

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wānanga o te Ūpoko o te Ika a Māui



School of Engineering and Computer Science
Te Kura Mātai Pūkaha, Pūrorohiko

PO Box 600
Wellington
New Zealand

Tel: +64 4 463 5341
Fax: +64 4 463 5045
Internet: office@ecs.vuw.ac.nz

**Interactive 3D Visualisation of
Exoplanets**

Owen Bannister, 300172912

Supervisor: Stuart Marshall

Submitted in partial fulfilment of the requirements for
Bachelor of Software Engineering with Honors.

Abstract

We have large amounts of information about planets outside of our solar system. This can be accessed from a database by anyone. However this information is complex and cannot be easily understood by laypeople. This is a problem as it means that the information gained about these planets is not being used effectively to convey the intended information to the masses. To resolve this a visualisation has been created that can convey this information in a way that interested lay people can understand. This visualisation can be used as an information source for those wanting to increase their knowledge about the planets residing outside of our solar system. This report outlines the project carried out, the planning and decisions made, the visualisation created, and its evaluation to discover its effectiveness at fulfilling the goals driving its creation.

Acknowledgements

I would like to thank Dr Stuart Marshall the supervisor of this project. He provided me with endless advice and mentorship over the course of this project making it a very positive experience. Thanks also goes to the Victoria University of Wellington Human Computer Interaction Group who were always free to help and provide me with assistance.

Contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 1 |
| 1.1 | Motivation | 1 |
| 1.2 | Problem statement | 1 |
| 1.2.1 | Understanding the content in the dataset | 1 |
| 1.2.2 | Comprehension of planetary information | 2 |
| 1.2.3 | Existing solutions lack functionality | 2 |
| 1.2.4 | Effective user interaction with visualisation | 2 |
| 1.3 | Key issues project addresses | 3 |
| 1.4 | Contributions of this project | 3 |
| 2 | Requirements Analysis | 5 |
| 2.1 | User models | 5 |
| 2.1.1 | John Truman (Primary Persona - The interested layperson) | 5 |
| 2.1.2 | Cara Thompson (Secondary Persona - Likes gesture based systems) | 6 |
| 2.2 | Scenarios | 6 |
| 2.2.1 | Scenario 1: View planets ordered by their similarity to Earth | 6 |
| 2.2.2 | Scenario 2: Select ranges for attributes of each planet displayed | 7 |
| 2.2.3 | Scenario 3: Select planets to display more information | 7 |
| 2.2.4 | Scenario 4: View planets in the same solar system | 7 |
| 2.2.5 | Scenario 5: View the Goldilocks zones of each planet | 8 |
| 2.2.6 | Scenario 6: Select two planets to compare against one another | 8 |
| 2.2.7 | Scenario 7: Navigate the visualisation with gestures | 8 |
| 2.3 | Requirements summary | 8 |
| 2.3.1 | Functional Requirements | 8 |
| 2.3.2 | Nonfunctional Requirements | 9 |
| 3 | Project Methodologies | 11 |
| 3.1 | Project management approach | 11 |
| 3.1.1 | Software Development LifeCycle (SDLC) | 11 |
| 3.2 | Other project methodologies explored | 12 |
| 3.3 | Supporting Tools for project | 13 |
| 4 | Existing systems and technologies | 15 |
| 4.1 | Existing systems | 15 |
| 4.1.1 | Worlds: The Kepler Planet Candidates - Non Interactive | 15 |
| 4.1.2 | The Kepler Orrery and The Kepler Orrery 2 - Non interactive | 16 |
| 4.1.3 | Celestia - Interactive | 17 |
| 4.1.4 | Kepler Visualisation Tool | 18 |
| 4.1.5 | Summary of Existing Applications | 19 |
| 4.2 | Analysis of technology options | 19 |

| | | |
|----------|--|-----------|
| 4.2.1 | D3 (Data Driven Documents) | 20 |
| 4.2.2 | Prefuse | 20 |
| 4.2.3 | Processing | 20 |
| 4.2.4 | Decision of technology | 20 |
| 5 | Solution Design: Improved Kepler Visualisation Tool (IKVT) | 23 |
| 5.1 | System design and structure | 23 |
| 5.2 | Visualisation Design | 23 |
| 5.2.1 | Functional Requiriement | 24 |
| 5.2.2 | Nonfunctional Requiriement | 28 |
| 6 | Visualisation implementation of the Improved Kepler Visualisation Tool (IKVT) | 31 |
| 6.1 | Additional Tools and artifacts used | 32 |
| 6.1.1 | Dataset used | 32 |
| 6.1.2 | Integrated Development Environment (IDE) | 32 |
| 6.1.3 | Keyboard and Mouse System | 32 |
| 6.1.4 | Microft Kinect sensor system | 32 |
| 6.2 | Main interface components | 33 |
| 6.3 | Problems encountered | 33 |
| 7 | Visualisation Evaluation | 39 |
| 7.1 | Why was evaluation conducted | 39 |
| 7.2 | Expectations of evaluation | 39 |
| 7.3 | Evaluation Environment | 39 |
| 7.4 | User Study Method | 39 |
| 7.4.1 | Participants | 40 |
| 7.5 | Pilot Study | 40 |
| 7.6 | Experiment on Keyboard and mouse visualisation | 40 |
| 7.7 | Experiment on Microsoft Kinect visualisation | 40 |
| 7.8 | Results | 40 |
| 7.8.1 | Analysis of requirements | 40 |
| 7.9 | Threats to validity | 41 |
| 8 | Conclusions | 45 |
| 8.1 | Future Work | 45 |

Figures

| | | |
|------|---|----|
| 1.1 | Fields of the dataset to be visualised | 2 |
| 2.1 | Matrix of project requirements to issues project is attempting to address . . . | 9 |
| 3.1 | Spiral process model followed | 12 |
| 4.1 | Image of Worlds Visualisation | 15 |
| 4.2 | Image of The Kepler Orrery Visualisation | 16 |
| 4.3 | Image of Celestia Visualisation | 17 |
| 4.4 | Kepler Visualisation Tool Orbital View | 18 |
| 4.5 | Kepler Visualisation Tool Graph View | 19 |
| 4.6 | Matrix of existing solutions mapped to scenarios | 19 |
| 4.7 | Table of technology choices | 21 |
| 5.1 | Sequence Diagram of IKVT render cycle | 24 |
| 5.2 | Class Diagram of IKVT | 25 |
| 5.3 | Mockup of the text area | 26 |
| 5.4 | Mockup selection process | 26 |
| 5.5 | Mockup highlight sister planets process | 26 |
| 5.6 | Mockup compare planets text areas | 27 |
| 5.7 | Mockup of ESI views | 27 |
| 5.8 | Mockup of range sliders | 27 |
| 5.9 | Mockup star habitability zone | 28 |
| 5.10 | Mockup of the interaction panel | 28 |
| 5.11 | Mockup of interactive buttons | 28 |
| 5.12 | Mockup of the Kinect system | 29 |
| 5.13 | Mockup of cursor images for Kinect system | 30 |
| 5.14 | Cursors for Kinect sensor | 30 |
| 6.1 | Original Horizontal Layout | 34 |
| 6.2 | Improved Vertical Layout | 34 |
| 6.3 | Navigation Panel Overview | 35 |
| 6.4 | Panel of interactive buttons | 36 |
| 6.5 | Panel of interactive range sliders | 36 |
| 6.6 | Text boxes in each possible state | 37 |
| 7.1 | Intuitivity of keyboard and mouse | 42 |
| 7.2 | Intuitivity Microsoft Kinect sensor | 42 |
| 7.3 | Perceived value of visualisation | 42 |
| 7.4 | User comprehension of visualisation | 43 |
| 7.5 | GUI layout intuitivity | 43 |
| 7.6 | Knowledge gained from the visualisation | 44 |

| | | |
|-----|---|----|
| 7.7 | Correct amount of information displayed | 44 |
| 7.8 | Summary of results | 44 |

Chapter 1

Introduction

This project seeks to design, implement, and evaluate an interactive 3D visualisation software system for displaying the content in the Kepler Exoplanets dataset ((REF)). This deliverable is intended as a standalone 3D visualisation with two modes of interaction, keyboard and mouse or Microsoft Xbox Kinect sensor ((REF)). The resulting visualisation will convey the information in the dataset in a way that the target users, laypeople who have an interest in astronomy, can understand and interact with.

1.1 Motivation

There are many planets that have been located outside of our own solar system, these are called exoplanets, these are referred to interchangeably as planets and exoplanets for the remainder of this report. Information about these exoplanets are stored in the Kepler Exoplanet dataset ((REF)). This project seeks to develop and evaluate an interactive 3D visualisation software system for the Kepler exoplanets dataset ((REF[28])). We can leverage how humans respond to visualisations by experiencing more enjoyment and more immersion, both of which result in improved learning and recall. This means that creating an effective and engaging visualisation will help convey the information in the dataset effectively to the users.

1.2 Problem statement

The complex nature of the data involved in this project causes a range of problems revolving around understandability to arise, which this project attempts to address. The following subsections outline these in detail.

1.2.1 Understanding the content in the dataset

Understanding and analysing large datasets whose size defies simplistic or trivial analysis by humans is a known issue that many areas of research are attempting to address. These areas of research range from data mining to visualisations in order to discover or highlight important features in the data so that people can more efficiently use or understand it.

Humans often rely on internal visualisation when we solve problems. We create an image in our mind of a situation in order to make sense of it <http://nrich.maths.org/6447>. This allows for a much more comprehensive understanding of the content being visualised. The content in the dataset used for this project is made up of records of every one of the 2234 exoplanets discovered by the Kepler Mission. Each of which contains 46 fields. It is next to

impossible for someone to internally visualise so much information, especially as most of it is floating point numbers. This means that an external way of visualising it is needed, which is one of the problems that this project attempts to address.

| | | | | | | | | | |
|-----------------|---------------|-----------------|----------------------|---------------------|--------------------|-----------------|-----------------|------------------|----------------------|
| KOI | Dur | Depth | SNR | t0 | t0_unc | Period | P_unc | a/R* | a/R*_unc |
| r/R* | r/R*_unc | b | b_unc | Rp | a | Teq | EB prob | V | FOP |
| N | P. Zone Class | P. Mass Class | P. Composition Class | P. Atmosphere Class | P. Habitable Class | P. Gravity (EU) | P. Esc Vel (EU) | P. Period (days) | S. Hab Zone Min (AU) |
| S. Hab Zone Max | P. HZD | P. HZC | P. HZA | P. HZI | P. Int ESI | P. Surf ESI | P. ESI | S. HabCat | P. Habitable |
| P. Hab Moon | P. Confirmed | P. Disc. Method | P. Disc. Year | S. Name | S. Constellation | | | | |

Figure 1.1: Fields of the dataset to be visualised

1.2.2 Comprehension of planetary information

Much of the information regarding planets is cryptic and unintuitive, this make its understandability difficult. Visualisations in general attempt to address this by displaying data in simplistic ways that allows improved user comprehension.

1.2.3 Existing solutions lack functionality

Existing data visualisation techniques using this exoplanet dataset lack the ability to display sufficient detail for each exoplanet and do not fully utilise the data in the dataset. Existing solutions display only the size, temperature, and orbital information about the exoplanets. While this is useful information that informs users of important facts about the planets, it does leave a lot of potential information unseen and overlooked. For example, information about the type of planet, planets with similar traits, solar system information, similarity to earth and habitability. This project will therefore be focused on researching, implementing, and evaluating a new interactive visualisation system that will display additional information to contained in the dataset but not included in existing visualisation systems.

