Visualisation of Design Space Exploration using Virtual Reality

Newcastle University School of Computing Science

2017-2018

Student: Hasan A.H.J.A.

Mark Student No.: 150454388

Degree Prog.: G400 Stage 3

Module: CSC3095

Exercise: Dissertation

Due: 23:59:59 Friday, 4th May 2018

Word Count: 11283

Abstract

Cyber and Physical co-models are complex heterogenous structures that are difficult to analyse and usually require the assistance of methods like Design Space Exploration, to be configured and improved. However, the results from Design Space Exploration process could be difficult to understand due to its systematic exploration of all potential results. Hence, to make the illustration of Design Space Exploration results easier, this project attempts to develop a solution that illustrates its results in a 3-dimensional environment in Virtual Reality. To achieve this, research was conducted into the effectiveness of data representation in Virtual Reality and how to overcome the limitations of developing such a solution in Virtual Reality. This research in project has found that data representation in Virtual Reality combined with motion controls can easier to understand in comparison to other data representation methods, and that development in Virtual Reality places special importance in the use of popular and open tools. Hence, the final solution developed by this project used the Unity engine and SteamVR libraries to illustrate Design Space exploration results. The solution according to undertaken user evaluation provides an easy to use interaction model and an easy to understand graphical display of data.

Declaration

“I declare that this dissertation represents my own work, except where otherwise stated.”

Signed: Ammar Hasan

Dated: 04/05/2018

Table of content

Table of Figures

Chapter 1 Introduction

1.1 Basic Introduction and Motivation

Design Space Exploration (DSE) is a data analysis technique that uses systematic alteration of parameters (Gries, Matthias 2004, p. 134) with the intent of finding a solution design that meets predefined constraints and objectives (Gries, Matthias 2004, p. 136). This technique is useful for exploring the designs of heterogenous and complex systems (Gries, Matthias (2004, p. 133) like Cyber Physical Systems (CPS), which are formed from diverse physical (continuous) and cyber (discrete) components connected by heterogenous networks (Al-Hammouri 2012, p. 8). By using model-based designs these systems can be abstracted and modelled together (co-models) and then explored for potential designs that fit requirements. (Fitzgerald et al. 2014, p. 17-21).

However, DSE can become difficult to manage due to the overwhelming nature of DSE size (Kokhazadeh & Fatemi 2011, p. 550). This presents a challenge to illustrate and interpret this process and its results due to the overwhelming size of parameters.

One way to simplify DSE interpretation would be the use Virtual Reality (VR) to illustrate its results and parameters. Data representation using VR is a topic that has been explored before, but not explored in the context of DSE of CPSs. The advantage of using VR for data representation is that it presents a living, interactive and deep experience (Valdés, Romero & Barton 2012, p. 13193), these advantages would help in managing the overwhelming nature of DSE.

The use of VR in DSE would be beneficial in Cyber Physical fields, where Cyber Physical modelling toolsets like INTO-CPS make use of co-modelling of Cyber and continuous parameters (Fitzgerald et al. 2016, p. 8-9). The use of a diverse set of parameters alongside DSE presents a challenge, as the number and diversity of parameters to illustrate and interpret becomes difficult to manage. Hence, a more interactive experience might be required to help users interact with the data more easily.

1.2 Dissertation Outline

This document is divided into 6 chapters:

1. Introduction

   This chapter goes briefly over the motivation for the project and introduces key concepts. The aims and objectives are also introduced along with the requirements and the plan to achieve them.

2. Background Research

   This chapter details the results of research into relevant topics and details potential solutions for the issues this project tries to solve and how it compares to similar solutions.

3. Tools, Design and Implementation

   This chapter goes through the essential tools used to construct the project, then goes through requirement analysis and design (specifications). This chapter also describes how the requirements were implemented in Unity, including any prototypes developed

4. Planning and Testing

   This chapter goes through the plan used throughout the project to achieve its objectives and how the implementation was tested.

5. Results and Evaluation

   This chapter evaluates the final product against the aims, objectives and requirements of the project, and summarises the user evaluation of the solution. Moreover, an evaluation of the tools, techniques and plans followed throughout the project are provided in this chapter as well.

6. Conclusion

   This chapter summarises the results of the project and issues faced in the process. Moreover, it provides suggestions for improvement and further work.

7. Appendix

   Supplemental material is included here:

   - Appendix A: Glossary
   - Appendix B: Acronyms
   - Appendix C: Requirements Interview Script with summarised answers
   - Appendix D: User Evaluation Interview Script with summarised answers

8. References

   Numbered referenced work citations (Harvard at Newcastle).

1.3 Aims and Objectives

This section of chapter goes through the aim of the project and what objectives with concise descriptions need to be achieved to complete this aim. The presented objectives slightly differ from the ones in the project proposal as one objective related to design candidates was removed as it was redundant with the 3rd objective.

- Aim: Illustrate the results of DSE of Cyber Physical Systems through visualisations using VR equipment.
- Objectives:
    1. Review and work around the risks and limitations of data representation with VR equipment
        - Since this project is using VR, it is crucial to understand the limitations of the technology and to figure out ways to mitigate these limitations, particularly since the technology is still developing.

    2. Illustrate the multidimensional Cyber Physical DSE in 3D graphs using VR hardware
        - The Design Space explored should be represented by 3D graphs that are visible to the user in a Virtual world, representing the results of the DSE.

    3. Illustrate the effects of changing Cyber Physical DSE objectives using motion controls in a VR environment on three-dimensional graphs
        - The user of the software should be able to change the objective values the DSE process seeks to achieve with the provided motion control while receiving live feedback on its effects via the 3D graph.

    4. Allow the user to switch to different sets of dimensions in the design space dimensions through VR motion controls
        - The user of the software should also be able to change the results of which objectives are shown in the Virtual Environment using his Virtual motion controls.

1.4 Solution High-Level Requirements

This section only goes through the basic high-level requirements for the project to achieve its objectives. For a more detailed description check section 3.2.3.

1. Read data from a co-simulation software
    - The solution software needs to be able to read DSEs on co-models' data results, which in this project context would be the CSV and JSON files produced by the INTO-CPS co-simulation engine.
2. Construct the virtual environment where the user would be able to see the 3D results of the DSE process and control it.
    - A space where the user can interact with the results of the DSE process needs to provide.
3. Provide control over graph
    - Basic axis, scale and label control needs to be provided by the software.
4. Draw 3D graphs in VR space to user hardware
    - Based on the received CSV data the engine needs to draw 3D representation of the DSE results in the virtual space.
5. Construct a motion-based interface
    - The interface should allow user to control the data visualisation and the DSE process, construct a motion-based interface. The interface should also be able to control the graph settings (axis, labels, etc.).

1.5 Plan and Changes

The plan for the project is fully detailed in section 4.2.1, but to summarise the basic plan is to divide the project into two routes, a project route and a dissertation route. The routes join in the dissertation results task and the project route with the results acts as the critical path of the project. Methods like prototyping are also introduced in the development process to allow for quick evaluation of project progress.

Throughout the progress of the project, the original plan was revised and changed multiple times to accommodate for changes in relation to prototyping, delays and other issues faced in the project described in section 4.2.2.

Chapter 2 Background Research

2.1 Background Research Introduction and Strategy

This section details the relevant research undertaken for the project, going through relevant concepts, tools and competing solutions. In the scope of the is project, this section goes through the INTO-CPS toolset and co-modelling, DSE and VR. A comparison of the proposed solution with similar VR data representation tools and with DSE result illustration techniques was also carried.

Research was conducted using published research only, located using Newcastle University Library Search and through published specification papers of INTO-CPS provided by the supervisor.

2.2 Co-modelling and INTO-CPS Toolset Research

Due to the heterogenous nature of CPS, this project must work alongside Cyber Physical co-models that consist of various connected Cyber and Physical components. To do this, a co-simulation tool that can combine the simulations these components together is essential. The co-simulation toolset that was selected for this purpose is the INTO-CPS toolset which is detailed in section 3.2.2. To utilise the features of the INTO-CPS tool properly an understanding of co-modelling and the tool's structure is essential.

2.2.1 Co-modelling and Co-simulation

As per the glossary section, a co-model is a model that typically consists of abstractions of both continuous and discrete systems (Fitzgerald et al., 2014, p 17). co-models are constructed and simulated by allowing the Continuous (CT) and Discrete (DE) models to communicate to be via a contract (Fitzgerald et al., 2014, p 18):

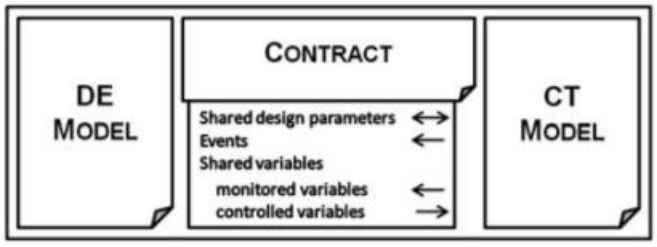

Figure 1: Co-model Structure (Fitzgerald et al., 2014, p18)

- Shared variables: Variables available to both DE and CT models
- Events: An action from a model that affects another model
- Shared Design Parameters: Initial design parameters shared by DE and CT models
- These are referred to as a contract

This requires the use of a co-simulation Engine that helps keep the semantics and syntax of the contract consistent and provides results of the simulation, while the models handle other domain specific information (Fitzgerald et al., 2014, p 18-19).

The benefits of constructing co-models is that it allows for simulations of complex CPSs, through handling the coupling of cyber and physical systems (Al-Hammouri 2012, p 8) in CPSs to carry simulation analysis for when non-practical analytical approach fall short. For instance, the analysis of CPSs carried in the following stages benefits from use of co-models (Fitzgerald et al. 2014, p 22-24):

- Requirements definition
- Requirements analysis
- Architectural design
- Detailed design
- Implementation/integration
- Operations and maintenance

2.2.2 INTO-CPS

The INTO-CPS Application refers to a set of tools that coordinates co-simulation (via COE) and working with a DSE driver to carry exploration in the co-model, this collection is referred to as the INTO-CPS tool chain. (Lausdahl et al., 2015, p. 24).

This toolset uses the FMI interface and FMU units (Lausdahl et al., 2015, p. 24-27). The FMI interface to allow the various FMU implementing models to communicate to carry co-simulations (Fitzgerald et al., 2016, p. 11). These co-simulations are managed by the COE engine, which ensures models can be co simulated via an algorithm which allows the models to communicate (Lausdahl et al., 2015, p. 33-34), as explained in the previous section.

The tools section in 3.2.3 goes into the details of how INTO-CPS structures its results and why it was chosen.

2.3 Design Space Exploration Advantages and Issues

As previously described in the introduction and in the glossary, DSE refers to the Systematic alteration of design parameters (Gries, Matthias 2004, p. 134) to reach a goal or solution that fits an objective (Gries, Matthias 2004, p. 136). This systematic alteration occurs in “Design Space”, which refers to the set of all possible solutions to a given problem, where every solution act as a potential design alternative (Fitzgerald et al., 2014, p 21). These design characteristics might be referred to as dimensions (Gries, Matthias, 2004, p. 135). The objective of the DSE algorithm is to the select the best candidate design from the set of alternatives based on a given set of criteria (Fitzgerald et al., 2014, p 21).

