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

A short description of the project goes here.

Contents

Chapter 1

Motivation

There are many planets that have been located outside of our own solar system, these are called exoplanets. This project seeks to create an interactive 3D visualisation for the Kepler exoplanets dataset [**?**]. This visualisation will convey information in a way that the target users, laypeople who have an interest in astronomy, can understand.

The aim of this project is to implement a visualisation system that effectively conveys information which is not understandable to lay people in its database form. This involves creating a visualization that uses the data in the dataset to answer a set of questions that users are predicted to have about exoplanets.

Chapter 2

Literature Review

2.1 Layout of Multiple Views for Volume Visualization: A User Study

2.2 Interactive Design Metric Visualization: Visual Metric Support for User Interface Design

Chapter 3

Research On Prior Solutions

Much of the research I have performed has been into other visualisations which focus on stars, space, or planets. I felt that these would be the most likely to offer insights into the techniques that I could use in my project.

3.1 Worlds: The Kepler Planet Candidates - Non Interactive

This animation shows the 2299 high-quality (multiple transits), non-circumbinary transiting planet candidates found by NASA's Kepler mission so far. These candidates were detected around 1770 unique stars, but 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. (http://kepler.nasa.gov/multimedia/animations/scien

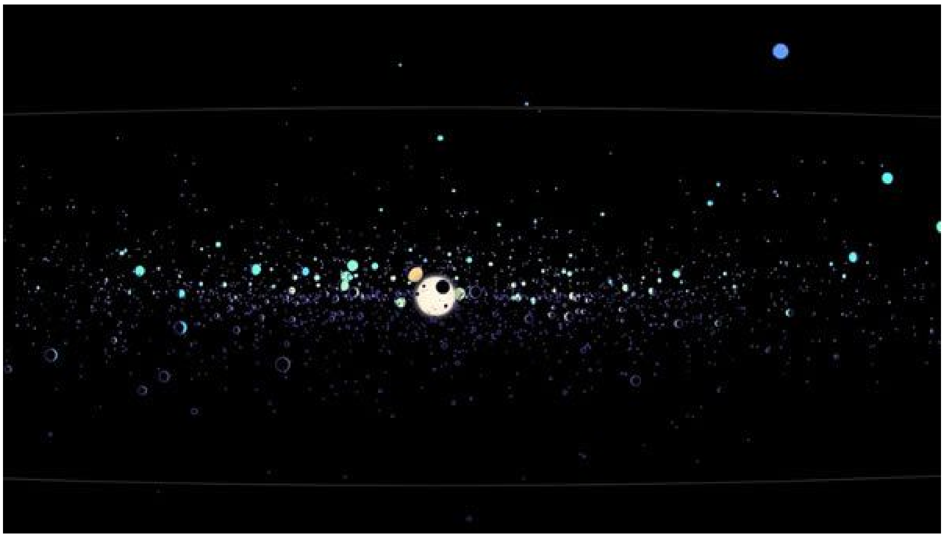


Figure 3.1: Image of Worlds Visualisation

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

This 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. http://kepler.nasa.gov/multimedia/animations/?In

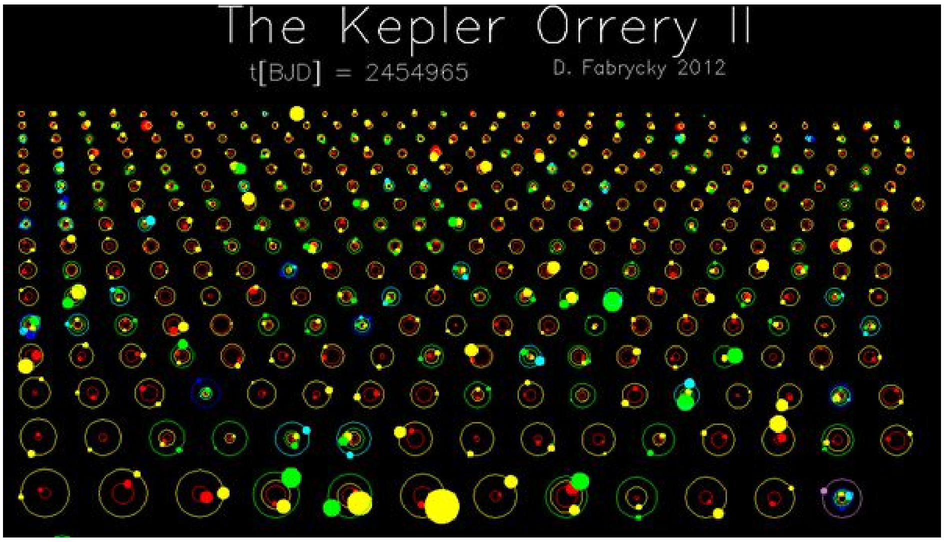


Figure 3.2: Image of The Kepler Orrery Visualisation

This system exhibits small multiples, a grid of small similar graphics or charts, allowing them to be easily compared.