1.2.4 Effective user interaction with visualisation

A visualisation that solely displays information without effective methods of interaction limits the immersive qualities that keeps users engaged. To address this interactive visualisations emerged, generally these visualisations allow users to modify the representation of information rather than the information itself. This means allowing users to control properties of how the data is represented, be it something as simple as the layout of elements or something more complex. Many mediums of interaction are possible from the mundane keyboards, mice, or touchpads to the more esoteric wired gloves, motion sensors, and omnidirectional treadmills or even a combination of devices.

With interactive visualisations, response time of the system to user actions is important and so changes made by the user must be incorporated into the visualization in a timely manner. Experiments have shown that a delay of more than 20 ms between when input is provided and a visualisation is updated is noticeable by most people ((REFERENCE)). Thus

it is important for an interactive visualization to provide a rendering based on human input within this time frame or else risk breaking user immersion.

1.3 Key issues project addresses

To summarize the above sections, this project addresses the following key issues:

- I1. Content in database form is difficult to view and understand.
- I2. Planetary information is complex and difficult to comprehend without a visual reference.
- I3. Existing visualisations for this dataset have minimal functionality and have lacking usability.
- I4. User interaction is needed in a visualisation to make the most of data displayed.

1.4 Contributions of this project

This project provides a visualisation that conveys more information and contains better interactivity than other visualisations in the same area. This extension is evaluated qualitatively by a user experiment to ensure that it is successful in conveying the information contained in the dataset.

The interactivity that is provided by this project is a mix of keyboard and mouse as well as a more novel approach using a Microsoft Kinect sensor. This sensor provides a further contribution of gesture based interactivity of a 3D visualisation.

The work and research completed for this project will provide the opportunity for further improvement of the created visualisation by other developers and researchers. This will create further exposure of the Kepler dataset which will encourage learning about exoplanets and the Kepler mission.

Chapter 2

Requirements Analysis

To guide the creation of the visualisation a user oriented design approach was used, in particular making use of user models (personas). These personas were created to give a sense of empathy and understanding for the foreseen users of the visualisation in order to better understand the requirements and design decisions to be made.

The design of the visualisation was based heavily on User Centered Design as it provided a method of user interface design as well as visualisation design. User Centered Design is a process in which the needs, wants, and limitations of the end users of a system are given extensive attention. To achieve this, personas were created (also known as archetypal users), which are a personification the needs of a larger group of related users. These personas act as stand-ins for real users, describing them in terms of their goals and personal characteristics, and although they are fictitious, they are based on knowledge of real users. This design methodology supported my understanding of how users were likely to use the visualisation.

An additional tool used during requirements analysis was User Scenarios which describe the foreseeable interactions of the user personas with the visualisation. A scenario is made up of a functional goal for the visualisation and describes how it is carried out by a persona. Both of these tools force you to think about the tasks needed for the visualisation and their context in the system as a whole. Once the personas and scenarios have been completed you can then start to design specific elements of the user interface and visualisation based on the requirements and interactions described in the scenarios.

2.1 User models

Below are the two personas that were used in the design of the visualisation for this project. They depict users that would use the visualisation in the context of a terminal or display in an observatory environment. These personas can be validated during evaluation of the visualization by finding real users that match the core values of the personas.

2.1.1 John Truman (Primary Persona - The interested layperson)

24 year old John is interested in planets and space and has a basic knowledge about both. He frequently visits attractions catering to this interest at locations such as planetariums and observatories. Some of his favourite things to do when visiting these attractions is to go to the computer terminals that allow users to choose what information they see.

John is used to playing computer games and using visualisations and is not overwhelmed understanding and using new systems. He finds that he learns better when provided with

visual examples than when reading or listening to information. John is most comfortable using keyboard and mouse when interacting with a computer.

2.1.2 Cara Thompson (Secondary Persona - Likes gesture based systems)

23 year old Cara likes using interactive visualisations when visiting attractions, she finds that they are more entertaining and provide a better level of interaction and more of a novelty experience with a visualisation than simply a keyboard and mouse.

Both of these users are similar in their need for information from the visualisation but differ in the methods that they wish to access the information and interact with the visualisation. John wants to interact with keyboard and mouse as it is more straight forward and accurate. Cara wants to interact with gestures as she finds it more of a novelty and more immersive.

2.2 Scenarios

A good use scenario does a number of things:

- Describes the user's goals and motivations.
- Describes a specific task or tasks that need to be accomplished.
- Describes some of the interaction, with enough detail to make it compelling, but not so much detail as to be overwhelming.
- Provides a shared understanding for everyone on your team about what a user might want to do and how they might do it.
- Helps you construct the sequence of events that are necessary to address in your user interface.
- Can be sketchy, as long as it provokes ideas and discussion.

2.2.1 Scenario 1: View planets ordered by their similarity to Earth

Primary Persona:

When John first sees the system the first thing he notices is that there are many planets orbiting what looks like a star. He doesn't have any point of reference for these planets so their sizes, colours, and movement speeds are meaningless. By providing a way of comparing the planets to Earth it gives a point of reference which is well documented and known by most.

Procedure:

1. John clicks a prominent button stating view planet similarity to earth.
2. The planets on screen move so that the earth is located at the center and top of the screen with all others orbiting it. The planets with a high internal similarity to earth are higher on the Y axis whilst the planets with a high surface similarity to earth are closer to the center of the orbiting planets on the X axis.
3. From here John can select any of the planets for further analysis.

2.2.2 Scenario 2: Select ranges for attributes of each planet displayed

Primary Persona:

John has become comfortable with selecting the planets and has some idea of the scale and basic attributes of the planets. Now he wants to select more planets to find out more information. However due to the large number of planets he finds it difficult to accurately select them due to overlapping and fast moving small planets.

Procedure:

1. John uses a range of filters to remove planets from his view that don't match the criteria he chooses (temp ,size ,KOI , ESI).
2. As planets disappear the graph of planets expands into the space that frees up, this causes more space to appear between planets making them more selectable.

2.2.3 Scenario 3: Select planets to display more information

Primary Persona:

John wants to see more information about each of the planets he can see orbiting in the visualisation. To do this he wants to be able to select the planets and have textual information appear on screen.

Procedure:

1. John has the option to pause the rotation of planets in order to make more accurate selections.
2. John clicks on a planet orbiting a planet.
3. The planet selected becomes larger and its outline grows, making it more visible.
4. The text window has all of the information about the planet selected added to it.

Secondary Persona:

Cara wants to see more information about each of the planets she can see orbiting in the visualisation. To do this she wants to be able to hover her hand over a planet to get the information to display on screen.

Procedure:

1. Cara hovers her hand over a planet to make a selection
2. The planet selected becomes larger and its outline grows, making it more visible.
3. The text window has all of the information about the planet selected added to it.

2.2.4 Scenario 4: View planets in the same solar system

Primary and Secondary Personas:

John and Cara are curious about which of the planets they can see in the visualisation are in the same Solar System. To discover this they want that when a planet is selected all other planets in the same Solar System as the selected planet to become highlighted.

Procedure:

1. When a planet is selected, all planets in the same Solar System become larger and its outline grows, making it more visible.
2. A label appears on these planets indicating that they are related planets.

2.2.5 Scenario 5: View the Goldilocks zones of each planet

Primary Persona:

Looking at the planets orbiting the sun in the visualisation John wonders whether any of them could support life. To see this John wants to see which planets are in the habitable zones of their stars.

Procedure:

1. John clicks a button saying "Show habitable zones"
2. Coloured rings appear showing the cold (blue), habitable (green), and hot (red) zones of the selected planet's star.
3. When a planet from a different star system is clicked the coloured rings will change to that star's zones.

2.2.6 Scenario 6: Select two planets to compare against one another

Primary Persona:

When John is selecting planets to view more information he often finds that he wants to compare his selections against another planet. To do this John wants to be able to make multiple selections to compare two planets against one another.

Procedure:

1. When John selects a planet a button becomes ungreyed called "Compare" with a note next to it saying "Please select another planet to compare to".
2. When the second planet is selected a second text box fills up with the information about the second planet. This information can be compared with that in the first text box.

2.2.7 Scenario 7: Navigate the visualisation with gestures

Secondary Persona:

Cara doesn't find using keyboard and mouse interesting enough for interacting with the visualisation. She would rather navigate around the visualisation by using hand gestures as it's more immersive.

Procedure:

1. By moving her hand to the edges of the screen the visualisation will pan in the corresponding direction, i.e. if the hand goes to the top of the screen the visualisation pans up.
2. By moving her hand backwards and forwards the visualisation will zoom in and out.

2.3 Requirements summary

2.3.1 Functional Requirements

Functional requirements define the functions of a system. These functions are described as a set of inputs, the behavior, and outputs from the system. The functional requirements for this visualisation are as follows:

- R1. The visualisation needs to display planetary information to convey knowledge to users.
- R2. The visualisation needs to allow exoplanets to be compared against one another.
- R3. The planets need to be able to be ordered by their similarity to earth (ESI) and by their Kepler Object of Interest number (KOI).
- R4. R3. The visualisation needs to allow users to define ranges of planetary attributes to filter which planets are displayed.
- R5. Users need to be able to view the habitable zones of stars in relation to the planets orbiting them.

2.3.2 Nonfunctional Requirements

Functional requirements are supported by non functional requirements. Non functional requirements impose constraints on the design or implementation (such as performance, security, or usability) of a system.

The non functional requirements for this visualisation are as follows:

- R6. All interaction methods must be visible and intuitive.
- R7. The visualisation must remain uncluttered to reduce information overload.
- R8. There needs to be two modes of interaction with the system, keyboard and mouse vs gesture based.



















| Key: | Functional Requirements | | | |
|-------|---|---|--|---|
| | Nonfunctional Requirements | | | |
| | Issue 1 | Issue 2 | Issue 3 | Issue 4 |
| Req 1 |  |  |  | |
| Req 2 | | | |  |
| Req 3 |  |  |  |  |
| Req 4 | | |  |  |
| Req 5 |  |  |  |  |
| Req 6 | |  | |  |
| Req 7 | |  | | |
| Req 8 | | | |  |

Figure 2.1: Matrix of project requirements to issues project is attempting to address

Chapter 3

Project Methodologies

3.1 Project management approach

Project management is the discipline of planning, executing, monitoring, and evaluating a project. It is a vital role as it is the glue between all of the different components that go into a successful project. It is beneficial as it encourages thinking about requirements, design, and testing before coding is commenced. For this project it helped to avoid the problem of following a code-and-fix approach described as 1) write some code. 2) Fix the problems in the code [?]. The code-and-fix approach may be suitable in some very small scale applications, but as soon as a system becomes complex it increases the chance of a system turning into a nightmare of high coupling, low cohesion, no modularity, minimal structure, and little consistency. (REF BIG BALL MUD). A widely used method of managing IT projects is to use a Software Development LifeCycle (SDLC) which details the stages that a project must pass through when being created.

3.1.1 Software Development LifeCycle (SDLC)

The project methodology/software development lifecycle chosen for this project was a customized Spiral Model (REF BARRY BOEHM) made up of requirements analysis, design, implementation, and evaluation phases as shown in Figure 3.1.

A software project repeatedly passes through these phases in iterations (called Spirals in this model). For each iteration of the spiral a piece of functionality is completed. The first stage of the spiral involves analysing the requirements of the functionality being created. Second the designs are created, including the element structure and visual elements. Third the functionality is implemented. Finally in the fourth stage the functionality is user tested by one or more people. This evaluation is in addition to the final user evaluation with multiple users.