DSE’s main advantage is its ability to explore complex heterogenous systems like co-models, however due to the size of the design spaces due to the various potential designs (Kokhazadeh & Fatemi, 2011, P. 550), it is complex to analyse the solutions and criteria in DSE.

2.4 Virtual Reality Data Representation

VR refers to a computer simulated environment that replicates the physical world, the user can be immersed via interactions with that environment (Velev, D. & Zlateva, P., 2017, p. 33). VR uses various complex pieces of hardware to achieve this, including HMDs and tracking devices (Velev & Zlateva ,2017, p. 33).

VR has a wide variety of application, but of relevance to the context of this project is its use in data representation. In this application area, it is being particularly beneficial in as seen in some papers. For instance, research found that VR HMD can make graphs more comprehensive by 3 times (Ware & Franck, 1994, p. 183). Also, other research has found that the use of motion controls can help improve controls of graphical visualisations (Huang et al., 2017, p. 44).

However, VR is still an emerging technology and hence still subject to many limitations. Some of the issues that face VR include the limited support and issues with proprietary technology (Velev & Zlateva, 2017, p. 36).

2.5 Summary of Research and Solution

As described in the previous sections in this chapter, CPS co-models are complex and heterogenous structures that are difficult to explore with analytical methods like DSE. However, DSE brings its own challenges as its own results are complex on their own.

This project proposes a solution to this issue by using VR to handle DSE result data representation. Since, as described in section 2.4, VR data representation is more comprehensible and easier to control (with motion/gesture control) than traditional 2D graphs, the solution aimed to make use of motion-controlled tracking devices and a 3D VR environment to display the DSE results. To mitigate the issues with limited support and standards, Unity Steam VR libraries are used as they are open and fully compatible with the Unity Game engine, which is the choice of engine for this project (see section 3.2.1). Also, no attempt was made to make the software compatible with more than one VR hardware yet.

To further improve the solution, lessons can be learned from other similar solutions. For instance, DatavizVR is an early access VR data visualisation tool available on steam (DataVizVR Inc., 2017). DatavizVR allows users to plot CSV files provided to in a 3D scatter plot with various handy controls for the axis (relabelling, inversion, etc.), also it provides an easy data selection tool via a laser from the VR tracking controller. Providing such controls in the solution would be essential for the user experience. However, Dataviz offer limited analysis tools (e.g. no pareto analysis) and limited use of space (i.e. only a scatter plot is provided and no other controls or buttons in the space) Hence, the solution might require additional data visualisation (e.g. more charts, pareto analysis, etc.) and better use of the scene space (e.g. other co-model configuration data).

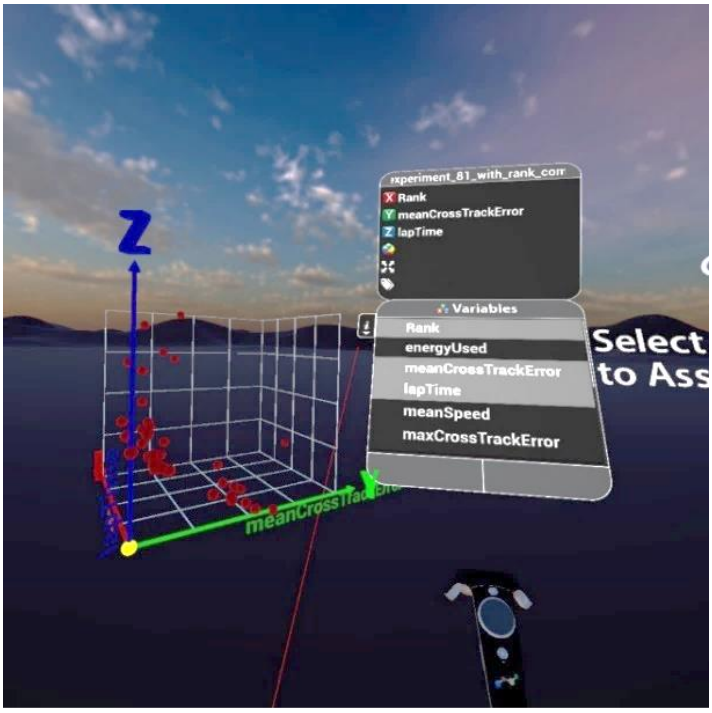

Figure 2: DataVizVR Axes Settings

Moreover, the INTO-CPS tool chain also provides some basic visualisation for its results, these are formatted by two scripts (Gamble, 2016, p. 31-32):

1. Ranking script: Produces ranking data for results and a PNG 2D graph that highlights the best designs (Pareto front)
2. Output script: Produces the static HTML page which includes a table of ranks and the web page

However, while the Pareto front is useful and could be incorporated into the solution, the results produced are still lacking. For instance, there is no way to link the results in the table with the graph (perhaps a selection feature can be added) and there is only 2 dimensions (avoidable limitation in a 3D environment) (Gamble, 2016, p. 31-32).

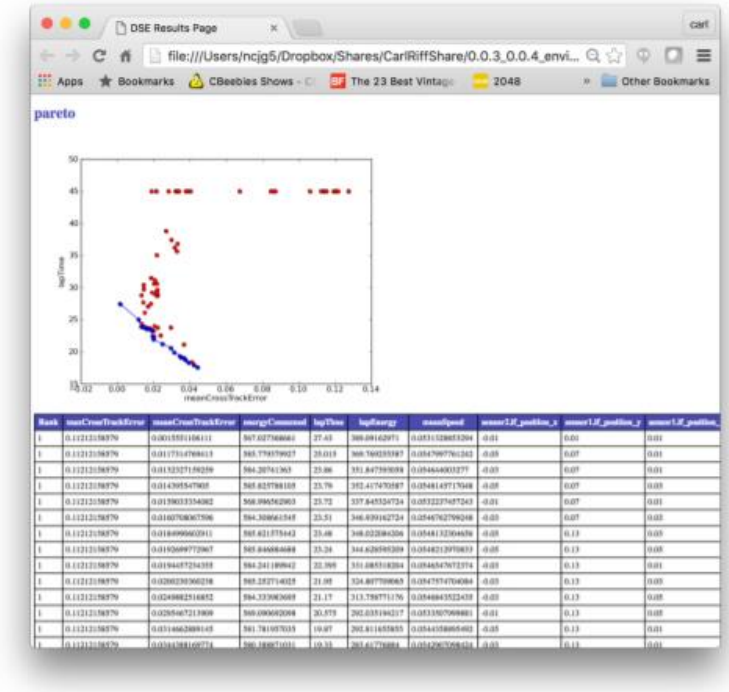

Figure 3: INTO-CPS HTML Results (Gamble, 2016, p. 15-16)

Chapter 3 Tools, Design and Implementation

3.1 Tools, Design and Implementation Introduction

This chapter initially introduces relevant and essential tools used for this project, and then goes through the design and implementation process of this project from requirement analysis to the specification and architecture (requirements and diagrams).

3.2 Tools

This section outlines the key tools tools used in the projects. To achieve that, this section refers to manuals and product sites for the tools for sources to describe the basic features of the tools, however INTO-CPS sources use published papers relating to its specification.

3.2.1 Unity

Unity is a 3D game engine used for various games as well as other applications that require a 3D world. Unity provides various tools to allow this, from sophisticated physics simulations to built-in UI controls (Unity, 2018b).

Unity is quite helpful in this project context because of its great VR support. Unity supports various VR development platforms like SteamVR which makes it the most popular platform for VR development (Unity, 2017d).

Scene Editor

This editor in the Unity application allows users to edit 3D/2D scenes by managing the project Assets and Game Objects. Game Objects and assets are represented in a hierarchical view (Unity, 2018a). The Assets represent all resources that can be used in a scene (Unity, 2017a), while the Game Objects represent objects in the scene whose behaviour is affected by attached components (Unity, 2017b).

C# Scripts

Scripts are a type of component that can be used to affect the behaviour of in scene Game Objects, but unlike other components they are fully programmable with either C# or UnityScript (JavaScript based) (Unity, 2017b). This can be used to enable wider and more flexible behaviour from the Game Objects they are attached to.

SteamVR SDK

SteamVR SDK is a VR platform that supports development for major VR headsets by providing an interface to various hardware and low-level functions like tracking controllers (Steam, 2018). The SteamVR SDK is available as a Unity Asset from the Unity Asset store (Steam, 2018). This library was chosen for its compatibility with Unity, openness and HTC Vive compatibility (Steam, 2018), which as explained in section 2.4 and 2.5 is crucial to avoid issues with the developing VR market.

3.2.2 INTO-CPS

As been stated in section 2.2.2, INTO-CPS is a Co-simulation tool chain with DSE support and is the selected tool for carrying DSE on Co-Models. The results of INTO-CPS DSE are visualised by the tool developed by the project.

This tool was chosen because Newcastle University has been directly involved with development parameters (Fitzgerald et al. 2016, p. 2), and hence the supervisor would be well acquainted with the tool and its various published papers.

Results

As previously stated in section 2.5, the results produced from the INTO-CPS tool chain are stored in a file hierarchy with the following files (Gamble, 2016, p. 15-16):

Summary (for all results):

- dseResults.csv: A CSV file summarising each designs parameters, objectives and Pareto rank. This is a new addition (C Gamble 2018, personal communication, 23 April)
- ranking.json: A JSON file with the Pareto ranks of all designs based on objectives
- results.html: A printed HTML summary of the above files, plus the Pareto graph of the ranking
- dseConfig.json: A JSON file with ranking and configuration information
- Config.json: A JSON file with general FMU configuration

Specific (for every setting):

- config.mm.json: A JSON file that containing co-simulation configuration.
- objectives.json: A JSON file containing the scores the design got based on certain metrics
- results.csv: A CSV file with raw results simulation results with rows for each time step

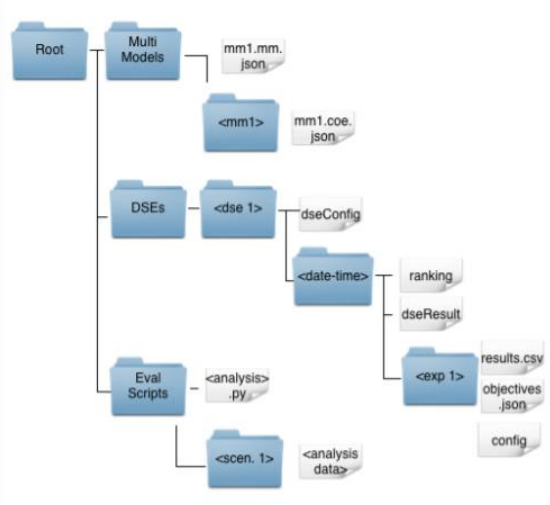

Figure 4: INTO-CPS Output Hierarchy (Gamble, 2016, p. 16)

The solution must read these results to provide the visualisations, the specifics of how this is done are shown in section 3.2.4.

3.2.3 HTC Vive

The HTC Vive is an HMD device used for VR and is used with a set of tracking controllers and a set of base stations that are used to track it and its controllers (VIVE™ United, 2018). The HTC supports the steamVR platform (VIVE™ United Kingdom, 2018), which is used in this project as stated in section 3.2.1.