3.3 The Kepler Visualization Tool

It displays all Kepler candidates, arranged as if orbiting a single star. Each candidate's estimated size, orbital speed, and orbital separation is accurately depicted, and each planet is color-coded according to its estimated effective temperature, with red being relatively hot and deep blue/violet being relatively cold. Mercury, Mars, Earth, and Jupiter are added for context. Two concentric rings plot distances of 0.5 and 1 astronomical units from the central star, and a pale blue line delineates Earth's location on two self-organizing charts. In the video posted above, the first chart in the sequence plots semi-major axis (i.e., average orbital separation) versus effective temperature, while the second plots semi-major axis versus planetary size.

Important data trends emerge from this visualization. The abundance of smaExplanation:

Using the vertical placement of exo planets to display their ESI while maintaining their horizontal placement should aid in comprehension by users about the comparisons to Earthller candidates and relative sparsity of larger ones clearly indicates that there are many tiny,

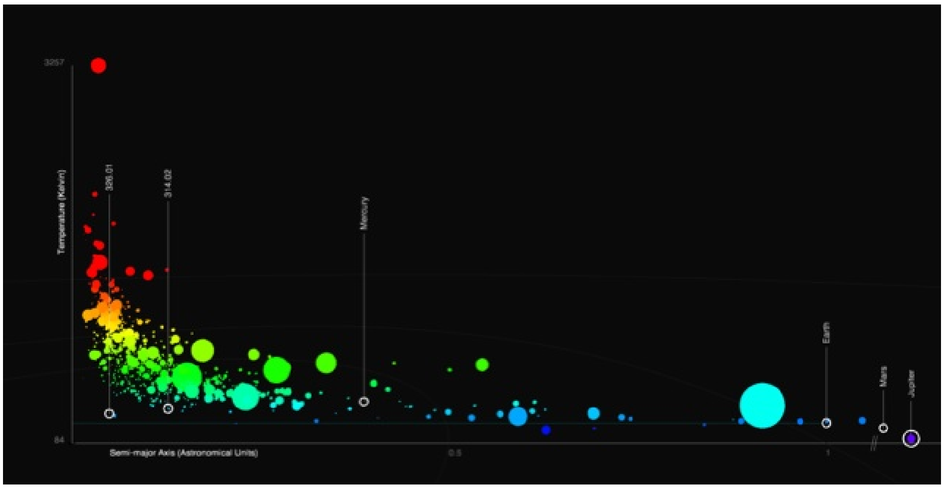


Figure 3.3: Image of The Kepler Orrery Visualisation

meek worlds for every giant planet. http://boingboing.net/2011/02/08/a-new-view-of-the-ga.html http://kepler.nasa.gov/multimedia/animations/scienceconcepts/?ImageID=135

3.4 Celestia - Interactive

Celestia is a free real-time space simulation that lets you visually experience the universe in three dimensions

Celestia is a computer program written in the computer language C++. The code is Open Source, and may be examined and modified by anyone under the terms of the GNU Public License.