This SDLC technique supported the creation of a visualisation as it allowed the flexibility to add and remove components into the visualisation as they were discovered to be beneficial or not. It also supported the expansion of the project brief to include using a Microsoft Kinect sensor in order to interact with the visualisation. The choice of this project management approach meant that whilst I had the freedom to explore visualisation options I also had a structured software development life cycle to guide me and the project through the necessary steps to end in the successful completion of each component. Using a spiral model also allowed me to produce a deliverable feature at the end of each iteration of the model which occurred each week to coincide with a meeting with my supervisor, thus ensured that I did not become delayed or stuck in my development with nothing to show. By using this methodology it also allowed me to prioritize the features that were the most

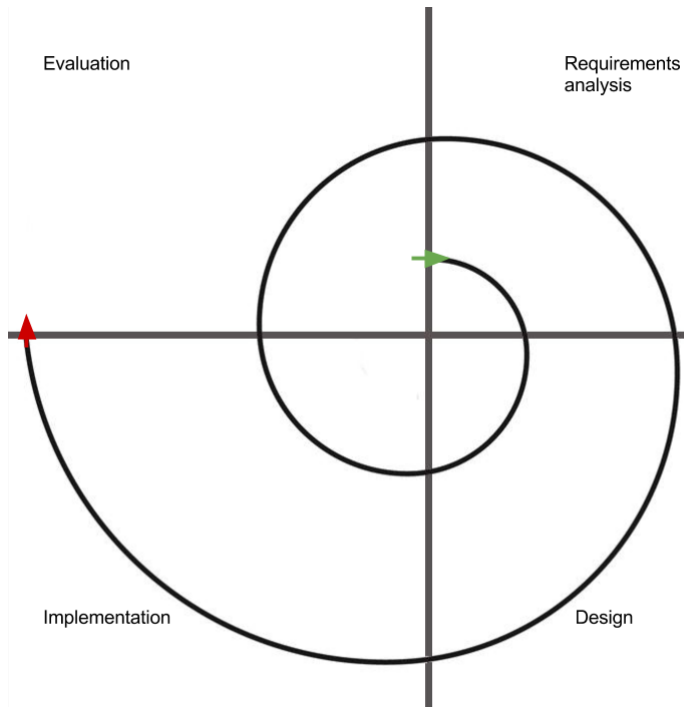


Figure 3.1: Spiral process model followed

important to the visualisation which reduced the risk that there would be missing or incomplete components at the end of the project.

Following this SDLC allowed me to achieve all of the goals that I set, although it did not always happen within the timeframes that I had planned and some deadlines had to be revised. The delays were caused by increasing the scope of the project to include the Microsoft Kinect sensor which required further planning, implementation and testing. Although there were these delays, due to the planning and project management approach that was used, all project elements were completed.

3.2 Other project methodologies explored

Two other project methodologies were explored before the choice to use the Spiral Model was chosen. These alternatives were a loose agile method SCRUM, and a strict Structured method Waterfall. The advantages of the Spiral model over these alternatives was that it provided me with the benefits of a structured work flow that is a feature of the waterfall model as well as a flexible iterative process that is a feature of Agile methodologies without needing to adhere only to the strict guidelines of either. Following a pure waterfall methodology would not have allowed me to iteratively design, develop, and evaluate each feature and would have forced more upfront design which limits flexibility and support for changing requirements as was needed for this project. Following a pure agile approach would not have been optimal either as most Agile methodologies (ie SCRUM [REFERENCE]) are more beneficial to projects with more than a single person working on them. As it was I was agile in my approach to the project as embraced changing requirements, collaborated with my supervisor (the customer) regularly, emphasized working software over large amounts of documentation (REFERENCE AGILE MANIFESTO).

3.3 Supporting Tools for project

By supporting this project methodology with other project management tools such as Gantt charts [APPENDIX] and work breakdown structures(WBS) [APPENDIX], it encouraged efficient documentation of planning and work completed in the project as well as displaying the upcoming stages required to complete the project.

Version control was important for this project as it mitigated against the risks of file system crashes and corruption, it also maintained effective revision history that could be used to backtrack to or view changes that were made earlier in the project. Version control was also valuable as it allowed me to maintain multiple branches (versions) of my project that I could swap between. This was important for the creation of the Keyboard and mouse system and the Microsoft Kinect system as they were on separate branches in a repository.

The version control tool that was used for this project was Git which allowed for the repository to be hosted on the repository hosting web service Github. This version control option was chosen due to the ease of use of as well as my prior experience with Git which has all been positive. There was a minor limitation of this choice as I used a free Github license which meant that the repository would be open to the public.

Weekly meetings with the supervisor of the project, Dr Stuart Marshall, were used to provide guidance and ideas for innovation of the visualisation throughout the project. These meetings ensured that vital components and deliverables were implemented in the required timeframe and also provided a sounding board for ideas for elements to be included in the visualization. Another important aspect of having an involved supervisor was that he provided me the guidance of an experienced academic which was indispensable when navigating the administrative side of organizing delicate matters such as ethics approval for human evaluation of the visualisation.

Chapter 4

Existing systems and technologies

4.1 Existing systems

This section discusses 4 existing visualisations that have a theme depicting space or planets. Following the description, each visualisation is analysed to discover which of the User Scenarios detailed previously exist within the system.

4.1.1 Worlds: The Kepler Planet Candidates - Non Interactive

This animation [?] shows planet candidates found by NASA's Kepler mission. These candidates are animated in orbit around a single star. They are drawn to scale with accurate radii, orbital periods, and orbital distances. They range in size from 1/3 to 84 times the radius of Earth. Colors represent an estimate of temperature with red indicating warmest, and blue indicating coldest candidates. The layout of this animation is very similar to the



Figure 4.1: Image of Worlds Visualisation

Kepler Visualisation Tool that I am extending. This means that it provides insights into how my visualisation can be improved as Worlds is a much more visually appealing system. By researching how it displays its Exoplanets I can further improve my own visualisation.

Scenario 1. View planets ordered by their similarity to Earth:

Worlds has comprehensive functionality for comparing the different Exoplanets to one another, however it does not offer any functionality regarding comparisons to earth

Scenario 2. Select ranges for attributes of each planet displayed:

Worlds does not offer any functionality for any filtering of Exoplanets, this means that a user can only see all planets at once which can be overwhelming and causes many of the exoplanets to be excluded from the user due to overlapping and clustering. The reason that this is done is to convey how many Exoplanets there are and how their scale differs among each Exoplanet

Scenario 3. Select planets to display more information:

Worlds is non interactive which means that users are not able to request further information about the visualisation elements that they are seeing. This ability to find out more is a key part of the interactive visualization that is needed for this project.

Scenario 4. View planets in the same solar system:

Worlds shows all Exoplanets as if they are orbiting a single star. This allows for each of them to be compared against one another easily. However this does remove the ability for a user to see which planets are together in the same solar system.

4.1.2 The Kepler Orrery and The Kepler Orrery 2 - Non interactive

The Kepler Orrery [?] illustrates the exoplanet candidates in their own solar systems. The orbit radii are to scale with respect to each other and planet sizes are to scale with respect to each other, but orbits and planet sizes are different scales. The colors are in order of semi-major axis: two-planet systems (242 in all) have a yellow outer planet; 3-planet (85) green, 4-planet (25) light blue, 5-planet (8) dark blue, 6-planet (1, Kepler-11) purple. This system

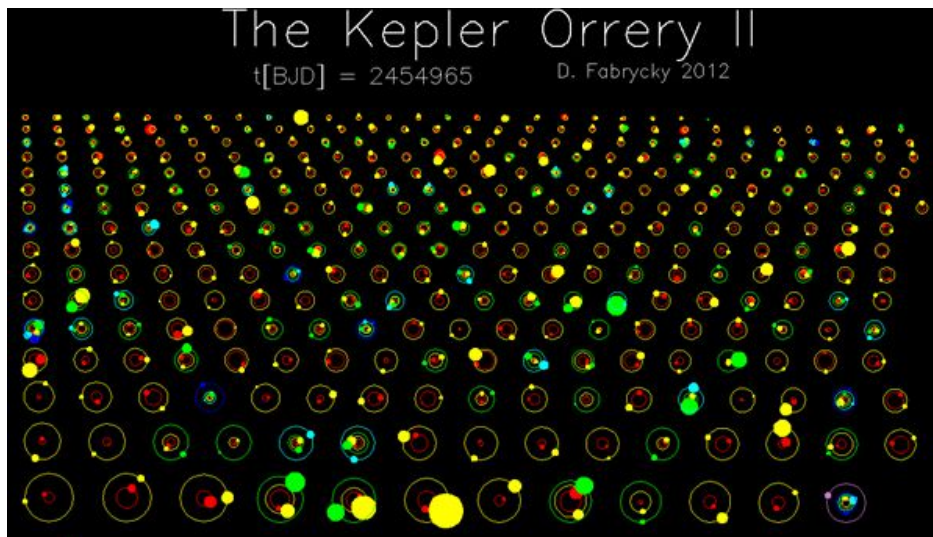


Figure 4.2: Image of The Kepler Orrery Visualisation

exhibits small multiples, a grid of small similar graphics or charts, allowing them to be easily compared. This provides insights into how I can use small multiples to display information about groups of planets. This will be important for displaying which planets share a solar system.

Scenario 4. View planets in the same solar system:

Maybe

Scenario 5. View the Goldilocks zones of each planet:

Maybe

Scenario 6. Select two planets to compare against one another:

Maybe

Scenario 7. Navigate the visualisation with gestures:

Maybe

4.1.4 Kepler Visualisation Tool

An existing system built with Processing is the Kepler Visualisation Tool[?, ?]. It is a simple visualisation focusing on displaying the candidate Exoplanets temperatures and their locations in relation to their distance from their nearest star, so that a sense of scale can be perceived. Each candidates estimated size, orbital speed, and orbital separation is accurately depicted, and each planet is color-coded according to its estimated effective temperature.

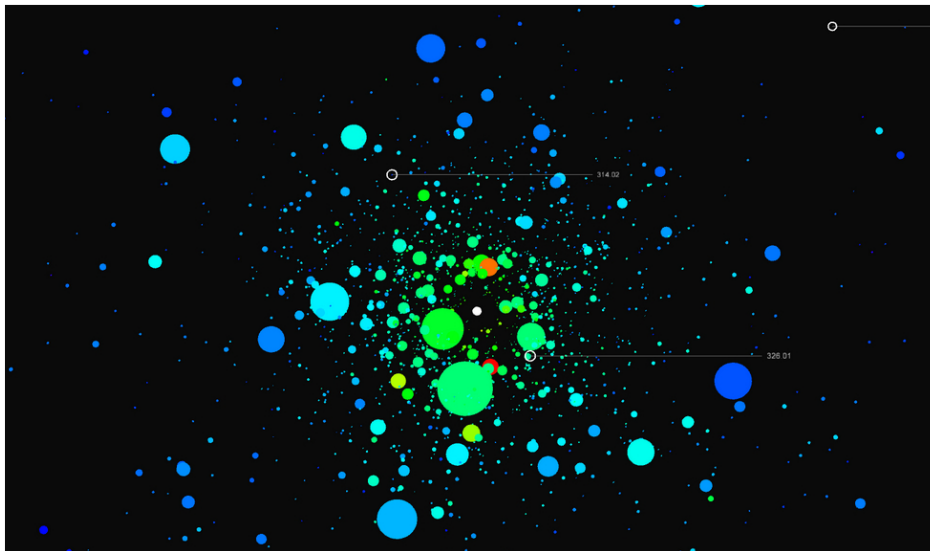


Figure 4.4: Kepler Visualisation Tool Orbital View

The existing work in this system would serve as foundation for this project. Because much of the visual aspects, and initial data manipulation of the existing system are already complete. It means that implementing the features needed for this projects completion could be focused on more heavily and larger improvements to the existing system can be undertaken, such as better labeling and information displays and user interaction methods.