3.3 Design and Implementation

3.3.1 Requirement Analysis

To achieve its objectives to reach its aim, the purposed solution of this project has followed a set of rigid requirements. These requirements were derived from a requirement analysis process that used several means to construct the specification with appropriate requirements. In the project this was done via:

1. Interviews
2. Analysis of Market
3. Analysis of Research

A set of high level requirements were listed in section 1.4 based on the brief research carried during the Project Proposal, these requirements are expanded upon in this chapter as shown in section 3.3.2.

Interviews

An interview was conducted with Frederik Forchhammer Foldageran, an Engineering Specialist at Agro Intelligence in Aarhus Denmark, with the script and answer summary provided in Appendix C. Frederik has used the INTO-CPS tool set to carry DSE to help configure agriculture controllers. Frederik used the results HTML page provided by INTO-CPS (described in section 2.5) and has found it to be limiting due to the lack of 3D axis and Selection. Moreover, he considers the Pareto analysis critical for finding the best designs.

Frederik considers analysing how well the various designs meet the set objectives as the most critical part of DSE results while analysing Co-Models, and rarely runs DSE algorithms with different configurations. Frederik uses the graphical representations of DSE mainly internally during the design process.

Analysis of Market

This had been difficult due to the lack of solutions that are use VR in the context of DSE. So, instead of analysis direct competing solutions, similar solutions were look at instead. In section 2.5 in the Research chapter, two solutions (DataViz VR a 3D VR data representation tool and INTO-CPS DSE results) were analysed. The result of the research was as stated in 2.5 that the solution must:

1. Offer multiple axis for extra flexibility
2. Offer a data point selection feature
3. Analytical tools like pareto fronts
4. Provide various axis controls like axis inversion and relabelling

Analysis of Research Publications

The results of the background research as summarised in section 2.5 formed the basis of both the motivation and requirements of the project solution. The research has proven that VR makes understanding visualised data easier as described in section 2.4, which would be especially useful for DSE results which as described in section 2.3 can get complex and convoluted. This makes providing a VR environment to present the data a crucial requirement for this project.

3.3.2 Specification Requirements

This section provides a low-level description of the list of requirements derived from the analysis process to complete the solution.

1. Read the dseResult.csv generated by the INTO-CPS tool set into a script, separating design objectives, design parameters and ranks from it:

    - Reasoning: The results need to be read first and stored, so that it can be used later to visualise the DSE results. But, design objectives need to be separated from design values and ranks. As, the objects are used to display how well the design performed, design values are used to show which design has performed and ranking is used to carry pareto analysis.

2. Read .json configuration files into a script to record and display pareto ranking:

    - Reasoning: As described in section 3.3.1 the interview conducted has shown that pareto analysis is important, hence reminding the user of which objectives are being used to carry it is important.

3. Read .json configuration files to record and display model configuration information (i.e. FMU information):

    - Reasoning: The Interview conducted has also shown that DSE results are typically analysed to configure models and their controllers. Hence providing users with information about the configuration of the co-model is vital.

4. Construct a scene that can be viewed from a user HMD using library assets:

    - Reasoning: As described in the research results analysis in section 3.3.1, VR makes understanding data visualisation easier. Hence, there is a need for a VR space to visualise the data and provide controls.

5. Draw a 3D plot graph axis using Unity Game Objects that can be viewed in the Unity Scene:

    - Reasoning: The main limitation of the INTO-CPS toolset HTML results as described during the interview was its lack of 3D graphical display. Providing the user with a 3D graph with 3 axes would help alleviate these issues.

6. Draw the points from the objectives by taking the readings from the dseResult.csv file into the 3D plot graph taking pareto rankings, minimum and maximum values:

    - Reasoning: To complete the graph, the stored results must be constructed in the VR scene. Hence, points must be drawn in the correct coordinates according to their stored values and relative position to graph axis.

7. Allow user to select points in the plot to read their values for interpretation with motion control via controller laser pointer:

    - Reasoning: The interview has shown that a selection feature would have been a helpful addition to the HTML display provided by INTO-CPS (this was also raised in its published documents as a limitation as described in section 2.2.2). On way to implement this feature, would be to select the points using a laser pointer as done by solutions like DataVizVR as described in section 2.5.

8. Add UI elements to show user selected values (variable values and model design parameters):

    - Reasoning: The selection feature would need to display the results of the selected point somewhere. Hence, UI elements with select variable value and model design parameters need to shown to the user.

9. Add buttons to allow users to control selected variables to display in axes

    - Reasoning: As described in the research analysis summary in this chapter, selection of axes variables offers helpful flexibility to the user to select which variables they want to compare.

10. Add buttons to allow users to control graph attributes (scale and axis inversion):

    - Reasoning: Another advantage observed in DataViz in the analysis carried on similar solutions in section 2.5 was added controls like axis inversion. Hence, their addition is vital to the usability of the project solution.

11. Allow axis to rescale according to user set proportions:

    - Reasoning: To allow easy reading, the axis and their points must be rescaled by the user as he sees fit.

12. Allow graph to enable and disable negative and positive graph elements according to the data set:

    - Reasoning: Negative and positive parts of a graph are not needed if all values are negative or positive respectively. Hence, they need to be hidden away to allow the user to concentrate on the data.

13. Allow graph to enable or disable certain axis as per user request:

    - Reasoning: Like the previous requirement, the user might not require 3 axes. Hence, they should be able to disable a certain one (x, y or z) to concentrate on the only two he needs.

14. Allow graph to scale negative axes against positive axes according to minimums and maximums.

    - Reasoning: To ensure points drawn look proportional axes need to be scaled.

3.3.4 Architecture, Diagrams and Implementation

As described in the previous section, a room based on a Unity Scene was constructed to form the 3D VR Environment. This environment consisted of:

1. UI Elements (e.g. selection read outs, objectives, etc)
2. Buttons to allow control (scale, axis inversion, etc)
3. Graphical representation of results with Paetro analysis (i.e. a 3D graph with points), consisting of:
    a. Axes
    b. Results (i.e. points)
4. Representation of user controllers to carry motion control (selection of data and UI elements)
5. Other elements required for SteamVR and the camera or lighting model

These elements would be presented as single Game Objects or created from multiple Game Objects which are described in section 3.2.1. Also, to allow these elements to communicate and to allow them to read DSE results Unity C# scripts (which are described in section 3.2.1) are attached to Game Objects as components. The structure of the Scene, Game Objects and how results are read are described in the next sections with diagrams for illustration.

The final implementation results will then be evaluated for their usability through a user evaluation test based on the Test Plan in section 4.3.2 and an interview shown in Appendix D. The results of the evaluation are shown in section 5.1.1.

Unity Scene Structure

Unity manages its 3D and 2D scenes as Unity Scenes with Game Objects as described in section 3.2.1. Hence, to construct the 3D environment specified in section 3.3.2, a Unity Scene with the appropriate Game Objects needs to be constructed. But before getting into details into what Game Objects are used and in what hierarchy, the Scene needs to be laid out.

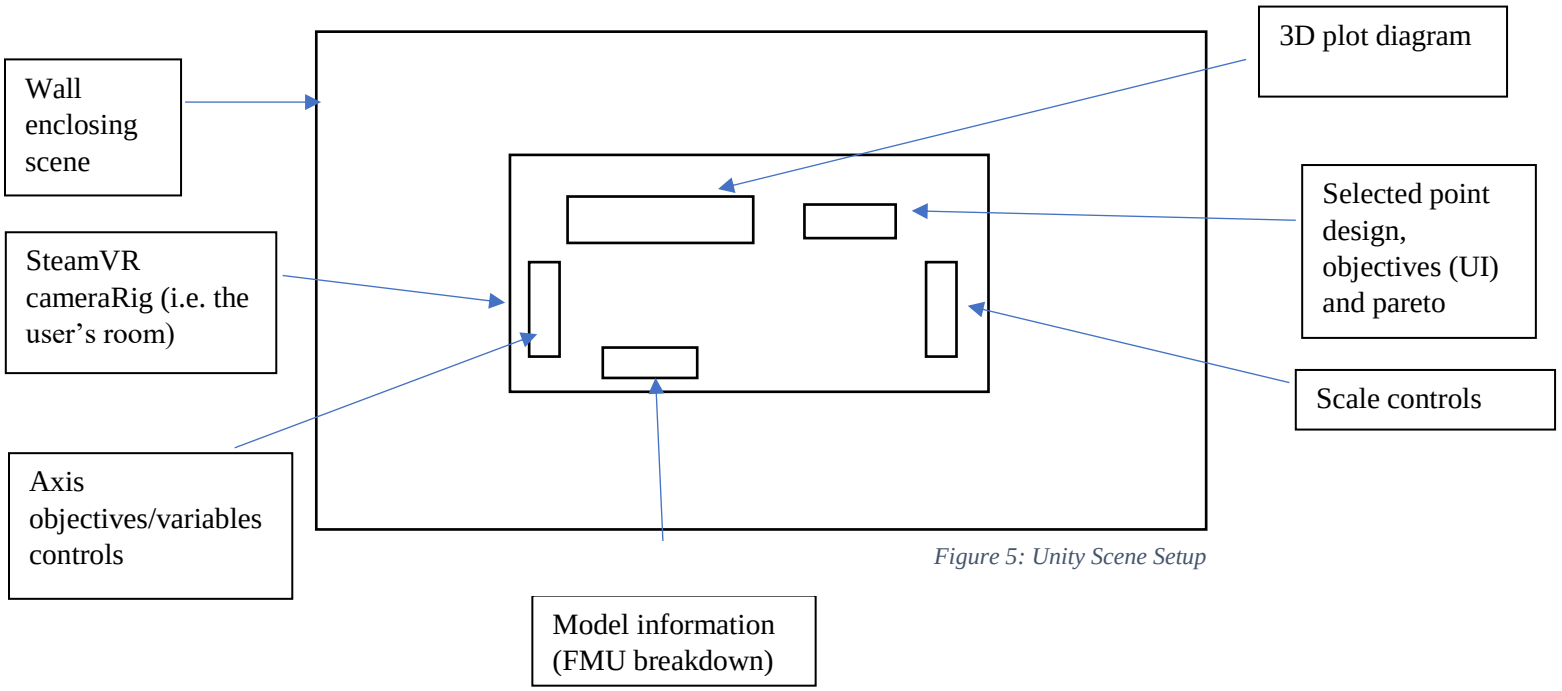

Figure 5: Unity Scene Setup

Unity Scene Game Objects Hierarchy

The Game Objects in the scene are arranged in a hierarchy to ensure that game objects can be appropriate categorised and grouped (e.g. x, y and z axes are all children of the graph).

Note: Certain elements are not shown for simplicity or to avoid repetition.

- Directional Light: This represents the light source to the scene.
- CameraRig: This a SteamVR asset that provides the interface to the HTC Vive hardware (HMD and the two controllers).
- Walls: These are 3D cube objects scaled to look like walls that enclose the unity VR scene to add a sense of depth. “Walls” is the empty Game Object that parents all the walls (northern, southern, eastern and western).
- Axis: This empty Game Object represents the graphical scatter plot graph with its UI elements:
    - Axis Elements: These are 3D cubes scaled to look like axes and arranged to form a scatter plot axes.
    - Label Canvas: This contains the UI labels used to display axes labels.
    - Selection Canvas: This contains the UI labels used to display the selected objectives values for the laser pointer selected point. Also, contains labels for pareto information.
    - Design Canvas: This contains the UI labels used to display the selected design parameters of the laser pointer selected point.
- Scale Buttons: An empty Game Object used to keep all scale control related objects:
    - Scale Down Button, Scale Up Button: These are 3D cubes scaled and coloured to look like buttons, they scale the plot when selected with the laser pointer.
    - Scale Buttons Canvas: This contains all the button UI labels.