http://www.shatters.net/celestia/

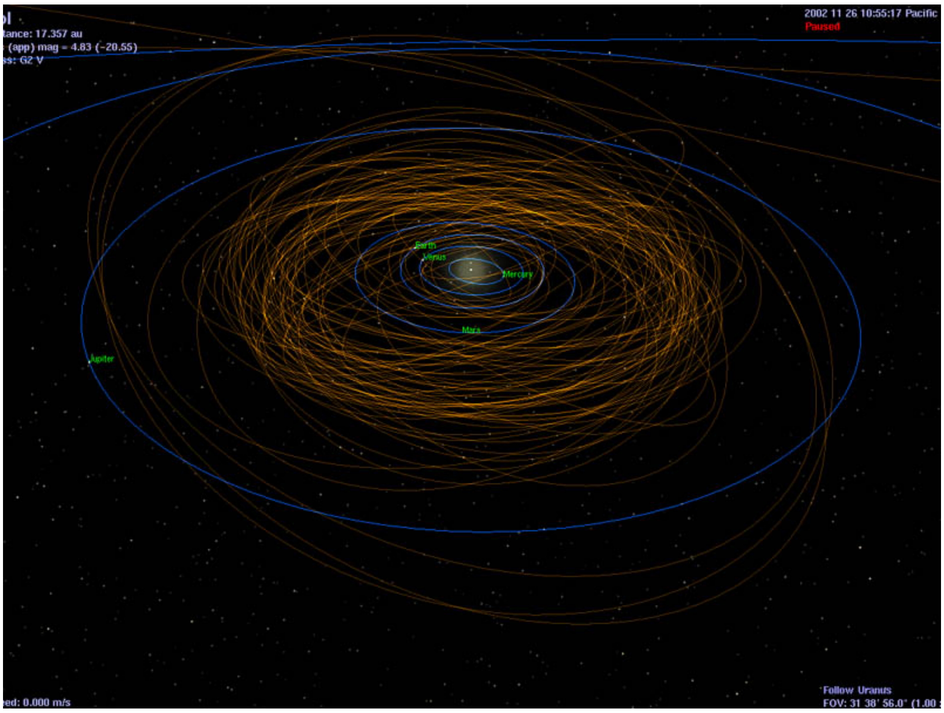


Figure 3.4: Image of Celestia Visualisation

Chapter 4

Technology

4.1 Requirements of Technology

It needs to have a low learning curve as the time available to learn a technology is small

3D is an advantage as it is more immersive(REF)

It needs to be suited towards visualizations with prior evidence that it can produce quality visualisations.

It needs to allow user interaction with dynamic transitions.

It would be advantageous if there is a pre existing solution with a similar dataset as it would mean that in depth design work would not be required.

4.2 Technology Option Exploration

4.2.1 Java using Swing or Processing

Building a visualisation in Java would mean that I would be able to design it from the ground up. This would mean that there would be an almost non existent learning curve as all of the system would be created by me. If I were to use the GUI library Swing, it would involve using a drawing canvas or another graphic rendering tool. However by using this solution over tailored and proven visualization toolkits I would be limiting the quality of the system.

A better option would be to use 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 proven visualisation framework. The most complete existing visualization using the same exoplanet dataset (Kepler Visualization Tool) is built using Processing.

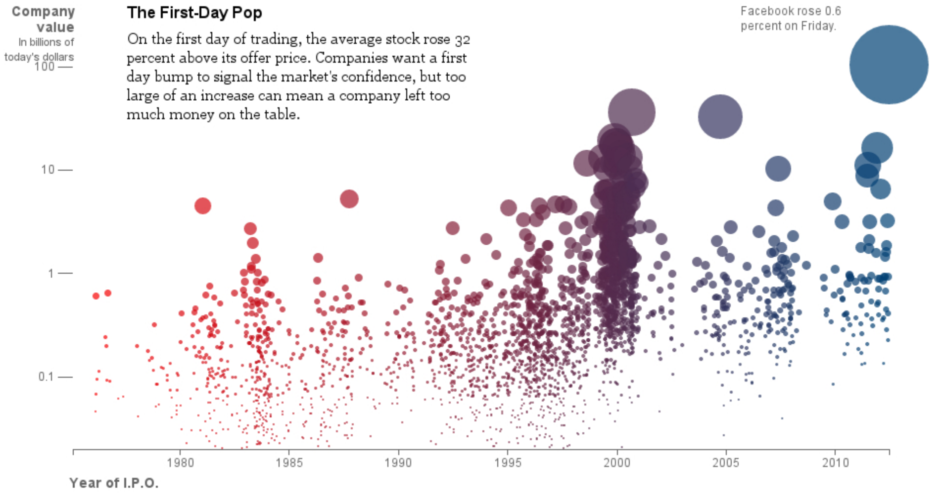
image

4.2.2 D3 (Data-Driven Documents)

D3 is a JavaScript library that allows displaying of digital 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) 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.

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 painter's algorithm and textures.



On the first day of trading, the average stock rose 32 percent above its offer price. Companies want a first day bump to signal the market's confidence, but too large of an increase can mean a company left too much money on the table.

