically according to the director's needs, and can be overstage (e.g. flying devices) or understage (e.g. stage lifts) [23, 1].

1.2.1 Understage Machinery

Understage machinery is the equipment placed under the stage, most developed in the entertainment industry once it was first used many years ago. For example, the Coliseum in ancient Rome had lifts which were used to raise animals into the arena for their games. Also, in the Roman Empire, some theatres had lifting systems to raise and drop the curtain at the start of a play.

However, the use of understage machinery had its maximum point in the Baroque Period where some ordinary pieces of scenery were more likely to be raised from the understage, as shown in Figure 1.2.

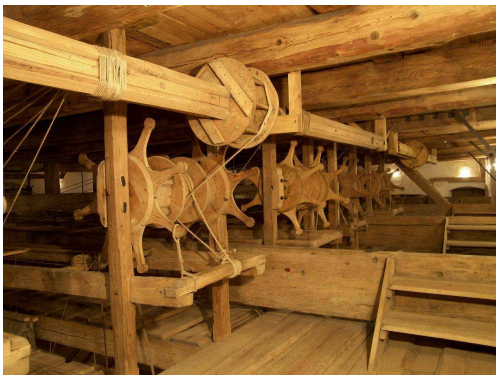


Figure 1.2: Understage of a Baroque theatre in Czech Republic [12].

In the 1880's, hydraulic lifts started to be installed in theatres using water under pressure from tanks mounted in the roofs. In 1932, the Radio City Music Hall built a famous installation of lifts, so advanced for their time that the World War II aircraft carrier lifts were inspired on it.

Despite being used in a large scale, understage equipments are very dangerous due to the open hole in the stage, usually with no protection, and where the performer can fall in the darkness of a blackout and it is not surprising that deaths have occurred [23].

1.2.2 Overstage Machinery

Only in the last few years, overstage machinery had some developments with the efforts of big companies, and with the necessity of big production companies to have more control on the flying scenes. Until a few years ago there was no system which enabled the director to create a 3D flying scene. Nowadays, there is only a couple of companies working on this subject.

Flying systems are used to lift (or fly) pieces of scenery or even actors into a space above the stage or above the audience using mechanical devices. This is the key point that makes this kind of equipment so important in the new Era of the entertainment industry.

It is important to refer that there are two types of flying systems: the hand-operated and the machine-operated. The hand-operated system is archaic and inherently dangerous, having its massive use in the early 1600's. At the turn of the 21^{st} century appeared the machine-driven flying systems which replaced the human strength for the electrical power.

In some of these systems, electricity may provide both the lifting ability and the force needed to counterbalance the carried load using a motor, a brake, a reducer gearbox, and a drum around which several hoisting lines are wound.

The appearance of the electric-powered flying systems along with the developments in the late 20^{th} century of the computer-driven controller, generally known as show control (computerized process for motion control), enhanced the flexibility and usefulness of drive systems in the theatre, as shown in Figure 1.3.

Figure 1.3: Flying car on the show Chitty Chitty Bang Bang [11].

The console operator aided by motion control units can therefore determine with strict precision the operation parameters, such as start time, speed, duration, and so forth, for every piece of powered equipment used in a show.

In the beginning of the 21^{st} century, production designers started to choreograph complex scenes as part of the visual spectacle of a production, in order to create the surprise effect that originates the so called 'standing ovation' [14].

Chapter 2

Objectives

'I really do believe that if you don't challenge yourself and risk failing, that's not interesting.'
Julie Taymor, film & theatre director

In the previous Chapter, the problems the producer may find while searching for a company which can help him to create an overstage scene were highlighted. The technology associated to overstage scenes is very recent, so that, there are few companies capable of doing it in an affordable way so it becomes mandatory to look for other ways of doing the same. As Julie Taymor said, it is important to challenge ourselves in a quest we do believe that is possible. The latter establishes that the main focus of this project is concerned with the overstage machinery, since the currently available 3D flying devices show a lack of development. Only in the last few years this kind of equipment had some developments with the efforts of the big companies of the field and with the needs of the big production companies of having more control over the flying scenes.

Thus, the aim of this project is to develop a system to control a five degrees of freedom system (three main controlled directions, one commanded rotation axis and one free rotation axis), the most affordable as possible. Nevertheless, the system itself, as far as the hardware is concerned is not the focus of the project. To be a winning product, the entire system has to be unique and very complete, so the ideal solution is to join both hardware and software in a single package since they are 'the key components in most of today's automated stage systems which allow sophisticated computer-based control and synchronization of many different stage systems, all running at different time intervals and speeds' (Richard Brett, theatre engineer).

Therefore, the project was divided into three main parts: the graphic interface, the control unit and the hardware. However this last point will not be dimensioned because it is not relevant for the controlling task which is the aim of the project.

2.1 Graphic Interface

The main challenge in graphic interfaces is to create animation systems that allow a more intuitive control of the movement. The graphic interface (Figure 2.1) is supposed to be responsible to interact with the user and therefore allow the creation of paths inside the predefined workspace (the volume of the real venue which the director can work with) according to the director/choreographer ideas.

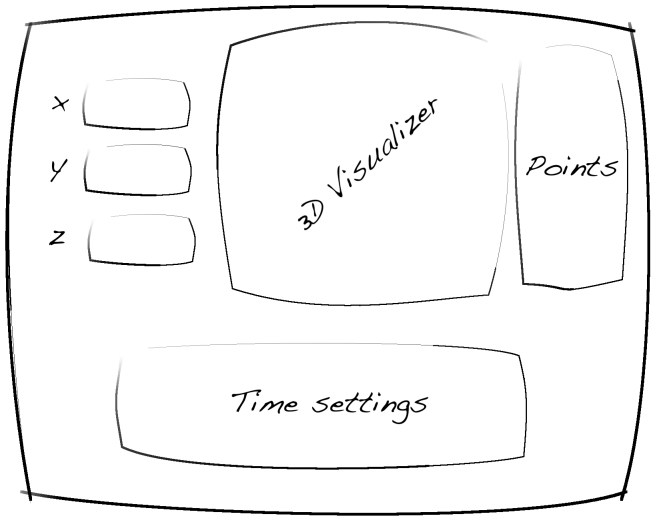


Figure 2.1: Illustration of a generic graphic interface.

First of all, the user may be asked to create the workspace such as the dimensions and the general specifications of the venue in order to have a precise idea of the room to work with.

The paths will be drawn and the user will be able to constantly visualize the current position and save it whenever wanted. Thus, every time a point is saved, an interpolated path will be created between them and depicted in the interface.

When creating the path, the user may be also able to customize the duration of the entire scene, making it slower or faster according to what it is required. The user may also be able to add different speeds on the flying scene, creating acceleration and deceleration movements.

Once finished this conceptualization task, the user will be able to have a more realistic view with a rendering tool where the entire scene may be simulated with much better graphics. Therefore, if the user is satisfied with the result, the movement may be exported to the motion control unit.

2.2 Motion Control Unit

The goal of the development of the motion control unit is to create a unit capable of receiving the information sent by the created software.

Therefore, the motion controller has to 'understand' this information, process it and send the corresponding signals to the motion units (winch motors and drives).

Summing up, the motion control unit has to be able to process movement information with incoming data from the graphic interface in order to generate signals that can be understood by the motion units.

These motion units are four winch motors and the corresponding drives, which are mounted on the corners of a rectangular structure, and the system hanging the harness will gather the cables that come from each motor, like it is shown in the Figure 2.2.

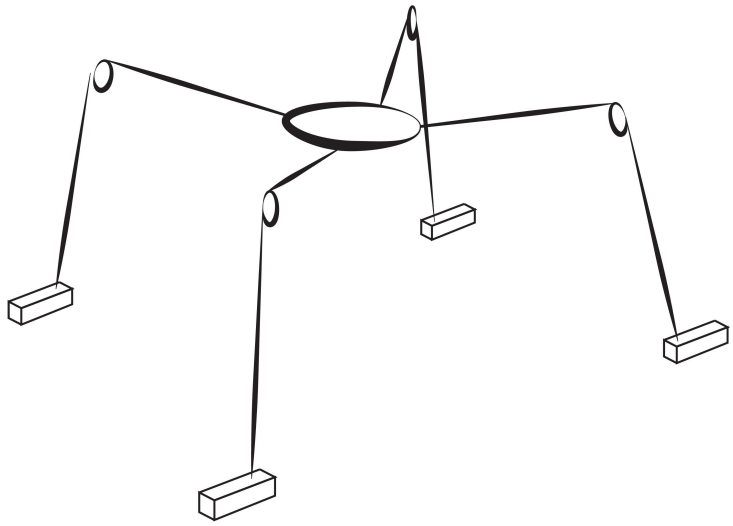


Figure 2.2: Illustration of a generic 3D motion system.

Chapter 3

State of the Art

Control systems for 3D motion are not a novel item in the entertainment industry, having its massive use in the cinematographic and musical theatre industry, with Hollywood and Broadway as the greatest developers of this technology.

Therefore, nowadays, directors can record an entire scene in a single take since this type of equipment (Figure 3.1(a)) is capable of covering up a larger area without the physical restrictions that traditional equipments have. With this technology, musicals in New York have dramatically changed because instead of having the scene entirely on stage as it was usual, it started to become common to have a scene up in the air, right above the audience, providing astonishing flying scenes (Figure 3.1(b)).

(a) Camera in a set [3].

(b) Flying performance [9].

Figure 3.1: 3D motion control systems.

3.1 Existing Solutions

It is important to have a look at the market offer, either for 3D flight simulation or for image capture, in order to understand which are the real needs in this area.

3.1.1 Spidercam™

This system, whose design was inspired by the Star Wars saga, is used to record scenes and it is largely used in cinema and live events. It gives the operator a complete sense of freedom in the entire space provided by four motors placed in the four corners of the

rectangular area. Kevlar and optical fibre cables are connected to the motors in order to give motion to a camera (stabilized through a gyroscope) that provides images that are transmitted in high definition through the optical fibre cables.

The camera, which can be seen in action in Figure 3.2, is mounted in a structure which allows rotation in two axes. This means that this is a system with five degrees of freedom (three principal directions and two rotation axes). All of them are remotely commanded [10].

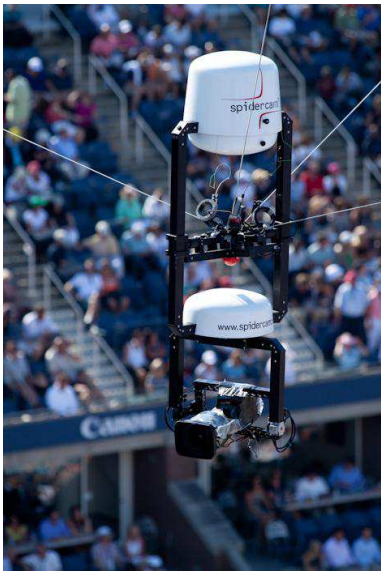


Figure 3.2: A Spidercam^{TM} recording a show [16].

This system works in 'live mode', which means that the movement of the camera is controlled by means of two joysticks while the action occurs. However, there is no information about the interface between the operator and the camera, or the way this camera is powered, but it is thought that this is by means of batteries.

3.1.2 Stage Technologies^{TM} Systems

Stage Technologies^{TM}, commonly known as StageTech, is a British company well known all over the world. It offers automation solutions for entertainment events supplying everything from the software to the hardware.

Software

Visual Creator^{TM}

The Visual Creator^{TM} was created about twenty years ago to offer a graphic interface between the operator and the mechanical system itself for flight simulation. The interface enables up to three degrees of freedom, the three main directions, which allows the operator to create profiles according to the movements the director came up with. The operator can also define either flying times or accelerations during the whole scene.

To define the movement profile and the scene itself (Figure 3.3) the operator can insert all data by hand or use a joystick [11].

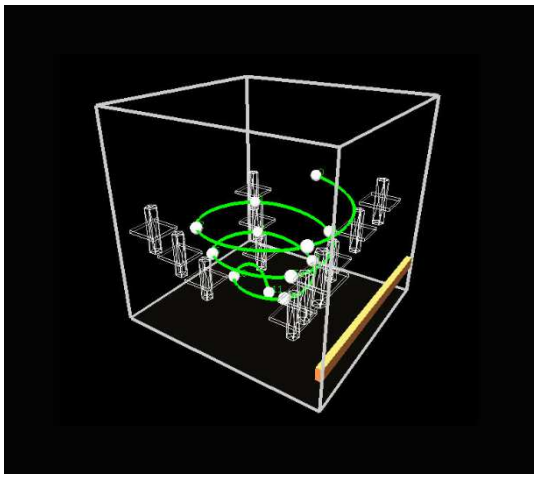


Figure 3.3: Movement profiling tool from Visual Creator^{TM} [11].

However, this software started to be deprecated and it is not so largely used on StageTech projects. It has been slowly replaced for the eChameleon, which was presented to this project team in a meeting with the Automation Engineer Alex Hitchcock, in London.

eChameleon^{TM}

The eChameleon^{TM} is a software which offers a real-time, 3D and tabular views of automation systems (Figure 3.4), and is used in all StageTech consoles. It supplies a full logical command line interface for a fast programming.

The eChameleon^{TM} runs on Windows Embedded Standard, which means that the user is only given access to the eChameleon launcher. This allows the user to only run Stage Technologies software.

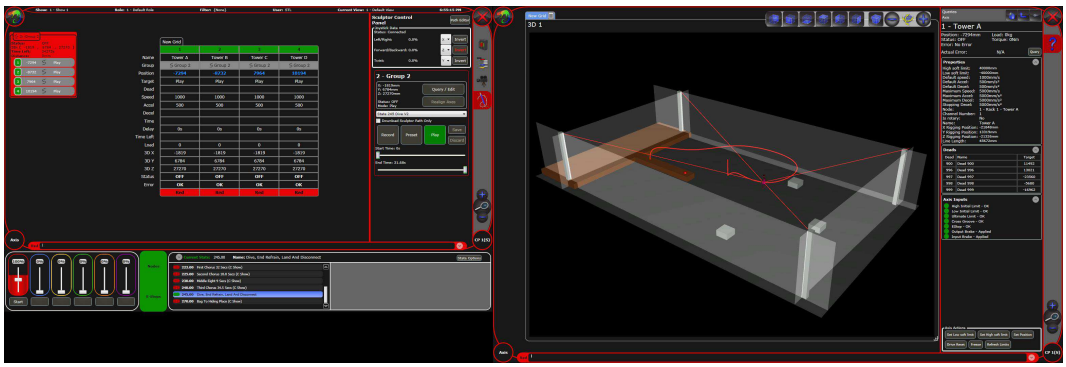


Figure 3.4: Aspect of the eChameleon^{TM} interface [31].