- Axis Buttons: An empty Game Object used to keep all scale control related objects:
    - X Button, Y Button, Z Button: These are 3D cubes scaled and coloured to look like buttons, they change which objective is shown in the axes when selected by laser pointer.
    - Invert Axis Button: Like the other buttons, this is also a manipulated 3D cube, but selecting it with a pointer will rearrange the x, y and z axes.
    - Axis Button Canvas: This contains all the button UI labels.
- FMU Canavas: Holds the labels used to display model configuration information

User Interface

As was described in section 3.3.2, various UI elements are required for this project, which include:

1. Axis labels
2. Scale buttons (Up and Down)
3. Axis objectives control buttons
4. Scale labels
5. Objective values for selection
6. Design values for selection
7. Co-model configuration labels (i.e. FMU Configuration Display)

To ensure continuity in user interaction, the operation of UI buttons would be like point selection operation. The user must point the laser pointer of his control to the object and hit the trigger to interact with or engage the object.

Other controls (i.e. proportion of axis and enabled axis) are provided via the inspector window in the Unity Editor. This is to avoid cluttering the Unity Scene with control and because writing numeric values in VR environment is difficult without a keyboard.

Selection Results

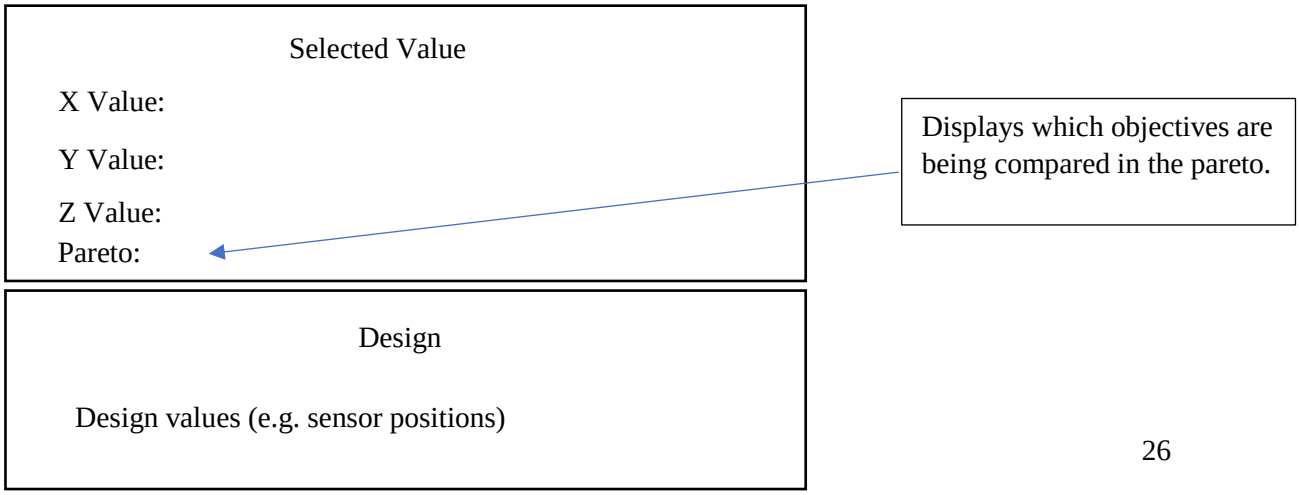

Figure 6: Selection Results UI

Settings Interfaces and Buttons

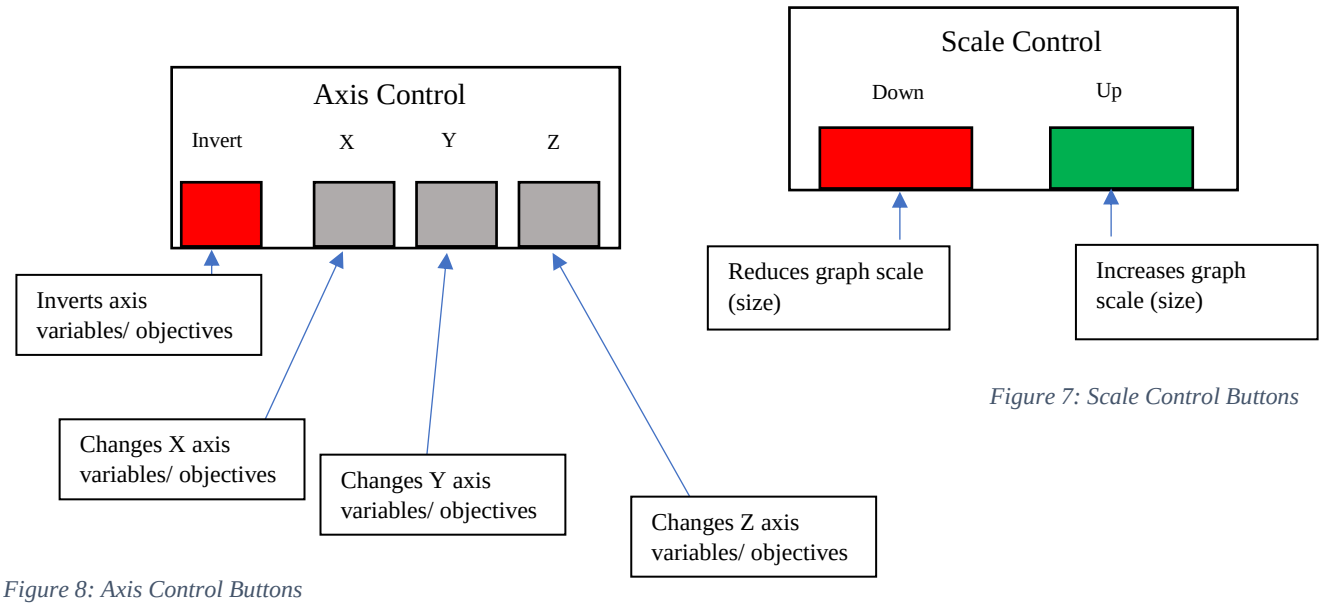

Figure 7: Scale Control Buttons

Figure 8: Axis Control Buttons

FMU Configuration Display

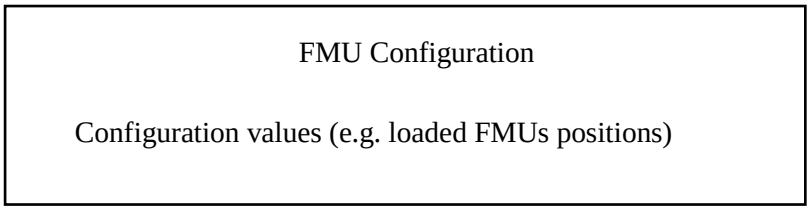

Figure 9: FMU Configuration Output

Unity C# Scripts

Unity C# as described in section 3.2.1 can affect the behaviour of Game Objects and allow game objects to have programmable functionality. Various scripts were written for this project and attached to certain game objects:

- Plotter: This script is attached to the Axis Game Object, and is responsible for enabling, scaling and drawing axes. Moreover, this script generates the points via reading the dseResults.csv files and reads pareto information from the dseConfig.json file This is done using JSON.NET libraries. both described in section 3.2.2. The points are coloured based on their rank (1-> green, 2-> yellow, 3->red and no rank -> blue).

- LaserPointer: This script is attached to the user controllers. It controls interactions using the laser pointer, which is done via collision detection. After collision and user response via the controller trigger, the script calls the appropriate methods for the collided objects.

- Point: This script is attached to the point prefab and is used to store the values (objectives, design values, etc.) that used when the point is selected.

- AxisButtonsController: This script is attached to the Axis Buttons Game Object and is used to disable buttons for axes that are not in use.

- AxisButton: This script is attached to axes buttons, it changes the objective that the axes represents to a new unused object ivewhen called by LaserPointer. It then calls the Plotter to redraw the graph.

- InvertAxisButton: This script is attached to the invert button, it switches the objectives between the axes when called by LaserPointer. It then calls the plotter to redraw the graph.

- ScaleButton: This script is attached to the scale buttons, it then calls the plotter to redraw the graph with a new scale when called by LaserPointer based on a scale factor.

- FMUDisplay: This script is attached to the FMUDisplay Canavs, it reads the config.json file (described in section 3.2.2) when the program runs to display FMU configuration information to the user. This is done using JSON.NET libraries.

To better describe the relationships between the C# scripts, a basic UML class diagram is used here with getters, setters, fields not related to other scripts and private methods removed.

Note: Some fields are public so that the Unity Editor inspector window can be used to set the values and that the LaserPointer will store null references (hence the 0.. 1) when no collision occurs.

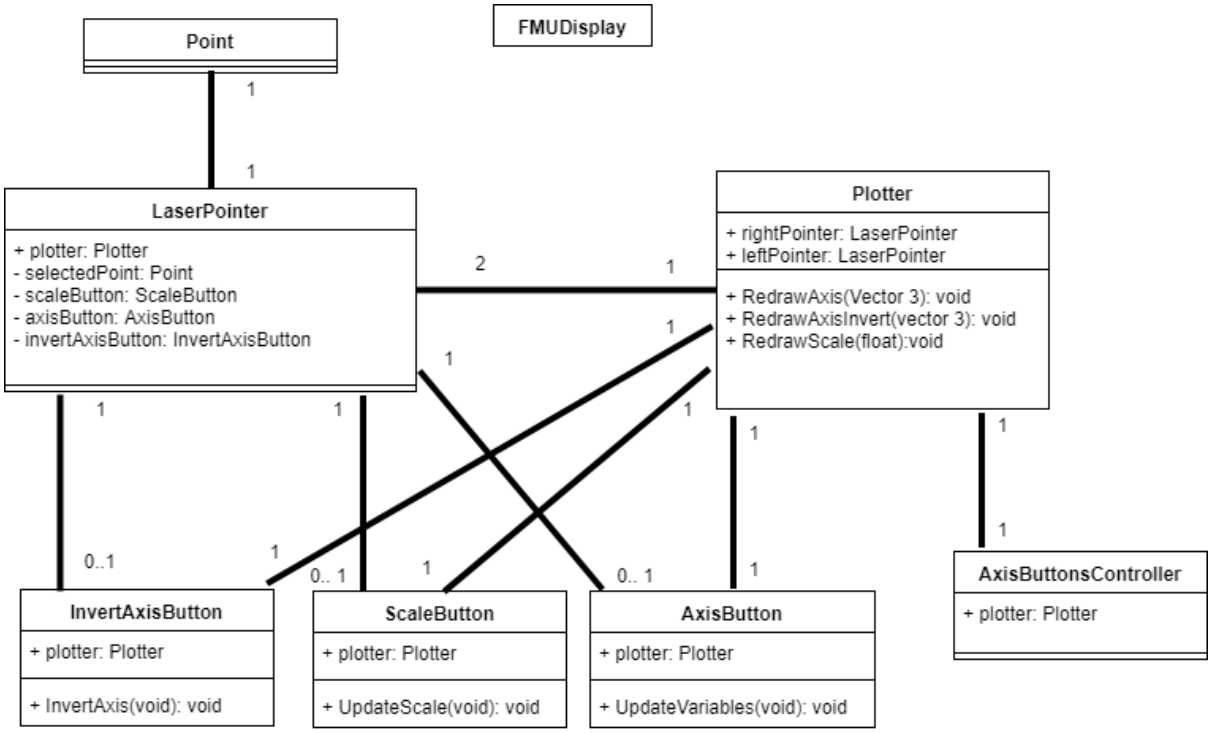

Figure 10: C# Scripts Class Diagram

Unity Assets

Various user generated and store assets were used for this project, from materials for colour and transparency to steamVR assets.