Figure 4.1: Example system in D3 1 *http* : //*www*.*nytimes*.*com*/*interactive*/2012/05/17/*business*/*dealbook*/*how*
the − *facebook* − *offering* − *compares*.*html*

4.2.3 Prefuse

http://prefuse.org/doc/faq/ Prefuse is a set of software tools for creating rich interactive data visualizations. 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 intends to greatly simplify the processes of representing and efficiently handling data, mapping data

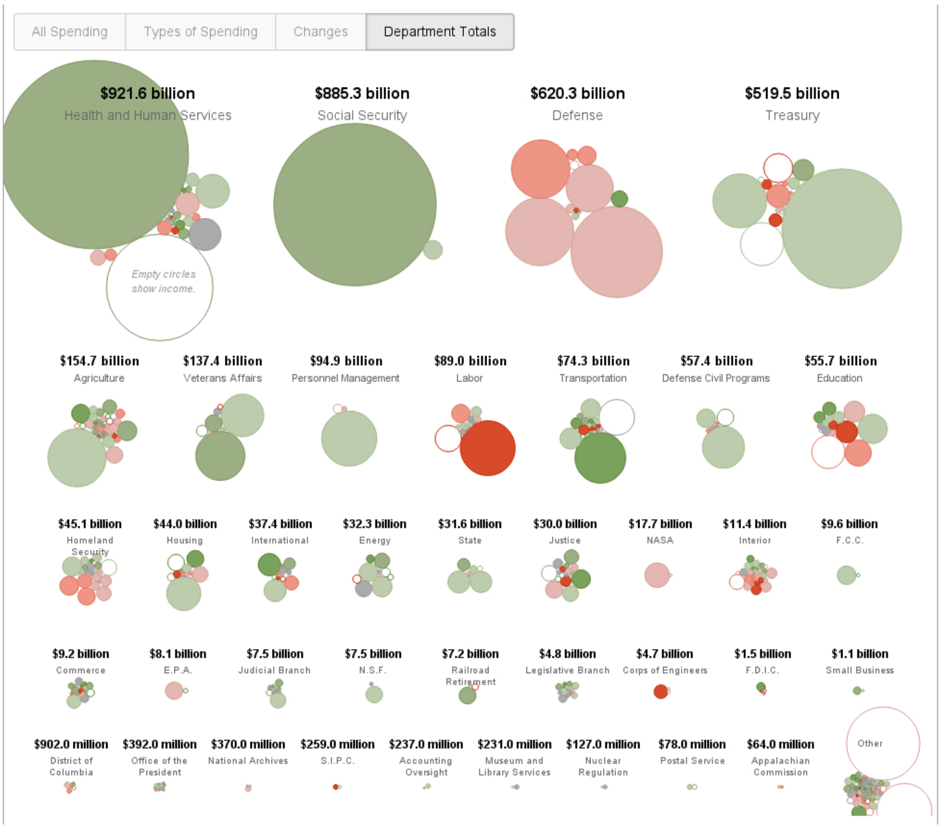


Figure 4.2: Example system in D3 2 $http://www.nytimes.com/interactive/2012/02/13/us/politics/2013-budget-proposal-graphic.html?_r = 0$

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.

4.3 Choosing My Solution

[[table]]

The technology that was chosen for this project was Processing. This was because of the advantages it had over the other options such as being 3D capable as well as having an existing solution with the Kepler dataset. It also has a shallow learning curve as it uses Java as the programming language. The only apparent downside to the language is that to effectively use the 3D functionality it requires knowledge of 3D transformations. However this negative aspect is overcome by its other positive features.

As well as using Processing as the tool for the visualisation, I will also extend the existing Kepler Visualization Tool. This is because it provides a basis for a successful 3D visualisation.

//TODO open source, further improved

//TODO To mitigate the risk of inadequate knowledge of 3D transformations

//TODO Because this solution has these advantages over the other solutions, and even though it does still have disadvantages. It has the highest chance of providing the best solution because

4.4 My Chosen Solution

The existing system built with Processing is a simple visualisation focusing on displaying the planets temperature and their location in relation to their distance from their nearest star so that a proper sense of scale to 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.

By choosing to extend this existing solution the benefits of using Java and processing would be joined by the benefits of using an already proven to be effective tool. The existing work in this system would serve as foundation for this project. The visualisation elements needed to convey answers to the proposed questions would be included into this system.