Scenario 1. View planets ordered by their similarity to Earth:

Like Worlds and the Kepler Orrery, The Kepler Visualisation Tool has functionality to display the similarity of each Exoplanet to each other. However, it also has some limitted

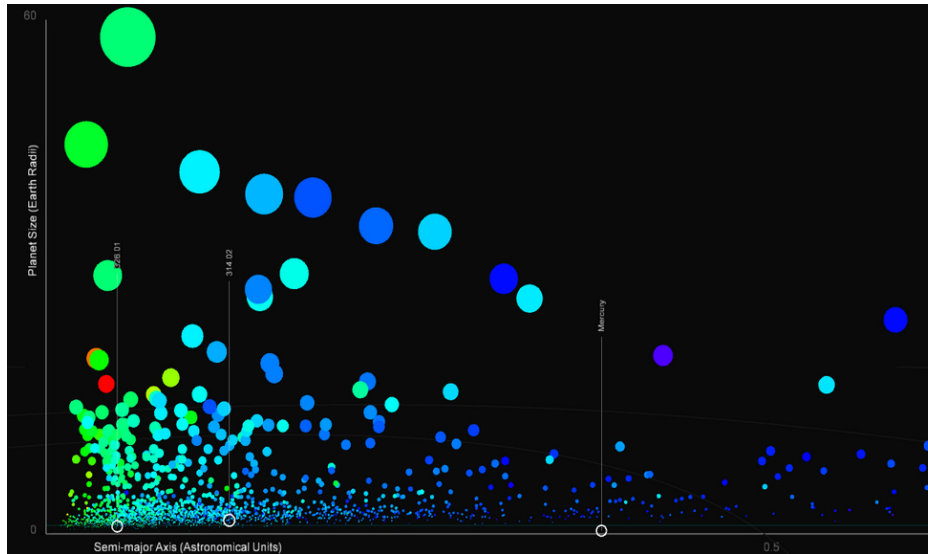


Figure 4.5: Kepler Visualisation Tool Graph View

functionality of comparing these to earth which the others lack. It does this by displaying Earthm, Mars, and Jupiter with the same colours, size, and orbit speed as the Exoplanets. This gives a user a chance to comprehend the scale and difference of some of the Exoplanets.

Scenario 2. Select ranges for attributes of each planet displayed:

The Kepler Visualisation Tool allows users to change which attributes the planets are ordered by on the vertical plane (Y axis) ie, by size or temperature, but does not allow users to select ranges of these attributes to filter the Exoplanets

4.1.5 Summary of Existing Applications

| Existing System | Scenario 1 | Scenario 2 | Scenario 3 | Scenario 4 | Scenario 5 | Scenario 6 | Scenario 7 |
|---------------------------|------------|------------|------------|------------|------------|------------|------------|
| Worlds | No | Partial | No | No | No | No | No |
| The Kepler Orrery | No | Partial | No | Yes | No | No | No |
| Celestia | No | No | Yes | Yes | No | No | No |
| Kepler Visualisation Tool | No | Partial | No | No | No | No | No |

Figure 4.6: Matrix of existing solutions mapped to scenarios

4.2 Analysis of technology options

Many technologies were looked into, experimented with, and the positives and negatives of each weighed up before a decision was made about which would be the choice for the

visualisation. It came down to 3 potential technologies that would be suitable for the project, the next 3 subsections outline these in detail.

4.2.1 D3 (Data Driven Documents)

D3 is a JavaScript library that allows the displaying of data in dynamic graphics. Embedded within an HTML web page, the JavaScript D3.js library uses pre-built JavaScript functions to select elements, create Scalable Vector Graphic (SVG)[17] objects, style them, and add transitions, dynamic effects and tooltips. Large datasets can be easily bound to SVG objects using simple D3 functions to generate rich charts and diagrams. D3 was created because of the need for a balance of expressiveness, efficiency, and accessibility that previous visualization toolkits did not allow [4].

D3 allows the binding of input data to arbitrary input elements. This means that the exoplanet dataset can easily be bound to SVG elements for creating visualizations. D3 adopts the W3C Selectors API to identify document elements queried. This results in a rich but concise selection method of elements in a visualisation.

D3 allows debugging thanks to Google chrome and other modern browsers development tools. A downside to D3 is that it does not allow 3D diagrams, although it does allow pseudo 3D by using the painters algorithm and 3D textures.

4.2.2 Prefuse

Prefuse is a set of software tools for creating rich interactive data visualizations [13]. The Prefuse toolkit provides a visualization framework for Java. It supports a set of features for visualizing and interacting with data. It provides optimized data structures for tables, graphs, and trees. It can be used to build standalone applications, visual components embedded in larger applications, and web applets. Prefuse greatly simplifies the process of representing and efficiently handling data, mapping data to visual representations (e.g., through spatial position, size, shape, color, etc), and interacting with the data. To use Prefuse a basic familiarity with the Java is required, including setting up and building Java projects. A knowledge of Swing or another similar user interface toolkit is also useful for understanding some of the concepts behind Prefuse and for integrating Prefuse visualizations into larger applications. Experience with database systems is also helpful. However the complexity of Prefuse means that the learning curve will be out of scope for this project.

4.2.3 Processing

Processing is an open source programming language and development environment that was initially created to serve as a software sketchbook and to teach the fundamentals of computer programming with a visual context. Using processing would mean that the visualization could be built with Java while still using a successful visualisation framework. The most complete existing visualization using the same exoplanet dataset (Kepler Visualization Tool) is built using Processing. Using this solution would involve learning the Processing language, however Processing is a library built in Java so the syntax is the same. This means the learning curve should be shallow. Using processing means that 3D elements could be included, this wouldnt be possible with D3.

4.2.4 Decision of technology

The final decision of technology was to use Processing, this is because it had many positive aspects that the others did not and minimal negatives as the below table illustrates.

| | D3 (Appendix A.1.2) | Processing (Appendix A.1.1) | Prefuse (Appendix A.1.3) |
|---|------------------------|--------------------------------|-----------------------------|
| Potential for 3D | No | Yes | No |
| Has low learning curve | Yes | Yes | No |
| Prior evidence of successful visualisations | Yes | Yes | Yes |
| Interactive | Yes | Yes | Yes |
| Dynamic transitions | Yes | Yes | Yes |
| Has existing solution related to planets | No | Yes | No |

Figure 4.7: Table of technology choices

As this project was created using Processing, it allowed me to extend the previous visualisation using the same data set, The Kepler Visualisation Tool [?, ?]. As the time was short for this project building upon a previous solution increased the amount of progress that could be made in the time afforded.

Taking this approach meant that the languages being used would be Java using processing libraries.

As this is such a large project involving many different iterations, version control was important for maintaining records and backups of important changes which was stored on remote servers to ensure against file loss in system failures.

Chapter 5

Solution Design: Improved Kepler Visualisation Tool (IKVT)

This section discusses the design of the deliverable visualisation, The Improved Kepler Visualisation Tool (IKVT). It details the key design decisions revolving around structure, aesthetics, and functionality that were made about the visualisation. This project aims to improve an existing visualisation, The Kepler Visualisation Tool which was discussed in the previous chapter. Whilst this existing visualisation displays exoplanets and some of their features, it lacks interactivity for users trying to use it to gain information contained in the Kepler Exoplanet Database effectively. The IKVT expands on this pre-existing visualisation by adding key elements of interactivity missing in the existing visualisation as well as further enhancing the range and amount of data that is available to users about exoplanets. The IKVT also incorporates a novel gesture based interactive mechanism to the visualisation.

5.1 System design and structure

Because this project builds upon an existing system complete comprehension of how it is designed and how it functions is important. Going ahead in the creation of the visualisation without this knowledge would create opportunities for mistakes and incorrect assumptions about how the visualisation needs to be created. To assist with this issue the following two tools were used.

1. UML Sequence Diagram
image
description of
what it shows about visualisation
2. UML Class Diagram
image
description of
what it shows about visualisation

5.2 Visualisation Design

The requirements produced in the Requirements Analysis chapter provide a description of the functionality that needs to be designed for this visualisation. By adding additional details to these requirements we can design how the visualisation should look, behave, and function.

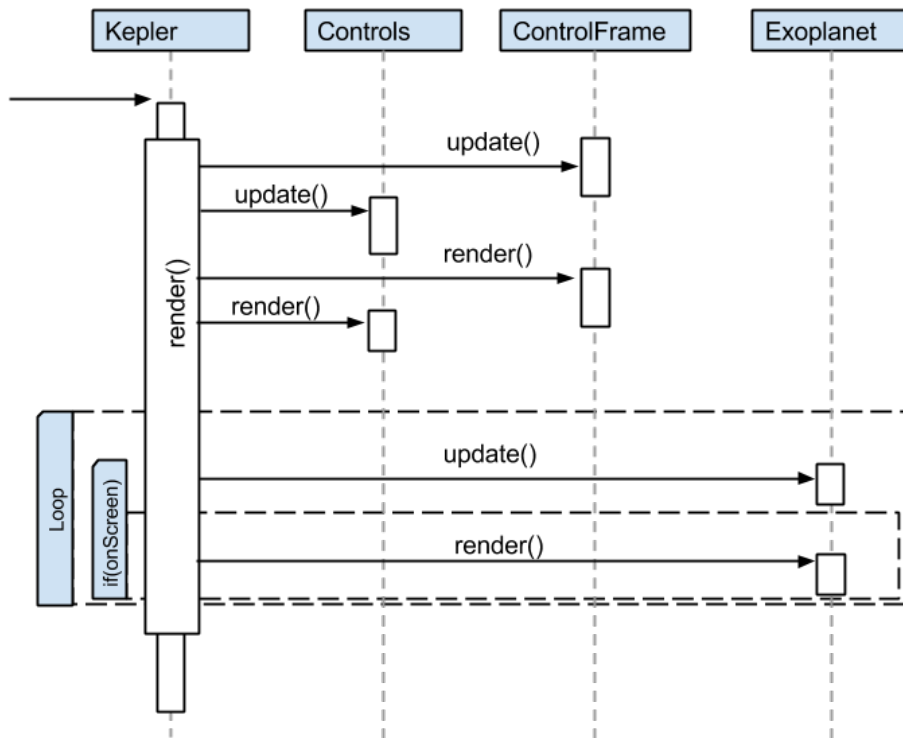


Figure 5.1: Sequence Diagram of IKVT render cycle

By using abstract user interface design the layout and configuration of each element can be planned and coordinated without the need for excessive details which are likely to change throughout the course of the project (e.g. colours and content) ((REF))[18] .

5.2.1 Functional Requirement

R1. The visualisation needs to display planetary information to convey knowledge to users.

This requirement needs some form of textual display in order to convey enough of the information about exoplanets to the user. The most obvious choice for this would be to use a Java TextArea object to display each of the key attributes of each Exoplanet. The following figure is a mockup of the text area showing the information about each planet that will be displayed and the method calls that will be used.

SELECTIONG All planets need to be selectable and react appropriately when clicked.

To fulfill this requirement every planet needs to be able to be selected. This involves detecting when a user clicks and then finding whether any planets are located in that space. This is more complex in this system as it requires detecting where each planet is in a 3D space and checking whether the 2D location of the mouse click coincide.

When a planet is successfully selected it needs to provide feedback and information to the user to inform them that it has been selected and also to provide relevant information.

When a planet is selected all other planets in the same solar system need to become more visible.