- Materials: The various unity materials used to control transparency and appearance of various game objects (e.g. making the axes transparent)

- Prefabs: These are Game Objects that are saved in the file systems alongside assets for repeated use.

    - Point: This prefab is used to represent the points that would be generated by the plotter C# Script

- Scenes:

    - MainScene: The main 3D VR compatible scene the user would be placed in with all the appropriate Game Objects

- Scripts:

    - CSV: CSV files from INTO-CPS results
        - dseResults.csv: Summarises all design parameters, objectives and ranks (section 3.2.2)

    - JSON: JSON files from INTO-CPS results
        - dseConfig.json: Stores some configuration information and ranking (pareto) data (section 3.2.2)
        - config,json: Stores FMU configuration information (section 3.2.2)

    - Unity C# Scripts: All Unity C# scripts are kept here

- SteamVR: A store asset that contains all steamVR assets which are used to interface with the HTC Vive

3.4 Prototypes

To experiment with features and try out new tools, prototypes were developed throughout the project. These prototypes were built using the Unity Engine and were crucial in providing the required experience to use Unity C# scripts and the SteamVR library.

3.4.1 Controller Interaction Prototype

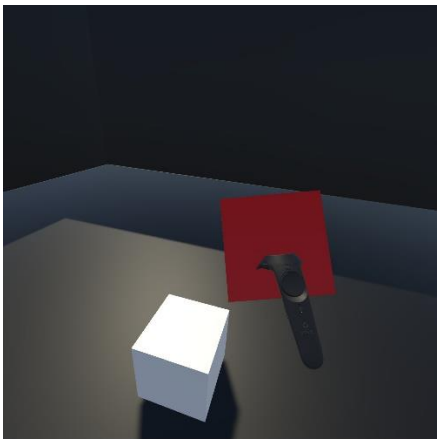

Figure 11: Controller Interaction Prototype

Understanding how the controllers are interfaced with the Unity Game Engine via SteamVR is of crucial importance to this project, as various controls would be handled through the controllers as described in section 3.3.2. To ensure familiarity with SteamVR controller interfaces a prototype with basic controller interaction was constructed. The prototype had the following basic features:

- A basic 3D environment to be presented to the user through the HMD including a grid to limit the user to his HTC Vive room setup

  This was done by using a Unity 3D scene with a SteamVR CameraRig which sets up the VR room in the scene and allows the HTC Vive HMD to display the room.

- User controllers are rendered, and their position is displayed in real time

  This was also done by the SteamVR CameraRig as well, which displays any connected controllers to the scene.

- 3D Cubes with collision detection that can be picked by the user through trigger control

  By using SteamVR library is a Unity C# script it was possible to detect controller button presses using provided events. So, when a side trigger button was pressed near a cube, using the collision box in the cubes a collision is detected and the cube transform is adjusted to follow the controller.

3.4.2 Cube and Sphere CSV Reading Prototype

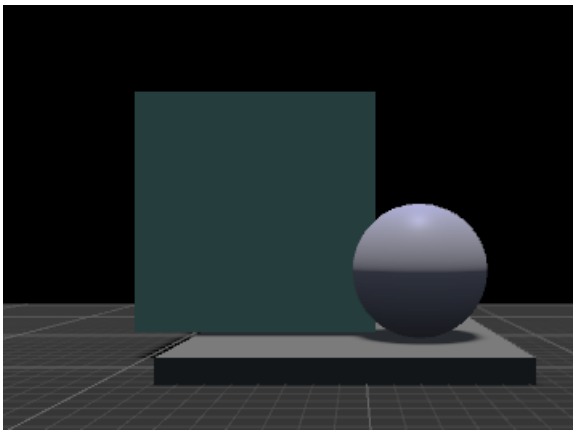

Figure 12: CSV Reading Prototype

Since reading results is an important requirement of this Project as described in section 3.3.2 a prototype that reads csv files was constructed. This prototype was built using files from the previous prototype but adding addition Game Objects and C# Scripts to allow csv files to be read.

The prototype has the following additions over the previous prototype:

- A Cube and a Sphere object

These are basic 3D Game Objects added to the 3D Unity Scene.

- A script controlling the colour and scale of the objects through two .csv files

A Unity C# was attached to the cube and sphere Game Objects which would read a CSV file with a row for colour and another for scale. Colour would be set by changing the material component and scale via the object's local scale.

3.4.3 Basic Plot Prototype

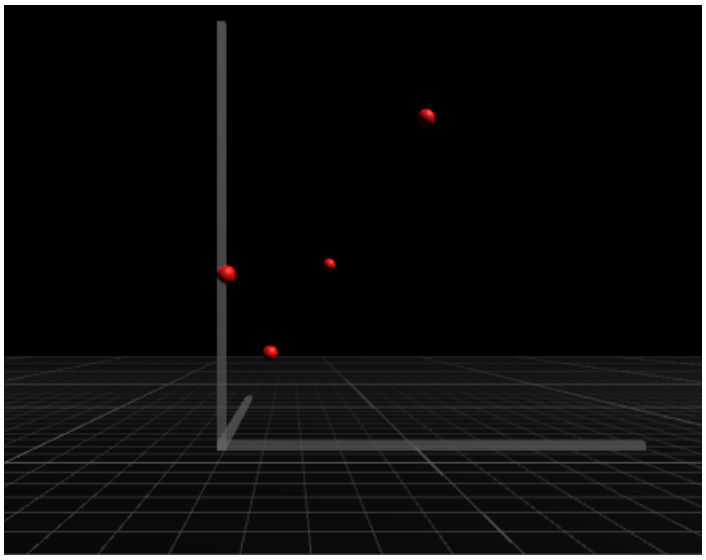

Figure 13: Basic Plot Prototype

After the previous tutorial was tested and .csv files were read correctly, a newer prototype was constructed based on the previous prototype. A CSV file with 3 axes was constructed and the prototype was tasked with reading them and providing the user with a 3D graphical display of the results in a plot graph. To achieve this the prototype would:

- Display X, Y and Z positive axes

  These are scaled cubes which are contained in a common parent with a script. They use a transparent material via their material components.

- Carry points creation

  This would be done using the script attached to the axes parents, which would read the exact values for each axis row by row and draw a point prefab at that point.

This prototype forms the basis for the project graphical 3D scatter plot graph.

3.4.4 Laser UI prototype

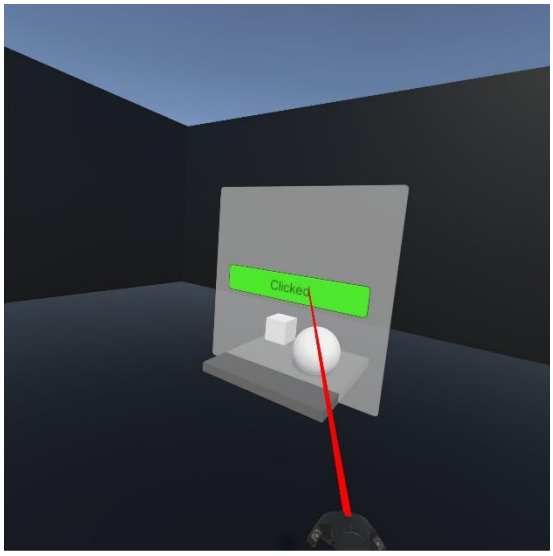

Figure 14: Laser UI Prototype

This project must allow the user to control the graph through motion control as described in section 3.3.2. A suggested method based on DataVizVR selection feature described in section 2.5 was considered. So, following a new prototype with the following features was constructed based on content from previous prototypes.

1. A UI based on canvas with labels

   These UI labels would contain a box collider.

2. A controller that with laser pointer that can highlight buttons and engage the button with the trigger

   This is done by adding a SteamVR_LaserPointer to the controllers in the camera rig and using it to detect collisions with the box collider place in the labels. If that occurs the button is selected and via the on click property, and if the trigger is pressed an action can be engaged and the button is clicked.

This prototype forms the basis for UI interfaces to control variables and graph configuration. However, as shown in section 3.3.4 the use of cubes to form buttons was preferred over UI labels, ths was done to differentiate between output (labels) and input (buttons).

Chapter 4 Planning and Testing

4.1 Planning and Testing Introduction

This chapter goes through how the design and implementation process described in the previous chapter was planned and tested. Specifically, going over plan, testing and going through any changes in plans during development.

4.2 Planning

This section would go over how that design in section 3.3 was implemented, going through how the requirements were planned to be completed. This would be done through the illustration (through a PERT Chart) and explanation of the original plan in section 4.2.1, and through covering any changes applied to that plan with their motivation in section 4.2.2.

4.2.1 Development Techniques and Planning

Pert Chart with Table Key:

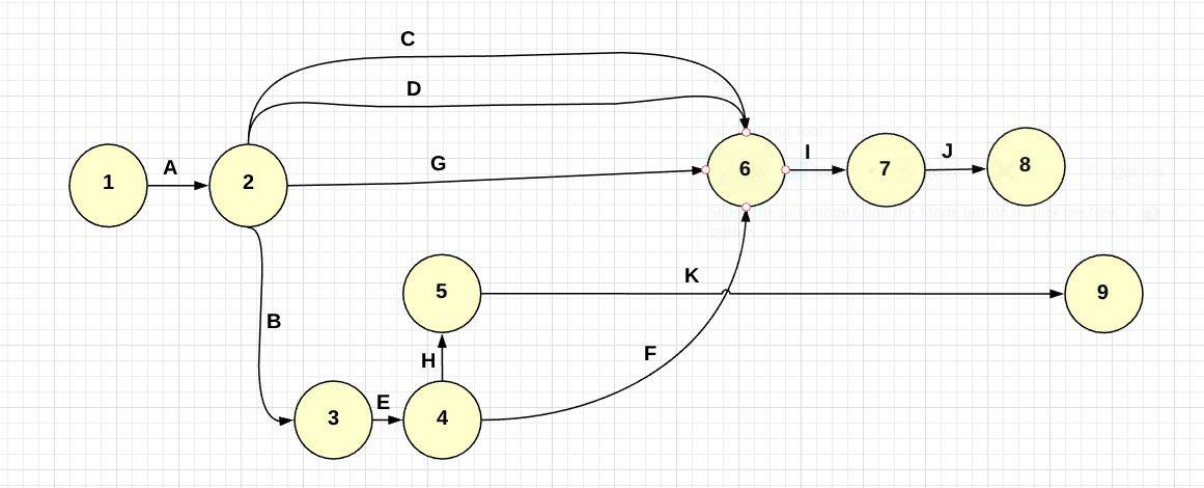

Figure 15: Project Plan PERT Chart

Index	Activity Description	Required Predecessor	Duration (weeks)	Internal Deadline	Deadline
A	Project Proposal	(none)	2	29/11/2017	01/12/2017
B	2D VR graphing program	A	2	28/02/2018	-
C	Dissertation (Introduction and background)	A	2	21/02/2018	-
D	Dissertation (Project Tasks and reasoning)	A	2	05/04/2018	-
E	Multiparameter DSE illustration VR program	B	4	14/03/2018	-
F	Motion control support	E	2	28/03/2018	-
G	Dissertation (Results and evaluation	F	2	19/04/2018	-
H	Poster	E	2	17/04/2018	20/04/2018
I	Dissertation (Conclusions and appendix)	C, D, G	1	26/04/2018	-
J	Dissertation final checks and proofing	J	1	01/05/2018	04/05/2018 (final dissertation)
K	Demonstration	H	2	(rehearsal) 08/05/18	10/04/2018

Note: Tasks start date is typically (Internal deadline – duration), but some flexibility might be allowed.