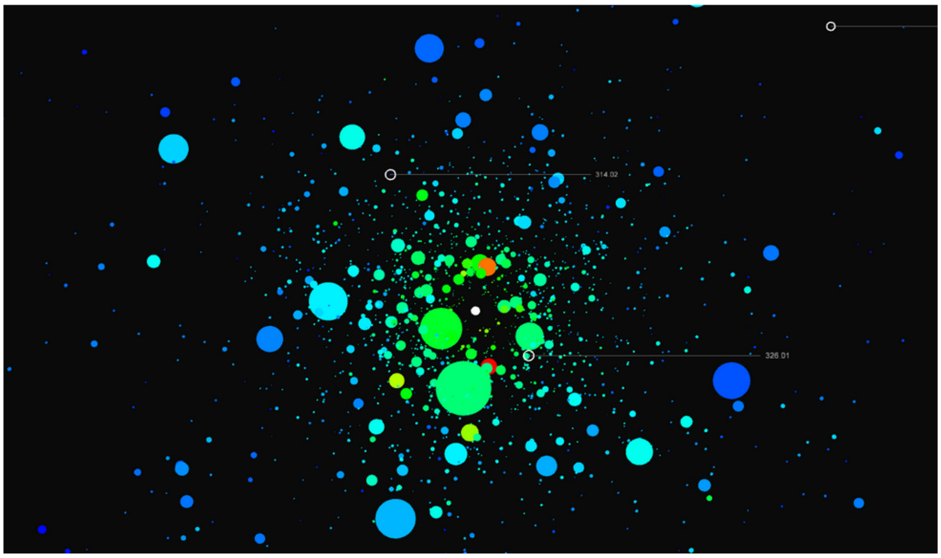


Figure 4.3: Kepler Visualisation Tool Orbital View

Using this solution would involve learning the Processing language, however Processings is a library built in Java so the syntax is the same. This means the learning curve in in regards to the program itself should be shallow.

Using processing and the existing visualization would mean that 3D elements could be included, this wouldnt be possible with D3. However it does require a strong knowledge in 3d transformations which I do not possess. This may be a limiting factor in the speed at which I could understand the existing code and may push using this solution out of scope in favour of D3.

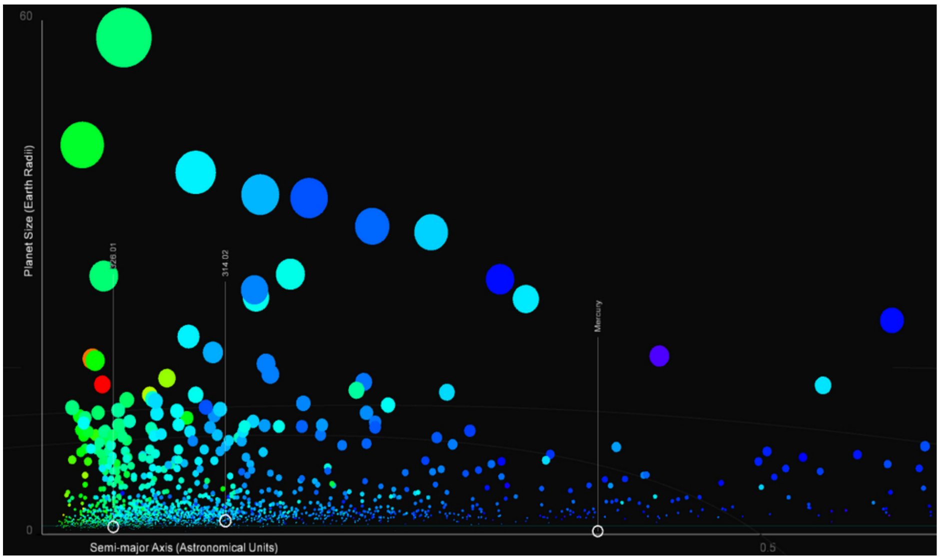


Figure 4.4: Kepler Visualisation Tool Graph View

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 focussed on more heavily and larger improvements to the existing system can be undertaken, such as better labeling and information displays and user interaction methods.

4.5 Existing work on Kepler Visualization Tool

The previous work that will be extended in this project is a 3D visualisation of the Kepler dataset.

