**Project Recap**

Technically challenging and interesting topic which has real world value and application – fractal rendering software is used in the discipline of fluid mechanics, data visualisation

Project was well defined, it had aims and objectives, a flexible scope with stretch goals, through requirements specification where requirements were linked to specific objectives, risk analysis with mitigation plans, and project plan with project timeline

All in all the project was planned quite well

**Work Completed**

Created a real-time rendering engine for signed distance functions (SDFs)

The use of signed distance functions makes the renderer very versatile as it is capable of rendering 3D fractals, Constructive Solid Geometry (which is primitive or simple shapes being combined using boolean operations) (CAD software), and algebraic surfaces (which is a branch of mathematics which focuses shapes created by the roots of polynomial expressions, an example of which is the Klein Bottle though there are many more)

**Additional Work Completed**

Added a Klein bottle scene

Updated the image gallery with some more screenshots, and added descriptions for some videos that were missing

Neglected to mention this in the report, application has debugging tools for scenes. I Added debugging tools section to documentation

**Evaluation**

Application

Unit tests for code correctness

Performance of the application was benchmarked to analyse the value of optimisations and visual features in respect to the performance cost

The scalability over different systems was also analysed which determined that the application performance scaled logarithmically as expected because the performance is limited by the amount of sequential code it contains

Brief discussion here and there about the visual artifacts that were created for certain features, I would have liked to go into more depth about this but didn’t have enough space in the report

Project

Requirements were marked using a status of achieved or not achieved which made sense in this case as many requirements defined visual features of the application, so it was enough to view output from the application and say: yes it is there, or no it isn’t there

Goal based evaluation strategy was used for evaluating aims and objectives, as requirements were grouped by the objective that they worked towards, and so once all MUST and SHOULD priority requirements were implemented the objective was also completed

Finally a comparison with existing pieces of literature was completed which put into perspective how this piece of work builds upon existing work, though unfortunately there are few real-time renderers and those that do exist weren’t too good at reporting runtime results

**Future Work**

As discussed in the report, this project is a valuable foundation for future work and there are lots of improvements that could be made

Several steps have been taken to make it easier to complete future work, such as

Using the GPL license which is open source compliant and allows modification as long as the modified code is made available and documented fully, and released under the same license

GitHub repository for version control which makes it trivial for someone else to fork the project to work on

Image gallery was created which contains screenshots and videos of the application

Thorough developer and user documentation created using which uses Doxygen as an automated tool to generate an interactive web page, which is hosted using GitHub pages

<https://mathworld.wolfram.com/AlgebraicSurface.html>