For applications in more than one dimension, tools like Sculptor Animation Toolkit

(Figure 3.5) are available to generate 3D motion for any rigging scenario. This is an Autodesk 3ds Max toolkit which can be tightly integrated with eChameleon to allow partial or complete animation of the show.

The Sculptor Animation Toolkit offers a full visual representation of the automation system and can be imported from CAD models, or rapidly composed using a versatile library of standard automated objects.

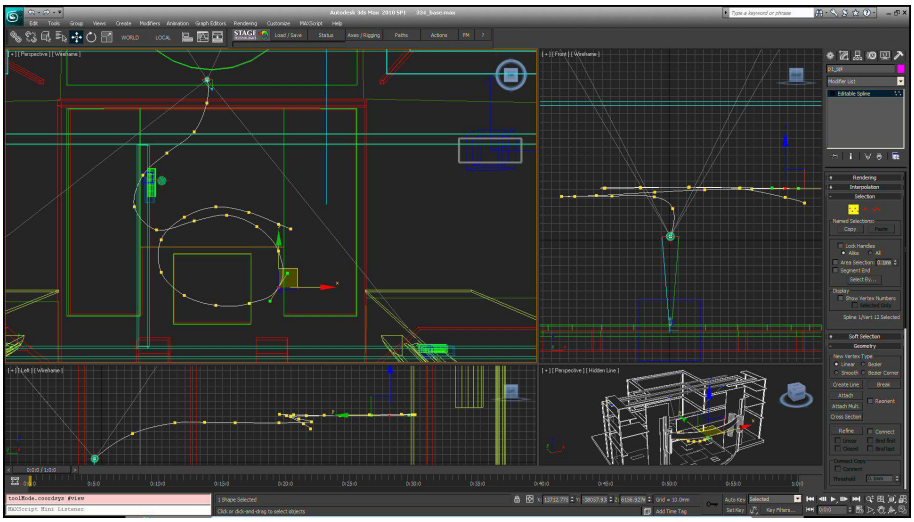


Figure 3.5: Autodesk 3ds Max's Sculpture Animation Toolkit [30].

One problem of this type of software is the price, since it is a closed toolkit of a very expensive dedicated software.

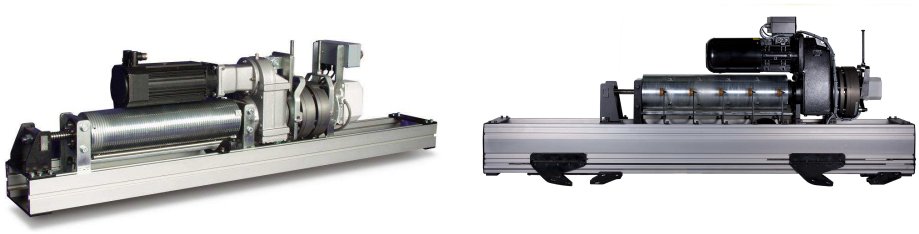
Hardware

Stage Technologies has a wide range of hardware and they are the pioneers of the zero-fleet technology - the exit angle of the cable is always zero, when the cable winds out of the drum.

So that, StageTech offers two types of winches: one is called BigTow (Figure 3.6(a)) and it is composed by a brushless servomotor engineered for high speed and high capacity, carrying up to 2 t at 2 $m.s^{-1}$ on multiple lines; the other winch is called BigTow Lite (Figure 3.6(b)) which has a lightweight chassis and a three-phase induction motor with an absolute encoder and an electromagnetic secondary brake [11].

In order to make the link between the software and the winches, this company uses Siemens®'s Programmable Logic Controllers (PLC) as the control unit and interpolation drives.

One problem of this technology is that for each show, a new ladder program has to be done in order to relate all axes in use, since this system enables up to hundreds of axes at the same time.



(a) BigTow winch. (b) BigTow Lite winch.

Figure 3.6: Stage Technologies winches [11].

3.1.3 Navigator Automation System™

The Navigator Automation System™, developed by FisherTech is largely used in live events all over the USA due to its flexibility, which enables the show to have as many components as the venue's needs. So that, from a small theatre to a larger one the system can vary from a simple configuration of one machine to a really complex one made up of thousands of components.

The Navigator Automation System™ has the particularity that there are no core where the power systems are plugged in. Each separated network has its own processor which can work either by itself or as part of other network. Once this separate modules (networks) are connected they are capable of sharing information in real-time, avoiding the boundaries between processors, making the whole system (all networks together) one computer out of many independent processors.

The flexibility provided by this system of various networks working together makes it almost impossible to kill, which is very important in big productions, because when a processor fails it does not mean that the entire show is affected and the show can go on.

So, in short, this technology allows the system to be part of any production, from a small one to a large one, but also very reliable due to the network system the FisherTech came up with, as it was reported above [4].

Software

The Navigator Automation System™ has one of the most developed interfaces of the market (Figure 3.7). However, there is no information about its specifications, so it is unknown if it is just a toolbox for any existent software or a software built from the ground.

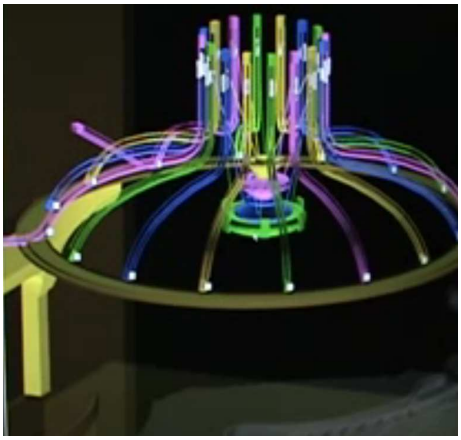


Figure 3.7: Aspect of the 3D profiling tool of the Navigator Automation System™ [4].

Hardware

Navigator Automation System™ has many winches to offer, from ones with servomotors to others with three-phase induction motors, similarly to the Stage Technologies™.

However, it is important to highlight one winch in particular, the G-Winch, shown in Figure 3.8, due to its simpleness and flexibility since it allows all types of actuators, either Direct Current (DC) motor, servomotors, Alternating Current (AC) motors or even hydraulic and pneumatic motors .

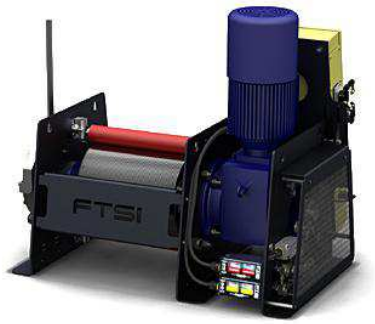


Figure 3.8: G-Winch Navigator Automation System [4].

3.1.4 Stage Command™ System

The Production Resource Group® (PRG®) set a new standard for stage automation in 1988, when it introduced the patented Stage Command™ System (SCS) motion control system on *Phantom of the Opera*.

At the core of the SCS there are three main components designed to work as a system or as an individual product. This system is very similar to the StageTech's one, using a self made console (Figure 3.9(a)), where all data are inserted by hand or by means of a joystick. Then, the information is sent to the PLC cube - the 'brain' of the operation

- which hosts as many PLC as needed (Figure 3.9(b)), and then sent to the drive rack (Figure 3.9(c)) where all the axis interpolation drives, which are responsible for sending the control signals to the motion effects, are installed [8].

Finally, the wide range of controlled effects include the SCS Winches, a standardized line of winches, and the SCS Turtle Rotator, which brings rotational motion to a wide range of scenic elements [8].

(a) Command™ console (b) PLC cube (c) Drive rack

Figure 3.9: Stage Command™ System features [8].

Another example of application of PRG®'s SCS was the most expensive musical ever on Broadway, *Spider-Man: Turn Off the Dark* (2009-present). This musical was a massive technological achievement that required both meticulous engineering and the development of new technologies to create unprecedented production effects, as mentioned in Chapter 1 (e.g. the 3D flying system used above the audience).

3.1.5 Pegasus Automation System from Flying by Foy™

Flying by Foy™ is a company which is emerging as leader of flying scenes simulation this century, competing side-by-side with other big companies like the ones already mentioned. They are already working on the core of the entertainment industry, having participated on TV shows like MTV Video Music Awards or Tony Awards, and musicals on Broadway such as *Tarzan* (2006-2007), *Mary Poppins* (2006-present) or even more recently the award winning *Ghost the Musical* (2011-present).

This company has a package to offer which includes everything needed to create a flying scene, as shown in Figure 3.10. They offer software to create virtual movements, consoles to interact with the software, and even the hardware which makes the 'dream' come into reality [5].

Figure 3.10: Flying scene using Foy's Pegasus Automation System^{TM} [5].

Software

The software used on Pegasus Automation System^{TM} is the Pegasus R3 (Figure 3.11), a Microsoft Windows Virtual Studio motion control package, which allows the creation of virtual paths, ready to be exported to the processing unit.

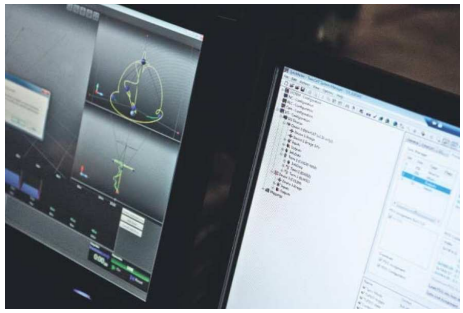


Figure 3.11: Foy's Pegasus R3 - Software [5]

Hardware

- **DMAC** - This is the Foy's largest console and it is ideal for multi-axes scenes once it has four submasters, three assignable joysticks and twelve programmable macro buttons. Together with the software Pegasus R3, this console became a powerful tool to design and control complex scenes.
- **LXE** - This is the medium console of the company, perfect for small environments with a maximum of five axes to manipulate. Although this console can work alone as master, it can also work as a satellite console of a larger installation.
- **SAC** - This console is the most basic one of the company, capable of managing only one axis. This suits perfectly to stunt effects and acrobatic jumps or flying scenes for films.

The Flying by Foy^{TM} has a wide variety of winches to offer, from slower to faster winches to push the performers to the limits. The most relevant winch is a battery powered one (BR-X Wireless 3-Axes Motion), which makes it capable of creating almost every configuration required [5].

3.2 Supporting Technologies

In order to create a 3D motion control system like the ones previously mentioned, it is important to have in mind which type of technologies are available and which are their positive and negative aspects. So, in Figure 3.12, a chart is presented to summarize the devices needed to develop a 3D motion control system.

First, the computational software gives instructions (position, speed and acceleration) to the control system (motion control unit), which processes the information to send thereafter to the drive. This last device is connected to the motors to supply the required power. The motor's position is controlled in a close loop by receiving the feedback from the encoders/resolvers.

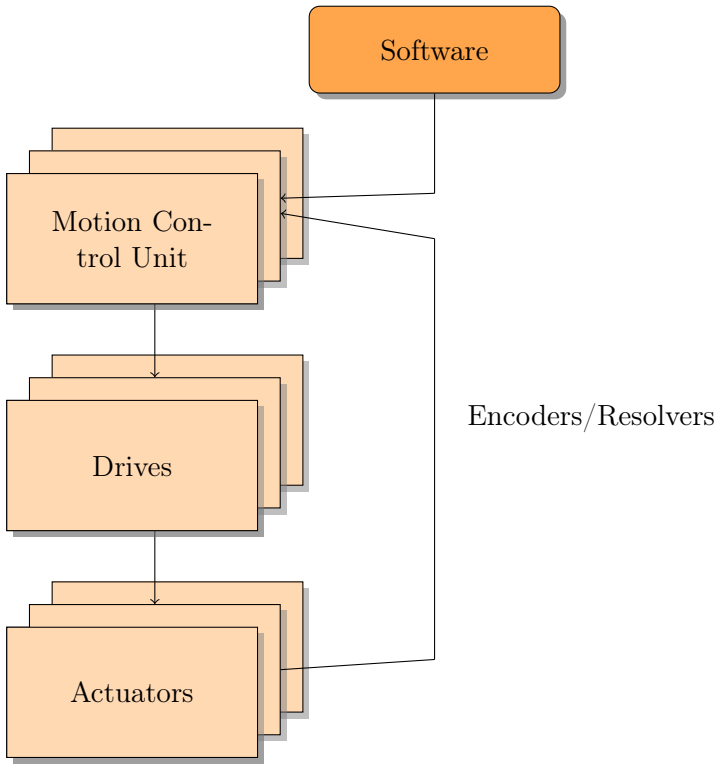


Figure 3.12: Drive and command chart for a 3D motion control system.

3.2.1 Software

For the graphic interface there are several options for programming such as C or C++ language or even Python. However, the key point while programming a software for a 3D motion control system is the 3D modeling tool.

For that, there are open source libraries like GIMP Toolkit (GTK+) and OpenGL, for C/C++, which are defined below.

GTK+

This is a multi-platform toolkit which enables the user to create graphic interfaces, offering a complete set of widgets. This library is suitable for a wide range of projects, from smaller to more complex applications.

GTK+ is multi-platform which means that the user will not have any restriction on using the software either in Windows or Linux, for example. Another positive aspect of this toolkit is that it was written in C, but was also designed from the ground up to support a large variety of languages, not only C/C++, but also languages such as Perl and Python (especially in combination with the Glade GUI builder) [13].

OpenGL

Since 1992, OpenGL is the industrial application most widely used for 2D and 3D graphic interfaces. Similarly to GTK+, OpenGL is an open source library, and it is also multi-platform.

From a simple 2D drawing to a complex 3D realistic animation, the user can exploit the OpenGL capabilities. These capabilities vary from creating shapes to geometrical transformations, or even playing with the lighting of the scene [7].

3.2.2 Motion Control Unit

After taking a look at some options for the development of the graphic interface, some solutions for the motion control unit are going to be presented.

Generally, in the industry, the movement of some type of machines is commanded either by Programmable Logic Controller (PLC) or by numerical controls (NC), so it is important to take a look at the advantages and disadvantages of both options.

Programmable Logic Controllers

Programmable Logic Controllers are largely use for controlling machine units in the manufacturing automation industry. They use a unique language called ladder logic. Although other languages may be used, ladder logic presently remains the dominant language of automation [20].