4.5.1 Existing Layouts

The existing visualisation has 2 different layout views for the data

1. Orbital Layout - Shows exoplanets orbiting a single stars

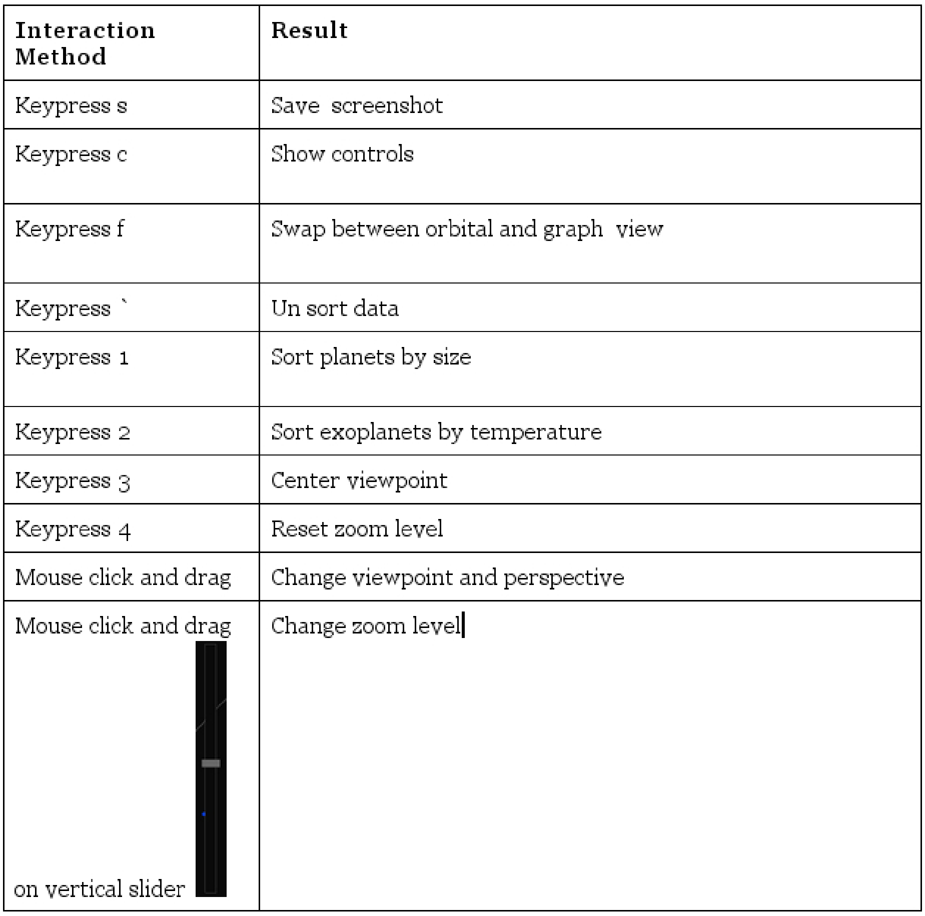
image

2. Graph Layout - Used to display planets on x,y scale using attributes

image

4.5.2 Existing Interaction Techniques

The only interaction techniques in the existing system are:



Interaction Method	Result
Keypress s	Save screenshot
Keypress c	Show controls
Keypress f	Swap between orbital and graph view
Keypress `	Un sort data
Keypress 1	Sort planets by size
Keypress 2	Sort exoplanets by temperature
Keypress 3	Center viewpoint
Keypress 4	Reset zoom level
Mouse click and drag	Change viewpoint and perspective
Mouse click and drag on vertical slider	Change zoom level

Figure 4.5: Table of exisiting interaction methods in system

Chapter 5

Visualisation Requirements

5.1 Required User Interactions

There are a set of interaction methods that will be important for this visualization. These methods are:

1. Select planets for further analysis
2. Select planet attribute to sort by

There are also improvements needed for the existing system to improve user interaction. These improvements are:

1. A panel containing buttons for each of the views changes
2. Include scroll to zoom functionality

5.2 Required Visual Elements

TODO

5.3 Layout Requirements

By using abstract user interface design the layout and configuration of each element can be planned and coordinated without the need for excessive details, as it is likely to change throughout the course of the project (e.g., colours and content).

The visualisation elements needed to convey the information for each question are broken up into 4 sections. This means that there should be a section in the layout of the visualization that corresponds to each. I have created 3 potential layouts and the optimal choice will be one that induces the lowest cognitive and load for users and is the most intuitive. I am able to focus on creating a layout that does not create unneeded extraneous load [cognitive layout citation] on users. Doing this means that each section of the visualization needs to be separated spatially from one another so that the cognition needed to visually separate each section is minimal.

For this visualisation there is the need for 4 user interface boxes.

image

1. Display the interaction components for the visualisation. This is because the existing solution has only keypresses as the way of modifying the visualisation. This is not sufficient as there is no indication in the system of how to use these. Also the proposed extensions to the system will require more advanced user interaction elements

2. Display the information about a selected planet.

3. Display a comparison between 2 selected elements.

4. (Main Panel) Display the main visualisation. The other panels will be used to support this panel.