Work Plan Explanation

The overall project plan relies on waterfall like development methods with clear steps to allow for a ridged structure and simple steps towards the goals of the project. However, some level of agility was used allow for quick revisions based on supervisor feedback and to allow routine testing to be integrated with the tasks themselves to detect issues early.

The plan is divided into two main routes during most of the duration of the project, a project route and a dissertation route. The two routes later join before finalising the dissertation results and conclusions. The main critical path extends from the project proposal, the project tasks, the poster and finally the final demonstration.

The project tasks involve the creation of the initial two parameterised graphing/charting VR tool for experimental prototyping and as a starting point, then the virtual Unity scene where the user lies is constructed starting with a basic 2D graph, and then eventually fully featured 3D graphs are added. User control and control over the co-simulation toolset beyond reading CSV files would be introduced later. Dissertation tasks follow the basic structure of introduction, background, results and conclusions.

Potential risks in the plan include the threat of delays in the main critical path (project to demonstration), limited supplies of VR hardware and the potential API/compatibility issues due to the recency of VR. To mitigate and work around these risks, various contingency plans have been considered. These plans include the use of generous internal deadlines (e.g. for the dissertation background and introduction), early appointments for the use of VR equipment in the Lab and the use of officially supported libraries like Unity.

4.2.2 Changes

Throughout the project, various requirements, plans and objectives were revised to better suit the progress or the tools and techniques used. These changes are highlighted in this section.

DSE Control

Earlier versions of the requirement specification suggested controlling the DSE algorithm directly (e.g. backtracking and changing objectives), however as the project progressed this was slowly rollbacked and eventually removed. This was due to several reasons, including the results of the interview in appendix C which showed that rerunning DSE algorithm was rare. Moreover, as would explained soon, interfacing to INTO-CPS is difficult making DSE controls hard to implement.

INTO-CPS Interfaces

Earlier versions of requirement specification suggested interfacing directly with INTO-CPS to retrieve results and control the DSE process. This was however deemed unnecessary and overly complex, as DSE control took less importance as the project progressed and was eventually removed, moreover INTO-CPS offers limited interfacing options.

Plans and Prototypes

The original envisioned plan shown in section 4.2.1 included only one prototype, a 2-dimensional VR prototype. As the project progressed this was deemed insufficient and the 2-dimensional VR prototype unnecessary, as the design of a 3D VR scene was crucial to the project other prototypes relating to the design of the 3D scene were considered as described in section 3.4. Moreover, a prototype related to user interaction (Laser UI prototype 3.4.4) were considered as user interaction and was deemed crucial due to the various controls required as described in section 3.3.4.

Menu, Errors and Configuration

Some form of a menu was originally intended instead of handling proportion and axes enabling through the Unity Editor and a full error display. However, these were scrapped due to delays and the program still runs in the Unity Editor with Debug logs used to alert user and the inspector used to set values.

Delays

Lastly, some of the plans suffered some delays. Due to changes from less DSE control emphasis to more user interaction and prototyping, which lead to delay the completion of the motion controls by two weeks. And since the delay was on the critical path, this lead to various consequence in other parts of the project (e.g. testing, as described in section 4.3).

4.3 Testing

This section goes over how the various implementable project elements were tested to ensure correct behaviour according to specified requirements. This chapter would specifically go over which testing techniques were chosen for the project, the test plan followed and a summary of the test results.

4.3.1 Testing Techniques

Other than static analysis of code, the Unity testing tool Unity Test Runner was proposed to be used to carry Unit testing to ensure that Unity C# scripts operate as intended and return the appropriate results. Furthermore, Black Box testing methods were applied on completed requirements from section 3.3.2 using a test plan, which is illustrated in the next section 4.3.2. This project relied mostly in Black Box testing and rarely ran the Unity Test Runner as it's a user-oriented tool and due to time constraints outlined in section 4.2.2.

The test plan was constructed based on the requirements in section 3.3.2, with each requirement having at least one test in the test plan, or more when the user actions can cause the issues with the functionality of the requirement (e.g. requirement 13). The test plan also attempts to cover all the user interaction methods through the UI described in section 3.3.4 (e.g. the scale up and down buttons are tested individually).

The test plan is also used to carry the user evaluation, acting as a method to present the user with tasks to test the resultant product from the project as shown in section 5.1.1. The user evaluation will also act as a second run for the test plan to ensure that any bugs that might have been missed in the original run might be spotted by users.

4.3.2 Test Plan:

Test	Requirement	Test Result	Comments
UI elements display objectives being compared in pareto	2	Pass	
Correct model FMU configuration can read from FMU UI	3	Pass	
Points are drawn for each row of data in dseResults.csv	6	Pass	
Points are coloured based on their pareto class	6	Pass	
Unity Scene is loaded and seen using HMD	4	Pass	
3D scatter plot graph is visible to user in the scene	5	Pass	
Hitting the trigger while pointing to a scatter point will reveal its design and objective values	7 & 8	Pass	
Hitting the X axis button with pointer while clicking trigger will change X axis objective	9	Pass	
Hitting the Y axis button with pointer while clicking trigger will change Y axis objective	9	Pass	
Hitting the Z axis button with pointer while clicking trigger will change Z axis objective	9	Pass	
Hitting the invert axis button with pointer while clicking trigger will invert axis objectives	10	Pass	
Hitting the scale up button with pointer while clicking trigger will increase the graph size	10	Pass	

Hitting the scale down button with pointer while clicking trigger will decrease the graph size	10	Pass	
Points are scaled according to user specified proportions	11	Pass	
Negative Axes are disabled if set minimum in CSV is not negative	12	Pass	
Positive Axes are disabled if set maximum and minimum in CSV are negative	12	Pass	
User can disable X axis	13	Pass	
User can disable Y axis	13	Pass	
User can disable Z axis	13	Pass	
Users can’t disable more than one axis	13	Pass	The program might enable more than two axes when fixing it, but this is acceptable as it still fixes the issue and meets the plan.
Having a larger negative minimum will scale down the positive axis relatively	14	Pass	
Having a smaller negative minimum will scale up the positive axis relatively	14	Pass	

Chapter 5 Results and Evaluation

This chapter goes over how the final product and research stacks up to the originally envisioned aims and objectives in section 1.3 through user testing and analysis of completed features. Lastly, an evaluation of the planning, testing and tools used throughout the project was provided.

5.1 Final Product and Objectives

A finalised product that is run in a Unity Editor was developed by the end of the project with all requirements specified in section 3.3.2 implemented as features and tested using the test plan in section 4.3.2. To evaluate the results of the project, a user evaluation was carried (explained with results in section 5.1.1), and the final product was put against the aims and objectives specified in Chapter 1 section 1.3 in section 5.1.3. Moreover, the project has also produced research that will also be evaluated against the aims and objective as well in section 5.1.3.

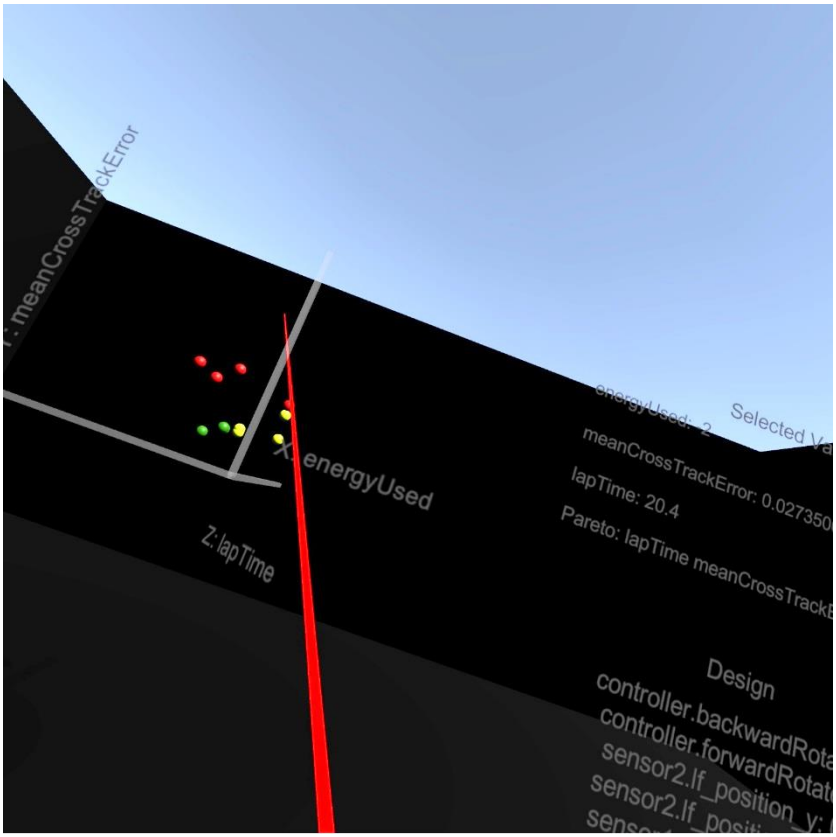

Figure 16: Selection Feature in Final Product

5.1.1 User Evaluation of the Product

Since this solution is user oriented, a user evaluation was carried to evaluate the final product's usability results. This was done though a user evaluation interview after the user has experimented with all the features of the product using the test plan outlined in section 4.3.2. The interview and user test were carried by individuals who experimented DSE beforehand, hence both participants have done research into the subject.

The participants were a PhD research associate in Newcastle University who conducted research into co-modelling and DSE, and a Computer Science undergraduate 3rd year student who was doing a dissertation about DSE as well. The questions and results of the interview are available in Appendix D for both participants, but a summary of the results of the evaluation is provided in this section.

Results Summary

PhD research associate:

This participant liked the overall user interaction model, as they found that the user selection feature was helpful and liked the overall consistency of the UI. Moreover, the participant has found that the buttons were easy to use and appropriate for the control model. However, this participant had suggested that the solution should be able to highlight the selected points themselves.

About the FMU configuration display, the participant has found that it could be useful under certain circumstances, but that it was not critical and could have used information about multi model connections.

The participant also found that the graph was easy to read but could use a function to move it around and needs numerical value labels.

This participant thought the 3D VR scene was well laid out with calm colours and no distracting elements and they liked the placement of non-graph related elements away from the graph. But, they found that some of the colour (output) clashed with the background.

They also provided an additional suggestion in that the axes objectives could be switched by pointing the laser at the graph axes and clicking the trigger rather than just through buttons.