The PLC is sometimes called a Programmable Controller (PC), but the abbreviation PLC is preferred in this study to distinguish it from the Personal Computer. In the 1960's, a typical automated manufacturing line had a cabinet to store the relays used for controlling the operation. However, the old relay system was hard to debug and was really time-consuming.

The architecture of a modern PLC is composed by a processor, a power supply, and input/output (I/O) modules. In order to program the PLC, a programming device (usually a Personal Computer) is connected to the processor through a serial port, or remotely through a local area network to send the program previously prepared and compiled [20].

As it was described in the previous section, about the existing solutions in the market, the PLC is the most widely used control unit in the entertainment industry.

Numerical Control

The NC first appeared in the 1940's and it is a type of equipment usually associated to large machine-tools for position control.

Nowadays, with the technological evolution, these commands have a more powerful processing capacity originating an integration with highest level devices. Thus, the manufacturing flexible systems arose, contributing to an improvement of the computerized production lines [21, 32].

Numerical Control is the equipment which gives the drive the reference parameters, sending analogical outputs to power it on and to set the forward and reverse signal, relating axes in use by providing interpolation (linear or circular) between them. This equipment can also look ahead, which means that it can adapt the next action based on an instruction that comes ahead in the code [21].

For a higher flexibility, the NC sometimes is associated to a PLC, sharing some variables. Thus, the PLC is responsible for managing input and output digital signals, which allows a robust control unit, since signals sent from the NC can be managed according to the signal the PLC had received.

Usually, NCs are the brain of the system, since it has the NC device and the PLC (if it is necessary) altogether. The user only has to make the NC program and sometimes the PLC program in ladder language. These consoles also use digital inputs for encoders and outputs to connect the motors [32].

3.2.3 Actuators

There are several types of actuators capable of executing the information that comes from the motion control unit. The most common systems in the entertainment industry use either electric motors or fluid power cylinders (either hydraulic or pneumatic). Then, the selection process starts by choosing whether a motor or fluid cylinder is more appropriate for a given application. So, it is important to take a look at the advantages of each type of motor and cylinder [23].

Electric Motors

Actuators most widely used in theatrical mechanical design are electric motors due to their advantages, such as:

- The widespread availability;
- They are well known, making it easier to get advise for a proper use and maintenance;
- The relatively low cost for the most commonly used motors, as well as the corresponding speed control;
- The wide range of sizes and configurations available.

In addition, these type of actuators use to be large and heavy and are mainly high speed rotary motion power source, which generally means that they are attached to a reducer gearbox in most applications [23].

In the market, electric motors are available in many sizes and styles. In this section it will be analyzed the three most common types of electric motors used in stage equipments:

- Permanent Magnet DC (DC Motor)
- Three Phase AC (AC Motor)
- Servomotor

Permanent Magnet DC

Before the appearance of the frequency inverter drives, the permanent magnet DC motors provided the best method for controlling the speed, using a powerful technique for controlling analog circuits called Pulse Width Modulation (PWM). The PWM supplies energy through a succession of pulses. By increasing/decreasing the pulse width, the controller can regulate the energy that flows to the motor shaft [15, 2].

Nevertheless, because of the frequency inverter drives and their decreasing price, the DC motor has become less common in some applications, but it still provides superior speed-torque characteristics compared to an AC motor/frequency inverter drives combination.

Some theatres and commercial shops still use the DC motors to avoid the cost of replacing them to AC motors and because of their proven reliability.

The DC motors are brush-type motors which means that a carbon brush makes electrical contact with the wings of the armature as the motor spins. However, due to the friction, the brushes have to be replaced regularly, for motors that are spinning for long periods of time. But for typical stage and film applications, this is not an issue since the operation cycles are very short, because shows last for a short period of time, unlike the general manufacturing industry, whose operation cycles can last for days in continuum [23].

Three Phase AC

The three phase AC induction motors - asynchronous motors - are the most commonly used nowadays because they are relatively inexpensive, extremely reliable and do not require a lot of maintenance in typical stage and film use.

These actuators combined with appropriate motor drives (e.g. frequency inverter drives) can provide:

- High torque for a certain speed range;
- Smooth reversing;
- Quiet performance.

These are the main advantages that make AC motors the most largely used by mechanical designers for their stage and film designs [23].

Servomotor

Servomotors, usually use AC synchronous motors and are commonly selected for high positioning accuracy, exceptional performance and quick reversing tasks. These motors are commonly manufactured with a shaft mounted feedback device (called resolver) and have a specific drive to control them. One of the advantages that servomotors present is that they provide higher power to size ratio than other motors. Servomotors are quite different from standard motors: they do not conform to the same mounting configurations as standards as other motors (either DC or AC), meaning that they are unable to share gear reducer and other equipment with standard frames [23].

However, unlike the above described servomotors, which are inherently servo-actuated systems, AC asynchronous motors with feedback devices can also be treated as one, without being a servomotor by itself.

Fluid Power Actuators

Fluid power actuators can be either hydraulic or pneumatic. They are very similar, but their operating fluid is different. For hydraulic actuators, the fluid is oil and for pneumatic ones it is the air.

Hydraulic Actuators

There are two main configurations for the hydraulic actuators: linear and rotary actuators. These systems are very similar to their pneumatic equivalent actuators. They differ mainly on the fluid type. Nevertheless, hydraulic actuators advantages are:

- Easy speed control;
- Immense force and torque are simply obtained;
- Speed reduction is not often needed;
- Generally quiet;
- The linear motion is easy to achieve;
- Very compact;
- Relatively safe and reliable.

However, there are disadvantages inherent to hydraulic actuators, like the following ones:

- High initial investment;
- Noisy pumps;
- It is nearly impossible to synchronize multiple actuators without mechanical or electronic means.

Pneumatic Actuators

As reported above, pneumatic actuators are available in same basic configurations as the hydraulic ones, as well as some other specific configurations, like air bearing and gas springs.

Pneumatic systems are designed to operate under lower pressure than the hydraulic actuators, so their components are generally smaller and rated for lower loads.

The main advantages of pneumatic actuators are:

- Less expensive than the hydraulic ones;
- Small, simple and relatively powerful (less than the hydraulic ones);
- High speed;
- Compact power source;
- Environmental friendly, since the fluid is the air;

However, there are also some disadvantages inherent to pneumatic actuators, such as:

- The speed control is a bit touchy, hard to predict accurately and sensitive to friction in the system;
- Compressibility can cause surging motion;
- Stick-slip effect at very low speeds;
- It is nearly impossible to synchronize multiple actuators without mechanical or electronic means. This behavior is similar to the hydraulic actuators.

3.2.4 Brakes

The actuators are usually attached to motor brakes, whose major use on theatre machinery is to hold loads in place rather than to slow or stop moving loads. These breaks keep pieces of scenery up in the air, when their winch motor is off. Usually, a flying system has two completely independent brakes as part of a single failure proofing scheme, as shown in Figure 3.13.

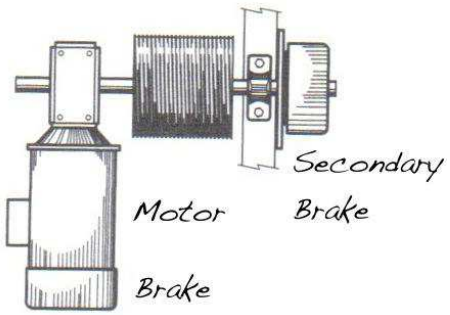


Figure 3.13: Aspect of the location of common brakes in a motor [23].

Brakes Placed on the Motor

Brakes assembled on the motor (Figure 3.13) usually have torque ratings in the range of 1 to 3 times the motor's torque at rated horsepower.

Brake motors are largely used in stage machinery, but due to their location they cannot hold the load if a chain or gearbox fails. So, if it is decided to use only one break in the system and it is placed in the motor, this system should only be used for movements without the need of lifting force [23].

Brakes Placed on the Drum Shaft

For situations where gravity would run scenery (Figure 3.13), a secondary brake is mounted on the drum shaft in case of gearbox failure. So, the torque ratings of these brakes need to be much larger than the brakes placed on the motor shaft. As an example, the drum shaft break used in Navigator Automation System^{TM}'s *Phantom of the Opera* was rated at 1017 N.m. Brakes like these are generally released by fluid power. These brakes give the systems more safety [23].

3.3 Main Challenges to this Thesis

After analyzing the market offer, the technologies proposed in the commercial solutions, and their strong/weak points, it seems relevant to try to identify the main challenges to take into account for the development of this project.

As previously described, a relevant issue appears for low budget productions, since the studied interfaces are attached to expensive closed software.

Another issue arises when analyzing the existing solutions. It is concerned to the flexibility of existing software. Highly flexible solutions have several strong points, such as allowing up to hundreds of commanded axes at the same time. Then, another important problem comes up, which is the extremely high computational cost, since all the calculations are done in real-time by the motion control unit, meaning a more expensive product.

Then, the challenges to this project are the quest for a system with:

- an open source and multi-platform interface, responsible for the majority of the calculations;
- a low cost graphic interface;
- a new control unit, with axes interpolation embedded;
- a less real-time processing;
- complete flexibility in the trajectory definition process.

Part II

Methods and Development

Chapter 4

Mathematical Definition

Before starting the development of the system itself, it is important to understand the mathematical concepts behind a 3D flight simulator (3D motion system).

Usually, a 3D motion system has either a triangular or a square prismatic workspace. So, for a larger cover of the space, a square prism was chosen in the present project. This system can be approximated to a parallel robot model, with four prismatic joints, schematically represented in Figure 4.1. It consists of a mobile and a static platform, and four prismatic joints attached to both platforms, using spherical joints. Nevertheless, these joints are going to be indirectly actuated [34].

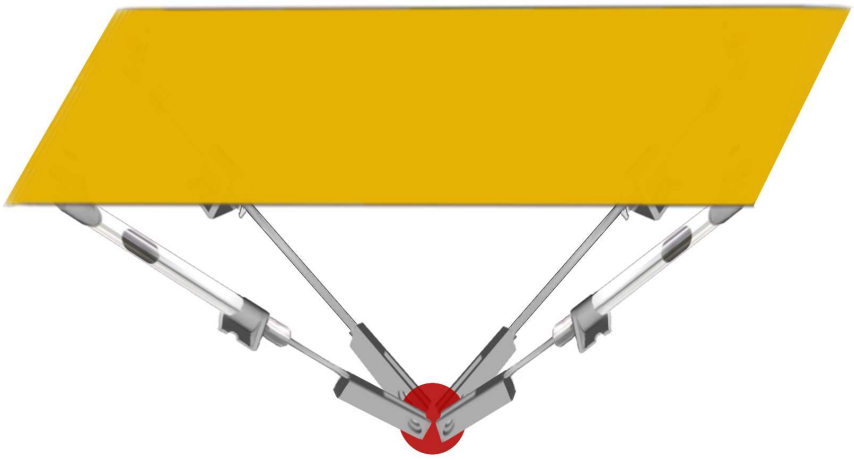


Figure 4.1: Parallel robot model approach, with four prismatic joints.

For a better understanding of the variables present in the system, a simple representation is shown in Figure 4.2, where the global reference is placed in the center of the static platform, and the local reference is in the mobile device.

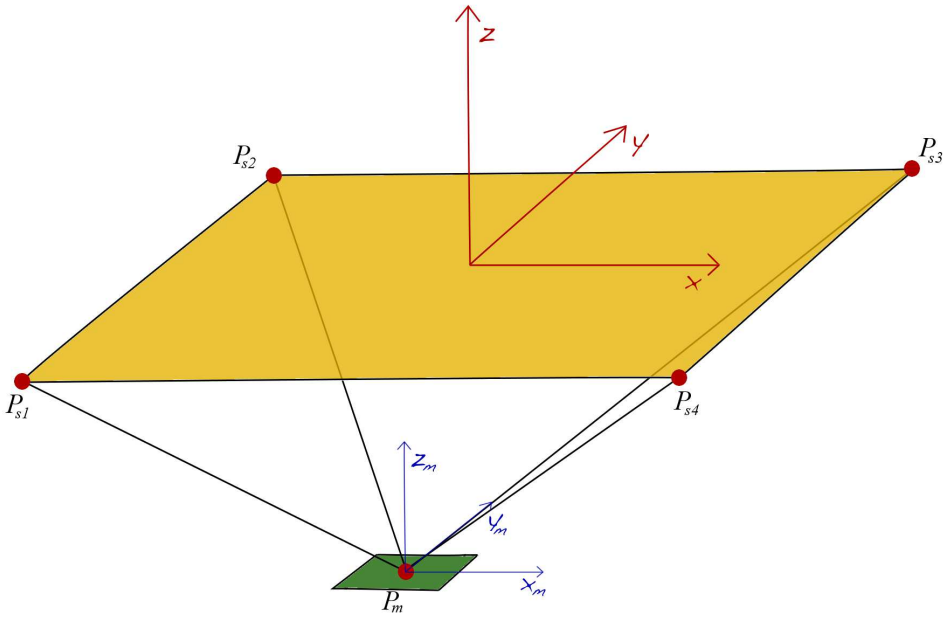


Figure 4.2: Schematic representation of a parallel robot system with four joints.

Thus, the P_{si} (i=1, 2, 3 and 4) - the four corners of the static platform - and the P_m - the intersection of all the prismatic joints into the mobile platform - define the length of each prismatic joint (L_i) and are given by equation 4.4.

$$L_i = ||P_{si} - P_m|| \tag{4.1}$$

The four prismatic joints were designed in this project to intersect the mobile platform in the same point, since this platform is supposed to be parallel to the ground with the help of its own weight.

4.1 Kinematic Relations

There are two types of kinematic relations to take into account when developing a robot: the direct and the reverse.

The direct kinematic describes the Cartesian coordinates of the end effector knowing the joints variables. Then, reverse kinematic is the reverse process, which means that the coordinates of the end effector are known and the joint variables can be determined.

Figure 4.3 shows the kinematic relations systematized for a better understanding.