These layouts will need to be evaluated by users in order to discover which will be the most effective. It is possible that none of these options are suitable and the layout may need to be redesigned, however as the functionality will be implemented, changing the layout will be a trivial matter.

Chapter 6

Visualisation Design

6.1 User Interface Design

$http : //web.cs.wpi.edu/\ matt/courses/cs563/talks/smartin/int_{d}esign.html$

6.2 Screen Mockup: Visualisation Interaction Panel(Panel 1)

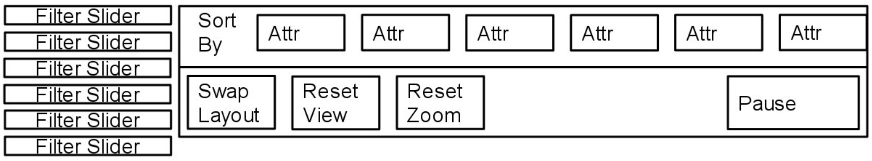


Figure 6.1: Mockup of interaction panel

By keeping the most widely used interaction elements at the edges of the visualisation it will allow mouse flicking for efficient component selection. Also by allowing GUI interactive elements as well as maintaining the existing key press interactivity it should allow a good balance between advanced and novice users. To improve this, each of the on screen interactive elements will also depict the keystroke that yields the same functionality as a way of transitioning novice users to advanced(REFERENCES)

FITS LAY AND STEERING LAW

6.3 Screen Mockup: Selected Planet Panel (Panel 2)

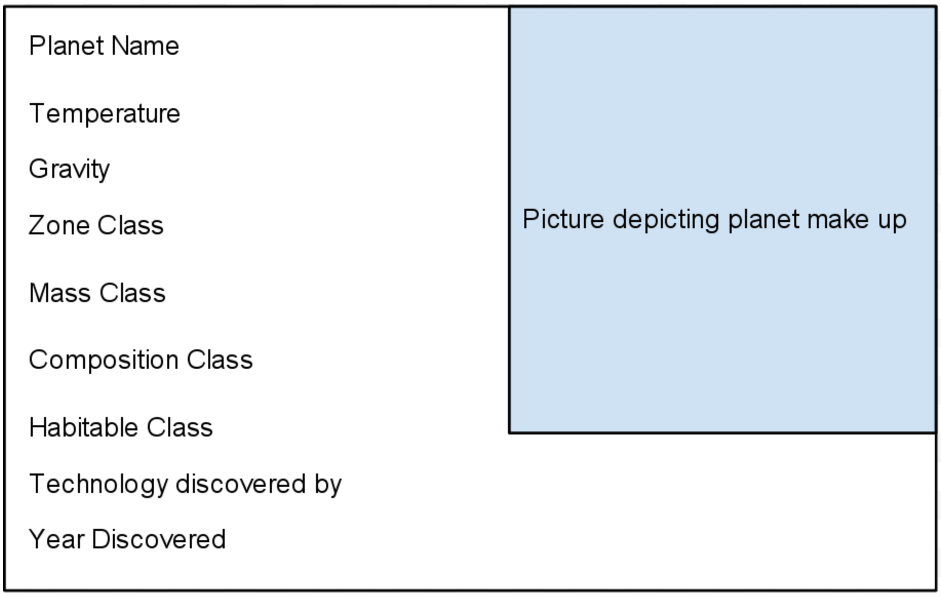


Figure 6.2: Mockup of planet information panel

Explanation:
Using the vertical placement of exo planets to display their ESI while maintaining their horizontal placement should aid in comprehension by users about the comparisons to Earth