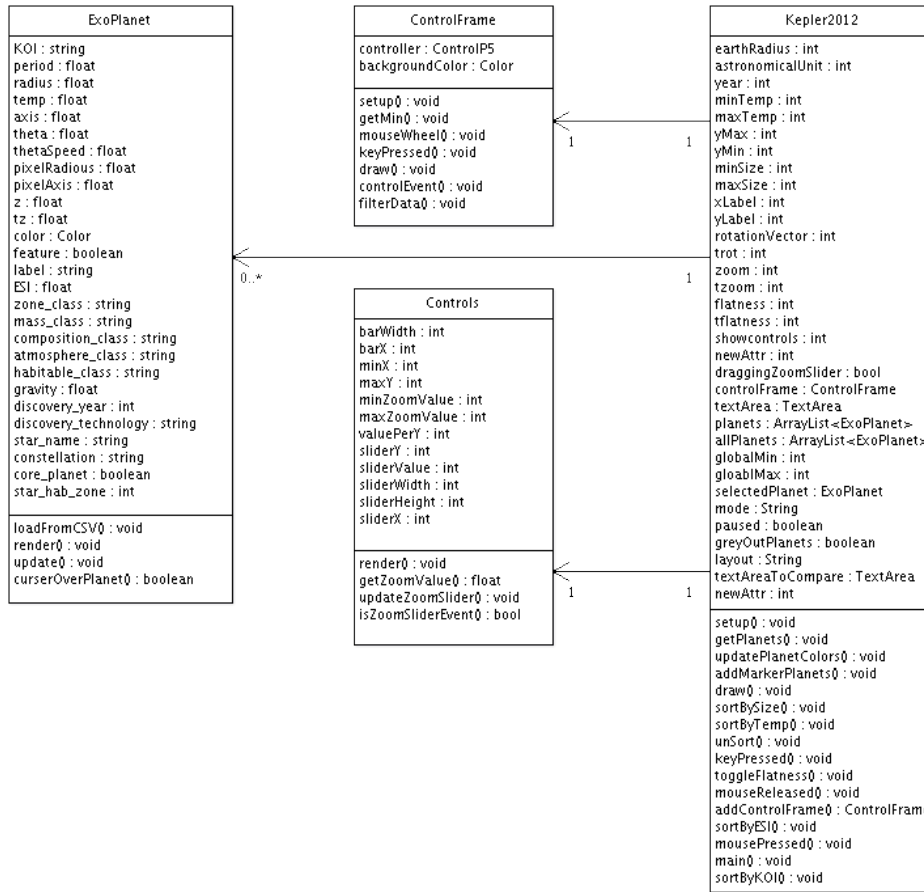


Figure 5.2: Class Diagram of IKVT

When a planet has been successfully selected all of the other planets in the same solar system (sister planets) need to become highlighted. This can be done by treating them as if they were selected and providing an additional indication that they are not actually selected.

R2. The visualisation needs to allow exoplanets to be compared against one another.

Using each of the previously discussed elements of interacting with the visualisation this requirement is fulfilled. The text areas coupled with the main visualisation window mean that all of the important and valuable information about each exoplanet can be conveyed to a user.

R3. The planets need to be able to be ordered by their similarity to earth (ESI) and by their Kepler Object of Interest number (KOI).

To fulfill this requirement the visualisation needs to allow users to view the exoplanets in a way that uses the Earth as a point of reference and their Earth Similarity Index (ESI) to order them and control their position. The following figures display an abstract mockup of how this would be done.

R4. The visualisation needs to allow users to define ranges of planetary attributes to filter which planets are displayed.

To fulfill this requirement a method of filtering exoplanets by their attributes in order to control the number and types of planets displayed to the user. A common method

```

KOI: planet.getKOI()
Temperature: planet.getTemp()
Gravity: planet.getGravity()
Radius: planet.getSize()
Zone Class: planet.getZoneClass()
Mass Class: planet.getMassClass()
Composition: planet.getComposition()
Habitability: planet.getHabitability()
Atmosphere: planet.getAtmosphere()
Method of discovery: planet.getDiscoveryMethod()
Year of discovery: planet.getDiscoveryYear()
Earth Similarity: planet.getESI()

```

Figure 5.3: Mockup of the text area

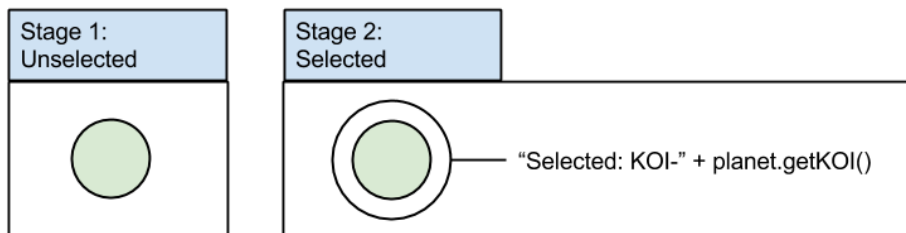


Figure 5.4: Mockup selection process

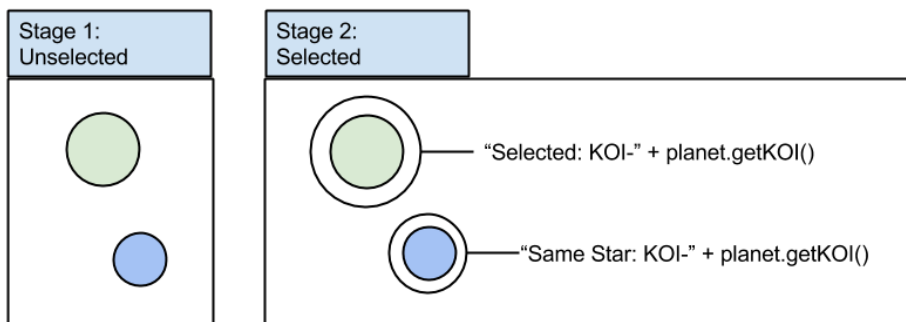


Figure 5.5: Mockup highlight sister planets process

of achieving this is to use a set of sliders that allow a user to filter something by a set of values. For example in this system a slider could be used to control the size of planets that are displayed.

R5. Users need to be able to view the habitable zones of stars in relation to the planets

| Selected Planet | Compared Planet |
|--|--|
| KOI: planet.getKOI() | KOI: compared.getKOI() |
| Temperature: planet.getTemp() | Temperature: compared.getTemp() |
| Gravity: planet.getGravity() | Gravity: compared.getGravity() |
| Radius: planet.getSize() | Radius: compared.getSize() |
| Zone Class: planet.getZoneClass() | Zone Class: compared.getZoneClass() |
| Mass Class: planet.getMassClass() | Mass Class: compared.getMassClass() |
| Composition: planet.getComposition() | Composition: compared.getComposition() |
| Habitability: planet.getHabitability() | Habitability: compared.getHabitability() |
| Atmosphere: planet.getAtmosphere() | Atmosphere: compared.getAtmosphere() |
| Method of discovery: planet.getDiscoveryMethod() | Method of discovery: compared.getDiscoveryMethod() |
| Year of discovery: planet.getDiscoveryYear() | Year of discovery: compared.getDiscoveryYear() |
| Earth Similarity: planet.getESI() | Earth Similarity: compared.getESI() |

Figure 5.6: Mockup compare planets text areas

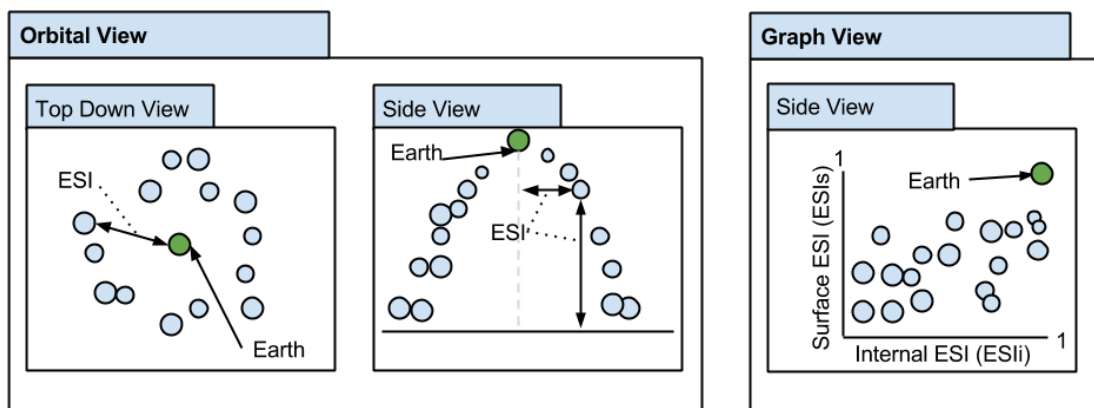


Figure 5.7: Mockup of ESI views


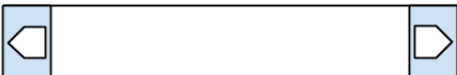

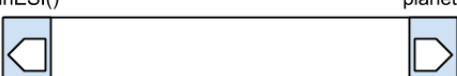
| Range Slider Mockup | |
|---|----------------------------------|
| planet.getMinKOI() | planet.getMaxKOI() |
|  | Kepler Object of Interest Slider |
| planet.getMinTemp() | planet.getMaxTemp() |
|  | Temperature Slider |
| planet.getMinSize() | planet.getMaxSize() |
|  | Size Slider |
| planet.getMinESI() | planet.getMaxESI() |
|  | Earth Similarity Index Slider |

Figure 5.8: Mockup of range sliders

orbiting them.

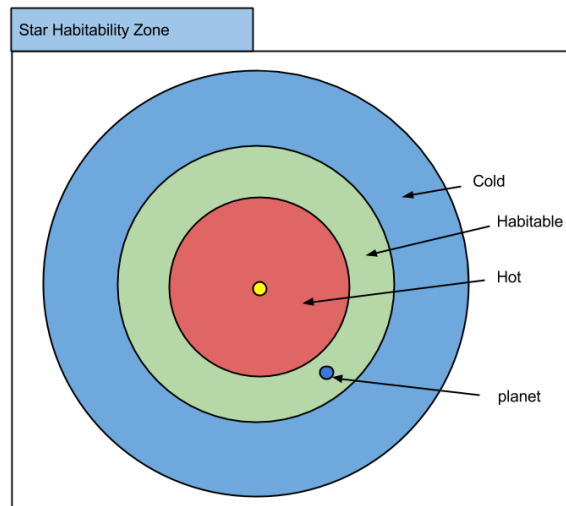


Figure 5.9: Mockup star habitability zone

5.2.2 Nonfunctional Requirement

R6. All interaction methods must be visible and intuitive.

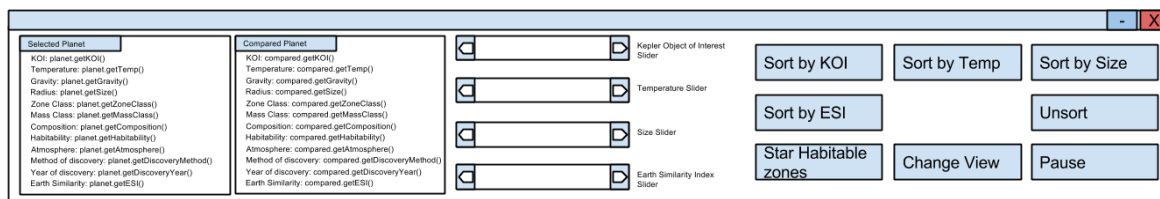


Figure 5.10: Mockup of the interaction panel

The visualisation needs to have a range of interactive buttons for each element of interactivity in the system to help inform users how to use the system.

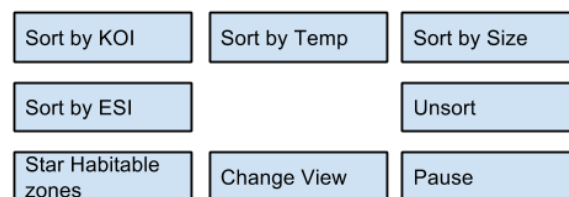


Figure 5.11: Mockup of interactive buttons

R7. The visualisation must remain uncluttered.