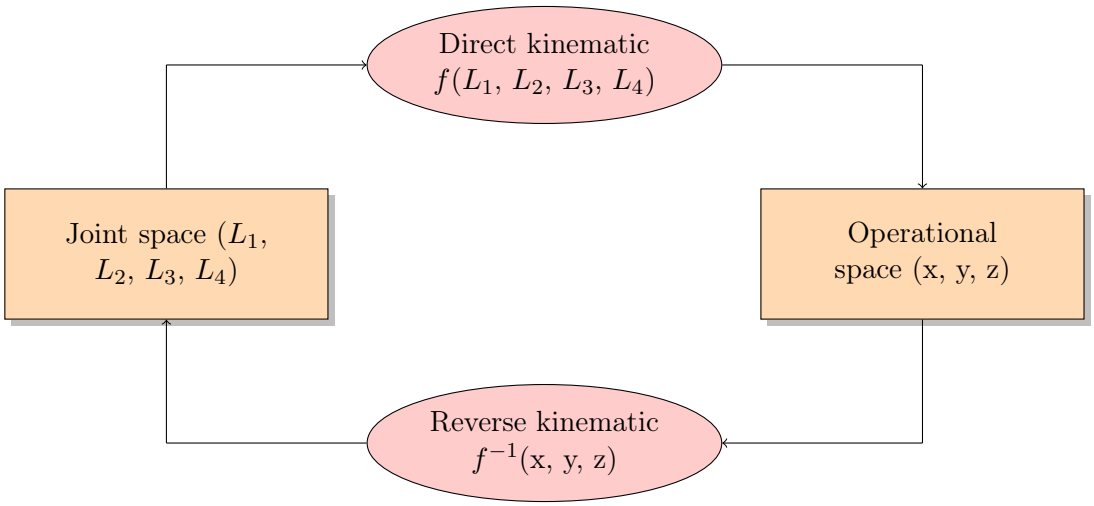


Figure 4.3: Kinematic relations.

For the development of the present project, the direct kinematic will not be considered since the movements are done after the knowledge of the Cartesian coordinates, in order to send the length of the prismatic joints to the control unit. The direct kinematic could be helpful in the beginning of the movement to position the system manually, when the mobile platform was set into the home position. However, it would be meaningless in the present study because the encoders reset their value every time they are powered on (relative encoders) which means that the positions of the encoders (directly related to the joint variables) are not memorized. So that, the mobile platform has to be placed in its home position by hand, and then axes are reset again.

4.1.1 Reverse Kinematic

After knowing the translation (x, y, z) of the mobile platform, the length of the prismatic joints (L_i) can be calculated, as long as the coordinates of the vertices of the static platform, P_{si}, remain fixed:

$$P_{si} = \begin{bmatrix} X_{si} \\ Y_{si} \\ Z_{si} \end{bmatrix}, \ i = 1,\ 2,\ 3\ and\ 4; \tag{4.2}$$

Then, the coordinates of the intersection of the four joints in the mobile platform, P_m, are a consequence of the translation from the origin to that point, which means that P_m is:

$$P_m = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{4.3}$$

So that, L_i is given by the expression:

$$L_i = \sqrt{(x - X_{si})^2 + (y - Y_{si})^2 + (z - Z_{si})^2} \tag{4.4}$$

These mathematical expressions will be useful for determining the new coordinates of the 3D motion system. These coordinates will be directly related to L_i.

Chapter 5

Software Development

A long-standing challenge in computer interfaces is to create performance animation systems that allow a user to intuitively control the motion [17].

The here proposed interface was chosen to be built from scratch rather than to create a toolkit for an existing closed software like 3ds Max or Microsoft Visual Creator, since the main goal of this software is to be free and multi-platform.

So, to develop a software there are some guidelines to follow, such as:

- to choose the programming language and to structure the code;
- to organize the graphic interface in order to be user friendly;
- to prepare the 'offline' calculations: the software does *à priori* all calculations the analyzed solutions do in real-time, as mentioned in Chapter 3. For example, the transformation of the Cartesian coordinates into the 'new' coordinates which will be read by the motion control unit.

5.1 The Choice of the Language and Code Structure

To develop the interface for the 3D motion control system, the chosen language was C since it has everything required and gives the programmer sufficient freedom to allow his ideas to come true.

The interface was built using the GIMP toolkit, GTK+, and the visual manager Glade which enables the programmer to create the interface by dragging buttons, text boxes, drawing areas, among others, into the main window.

However, the main purpose of this interface would not be achieved using the GTK+ by itself, because it only allows for 2D drawings. So it became mandatory to merge a 3D drawing area into the main window using the 2D drawing area provided by GTK+. Thus, a 3D drawing area was created using functions of the OpenGL library.

In addition to these two libraries, the header 'joystick.h' was also used in order to receive all the signals that came from the joystick.

In the present study, the code structure was split in two main parts: the Parent and the Child. By this way, the Parent was responsible for the communication with the joystick and the Child was responsible for managing the interface. The joystick data, which are received by the Parent can be used by the Child since they are allocated in a shared memory, as shown in Figure 5.1.

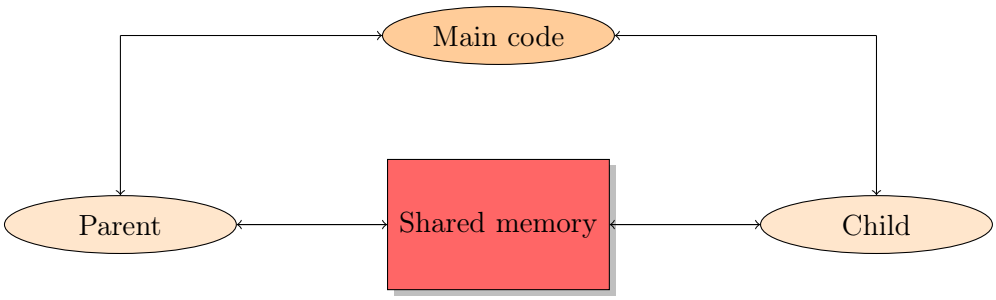


Figure 5.1: Proposed code structure.

The Child is the responsible for managing the interface, so it is also responsible for calling functions from another file called 'callbacks.c', where each required callback is created. Therefore, the Child is the one who makes all the calculations and where the callbacks are called to set values in the interface. It is also responsible for refreshing the drawing areas.

So, in short, the Parent code is responsible for:

- establishing the communication with the joystick;
- getting the signal from axes (16 bit) and buttons (binary) of the joystick;
- storing the information in the shared memory, in real-time.

On the other hand, the Child code has to:

- get the information from the shared memory, in real-time;
- manage the windows of the interface;
- store all the relevant information;
- create a user friendly 2D and 3D interface.

5.2 Areas of the Proposed Interface

After structuring the code it is important to organize the interface so that it becomes user friendly. A well-designed interface gives the user an efficient and effortless flow of information between the device and its operator. When users are given sufficient control over the creative process, they can develop efficient interaction strategies that optimize their communications to the machine [19]. Thereby, the interface was thought to be simple and intuitive, as shown in Figure 5.2, with each area clearly defined.

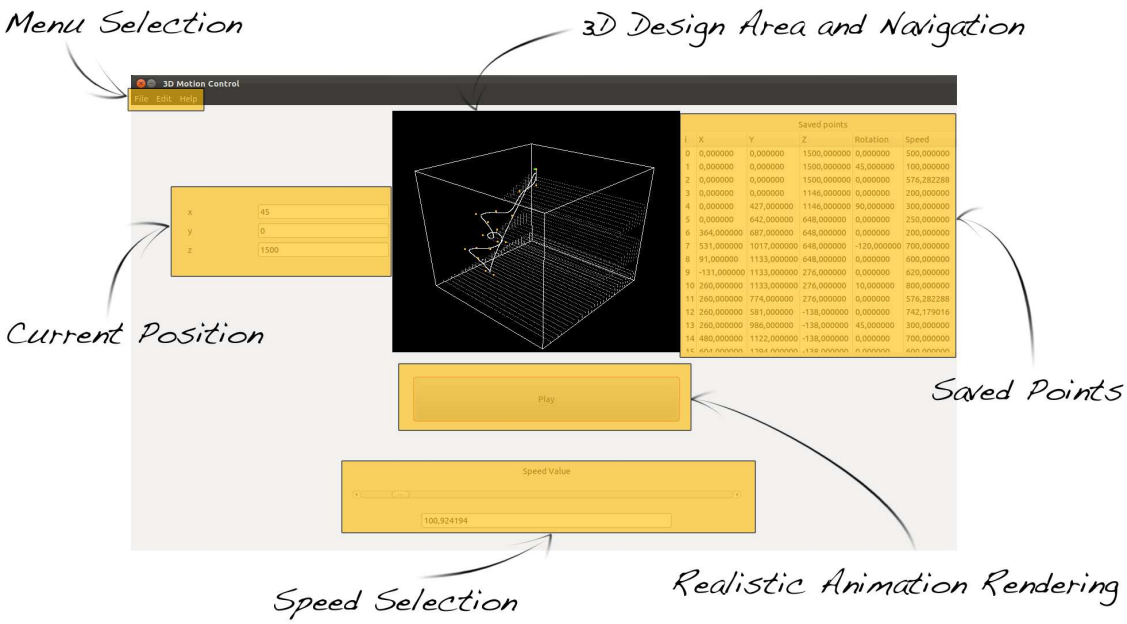


Figure 5.2: Areas of the proposed interface.

5.2.1 Current Position

The current position area (Figure 5.2) is composed by three 'entries' where the current point's coordinates are shown. However, these coordinates do not come from the joystick with the exact value, since they vary according to the venue (workspace) dimensions. The first step is to understand which signals come from the joystick. This device has two types of values, binary for buttons and numeric for axes. To vary the Cartesian coordinates of the current point, each coordinate has to be related to one axis of the joystick, so *X value*, *Y value* and *Z value* vary from -32767 to 32767. Nevertheless, this signal still has to be transformed into a coordinate. So, once the origin is in the center of the rectangular prism, which represents the venue, each increment (x_{inc}, y_{inc}, z_{inc}) must be according to equations 5.1, where the *X dimension*, *Y dimension* and *Z dimension* are dimensions of the venue in the corresponding axis.

$$\begin{aligned} x_{inc} &= \frac{Xdimension}{2}\frac{Xvalue}{32767} \\ y_{inc} &= \frac{Ydimension}{2}\frac{Yvalue}{32767} \\ z_{inc} &= \frac{Zdimension}{2}\frac{Zvalue}{32767} \end{aligned} \tag{5.1}$$

So, the instantaneous coordinates are:

$$\begin{aligned} x &= x_{prev} + x_{inc} \\ y &= y_{prev} + y_{inc} \\ z &= z_{prev} + z_{inc} \end{aligned} \tag{5.2}$$

The initial x_{prev}, y_{prev} and z_{prev} values are the coordinates of the device when it is on the top of the venue: (0, 0, $\frac{Z\ dimension}{2}$). Thereby, these coordinates are automatically sent to the drawing function which draws a green point representing the current point.

5.2.2 Saved Points

If the user wants to save the current point (Figure 5.2), a specific button on the joystick must be pressed, and hence, it becomes orange. Then, the point is added to a list which stores each saved point by order. However, just saving a point does not mean anything to the user. But a visualization of the path created through the saved points could be more enlightening. The latter could have been created by simple linear interpolations between points (Figure 5.3(a)), but the visual effect would not be realistic since the quest requires smooth movements [22]. So, this path is designed using a clamped cubic B-Spline curve, according to equations 5.4 to calculate the new function of interpolated points, $S_i(t)$, where i varies according to the number of saved points, $t \in [0, 1]$ and is given by the equation 5.3 [25, 27].

$$t = \frac{j}{n\ points} \quad 0 < j < n\ points \tag{5.3}$$

where $n\ points$ is the number of interpolated points between two consecutive saved points.

$$S_i(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} P_i \\ P_{i+1} \\ P_{i+2} \\ P_{i+3} \end{bmatrix} \tag{5.4}$$

Then, the created point in each iteration has coordinates according to equations 5.5, where i is the previous saved point and $i+1$, $i+2$ and $i+3$ are the three following ones, whenever it is required.

$$\begin{aligned} x &= b_0 \times x_i + b_1 \times x_{i+1} + b_2 \times x_{i+2} + b_3 \times x_{i+3} \\ y &= b_0 \times y_i + b_1 \times y_{i+1} + b_2 \times y_{i+2} + b_3 \times y_{i+3} \\ z &= b_0 \times z_i + b_1 \times z_{i+1} + b_2 \times z_{i+2} + b_3 \times z_{i+3} \end{aligned} \tag{5.5}$$

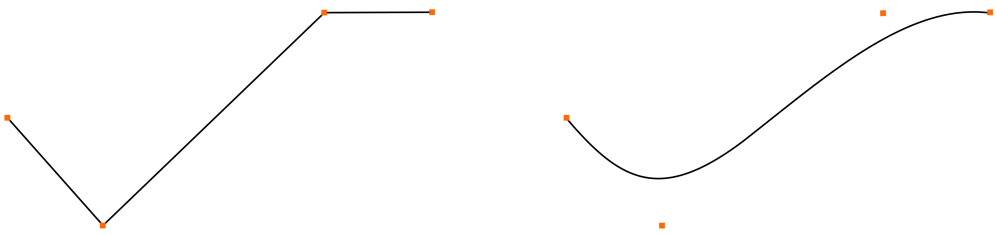
Constants b_0, b_1, b_2 and b_3 are given by equations 5.6 [25].

$$
\begin{aligned}
b_0 &= \frac{it^3}{6} \\
b_1 &= \frac{3\times t^3 - 6\times t^2 + 4}{6} \\
b_2 &= \frac{-3\times t^3 + 3\times t^2 + 3\times t + 1}{6} \\
b_3 &= \frac{t^3}{6}
\end{aligned}
\tag{5.6}
$$

where,

$$it = 1 - t \tag{5.7}$$

Once the interpolated line is created, it is stored in a new list of interpolated points in order to generate the white line that illustrates the movement of the flying system, as shown in Figure 5.3(b).



(a) Linear curve. (b) Clamped cubic B-Spline curve.

Figure 5.3: Examples of interpolated lines.

Every time a point is added, it is appended to a 'treeview' which shows all points already saved. However, the user may wish to delete a point during the creative process so that the point can be selected with the mouse and deleted by pressing the key 'delete'. Along with this, every time a point is appended to the 'treeview', a pop up window shows up in order to select the angular position of the harness on that coordinate. This angular position is the fourth degree of freedom of the system.

When a row is selected and the key 'Enter' is pressed, the angular position is changed.

5.2.3 Speed Selection

To simulate the entire scene as accurate as possible, it is important to set up the speed of the flying device in each saved point (Figure 5.2). To accomplish the latter, a slide bar has been added so the user may act upon its adjustment. There are two ways of attaching the speed to a saved point:

- **Predefined mode** - prior to save a point, the user adjusts the slide bar to the intended value and then, when the button for saving points is pressed, this speed is saved.
- **Replacing mode** - after saving a point with a certain speed, the user can change this value by adjusting the slide bar and pressing the key 'insert' of the keyboard.