Undergraduate 3rd Year Student:

Similarly, this participant also liked the overall user interaction model and found the use of buttons convenient and easy. They also liked that the numerical controls (e.g. control of proportions) were kept away from the VR space and moved to the Unity inspector, where they are easier to control with a keyboard. However, while they liked selection feature as well, they thought that it could have been possible to add the rank of the selected point to the selection output.

About the configuration display, this participant thought that it would useful when configuring multi models DSE simulations and couldn't think of any other additions to this feature.

About the plot, this participant has found that it was easy to read generally but can be difficult to read at times when points cluster together and had suggested that the minimum and maximum should be set based on the data set not values provided directly in the dseResults.csv alongside the output.

Lastly, the participant liked the layout of the VR scene, but thought that it was a bit dark which made some elements of the scene hard to interact with.

Results Evaluation

To summarise the results, the user interaction model of laser selection and buttons was very well received by the participants and the graph was easy to understand. Moreover, the FMU configuration was helpful to users, and the users also thought that the overall room layout was tidy.

However, several key issues need to be recognised:

1. The scene colours clash

   This makes reading the output difficult, which is worrisome since the selection feature is key to understanding the DSE results. This could have even caused the poor lighting for some users to look worse, as the users would struggle more with colour clashes in poor lit scenarios.

   Suggested solution: Switch output colours to vibrant colours like the ones used in some buttons (green and red) that don't clash with the dark grey background.

2. Numerical labelling of axes values is missing

   This is a consequence of concentrating on the selection feature which lead to the misconception that it would mean numerical labelling would not be as necessary. However, as demonstrated by the user evaluation, some users find it helpful.

   Suggested solution: Add labels that are set based on minimum and maximum axes values and are scaled alongside the axes.

3. Co-model configuration information is limited for some advanced users

   As demonstrated by the user evaluation carried by the PhD research associate, while the FMU configuration display was still helpful it still felt limited. And as described in section 2.2 and 2.3 Co-modelling and DSE go in hand in CPSs, hence the addition of more co-model information is crucial.

   Suggested solution: Display FMU and co-model connection information alongside used FMUs.

4. Graph scaling and scaling control need improvement

   The graph still does not scale itself as well as it could as the minimum and maximum values provided by the dseResults.csv are not accurate enough which can lead to some points to cluster. This can be eased with changing the proportions of some axes in the editor, but no such control is available for individual axis in the Unity VR Scene.

   Suggested solution: Set minimum and maximum based on provided data set and/or provide the user with the ability to change the scale of each axis individually in the Unity scene space.

5.1.2 Completed Product Against Requirements

The requirements described and explained in section 3.3.2 form a guideline for what the final product should do, and hence revisiting them is critical to understanding how well the project progressed.

Evaluation of Requirement Completion

As shown in the test plan in section 4.3.2, all the requirements were tested and passed. Hence, it is safe to assume that the product has met all the specified requirements on paper. However, certain issues were recognised with some of the completed features. For instance, as highlighted in the changes in section 4.2.2 no menu was developed as previously planned and user interaction elements of requirements (13 and 11) are still done via the Unity Editor in the inspector

Moreover, the test plan in section 4.3.2 for requirement 13 has highlighted some minor issues with the implementation of that requirement, as the program would enable two axes when the user tries to enable only one axis sometimes instead of just enabling one extra axis. However, since that behaviour still fits into requirement 13, it was considered acceptable.

Regardless, it can be assumed with regards to the base requirements on paper, this project has succeeded at meeting them even considering the caveats described here.

Evaluation of the Requirements

An evaluation of the requirements themselves might also be required to evaluate the scope of the project itself. Looking at section 4.2.2 again, various changes was made throughout the project progress which lead to the requirement list missing various features.

For instance, no full error message and menu implementation was done, and objectives can't be changed due to INTO-CPS toolset interfacing limitations.

Moreover, as described in the result evaluation in section 5.1.1 some additional features could have been considered for the requirement specification list. For instance, the addition of individual scale controls and numerical labelling could have been considered.

Lastly, one point raised in the 2.5 research was potential for the use of other graphical charts and graphs in the Unity Scene space like pie charts for instance. This however, was not realised in the requirements in the specification.

Regardless, the requirements still managed to achieve most of the objectives as shown in the next section 5.1.3

5.1.3 Completed Project Against Aims and Objectives

To better understand the overall project progress, it is crucial that the original aim and objectives shown in section 1.3 are revisited and contrasted with the completed solution to evaluate project progress. The original aim of the project was:

"Illustrate the results of DSE of Cyber Physical through visualisations using VR equipment"

And this aim was supposed be reached through the completion of the following objectives:

1. Review and work around the risks and limitations of data representation with VR equipment

2. Illustrate the multidimensional Cyber Physical DSE in 3D graphs using VR hardware

3. Illustrate the effects of changing Cyber Physical DSE objectives using motion controls in a VR environment on three-dimensional graphs.

4. Allow the user to switch to different sets of dimensions in the design space dimensions through VR motion controls

Looking at the solution and contrasting it with objectives:

Objective 1: In the scope of this project this objective can be considered completed, as having a functional VR solution means that various issues with VR development that were raised in section 2.4 were worked around. This was done via the use of the open SteamVR libraries and Unity to ensure well supported and tested VR libraries to avoid the limitations of VR development and proprietary hardware raised in section 2.4. However, the project should have considered more limitations and risks of VR and VR developed and proposed potential solutions (e.g. health, requirements, motion control and physical disability, etc.).

Objective 2: The results of the DSE exploration process in dseResults.csv file described in section 3.2.2 were drawn in a scatter plot graph as described by requirement 6 in section 3.3.2 which was contained in the Unity 3D VR scene. This functionality was handled by the plotter C# script described in section 3.3.4. This means that the DSE results with their multiple dimensions (i.e. objectives in this case) where illustrated 3D Scene over VR hardware (i.e. the HTC Vive) and this objective is completed.

Objective 3: This objective however was not achieved, as explained in section 4.2.2 various issues were faced with interfacing with the INTO-CPS toolset, which prevented the changing of DSE algorithm objectives.

Objective 4: This objective was completed, as the currently presented objectives in the x,y and z dimensional objectives can be switched around these axes. This was done though requirement 9 (section 3.3.2) with the interface controlled by the Vive controller described in figure 8 and the AxisButton script (both in section 3.3.4).

5.2 Evaluation of Development Tools and Planning

This is an evaluation of the software engineering aspects of the project, providing a review of how effective the tools, plans and test methods that were applied in the project were. This section also covers the issues faced in the software engineering aspects of the project with justification.

5.2.1 Evaluation of Tools

This section runs over the tools listed in section 3.2 and describes the experience of using them, including issues faced when using tools and the help provided by the tools to achieve the objectives in section 1.3.

Unity

As described in section 3.2.1, Unity was chosen for its VR support and versatile C# Scripts described in the same section. In that sense Unity has delivered as its C# scripts were used for various forms of functionality as described in section 3.3.4 and has provided full compatibility with the SteamVR library. However, unity has caused few minor issues with the JSON library (JSON.NET)) used in the project as it failed to import some of its libraries.

HTC Vive and SteamVR

The HTC Vive was fully compatible with the used SteamVR libraries and no issues were faced with regards to the hardware. The SteamVR libraries were crucial in developing controls and interfacing the scene with the HMD as described in the prototypes in section 3.4 and has not caused any issues worth reporting.

INTO-CPS

Due to issues with interfacing with the tool set described in section 4.2.2, direct interaction with this tool was rare and generally its results described in section 3.2.2 were the elements this solution interacted with. When it comes to its results, the results were well formatted.

5.2.2 Evaluation of Testing Techniques

This section contains the evaluation of the effectiveness and completeness of the testing techniques described in section 4.3.

As described in section 4.3, the testing techniques used was Black Box testing via a test plan and Unit Testing via the Unity Test Runner. However, the Unity Test Runner was rarely run due to time constraints and hence the testing was mainly done using the Test Plan.

In that regard the test plan was successful, as the participants of the user evaluation reviewed in section 5.1.1 were satisfied with the solution and did not find any major bugs to report. However, given the extra time various testing techniques would have been added and used alongside the Test plan.

For instance, more white box testing techniques other than Unit Testing could have been researched for Unity like Data Flow and Control Flow testing techniques. These techniques could have been particularly helpful judging by the level of communication between scripts described in diagrams in section 3.3.4.

5.2.3 Evaluation of Project Plan

This section contains an evaluation the of the project plan described in chapter 4, including discussion of any changes.

The project plan employed is described in section 4.2 (4.2.1 for the PERT diagram and the explanation) and was followed throughout the project with changes listed in section 4.2.2.

The project plan overall with the changes applied was clear enough to lead to all the requirements specified in section 3.3.4 completed but was not without its issues as described in the changes section. As, for instance the number of prototypes that were required to be developed were nowhere near the number nor the original description. Moreover, the DSE algorithm objectives run-time changes had to be abandoned due to poor understanding of the INTO-CPS toolset interfaces. This lead to delays in the critical path which caused issues in other parts of the project (e.g. testing as briefly described in the previous section).

These issues could have been avoided by dividing the current tasks into more abstract tasks and though more communication with the supervisor and research about the INTO-CPS toolset.

Chapter 6 Conclusion

This chapter finalises the project by providing a summary of the project through summarising the progress to the final product, highlighting faced issues and providing suggestions for future work to carry over progress done in this project.

6.1 Summary of Project Progress and Results

This project produced a Unity Project and research into various concepts and tools, which are summarised in this section. The results of the project have largely met the original aims and objectives with only a single objective left out and one partially met as described in section 5.1.3.

6.1.1 Completed Research

As described in section 2.5, the results of the research conducted about co-modelling, co-simulation INTO-CPS, DSE, VR and similar solution lead to the conclusion that Cyber Physical co-models are complex structures that require analysis with methods like DSE exploration. However, the research has shown the DSE results are complex and difficult to understand. Hence, a VR based solution that takes inspiration from solutions like DataVizVR and the advantage provided by VR data representation was proposed to make illustrating DSE results easier. The research has also shown the challenges in VR development, specifically the difficultly of standardised tools, which lead the choosing popular and compatible development tools like SteamVR and Unity to help alleviate these issues.

When the research was results were compared against the aims and objectives in section 5.1.3, it has been found that while the research has helped in finding the correct tools to work around VR limitations described in requirement 1, various other potential VR limitations have been left out (e.g. health, hardware limitations, etc.)

6.1.2 Completed Product

The final product is a Unity project file that can be viewed and manipulated through a Unity editor. This product provides all the features described in section 3.3.4 with the aim of providing the user with the ability to illustrate Cyber Physical Systems in a VR space. The that were implement according to designs produced in section 3.3, and fully tested through a Black Box testing via a Test Plan that is shown in section 4.2. This final product has achieved most of its objectives except for objective controls at the DSE algorithm level as described in the evaluation in section 5.1.3.

The completed product received largely positive feedback from the user evaluation carried in section 5.1.1, with the standard interface model, scene design and graph readability being well received. Moreover, the evaluation has provided some suggestion for improvements like better colour choices, numeric labelling and more co-model configuration information.

6.2 Project Issues

This section highlights any issues faced with the final product or the software engineering process to reach the final product.

6.2.1 Project Delays and INTO-CPS Interfacing Issues

As described in the project plan evaluation section 5.2.3, various issues with the way the project was planned had been faced which lead to delays and objectives related INTO-CPS interfaces being scrapped due to issues. As described section 5.2.3 as well, these issues could have been avoided though more research, communication and a more abstracted plan.