By separating each of the interactive components into a separate window inside the visualisation it provides

The visualisation must not show so much information that it causes information overload for users.

The ability to filter and sort the exoplanets gives a user the tools needed to reduce the quantity of planets displayed in the visualisation and thus the information load

imposed on users. In addition to this, by having the text area that displays the information about selected planets separate from the main visualisation it reduces the cognitive load on users as they don't have to use this component until they want to.

R8. There needs to be two modes of interaction with the system, keyboard and mouse vs gesture based.

The above requirements mostly relate to the keyboard and mouse system although some of the requirements do carry through to the Microsoft Kinect system as well.

For the Kinect system the key difference is that the user will be able to control the main visualisation window with gestures. This means incorporating a means of detecting the gestures.

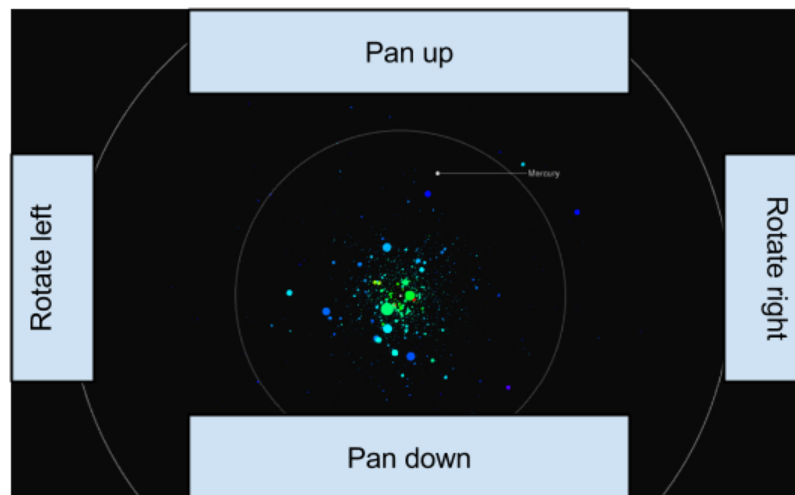


Figure 5.12: Mockup of the Kinect system

As the Kinect sensor no longer requires the use of a mouse the visualisation design needs to be modified to accommodate the use of gestures. This meant incorporating new cursors to indicate the state of the visualisation. There are 7 states that the cursor needs to be able to be in to inform the user of what action they are performing. These states are

- (a) default cursor, hand is at rest
- (b) panning up, hand is raised
- (c) panning down, hand is lowered
- (d) rotating left, hand is to the left
- (e) rotating right, hand is to the right
- (f) zooming in, hand is pressed forward
- (g) zooming out, hand is pulled backwards

Having a range of icons that clearly display these states is vital for keeping the user informed of what they are doing. The icons designed for this purpose are in the following figure

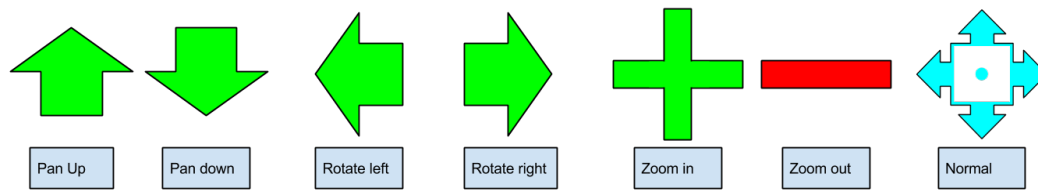


Figure 5.13: Mockup of curser images for Kinect system

Figure 5.14: Cursors for Kinect sensor

Chapter 6

Visualisation implementation of the Improved Kepler Visualisation Tool (IKVT)

This chapter discusses the implementation of the visualisation using the designs discussed in the previous chapter to fulfil the requirements for this project. It details the tools used, the deliverable features produced, and the problems encountered. IKVT displays all 2234 exoplanets in the Kepler exoplanets dataset ((REF[28])) dataset. Each of these exoplanets are represented as coloured ellipses, of which the colour and size are representative of the exoplanets temperature and size respectively. IKVT displays all of these exoplanets as if they are orbiting a single star which in reality would result in planetary collisions but in the visualisation provides users with a way to effectively make observations and comparisons about each of the exoplanets in a single view.

To make this selection, a user can click on any of the orbiting exoplanets. A further effect of this selection is that a text box will have further textual information about the selected exoplanet appended to it to provide the user with more detailed information. When a user is unable to accurately select an exoplanet due to clustering or overlapping of exoplanets they can move the camera around in space to gain a better viewing position with which to make their selection. If this is not enough, the user can use a set of range filters to filter the exoplanets displayed. These filters are Kepler Object of Interest number (KOI), temperature, size, and Earth Similarity Index (ESI). These filters allow for users to fine tune the exoplanets they wish to see which allows them to work with small multiples rather than the entire dataset.

In addition to the orbital view already discussed, there is a graph view that displays the exoplanets on a graph with the exoplanet attributes mapping to the x,y coordinates. This allows the user to modify what they want on each axis of the graph. Having these two views allows the user the option of how they wish to visualise the exoplanets, the less exact and more visually appealing orbital view, or the more exact and less exiting graph view.

There are two panels that make up the visualisation, the visualisation panel, and the control panel. The visualisation panel is where all of the exoplanets are displayed as well as text boxes describing the state of the visualisation to keep the user informed. The control panel contains all of the interactive components that the user can use to change the state of the visualisation. The components it contains are; two text areas that can be used interchangeably to display information about selected planets, four range sliders that are used to filter the exoplanets as discussed previously, and eight buttons to toggle the state of the visualisations. These buttons are "Sort by KOI", "Sort by Temp", "Sort by Size", "Sort by ESI", "Change

View", "Suns Habitable Zone", "Pause", and "Unsort".

The Kepler Orrery visualisation as discussed in the Existing Systems section of Chapter 3 details a contrasting system that displays each exoplanet and its sister planets orbiting its own star. By incorporating this idea into IKVT, when a planet is selected it should be highlighted and all of its sister planets should also become highlighted. This will provide the same effect as in the Kepler Orrery.

6.1 Additional Tools and artifacts used

6.1.1 Dataset used

The dataset for this project is a comma separated values file (CSV) that contains all of the data pertaining to the Exoplanets. At runtime this dataset is read into the system and an Exoplanet object is created for each element and contains all of the information from the dataset record.

6.1.2 Integrated Development Environment (IDE)

The IDE used for this project is one that is provided with Processing. This IDE provided all of the tools and functionality that were required to effectively implement the solution for this project. Another option that could have been taken was to use Eclipse with the processing package and external libraries imported. However I found that the processing IDE was entirely suitable for the majority of my needs creating the visualisation. The key things that I did not have access to with this choice was the Eclipse Debugger and JUnit tests.

6.1.3 Keyboard and Mouse System

In addition to Processing in the main system there was an additional opensource library required for effective user interface components, this library was called ControlP5 REF . This library is a customisable and intuitive interactive user interface. It allows for easy creation of visually appealing and precisely layed out interactive GUI components.

6.1.4 Microft Kinect sensor system

For the version of the IKVT system that uses the Kinect sensor for user interaction two additional libraries were required to integrate the hardware with Processing, these were:

1. NITE <http://www.primesense.com/news/our-blog/the-dld-experience/attachment/photo4/>
2. SimpleOpenNi <http://code.google.com/p/simple-openni/>

These libraries provided drivers to run the Kinect sensor in Processing as well as basic gesture recognition and body tracking. However as the libraries were opensource due to the official Microsoft Kinect SDK not being compatible with Processing, the gesture recognition was not as user friendly or effective as the official libraries. The effect of this was that the gesture tracking used in the system had to be created supoptimally from the opensource libraries.

6.2 Main interface components

The visualisation was designed to emphasis small multiples and filtering of the Exoplanets to display the information more clearly to users.

However I will need to ensure the effectiveness of the visualisation does not become diminished by trying to convey too much information which would lead to cluttering and overlapping in the visualisation, as well as information overload for users. There will also be larger emphasis placed on making the existing system more usable by improving the interaction methods for users. The following list outlines the new requirements for the visualisation being developed. This will be done by providing GUI elements for each form of interaction with the system, as well as ensuring all interaction methods are intuitive for users.

Spacial arrangement of components

As the majority of the interaction and movement of visualisation elements occurs in the center of the window it caused a aspect ratio that was not suitable . It was BETTER to use 2 vertical columns to view and control the visualisation as it had a higher aspect ratio which allowed more of the content to be seen on the screen at once thanks to the fact that the majority of computer screens have a wide ratio.

Navigation Window(BETTER TITLE)

Description of navigation window Description of each button Description of each slider Description of text boxes Due to the need for increased user interaction with the visualisation a window is required to house the buttons, range selectors, and text areas. These elements are needed as the different methods that users can use to interact with the visualisation need to be visually apparent to ensure that the system can be easily used without prior experience. A way to do this is to provide clearly labelled interactive elements and tooltips explaining what they do.((REFERENCE)) . These tooltips are widely used as a method of informing a user about the purpose of an item by hovering over it. This removes the need to click on a button to discover its effect.

In addition to this, the screen needs to display the user in relation to the screen, an effective way to do this is to display a washed out representation of themselves in the background of the visualisation.

6.3 Problems encountered

Due to the number of elements that needed to be displayed on screen at any one time (ie 2234 exoplanets), the load placed on a system is very high due to the need to render 2234 eclipses to represent the planets. This uncovered a bug in the processing library in which the memory use of the visualisation would periodically increase until it crashed due to an out of memory exception. After much experimentation of how to overcome this issue, I discovered that rather than trying to render a native ellipse shape in processing, if I instead rendered a Scalable Vector Graphic this bug would not manifest.

Libraries used for gesture detection in kinect are opensource in order to work with processing did not have decent detection

Using the Processing framework meant using a non industrial??? IDE that had many bugs,