This slide bar has a minimum value which is almost zero and a maximum of 200 cm.s^{-1}, the limit speed set in the numerical control, for safety reasons. Centimeter was the chosen metric unit for the software development.

Once this value is chosen, the animation time is calculated and it starts counting when the play button is pressed. So, to get the initial time, the header file 'time.h' is used to get the current time of the day. Then, after knowing the distance that the flying device has to travel in each part of the trajectory (by adding the distance between points) and the speed of that part of the animation, the software estimates the time required to complete that path, according to the equation 5.8.

$$animation\ time\ (s) = \frac{total\ distance\ (cm)}{speed\ (cm.s^{-1})} \tag{5.8}$$

Thus, the software has to get the coordinates of the flying device in each pulse. This pulse is responsible for refreshing the window every 20 ms (a proposed refreshing time), updating the position of the device, creating the illusion of virtual movement. Moreover, in each pulse the elapsed time from start (*time since start*) is calculated by getting the current time and subtracting the initial time. Therefore, in order to know the position of the flying device, the software chooses one point, from the list of interpolated points (with n points), according to the equation 5.9.

$$point = \left(\frac{(time\ since\ start\ (s) + time\ added\ (s))}{animation\ time\ (s)} \times n \right) + i \times n\ points \tag{5.9}$$

where i is defined in equation 5.10, and $n\ points$ is a variable previously defined in the subsection 5.2.2.

$$i = number\ of\ saved\ points - 1 \tag{5.10}$$

The *time added* is the parameter that allows for the forward and backward option during the animation. Every time the forward or the backward button is pressed that parameter increases or decreases 5 s, respectively. This parameter was fixed in the present study and can be obtained by dragging the cursor of the timeline.

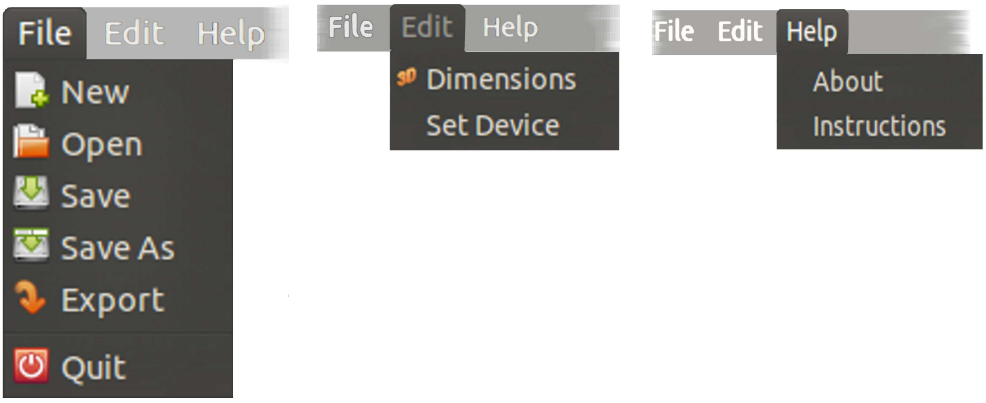
5.2.4 Menu Selection

This interface (Figure 5.2) replicates the most usual ones and therefore includes menus such as the 'File', 'Edit' and 'Help', as shown in Figure 5.4.

- The 'File' menu (Figure 5.4(a)) enables the user to create a new file or open an existing one, to save the new project or changes already made, and also to export the ISO file.

- The 'Edit' menu (Figure 5.4(b)) enables the user to reset dimensions of the venue and to choose the correct port of the joystick to start the communication.

- The 'Help' menu (Figure 5.4(c)) is only for showing the credits, and to help the user with some basic instructions about the software.



(a) File menu (b) Edit menu (c) Help menu

Figure 5.4: Proposed menus for the graphic interface.

To start a new project, the user has to select the 'File' menu and press the 'New' icon. Therefore, for a better approximation to the reality, the possibility of the user to choose the intended type of venue was added to the proposed graphic interface, from a single blackbox to an amphitheatre or even an arena. The user is also asked to set dimensions of the venue, as well as the existence, or not, of the Grand Circle and the Upper Circle in the amphitheatre mode (i.e. two types of sitting areas within the amphitheatre), and its angle and distance to the stage, as shown in Figure 5.5.

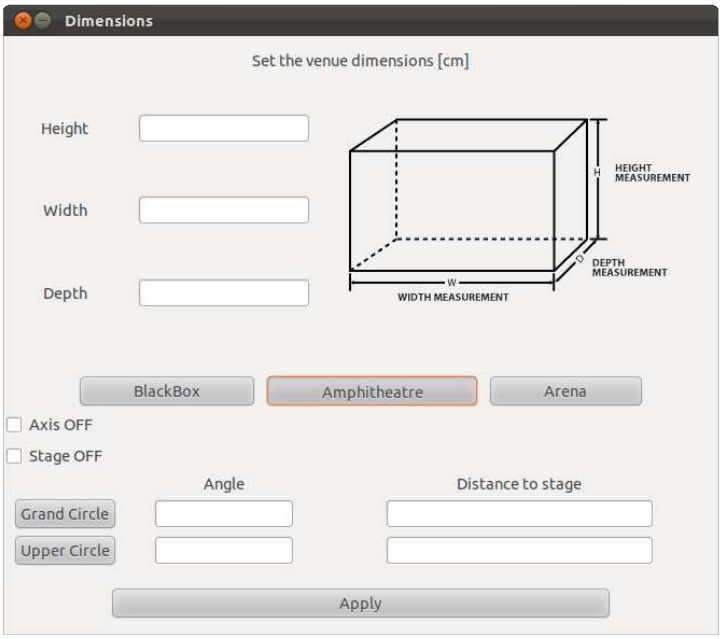


Figure 5.5: Proposed pop up window to set up the features of the venue.

With this additional information about the real constrains of the venue, every time a position is not physically reachable the user is alerted. Once these parameters are defined they are stored in a list for future use.

For saving the file by using the command 'Save' or 'Save As', the program registers the whole information, about the physical restrictions of the venue, before writing down the list of saved points (Figure 5.6).

```
//NC
/**********************
//
//FILENAME:3D_MOTION_CONTROL
//
//**********************
3000 3000 3000
1 20 1500
1 25 2000

0.0 0.0 1500.0 0.0 300
-13.0 292.0 1500.0 0.0 350
-13.0 796.0 1068.0 45.0 400
-13.0 954.0 798.0 45.0 400
-13.0 1058.0 498.0 90.0 300
5.0 915.0 420.0 90.0 320
5.0 -537. 234.0 90.0 350
0.0 0.0 1500 0.0 300
```

Figure 5.6: Example of a saved file.

After the header, the information about the settings of the venue is stored: firstly, with the dimensions and its type, and secondly, the information of the Grand Circle and the Upper Circle. Once the latter has been accomplished, the X, Y, Z coordinates, the angular displacement of the harness and the speed in each saved point are stored in lines.

The program only has to do the reverse process to open a file. But, first of all, it must be able to detect whether the file is supported or not by the proposed graphic interface. The header of each saved file is quite relevant since the software compares it with a predefined one. If it matches the expected header, it means that the file is supported. Then, it reads each line, picking all the information up and storing it in the corresponding list.

The last icon of the 'File' menu ('Export', Figure 5.4(a)) will be explained in detail, in the last section of this Chapter.

In the 'Edit' menu, the first icon shows a window similar to the Figure 5.5. The other option of this proposed menu allows the user to select the correct communication port, as shown in Figure 5.7(a). If the communication is not established, a window will appear to warn the user (Figure 5.7(b)).

(a) Window to chose the correct port (b) Warning window

Figure 5.7: Proposed communication established with the joystick.

5.2.5 3D Design Area and Navigation

In this area (Figure 5.2), the user has a basic interface to visualize the created movement. In the present project, the benches, the stage and the system are represented in a minimalist way with points and lines using functions from the OpenGL library, as shown in Figure 5.8.

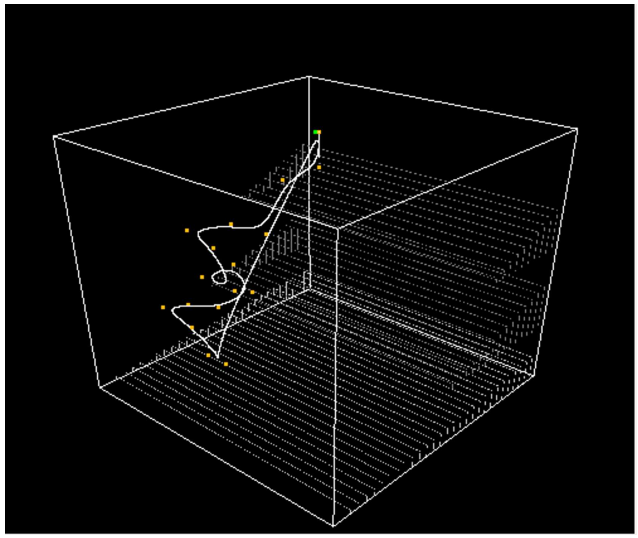


Figure 5.8: 3D design area.

Furthermore, in order to create the trajectory in the proposed 3D design area, it is mandatory to understand how to operate the joystick. So, the commands are explained in detail through the Figure 5.9 and the following items.

Figure 5.9: Instructions of the joystick.

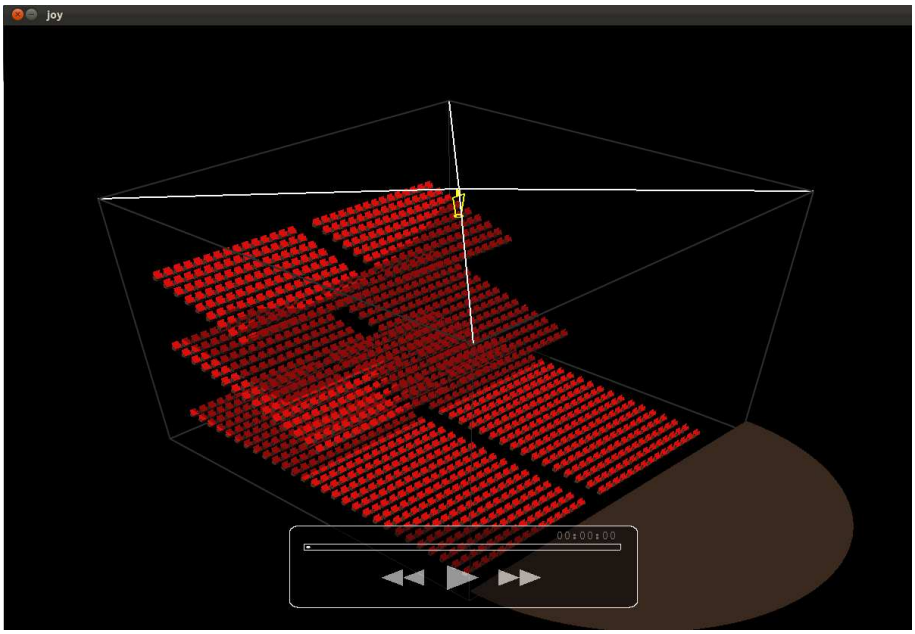
- *X axis* - to change the X coordinate, the user has to handle the joystick moving it to the left or right, to decrease or increase the corresponding coordinate.
- *Y axis* - to change the Y coordinate, the user has to handle the joystick moving it back and forth, to decrease or increase the corresponding coordinate.
- *Z axis* - to change the Z coordinate, the user has to push or pull the button forth or backward to increase or decrease the corresponding coordinate.
- *Save* - Once the user is satisfied with the point, it can be recorded by pressing the yellow button.
- *Remove* - This button allows the user to remove the point, not only from the 3D drawing area, but also from the list that stores the saved points.
- *Zoom* - To zoom in and out the image, the user only has to move the handle up and down.
- *Rotation* - This button is useless by itself. Once it is pressed and the joystick moves left or right or up and down, the image rotates, changing the view point.

5.2.6 Realistic Animation Rendering

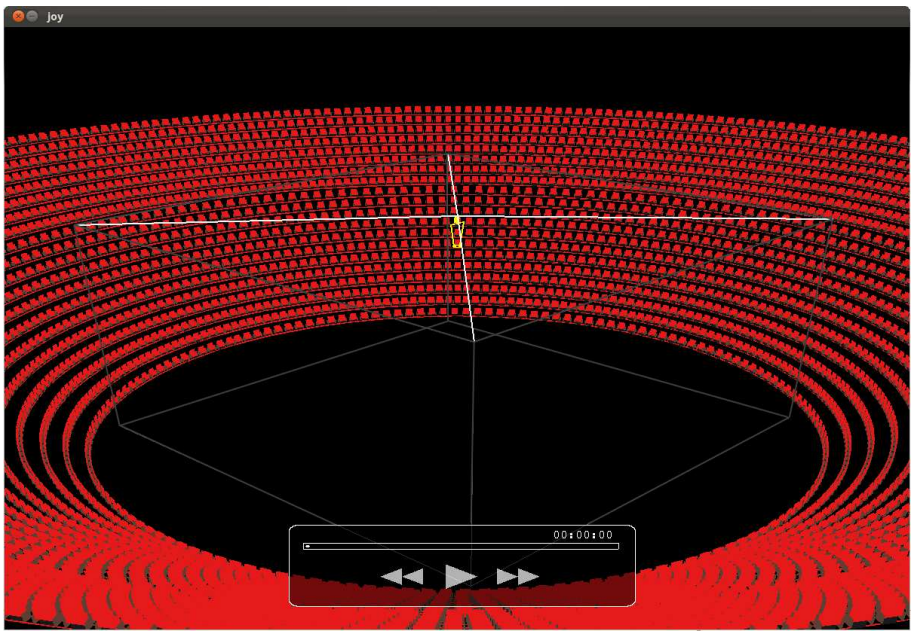
To compete side-by-side with the products already in the market, the developed software must include a realistic view to virtually simulate a real environment (virtual or synthetic environments -VEs) [26]. These environments are three-dimensional computer generated that occur in real-time as manipulated by the user (Figure 5.2). A VE is a projection

of either some real environment, a fairly realistic environment that does not exist, or an unreal environment [33].

In addition to these virtual recreation of a real scenario - generated by clicking the 'Play' button of the main window - a motion display for starting the animation (with play, backward, forward, and time counter options) inspired by some existing video visualizers, was created, as shown in Figure 5.10. Then the user has the opportunity to have a good perspective of the effect created.