6.2.2 Program Menu and Release

The final product is still a Unity Project and lacks a full menu as described in the evaluation of completed product against the requirements in section 5.1.2. And while this did not lead to issues in the User Evaluation in section 5.1.1, a completed menu and a full release executable would have made it easier to use and more portable. Avoiding the delays with a better project plan might have helped avoid this issue.

6.2.3 Limited Testing

Due to delays, organised testing was mostly limited to Black Box testing as described in the Testing section 4.3, which while makes more sense in a user-oriented solution, it might not be always sufficient nor provides enough guarantee of the product reliability. Avoiding delays with a better project plan might have also helped avoid this issue as well.

6.3 Future Work

This section highlights potential additions to the solution in the future.

6.3.1 3D Simulations Integration

The interview conducted with Frederik Forchhammer in Appendix C has shown that when results of DSE algorithm are to be shown externally, Unity is used to create 3D models that are shown to individuals less accustomed to DSE and Cyber Physical concepts. This means that the solution provided by this project can be interfaced with provided 3D simulations and models, perhaps through switching Unity Scenes that are described in section 3.2.1 through a menu.

6.3.2 INTO-CPS Integration and DSE Controls

If more research and more improvements for the INTO-CPS interfaces is done, it might be possible to get the Unity solution interfaced with the INTO-CPS toolset. This might enable better control over the DSE algorithm, allowing reruns with different objectives and showing the results on the ranking of designs all in the Unity Scene. Moreover, it might even be possible to switch out different components in the Co-Model through control over used FMU in the scene as well.

References

1- Gries, M. (2004) 'Methods for evaluating and covering the design space during early design development', *Integration, the VLSI Journal*, 38(2), pp.131–183.
2- Al-Hammouri, A.T. (2012) 'A comprehensive co-simulation platform for cyber-physical systems', *Computer Communications*, 36(1), pp.8–19.
3- Kokhazadeh, A. & Fatemi, O. (2011) 'Design space pruning of MPSoCs using weighted sub-sampling', *Electronics, Circuits and Systems (ICECS), 2011 18th IEEE International Conference on*, pp.550–553.
4- Valdés, Romero & Barton. (2012) 'Data and knowledge visualization with VR spaces, neural networks and rough sets: Application to cancer and geophysical prospecting data', *Expert Systems With Applications*, 39(18), pp.13193–13201.
5- Fitzgerald, J., Gamble, C., Payne, R., Pierce, K. (2016) 'Method Guidelines 2. D3.2a, INTO-CPS project (644047)'.
6- Lausdahl, K., Larsen, P. G., Wolf, S., Bandur, V., Terkelsen, A., Hasanagić, M., Hansen, C. T., Pierce, K., Kotte, O., Pop, A., Brosse, E., Brauer, J., Möller, O. (2015) 'Design of the INTO-CPS Platform. D4.1d, INTO-CPS project (644047)'.
7- Velev, D. & Zlateva, P. (2017) 'VR Challenges in Education and Training', *International Journal of Learning and Teaching*, pp.33–37.
8- Fitzgerald, J., Larsen, P., & Verhoef, M. (2014) 'Collaborative design for embedded systems : co-modelling and co-simulation'.
9- Ware, C., & Franck, G. (1994) 'Viewing a graph in a virtual reality display is three times as good as a 2D diagram', *IEEE Symposium on Visual Languages, Proceedings*, 182-183.
10- Huang, Y.-J., Fujiwara, T., Lin, Y.-X., Lin, W.-C., & Ma, K.-L. (2017) 'A gesture system for graph visualization in virtual reality environments', *IEEE Pacific Visualization Symposium*, 41-45.

11- Steam (2018) *SteamVR Plugin - Asset Store*. Available at: https://assetstore.unity.com/packages/templates/systems/steamvr-plugin-32647. (Accessed: 30 March 2018).

12- Unity (2018a) *Unity - Manual: GameObjects*. Available at: https://docs.unity3d.com/Manual/GameObjects.html. (Accessed: 30 March 2018).

13- Unity (2018b) *Unity: The world-leading creation engine*. Available at: https://unity3d.com/unity. (Accessed: 30 March 2018).

14- Unity (2017a) *Unity - Manual: Asset Workflow*. Docs.unity3d.com. Available at: https://docs.unity3d.com/Manual/AssetWorkflow.html. (Accessed: 30 March 2018).

15- Unity (2017b) *Unity - Manual: Creating and Using Scripts*. Docs.unity3d.com. Available at: https://docs.unity3d.com/Manual/CreatingAndUsingScripts.html. (Accessed: 30 March 2018).

16- Unity (2017c) *Unity - Manual: Learning the interface*. Available at: https://docs.unity3d.com/Manual/LearningtheInterface.html. (Accessed: 30 March 2018).

17- Unity (2017d) *Unity for VR and AR*. Available at: https://unity3d.com/unity/features/multiplatform/vr-ar. (Accessed: 30 March 2018).

18- VIVE™ United Kingdom (2017e) *Product*. Available at: https://www.vive.com/uk/product/. (Accessed: 30 March 2018).

19- Gamble, C. (2016) 'DSE in the INTO-CPS Platform. D5.3e, INTO-CPS project (644047)'

20- DataVizVR Inc. (2017) *DatavizVR Demo*. Available at: https://store.steampowered.com/app/551960/DatavizVR_Demo/ .(Accessed: 1 April 2018).

Appendix

Appendix A: Glossary 1.6

A1 Systems:

- Cyber: Computer related components (software/hardware)
- Cyber Physical Systems (CPS): Refers to systems of networked physical and cyber components.
- Functional Mockup Interface (FMI): A tool used by the INTO-CPS system to allow different models to communicate for the co-simulation.
- Functional Mockup Unit (FMU): Components (models) that implement FMI interface

A2 Models:

- Model: Abstract representations of real life systems that models
- Continuous Time model: Models of systems that are continuous in nature (e.g. models of physical systems)
- Discrete Event model: Models of systems that are discrete in nature (e.g. models of cyber systems)
- co-models: Models that consist of abstractions of continuous and discrete systems
- co-simulation: Simulations of co-models
- Shared variables: Variables available to DE and CT models
- Events: Model actions that affect other models
- Shared Design Parameters: Initial constant design parameters shared across a co-model

A3 Tools:

- INTO-CPS tool chain: A set of software tools using FMI interface for co-simulations
- co-simulation Orchestration Engine (COE): Implements algorithms that allow co-models to be co-simulated (the master algorithm).
- Virtual Reality (VR): A 3D simulation of reality

A4 Design Space Exploration, Analysis and Simulation:

- Design Space Exploration (DSE): Systematic exploration of Design Spaces for ideal solution through pruning dimensions by making appropriate Design Candidate choices.
- Design Space: Represents a space of all possible solutions to a given problem.
- Dimensions: In DSE, a dimension refers to a given parameter (e.g. Model, Algorithm used, speed, etc.)
- Design candidates: Candidates of dimensions (values of parameters) that are picked by the DSE algorithm.

Appendix B: Acronyms

- Design Space Exploration (DSE)
- Cyber Physical Systems (CPS)
- Virtual Reality (VR)
- co-simulation Orchestration Engine (COE)
- 3-Dimensional (3D)
- 2-Dimensional (2D)
- Continuous Time (CT)
- Discrete Event (DE)
- Head Mounted Display (HMD)
- User Interface (UI)
- Comma Separated Values (CSV)
- JavaScript Object Notation (JSON)

Appendix C: Frederik Forchhammer Foldageran Requirements Interview Script

1. What do you use the INTO-CPS toolset for and what kind of features do you use the most?
    - Answer:
        - Modelling vehicles
        - Issues in controller
        - Simulate to fix issues
        - Assist in development
        - Fault tolerance and finding issues
        - How accurate it is? How does it react to conditions?
        - Controller testing
        - Visualisation purposes (CAD)

2. Do you use DSE, if so where do you use DSE?
    - Answer:
        - Yes
        - Trying out controller settings to see what is ideal
        - Field with controller inputs, and how well does it decide the route to take

3. What kind of visualisation techniques do you prefer to use to analyse DSE results and why?
    - Answer:
        - Plot using python they made
        - HTML pareto

4. When working with visualisations of data, what do you usually look for in the visualisation?
    - Answer:
        - Depends, variable
        - Visual models
            - Unity, used externally
        - Graphs used internally
            - Building, must point to how well it meets the objectives
            - Pareto important

5. Do you use INTO-CPS to visualise results of DSE?
    - Answer: Yes, HTML pareto

   If so:
    - Why?
        - Answer:
            - Pareto objectives
    - What do you think are better aspects of it?
        - Answer:
            - Results
            - Good summary and easy to communicate (HTML document)
    - What does it lack?
        - Answer:
            - Export to CSV
            - Graph
    - Does 3D help?
        - Answer:
            - More variables
            - Helps in understanding what effects what
    - Will a selection feature help?
        - Answer:
            - Helps in understanding configurations to help optimisation.

6. Other tools used:
    - Answer: Python
        - Why?
            - Answer: 3D graphs

7. When working with DSE on Co-models, what is the most important configuration setting you need to control?
    - Answer:
        - Trying out various settings
        - Wait than try to check all results
        - Objectives important
        - Rarely ran multiple times

8. When working on graphs involved with Co-Models, what is the most important graph related settings you need to control? What kind of variables?
    - Answer:
        - Controller settings and configuration, how the model is controlled.
        - Vehicle configs considered (mass and height)

Appendix D: User Evaluation Interview

Key:

- S: Bachelor of Science Student Answers
- P: Phd Research Associate Answers

1. What do you think of the user interaction model?
    a. What did you like about it?
        S: Buttons intuitive, easy to code and use. Using config easy.

        P: Buttons easy to use and interface is consistent

    b. What did you not like about it?
        S: Pareto rank could be added to selection feature

        P: Highlight selected points (helpful for dense graph)

    c. What do you think of the point select feature?
        S: Easy and intuitive

        P: Very easy to use but could use a highlight feature

    d. Out of 10, how would rate it?
        S: 7/10

        P: 7.5/10

2. What do you think of the configuration information provided by the solution?

    a. Out of 10, how useful is it?
        S: 7/10, Useful when designing or configuring

        P: 5/10, Interesting to look at and well positioned. But occasionally used only

    b. What other configuration information do you think could be added?
        S: Looks sufficient.

        P: Perhaps model connections and paths to data.

3. Out of 10, how easy is the scatter plot graph to read?

    S: 6/10, Relatively easy to read, but some of the points tend to cluster. Axis should scale based on lowest point not minimum and maximum set in csv file.

    P: 7/10, could use a feature to move it and Numerical labels could be added however.

4. What do you think of the 3D VR Scene provided

    a. Are there any distracting elements?

        S: None.

        P: Nice and calm, grey works well for the environment. But, result colours clash and might need to be brighter.

    b. Are the scene objects well laid out and well-spaced?

        S: Well laid out and easy to find.

        P: Well-spaced and buttons easy to find. Configuration information and other non-critical information are away from the graph to not distract user.

    c. Is the scene well lit?

        S: A bit dark

        P: Nice and well lit, graph reflects well.

    d. What other improvements do you suggest

        S: None

        P: None

5. Any other suggestions?

    S: None.

    P: Have axes change when selected with pointer and clicked.