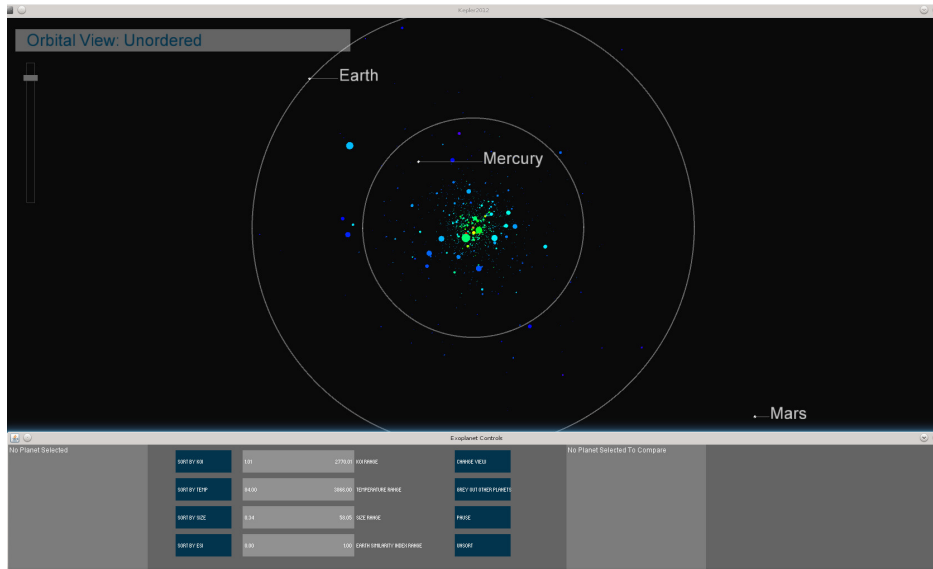


Figure 6.1: Original Horizontal Layout

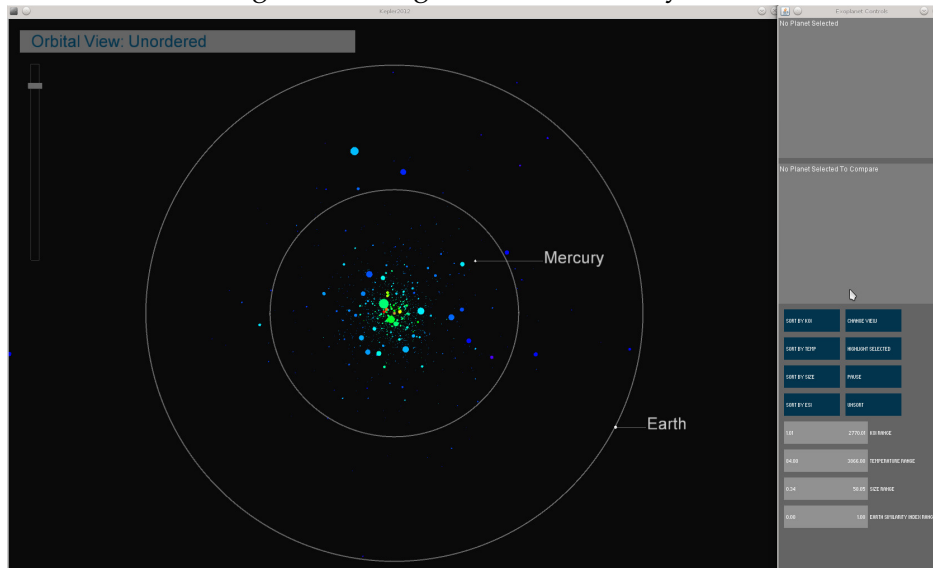


Figure 6.2: Improved Vertical Layout

for example when undoing multiple times in a row the file being modified would periodically become corrupted by lines of code being taken away or inserted into the wrong locations. The solution to this issue was to ensure that I regularly committed any changes to my version controlled system on Github ((REFERBTECE)). Doing this meant that if at any time a file became corrupted I could easily see the changes in the file when compared against the previous commit and manually fix the file.

Performance limitations SCREENSHOT

As this project builds upon a previous system much of the existing code and execution flow needs to be modified. This requires understanding of how the system was originally built and designed. Because this system does not have any unit or integration tests, going ahead without a comprehensive knowledge of the core functionality would have led to ineffective planning and errors being introduced into the system.



Figure 6.3: Navigation Panel Overview



Figure 6.4: Panel of interactive buttons

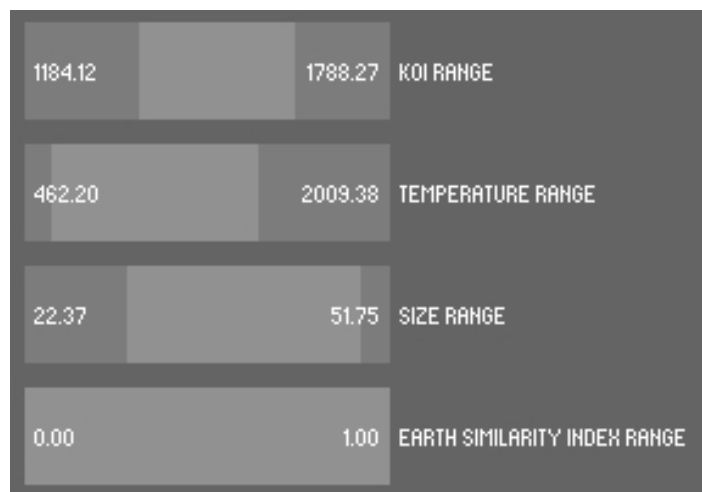


Figure 6.5: Panel of interactive range sliders

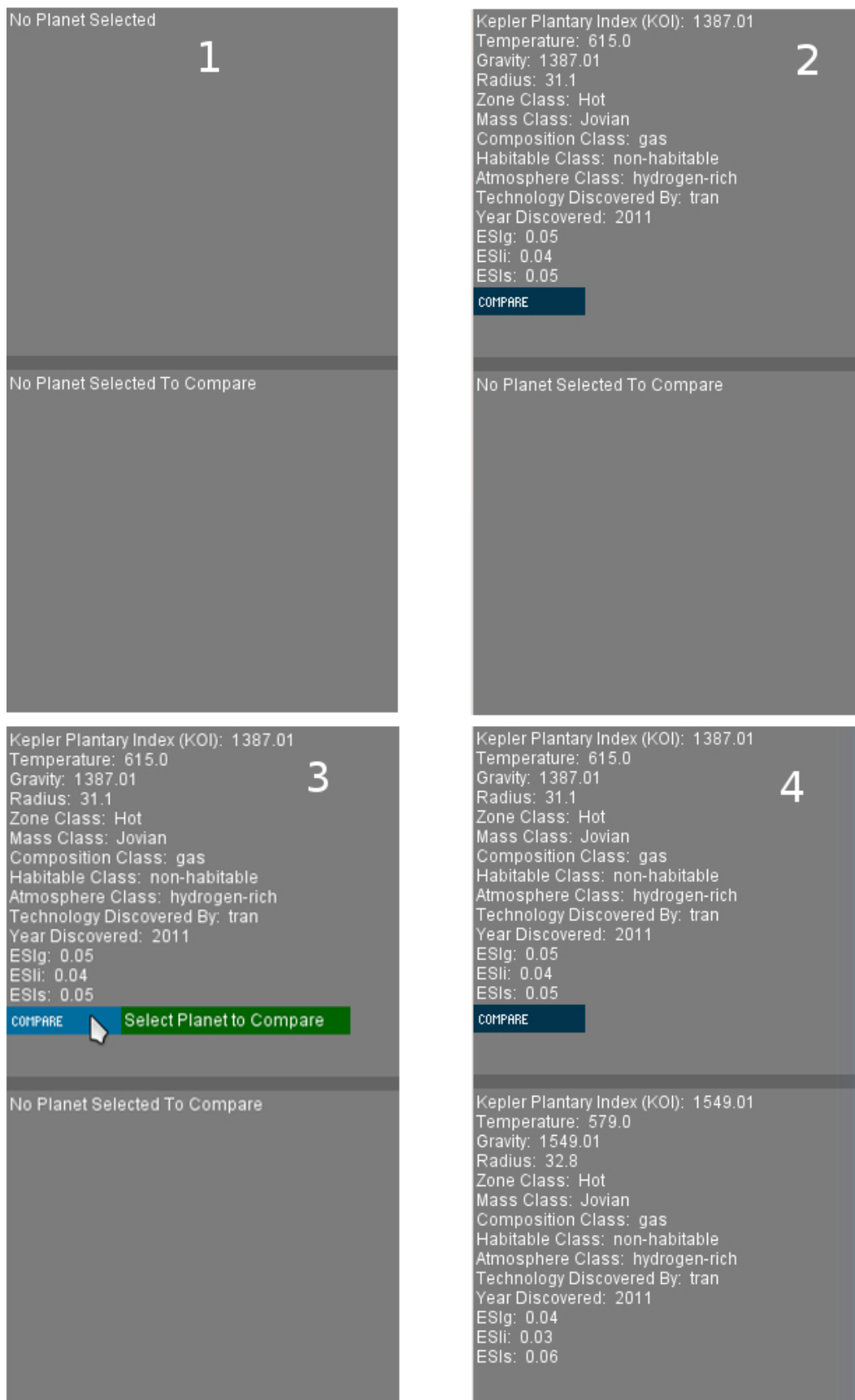


Figure 6.6: Text boxes in each possible state

Chapter 7

Visualisation Evaluation

Following the completion of the implementation stage of this project a final user evaluation was carried out on the visualisation. This evaluation was primarily designed to discover whether or not the visualisation created was successful in fulfilling the requirements of the project as well as conveying the information stored within the Kepler Exoplanet Database.

7.1 Why was evaluation conducted

7.2 Expectations of evaluation

7.3 Evaluation Environment

7.4 User Study Method

- The user enters the room and sits down at the computer.
- They are handed the consent form and information sheet.
- After these are completed they are handed the user questionnaire and the set of questions to answer while using the system. On this questionnaire there are two sets of questions, the first is for the keyboard and mouse system, and the second if for the Microsoft Kinect system.
- Following this they are advised that they have 5 minutes to get familiarised with the system but they do not need to use all of this time (the amount of time taken will be recorded for analysis of how user friendly and intuitive the system is).
- Following this the user is asked to complete the question sheet by first using the mouse and keyboard system. When they feel they have answered all of the questions they will notify the examiner who will move the user to the Kinect system to continue the questions.
- Once the user has completed both sets of questions they are asked to fill in the qualitative user questionnaire about their experiences using the visualisation.
- Following this if the examiner has no follow up questions the user is free to leave.

7.4.1 Participants

The number and demographics of participants in a user study is a key issue that can affect the results.

7.5 Pilot Study

One participant was asked to take part in a pilot study before any results were collected. This participant was asked to complete all of the activities that make up the main experiment. This pilot study took approximately 15 minutes as intended, this included the time needed for the explanation and completion of paperwork, as well as the experiment itself.

The reason for conducting this pilot study was to ensure that the experiment was producing the data required to evaluate the visualisation produced as well as taking the correct amount of time to complete. In addition to this it was used to discover whether there were any aspects of the study that would interfere with the results.

A result of performing this pilot study was that it revealed that the wording of some of the tasks users were asked to complete were ambiguous and caused unnecessary confusion for users during the experiment which could have interfered with the results in a negative way. These ambiguous questions and tasks were removed prior to the main user study. During the main study no users asked for clarification on any of the questions or tasks.

7.6 Experiment on Keyboard and mouse visualisation

Image of using kinect system

7.7 Experiment on Microsoft Kinect visualisation

7.8 Results

The user study was undertaken by 9 participants, all were either students or young professionals from a mix of specialities aged between 21 to 26 with a mix of genders with 5 females and 4 males. Each study took between 10 to 15 minutes.

7.8.1 Analysis of requirements

- R1. The visualisation needs to display planetary information to convey knowledge to users.
- R2. The planets need to be able to be ordered by their similarity to earth (ESI) and by their Kepler Object of Interest number (KOI).
- R3. The visualisation needs to allow users to define ranges of planetary attributes to filter which planets are displayed.
- R4. All planets need to be selectable and react appropriately when clicked.
- R5. When a planet is selected all other planets in the same solar system need to become more visible.
- R6. There needs to be the option to view the habitable zones of stars and show where the planets orbiting them are in relation.

R7. The visualisation needs to allow exoplanets to be compared against one another.

The non functional requirements for this visualisation are as follows:

R8. The visualisation needs to have a range of interactive buttons for each element of interactivity in the system to help inform users how to use the system.

R9. All interaction methods must be visible and intuitive.

R10. The visualisation needs to display attributes of each of the Exoplanets.

R11. The visualisation must remain uncluttered.

R12. The visualisation must not show so much information that it causes information overload for users.

R13. There needs to be two modes of interaction with the system, keyboard and mouse vs gesture based.