(a) Amphitheatre



(b) Arena

Figure 5.10: Proposed rendering tool.

Realistic Environment

To develop this interface in order to give the user a more realistic environment, the lines that were representing the benches were replaced for 'real' chairs and the two degrees of freedom harness was drawn according to the conceptual idea.

In order to create every row of benches, two 'for' cycles were used: one to set each bench in a row, and another one to move to the next row when the previous one was full. However, this first approach was not the best choice, due to the high computational cost involved. So, this procedure was replaced by the following one:

- to check the number of chairs in each row, according to the size of the row;
- to check the number of rows, according to the size of the venue.

Then, the total number of benches is known and therefore with only one 'for' cycle - which goes from zero to the number of benches - each chair is placed accordingly.

Cables were also added to this interface, and every time a position is not reachable, the chairs become translucent and the lines become red. Thereby, the user is warned to change the path.

Thus, to start the animation, a motion display is embedded in this rendering tool, as it is going to be explained in the next topic.

Motion Display

The motion display includes a time counter, so that the user can have an idea about the duration of the scene. User can also drag the time cursor to go back or forth in the time line.

This display has three buttons: the play, the backward, and the forward which allow the user to simulate the entire movement. In order to create this area, the interface intersects the click of the left button of the mouse and checks if the cursor is over one of the three areas creating the corresponding action. Once the play button is pressed, it turns into a pause icon.

5.3 ISO File Creation

Before creating the file that will be sent to the motion control unit (ISO file), it is mandatory to transform the Cartesian coordinates into a four coordinates of the system (one for each motor), as shown in Figure 5.11. Therefore, the created ISO file is ready to be sent to the motion control unit by pressing the 'Export' icon of the 'File' menu. The latter, includes the information of how much they have to wind or unwind.

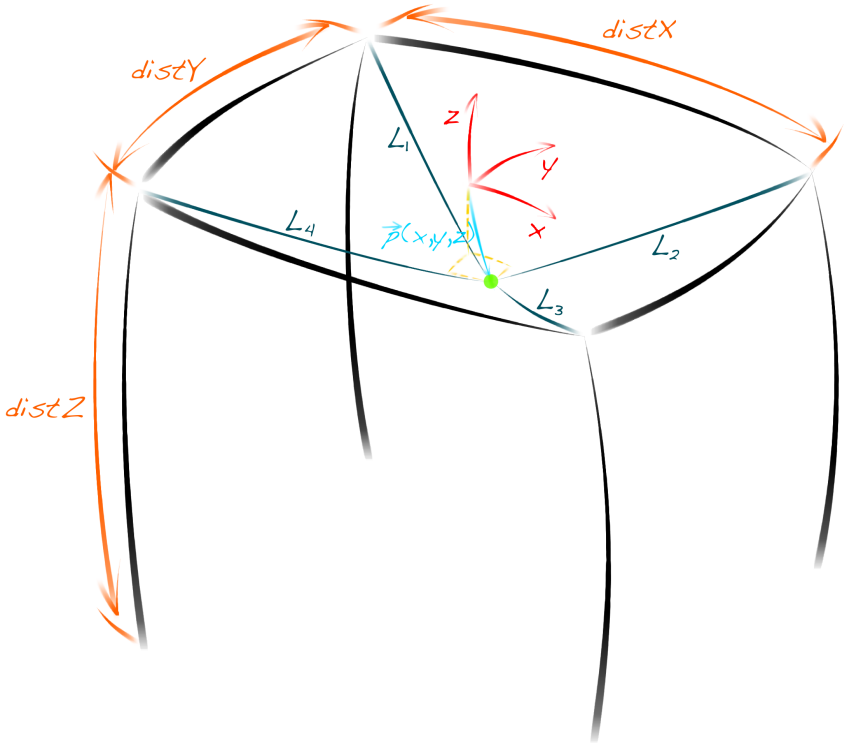


Figure 5.11: Cartesian coordinates *vs*. four coordinates of the system.

Every positioning system has to work in a closed-loop in order to guarantee an accurate position in each interpolation, so the motion control unit of this system will receive the feedback from the encoders. The new coordinates may be given as revolutions or as linear displacement. The linear displacement was chosen in the present project because the user is sensitive to this type of coordinates.

Knowing the current length and the initial one, the program gives the coordinate for such axis in the reported position, as it is demonstrated in equations 5.11, where equation 4.4 is applied.

$$\begin{aligned}
L_1 &= \sqrt{\left(\left(\frac{distX}{2}+x\right)^2+\left(-\frac{distY}{2}+y\right)^2+\left(\frac{distZ}{2}-z\right)^2\right)} \\
L_2 &= \sqrt{\left(\left(\frac{distX}{2}-x\right)^2+\left(-\frac{distY}{2}+y\right)^2+\left(\frac{distZ}{2}-z\right)^2\right)} \\
L_3 &= \sqrt{\left(\left(\frac{distX}{2}-x\right)^2+\left(\frac{distY}{2}+y\right)^2+\left(\frac{distZ}{2}-z\right)^2\right)} \\
L_4 &= \sqrt{\left(\left(\frac{distX}{2}+x\right)^2+\left(\frac{distY}{2}+y\right)^2+\left(\frac{distZ}{2}-z\right)^2\right)}
\end{aligned} \tag{5.11}$$

Thus, the initial point, (0, 0, $\frac{distZ}{2}$) has the following coordinates in the new system:

$$X_0 = Y_0 = Z_0 = U_0 = \sqrt{\left(\left(\frac{distX}{2}\right)^2 + \left(\frac{distY}{2}\right)^2\right)} \tag{5.12}$$

Using equations 5.11 and 5.12, the following coordinates are obtained:

$$\begin{aligned} X(x,y,z) &= L_1 - X_0 \\ Y(x,y,z) &= L_2 - Y_0 \\ Z(x,y,z) &= L_3 - Z_0 \\ U(x,y,z) &= L_4 - U_0 \end{aligned} \tag{5.13}$$

where $X(x,y,z)$, $Y(x,y,z)$, $Z(x,y,z)$ and $U(x,y,z)$ are the new coordinates of the system (X, Y, Z and U), an non-orthogonal system, which is written in the appropriate file, ready to be read by the motion control unit. This file needs other parameters such as the speed chosen for the animation as reported in section 5.2.3. This speed is given by linear displacement per second.

Chapter 6

Motion Control Unit Development

Instead of using a PLC it was decided to control the flying system using a NC, since it already provides interpolated axes and is largely used in positioning systems.

It is not very common to see this type of equipment in the entertainment industry, so it was thought that it might be interesting to try this new approach in flight simulators since they actually are positioning systems in a larger scale.

The NC used was the Siax 200 (Figure 6.1) from the Italian company Sipro. It is a four axes device and has a PLC for managing the digital I/O. The entire system includes thirty two digital inputs, thirty two digital outputs, ten analog outputs, four encoder inputs and a Rs232 port for communication.

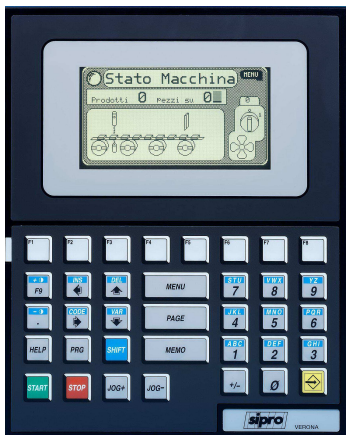


Figure 6.1: Aspect of the Sipro's Siax 200 NC.

6.1 Numerical Control

The parametrization of axes has to be done only if the winches change. So, it was decided to do it through the SiaxEd - a Sipro software - rather than creating a new software because Sipro uses a closed protocol to communicate with the NC. Before accessing the NC, it is necessary to start the communication by choosing the port (COM1) and the baudrate (9600 bit.s^{-1}).

6.1.1 Parametrization

Machine parameters

Once the communication is established, the first step is to set the machine parameters (Figure 6.2). Main parameters to take into account are:

- **Acceleration Factor** - This parameter determines how brusque the acceleration change is at the points of passage. The chosen factor was 1, which is the default number, because although it means a more brusque change it will not cause any practical effect since the accelerations and decelerations in this system are set by the frequency inverter drive [29].
- **Maximum Radius Difference** - This parameter sets the difference between the radius of departure and the radius of arrival while using G2 or G3 instructions (circular interpolations) [29]. However, this parameter and the previous one are not important for the present project because the interpolations are done by the developed graphic interface, so the G-code instruction used was the G1.
- **Tangent Tolerance** - This parameter sets how accurate the movement is. It sets the maximum distance between the virtual passing point and the real path [29]. The tolerance was fixed in this system in 0.100 cm to guarantee a reliable approach between the virtual and the real path.
- **PLC Port** - When the communication between the PC and the numerical control is the COM1, the port to communicate with the PLC is the COM2, because there are only two available ports.

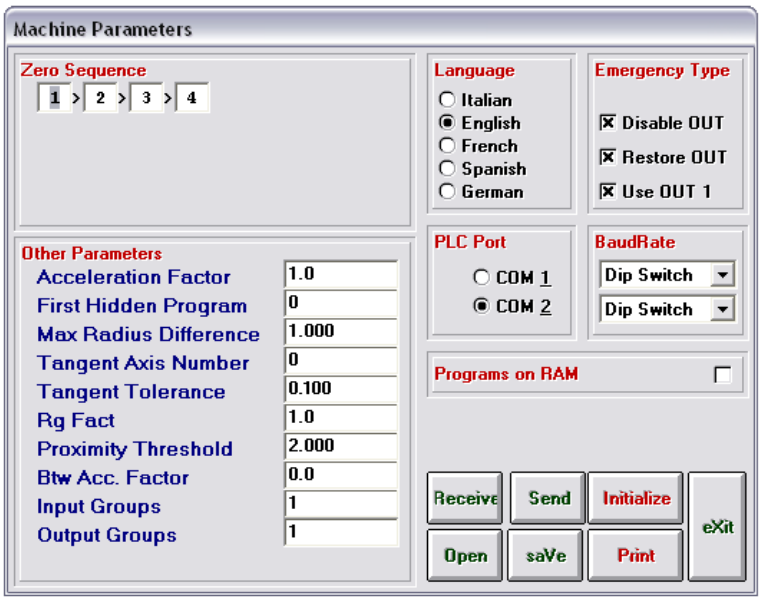


Figure 6.2: Machine parameters.

Axes parameters

After setting up the machine parameters, it is important to adjust the axes parameters in order to declare the hardware in use and establish the parameters of the system (Figure 6.3).

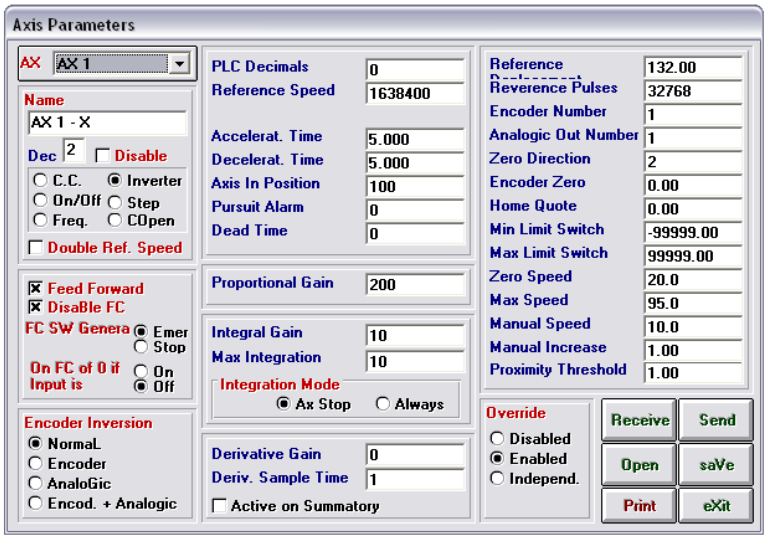


Figure 6.3: Axes parameters.

- **AX** - This list allows the user to parametrize each axis. In this system, all the axes will be parametrized equally.
- **Check Boxes** - In these check boxes, the type of winch motor is chosen. In this project the inverter was the checked one.
- **Reference Speed** - This parameter sets the reference speed of the system in encoder pulses per second.
- **Acceleration Time/Deceleration Time** - These two parameters set the time of acceleration and deceleration ramp, and were fixed at 5 s for safety reasons and because of the use of slow response asynchronous motors.
- **Proportional, Integral, Derivative Gain** - These three parameters enable the user to chose the convenient values for the Proportional, Integral, Derivative (PID) controller of the NC. The chosen values were reached by experimenting.
- **Reference Displacement** - This parameter sets the linear displacement of one revolution of the drum.
- **Reference Pulses** - This is the number of encoder pulses for the reference displacement. This parameter allows the NC to check the position of each axis and therefore correct the signal that is sent to the corresponding frequency inverter drive.

- **Encoder Number** - Since the encoders are incremental, the encoder number varies between 1 and 4, because this NC used in the project allows up to four incremental encoders [29]. Before having the system completely assembled and connected to the NC, the encoders 9 to 12 are also used in order to have a virtual encoder just for testing.

- **Analogic Output Number** - This parameter sets the analogical output that corresponds to each motor, to set the motor speed when it is supposed to be active.

6.1.2 Main program

Before running an ISO file, a main code was created in order to prepare the machine to start the movement. Firstly, the program has to be identified with a number to be called in the console (e.g. 1001), by the instruction *#prog* 1001, and call it the 'main cycle'. Secondly, urges to declare how many axes the controller has to interpolate, being four for the present system [29]. To do this, the instruction is: $INTP\ MODE\ =\ 4$. Thus, having everything prepared, the program calls the ISO file, already sent to the controller, with the instruction *gosub* 1000.

6.2 Programmable Logic Controller

As it was previously referred, the PLC is not the main controlling unit. It was used in this study just for managing digital inputs and outputs (e.g. external buttons) and replacing the console buttons. Therefore, the user has more intuitive buttons for starting and stopping the scene, or even the manual buttons for positioning the system.

The PLC program is divided into two, the manual and the automatic program. The automatic is to run the predefined movements and the manual is to move each axis in order to position the system in a certain position, but the main purpose of the manual mode in the present project is to drive the system to the 'home position'.

It is also important to refer that the graphic interface of the NC console which comes embedded in the NC, gently offered by Sipro, gives the user access to the manual and the automatic menus or even to reset the axis starting values, so it was not necessary to create a new interface for the NC.

6.2.1 Manual

When in manual mode it is utmost important to understand the procedure since axes (four) are chosen when the movement buttons are pressed (Figure 6.4).