6.4 Screen Mockup: Comparison to Earth panel (Panel 3)

6.5 Screen Mockup: Main panel (Panel 4)

6.5.1 Earth Similarity Index (ESIs, ESIi, ESIg)

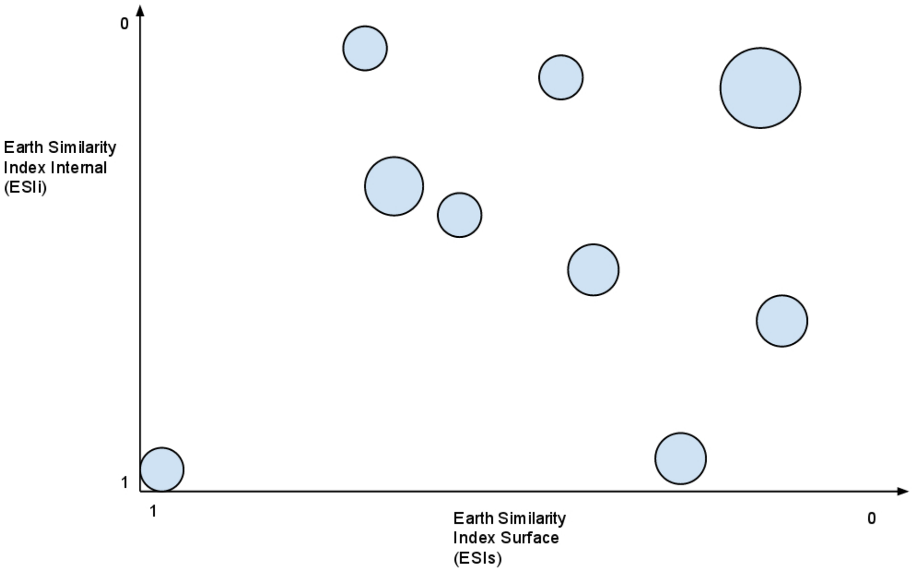


Figure 6.3: Mockup of planet ESI view in graph layout

Explanation:
Using the vertical placement of exo planets to display their ESI while maintaining their horizontal placement should aid in comprehension by users about the comparisons to Earth

Placement:
Exoplanets
Vertical Alignment = ESIg
Horizontal Alignment = Semi-Major-Axis(AU)

Earth
Vertical Alignment = 1.0 ESIg
Horizontal Alignment = Center of exoplanet orbit

Star
Vertical Alignment = 0.0 ESIg
Horizontal Alignment = Center of exoplanet orbit

Chapter 7

Progress

7.1 Progress Made

7.2 Difficulties encountered with implementation

Problem:
The existing solution was implemented in a way that only the z axis location of each exoplanet could be modified. This meant that to display multivariate data on more than one axis the method of translating planets needed to be modified.

Solution:
I extended the existing draw method of planets where the x and y locations of exoplanets were set so that it can be changed. However there is still a problem that it does not animate this change, this will require modifying the method further to incrementally move the planets when the axis are modified.

Problem:
Any on screen gui elements were placed inside the visualisation on the same plane as the planets, and so are zoomable as well as rotatable instead of being fixed as is needed for a GUI

Solution:
I currently have a partial solution which is to use multiple frames for each window component. However this is not optimal as the main window needs to be selected for key presses to work even though it does allow the user to rearrange the different windows as they wish.

Problem:
Due to the large number of items to be displayed on screen there is a large amount of performance loss and sometimes crashes the visualisation due to an out of memory exception or simply freezing.

Solution:
Filtering of data to reduce the amount of planets being rendered reduces this problem but does not solve it. This is still an outstanding issue.

Problem:
When filtering the exoplanets there is a lag in the rendering.

Solution:
I will find a more efficient way of sorting and filtering data. This is still an outstanding issue.

Problem:
When a planet is selected and a button is clicked a NullPointerException is thrown

Solution:
It was because the core planets of our solar system were being checked against due to them being considered feature planets when they should have been excluded. I refactored it so that they are now corePlanets and so the feature boolean can be used safely.

Chapter 8

Future Plans

8.1 Implementation

8.2 Evaluation

8.3 Report

Chapter 9

Conclusion

Bibliography

[1] DUMMETT, M. *The Seas of Language*. Oxford University Press, Oxford, England, 1993.

[2] DUMMETT, M. What is a theory of meaning? (i). In *The Seas of Language* [1], pp. 1–33. First published in S. Guttenplan, editor, *Mind and Language*, Oxford University Press, Oxford, England, 1975.

[3] DUMMETT, M. What is a theory of meaning? (ii). In *The Seas of Language* [1], pp. 34–105. First published in G. Evans and J. McDowell, editors, *Truth and Meaning*, Oxford University Press, Oxford, England, 1976.

[4] GOOSENS, M., MITTELBACH, F., AND SAMARIN, A. *The LaTeX Companion*. Addison-Wesley, Reading, Massachusetts, USA, 1994.

[5] GOOSENS, M., RAHTZ, S. P. Q., AND MITTELBACH, F. *The LaTeX Graphics Companion: illustrating documents with TeX and PostScript*. Addison-Wesley, Reading, Massachusetts, USA, 1997.

[6] LAMPORT, L. *LaTeX: a document preparation system: user's guide and reference manual*, Second ed. Addison-Wesley, Reading, Massachusetts, USA, 1994.