Image of using non kinect system

7.9 Threats to validity

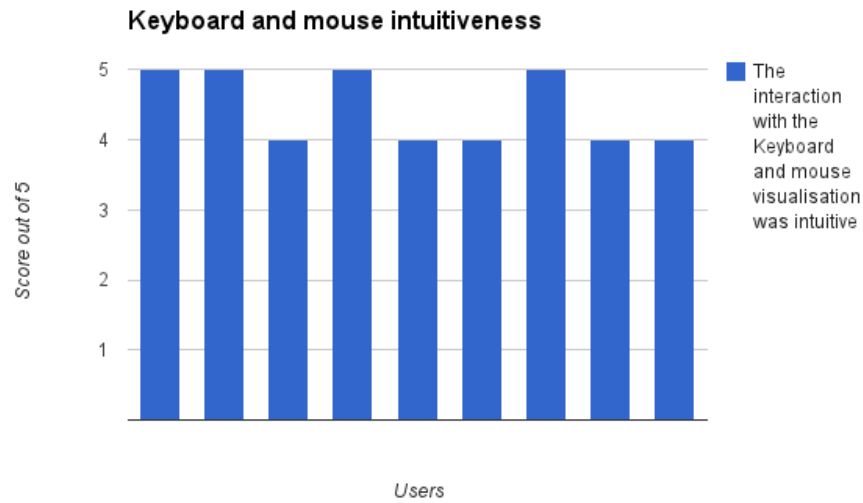


Figure 7.1: Intuitivity of keyboard and mouse

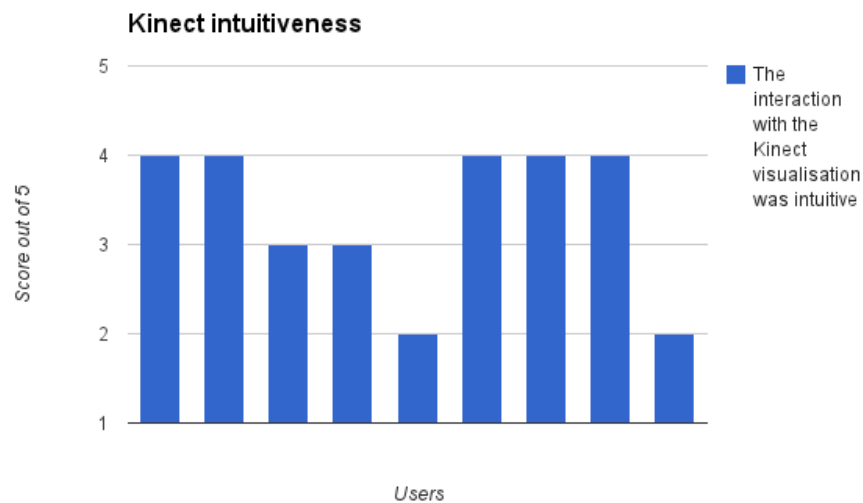


Figure 7.2: Intuitivity Microsoft Kinect sensor

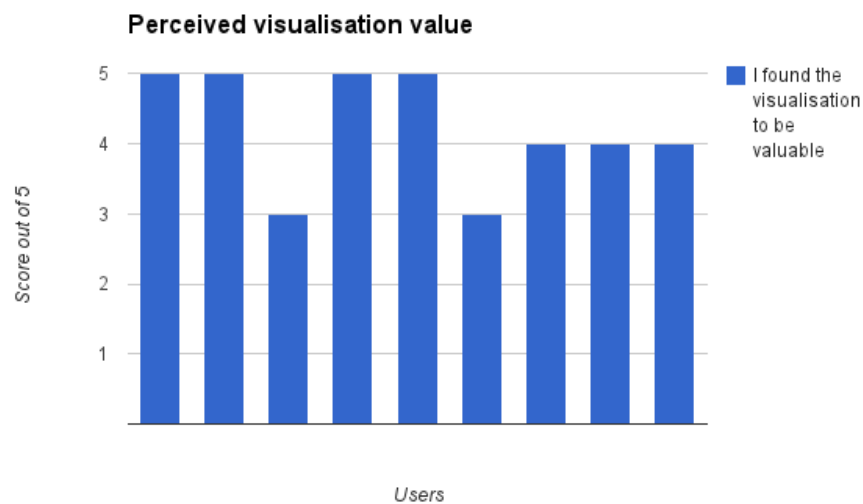


Figure 7.3: Perceived value of visualisation

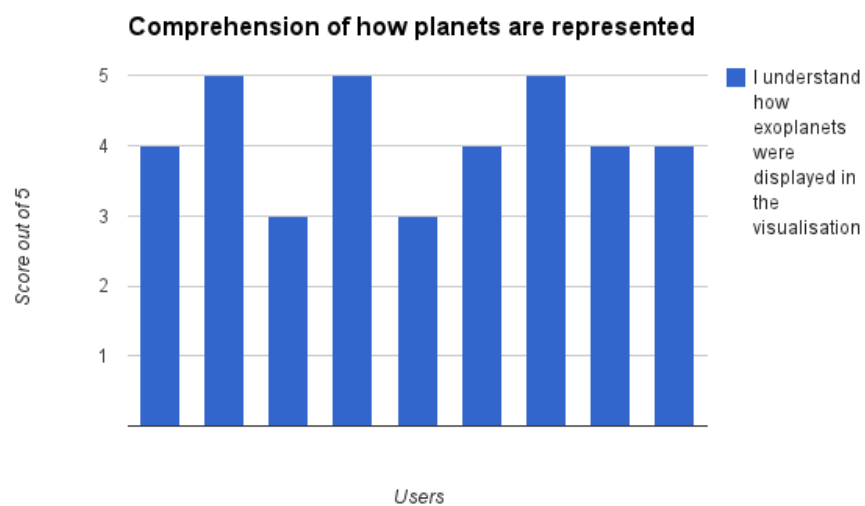


Figure 7.4: User comprehension of visualisation

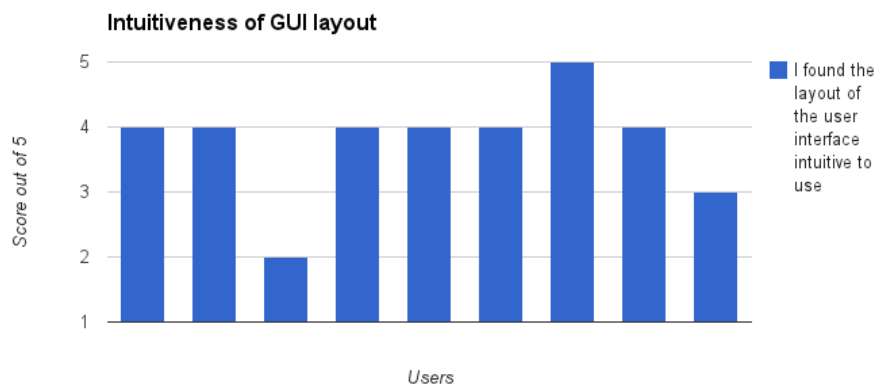


Figure 7.5: GUI layout intuitivity

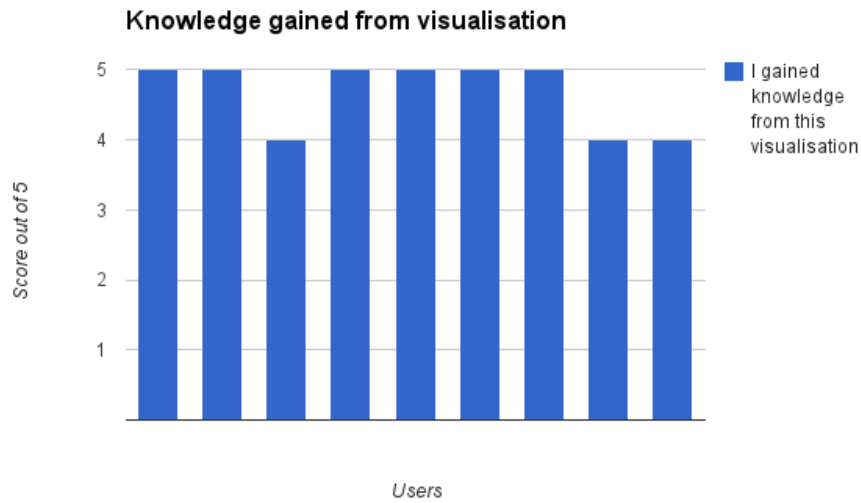


Figure 7.6: Knowledge gained from the visualisation

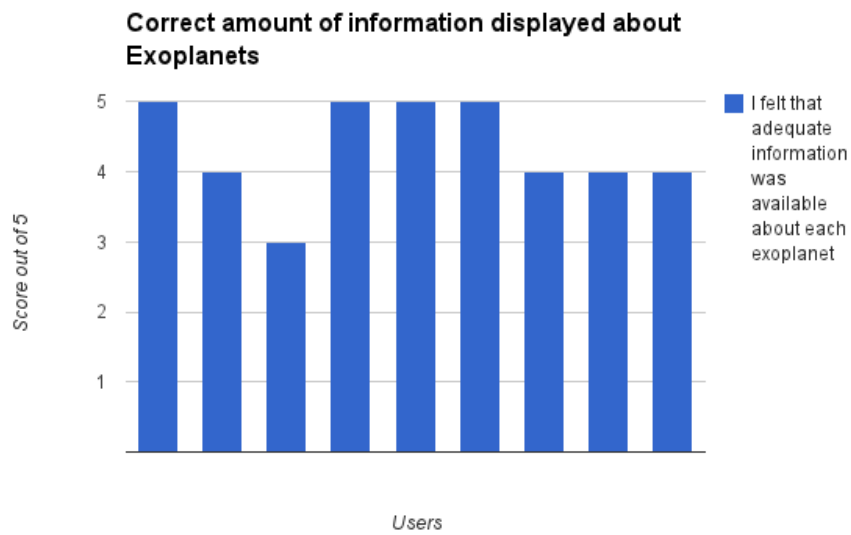


Figure 7.7: Correct amount of information displayed

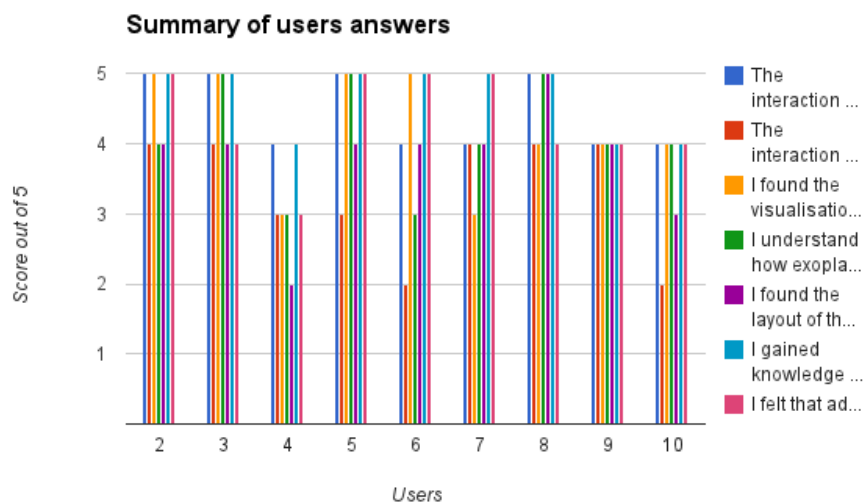


Figure 7.8: Summary of results

Chapter 8

Conclusions

8.1 Future Work

The work from this project can be taken further in many different ways depending on how it is intended to be used. There is the option of using the system as a terminal that users would use at an observatory or attraction where prior knowledge of the system is limited and amount of time users would spend on the system would be small. In this case further expanding the user experience and improved Kinect interaction would be beneficial as immersion would be the decider on its success. Another option for the system would be for a standalone desktop system that users would use multiple times and so prior knowledge of how to use the system could be expected. This would mean that more complex functionality could be introduced with the expectation that it could be used by users. The systems current state could be modified to fit into either of these two options.

Oculus rift, look at paper boy