Firstly, the user selects one axis and than moves it forward or backward by pressing the buttons JOG+ or JOG-, respectively. These buttons are on the customized motion desk (Figure 6.6).

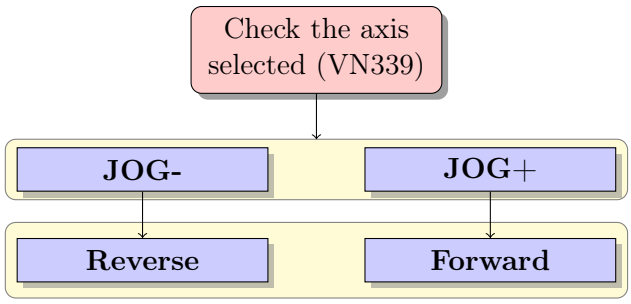


Figure 6.4: Scheme of the programming procedure for the manual mode

With this PLC program the user is able to pre-set the harness and drive it to the 'home position', one axis at the time.

6.2.2 Automatic

In order to run the system in the automatic mode, several steps have to be taken into account, as shown in Figure 6.5.

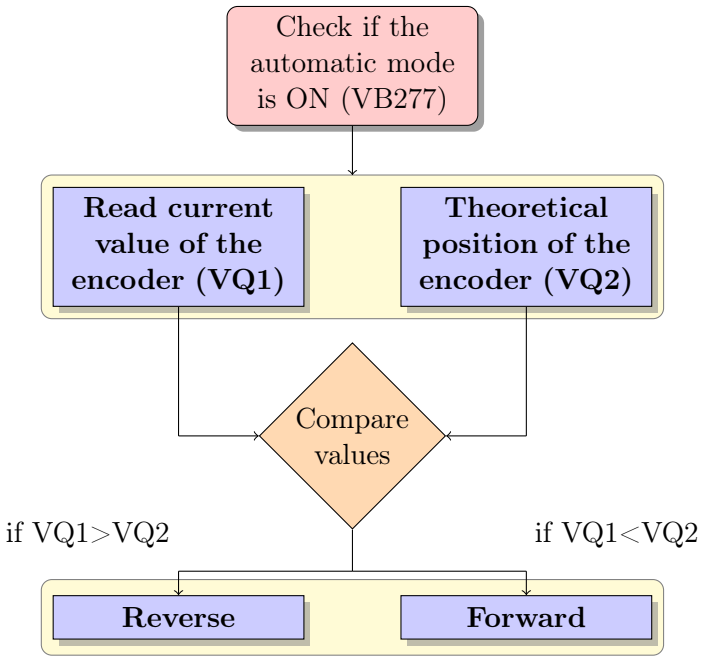


Figure 6.5: Scheme of the programming procedure using the automatic mode.

This procedure is taken for each motor, changing the VQs, where encoder data were recorded, and the digital outputs for the reverse or forward movement according to the winch motor the user wants to move.

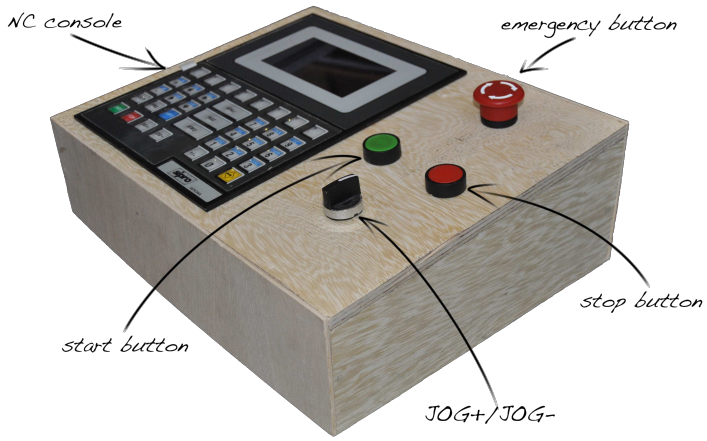
Once the start button is pressed, the NC starts running the ISO file, activating the analogical signals accordingly. Then, with the forward or reverse signal sets ON or OFF,

the motor starts moving. With this procedure repeated for the four axes, the virtual path becomes a reality.

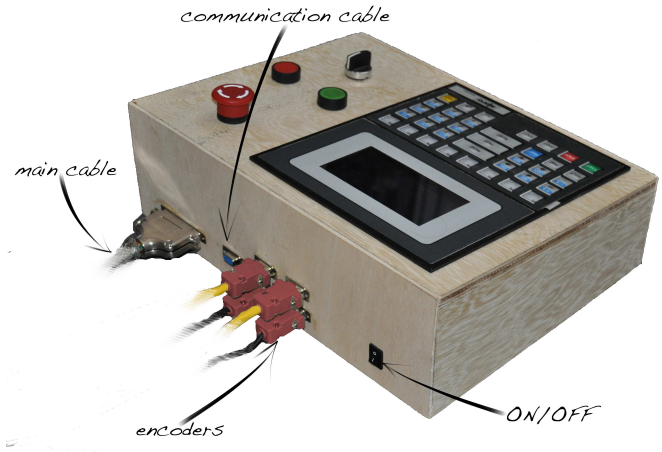
The PLC program protects the system, since it receives the emergency button signal. When activated, the system breaks down.

6.3 Assembling the Motion Control Desk

The NC and the PLC are already assembled in the console. A controlling desk was made specifically for this project, in order to receive all components that are going to be connected to the NC console (Figure 6.6).



(a) Front view.



(b) Back view.

Figure 6.6: Proposed controlling desk for the Motion Control Unit.

Thus, all the connections are hidden inside the desk. It is important to refer that the 'main cable' is responsible for supplying power to the desk and also to carry the motion

control signals, such as the forward and reverse signals, and all the digital and analogical signals to the electric panel.

6.4 Electric Panel

For a system like this, an electric panel is mandatory, since it plays a key role in the entire process. The electric panel is responsible for storing all the electrical components apart from the motors, as shown on Figure 6.7.

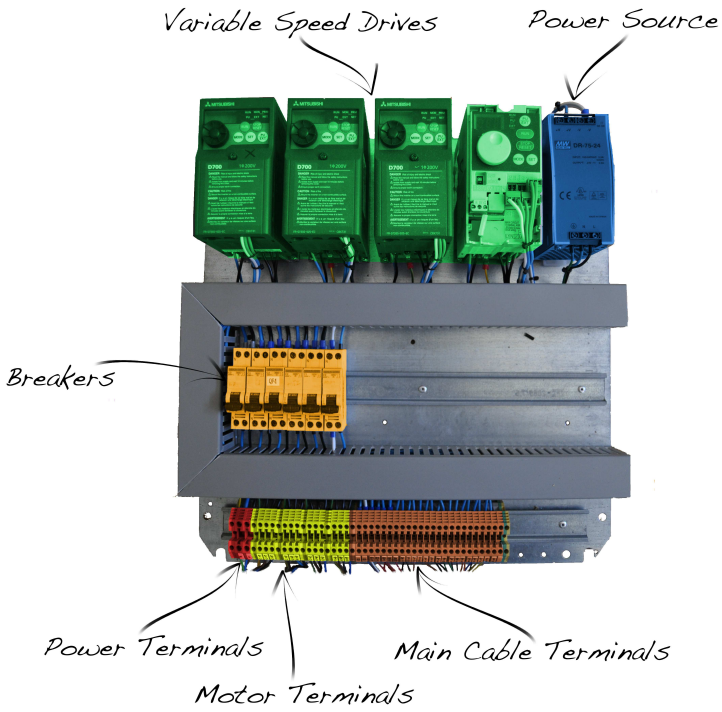


Figure 6.7: Proposed electric panel.

Through the 'main cable' the motion control signals is transferred, as previously reported. The movement data are forwarded to the terminals and then to the corresponding frequency inverter drive. The frequency inverter drives used in the present system are Mitsubishi FR-D700. Each one is connected to one motor. With this device some safety limitations are recommended, such as to the acceleration/deceleration ramp and the maximum/minimum speed, according to the Table 6.1.

In the present project, the acceleration and deceleration time was chosen to set to zero in order to follow the corresponding ramp, previously set in the NC axes parameters. The frequency covers the whole range, because the safety limits is guaranteed on the software.

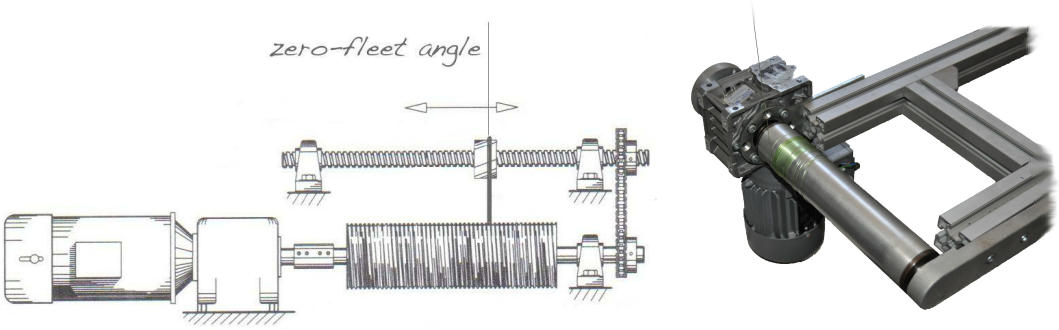
Table 6.1: Parameters of the frequency inverter drive [18]

Parameter	Name	Value
P1	Maximum frequency	120 Hz
P2	Minimum frequency	0 Hz
P7	Acceleration time	0 s
P8	Deceleration time	0 s

6.5 Winch Motor

In this project, the chosen motors were the three-phase induction motors, since they are the most commonly used in the entertainment industry due to their good responses to heavy loads.

These winch motors are composed by a motor, a reducer and a drum to wind the cable. Despite being supposed to compensate the exit angle of the cable when it rolls out of the drum, as shown in Figure 6.8(a), this system is represented here in a quite simple manner, since an appropriate design will be developed elsewhere. Therefore, it was decided that using very thin nylon cable in smooth drums (Figure 6.8(b)) would be a good representation avoiding the detailed mechanical design. This would not compromise neither the performance nor the final intended results.



(a) Exit angle of a winch motor's drum (b) Winch motors used in the project

Figure 6.8: Winch motors.

6.5.1 Assembling encoders

Unlike the servomotors which have embedded encoders, the three-phase induction motors need a feedback system for speed and position control. So that, in the present project encoders were assembled to each motor shaft, in order to send the feedback signal to the NC.

After reading the Siax200 manual, a DB9 plug was connected accordingly to each one of the encoder channels, as well as the power and the ground in order to communicate properly with the NC.

The mechanical assembling was finally made by connecting the motor shaft with the encoder shaft because the relation between the diameter of the two shafts was not

standardized, and it became mandatory to make customized couplers, as shown in Figure 6.9.

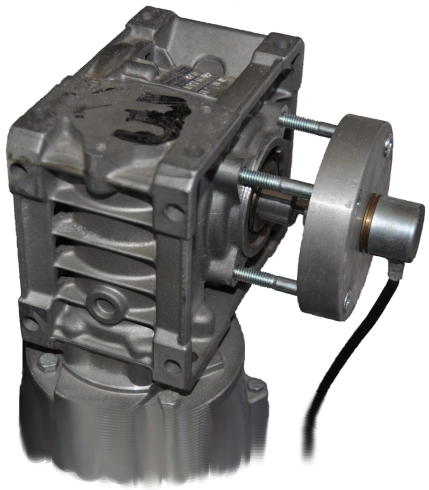


Figure 6.9: Encoder assembled in the motor shaft.

Chapter 7

Final Assembling

In this Chapter, both systems, software and hardware are ready to be assembled in a final configuration, as shown in Figure 7.1.

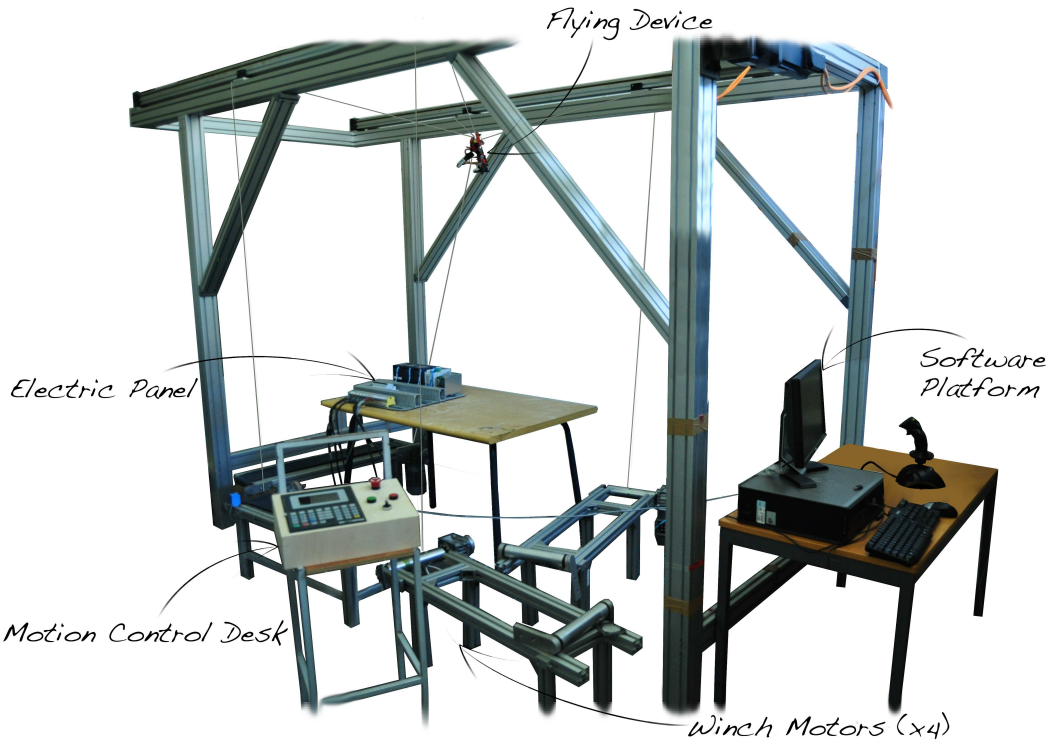


Figure 7.1: Proposed 3D motion control system.

As previously reported, this system is composed by four winches placed on the corner of a rectangle, on the floor. From each one, a cable passes through a pulley and then it is attached to the harness.

Then, each winch is connected to the electric panel in order to receive the signals from the frequency inverter drives, and the feedback devices (encoders) are connected

to the motion control desk in the corresponding port (from 1 to 4), following the order illustrated in Figure 7.2.

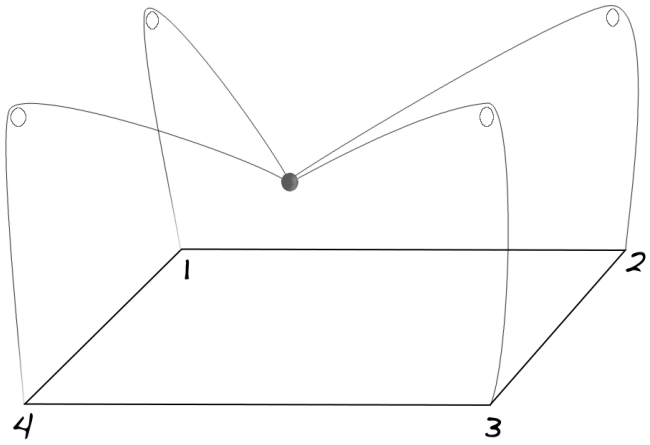


Figure 7.2: Positioning the winches.

Then, the main cable, responsible for the communication between the electric panel and the motion control desk is connected. Thus, the motion desk's buttons are connected to the NC console, in order to power it on and to give the corresponding digital signals to the PLC. With this partial assembly, the system is ready to work if it comprises a file with a predefined movement.

However, to send the ISO file to the motion control desk, it is necessary to establish the communication between the computer and the desk. Nevertheless, this file has to be primarily inserted into the Sipro's software previously mentioned, SiaxEd, and thereafter it is sent to the NC through a closed communication protocol.

Once everything is assembled and all the initial premises taken into account, the system is completely functional and successfully developed, as reported in the following section.

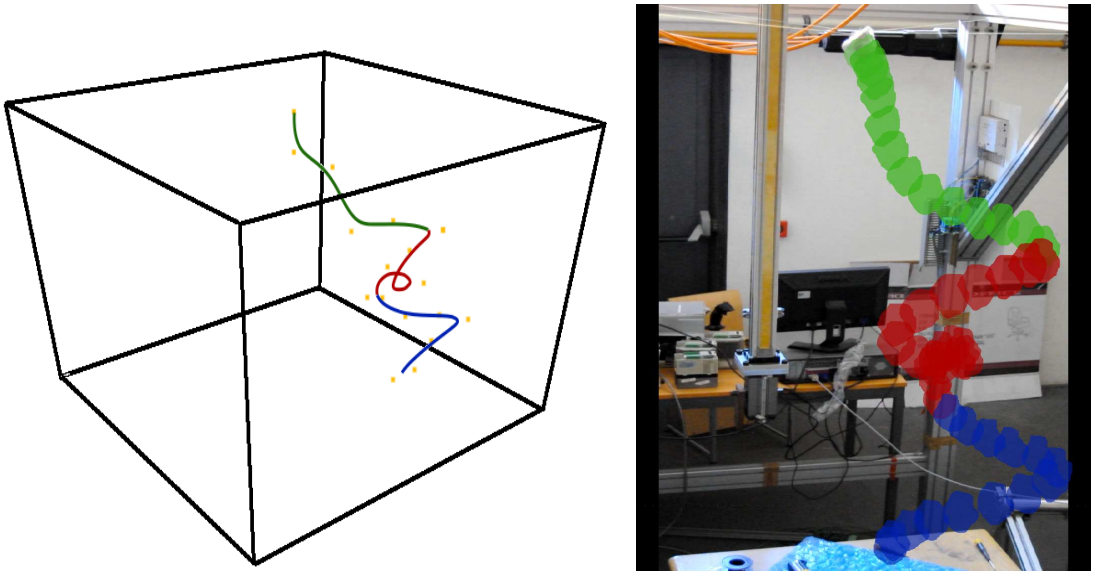
7.1 Performance Analysis

In this section the behavior of the entire system, when working altogether, will be described.

Firstly, the user designs the desired path using the joystick and checks if the result is the wanted. If the result is not the intended one, the user can go back and perform the required changes.

It is worth noting that, for the creative process, the user has a wide range of functionalities available on the graphic interface in order to assist a more realistic view of some real physical constrains, intended to optimize the final result.

The creative process takes time, but once reached the ideal trajectory, the path is exported into ISO file format to the desk. Then, with the motors and the harness already in position, the simulation begins, as shown in Figure 7.3, where the virtual movement (Figure 7.3(a)) and the real one (Figure 7.3(b)) can be compared.



(a) Virtual movement. (b) Real Movement.

Figure 7.3: Movement simulation.

The path was divided into three smaller ones, the green, the red and the blue one for a better analysis.

With the green path, the harness, was supposed to go downwards and forwards and, in the final part, slightly leftwards. From the real movement analysis, it can be observed that the object, which simulates the harness, does travel in a very similar way. Moving to the red path, it appears to be the trickiest one due to the small spiral (a complex movement) in the middle of the path. In general the system executed the movement very smoothly and accurate. The blue path, replicates a leftwards descendant movement followed by a rightwards descendant one. Similarly to the previous paths, it can be said that the last movement was fully accomplished.

Overall, the final system, was able to replicate, with a high degree of accuracy, the user's intentions, starting from the design aided by the customized software, fully reproduced by the physical system which was powered by the motion control unit.

Part III

Final Remarks

Chapter 8

Conclusions

During the entire process, some issues, worth mentioning, came across. The first one was related to the required 3D visualization and the 2D drawing area. However, with some research, the problem was solved, using functions of OpenGL in the GTK+ callbacks, and the user became totally free to start the creative process, having a complete flexibility for the designing process and to visualize the intended trajectory.

The objective of this project was to create a controlling system for a five degrees of freedom flight simulator. However, despite being ready to send the data for the three main directions and the vertical rotation, the interface was only tested according to the three main directions since the motion control unit only allowed fours axes (needed for controlling the three principal directions). However, a NC with five (or more) axes, is supposed to work properly with the five degrees of freedom.

When setting the PLC to receive the external information, a lot of compilation errors occurred - some of them without reasonable explanation - which means that the ladder program had to be done from the ground for several times. Nevertheless, at last the problem was overcome and it is now fully operational, capable of managing the external signals and relating them with the NC information.

The replacement of the PLC for a NC as the control unit revealed itself a very good choice, since the system became very easy to use, with the possibility of a quick change from one project to another one without the need of programming the control unit again. This means that the motion control unit is only responsible for the interpolation between points and for receiving the external inputs from buttons and encoders.This shows that the premise of reducing the real-time processing was successfully achieved, as well.

It must therefore be stated that all the project main goals were achieved. In fact, the two main points (development of the software and the motion control unit) were accomplished and a 3D flying system was assembled and it was fully operational.

It can also be stated that it would be a groundbreaking item in the market, capable of offering a low cost graphic interface and a motion control unit totally different from the market offers.

Due to the interaction of these two systems, the user of the final product has a complete sense of freedom during the creative process which means that in the end, the obtained movement is very smooth and accurate. These are the key points of a choreographed scene.

It is also important to refer that the highlighted drawbacks have, somehow, been overcome, but there is still some work to do in order to improve the overall proposed system. Some examples will be given in the following chapter.

Chapter 9

Future Work

'This is not the end. It is not even the beginning of the end. But it is, perhaps, the end of the beginning.'
Winston Churchill

Using the words of Winston Churchill, this Dissertation is the end of the beginning, which means that the 3D motion control system is completely functional, with some more details than it was initially expected. However, a product is never 100% developed. There is still some work to do in order to improve the overall system's behavior.

As far as the software is concerned, a few challenges remain. One lies upon the inclusion of the possibility to account for a non-prismatic workspace. Figure 9.1(a) and 9.1(b) illustrate the differences among these possibilities.

(a) Prismatic workspace.

(b) Pyramidal workspace.

Figure 9.1: Different types of workspaces.

Another challenge is to take advantage of the circular interpolation function that the NC provides, using the G-code, G2. To do this, the first task to take into account is to change the software in order to enable the user to choose the wanted interpolation. Currently it only provides a smooth interpolation between the points. In the future, for a circular movement, the user will be able to choose between this interpolation and the circular one that the NC can provide.

The most difficult task that still remains unsolved is to allow the software to accept external point clouds. The user would then be able to add details to the scene, for example pieces of scenery previously done in some kind of dedicated software.

In the end of this project, the Italian company, Sipro, supplied the closed communication protocol for the NC. So, it would be interesting to communicate directly from the developed software to the NC. Thus, it could also be added a live movement option, which would replicate the instructions of the joystick in real-time.

Another interesting task to accomplish is to take advantage of the 'Macros' provided by the NC. These 'Macros' would enable the user to call predefined movements (e.g. in the end of a trajectory, the user gives the instruction for the device to travel to the home position). These predefined movements would be on a list which could be accessed from the developed software and inserted anywhere within the trajectory.

Taking advantage of the digital signals of the motion control unit and using the 'M' command of the ISO language, the software could be more flexible, being responsible not only for controlling the flight device but also for changing lights or pieces of scenery at the same time.

As it was mentioned in the Chapter 2, the mechanical design was not one of the goals of this project. Nevertheless, it must be accomplished and should be one of the next big steps within the project.

Bibliography

[1] Ager M., Hastie J. 2009. *Automation in the Entertainment Industry.* Entertainment Technology Press Ltd.

[2] Anonymous. 2000. "Pulse-with modulation.". Accessed on 17th January 2012.
URL: *www.motionsystemdesign.com*

[3] Anonymous. 2011. "Canoe World Championship 2011 with the Spidercam.". Accessed on 16th December 2011.
URL: *http://www.mediatechnika.hu/broadcast/canoe+world+championship+2011+with+the+spidercam.html*

[4] Anonymous. 2012*a*. "Fisher Technical Systems.". Accessed on 2nd January 2012.
URL: *www.fishertecnical.com*

[5] Anonymous. 2012*b*. "Fly by Foy Systems.". Accessed on 15th January 2012.
URL: *www.flybyfoy.com*

[6] Anonymous. 2012*c*. "NYC Grosses.". Accessed on 20th May 2012.
URL: *www.livebroadway.com*

[7] Anonymous. 2012*d*. "OpenGL Overview.". Accessed on 15th January 2012.
URL: *www.opengl.org/about*

[8] Anonymous. 2012*e*. "Production Resource Group Systems.". Accessed on 2nd January 2012.
URL: *www.prg.com*

[9] Anonymous. 2012*f*. "Spider-Man, Turn Off the Dark: Original Broadway Cast Show Photos.". Accessed on 16th December 2011.
URL: *http://www.broadway.com/shows/spider-man-turn-off-the-dark/photos/*

[10] Anonymous. 2012*g*. "Spidercam Technical.". Accessed on 19th July 2011.
URL: *www.spidercam.net*

[11] Anonymous. 2012*h*. "Stage Technologies Systems.". Accessed on 2nd January 2012.
URL: *www.stagetech.com*

[12] Anonymous. 2012*i*. "Theatre Database.". Accessed on 20th January 2012.
URL: *www.theatre-architecture.eu*

[13] Anonymous. 2012*j*. "What is GTK+, and how can I use it?". Accessed on 15th January 2012.
URL: *www.gtk.org*

[14] Barker, C., W. Cruse and J. M. Gillette. 2011. "Stagecraft.". Accessed on 15th January 2012.
URL: *www.britannica.com/EBchecked/topic/562420/stagecraft/278572/Flying-systems*

[15] Barr, M. 2007. "Introduction to Pulse Width Modulation (PWM).". Accessed on 15th January 2012.
URL: *www.barrgroup.com/Embedded-Systems/How-To/PWM-Pulse-Width-Modulation*

[16] Burrage, L. 2012. "Spidercam on set.". Accessed on 7th June 2012.
URL: *http://www.lukeburrage.com/blog/archives/1062*

[17] Chai, J. 2004. Performance Animation from Low-dimensional Control Signals. Master's thesis.

[18] Electric, Mitsubishi. 2012. *FR-D700 Frequency Inverter Instruction Manual*.

[19] Ellis, S. R., D. R. Begault and E. M. Wenzel. 1997. *Handbook of Human-Computer Interaction*. North-Holland.

[20] Erickson, K. T. 1996. "Programmable logic controllers." *IEEE Potentials* .

[21] Ferreira, J. M. M. 1986. Sistema de Comando Numerico Para Prensas Hidraulicas de Duplo Efeito. Master's thesis Faculdade de Engenharia da Universidade do Porto.

[22] Ha, T., M. Billinghurst and W. Woo. 2011. "An interactive 3D movement path manipulation method in an augmented reality environment." *Interacting with Computers* .

[23] Hendrickson, A. and C. Buckhurst. 2008. *Mechanical Design for the Stage*. Focal Press, Elsevier Inc.

[24] Huntington, J. 2002. "Rethinking Entertainment Technology Education." *Theatre Design & Technology* 38.

[25] Macey, J. 2012. "Parametric Curves an Surfaces.". Accessed on 23rd April 2012.
URL: *http://nccastaff.bournemouth.ac.uk/jmacey/RobTheBloke/www/opengl_programming.html*

[26] Mills, S. and J. M. Noyes. 1999. "Virtual reality: an overview of user-related design issues revised paper for special issue on virtual reality user issue in interacting with computers." *Interacting with Computers* .

[27] Okaniwa, S., A. Nasri, H. Lin, A. Abbas, Y. Kineri and T. Maeakwa. 2012. "Uniform b-spline curve interpolation with prescribed tangent and curvature vectors." *IEEE Transactions on Visualization and Computer Graphics* 18.

[28] Pathak, K. B. and K. K. Acharya. 2011. "Automation in Entertainment." *International Conference On Current Trends In Technology* .

[29] Sipro. 2012. *Siax 200 Manual*.

[30] StageTechnologies. 2012*a*. *Autodesk 3ds Max's Sculpture Animation Toolkit*.

[31] StageTechnologies. 2012*b*. *eChameleon automation software*.

[32] Thilliez-Dunod, J. 1967. *La Commande Numerique des Machines*.

[33] Wasfy, T. M. and A. K. Noor. 2000. "Object-oriented virtual environment for visualization of fexible multibody systems." *Advances in Engineering Software* .

[34] Zhang, Y. and G. Cui. 2009. "Kinematic analysis on spatial rotation 4-SPS-1-S parallel manipulator with a passive constraining spherical joint.".