# 10004794 SET09115 Coursework Report

## Introduction

The aim of this coursework was to create a 3d scene using OpenGL. This report will detail the design of the scene, how the scene was implemented, an evaluation of how well the final scene meets the design and a performance evaluation using metrics from gDebugger.

## Scene Design

Whilst trying to think of ideas for a scene that I would like to implement I went to watch a firework display on bonfire night. During this display I was watching the fireworks and the bonfire and was thinking that it would present an interesting challenge to try and recreate the scene.

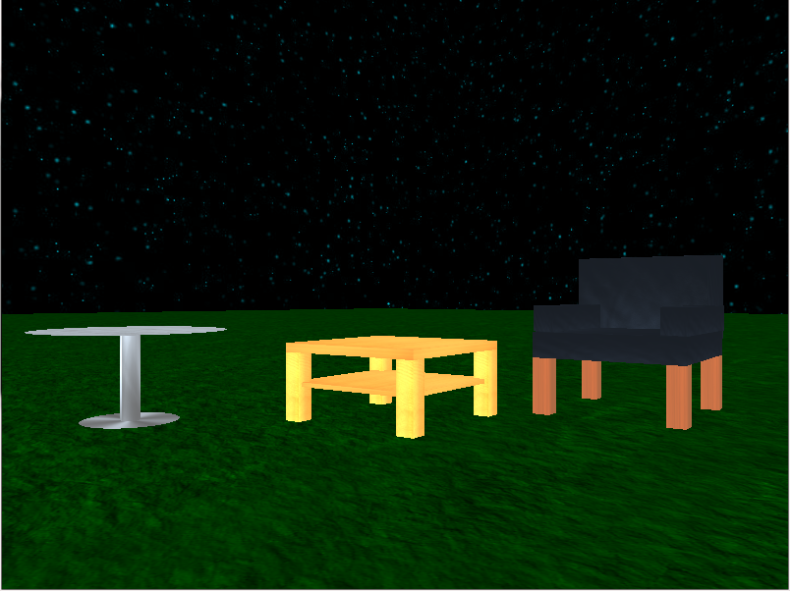
Expanding upon this idea further after researching some other scenes I decided that my scene would be set in a small park with houses and tress in the background with the fire set behind a park bench and some street lighting with some small hills. The images below are the images I choose to base my scene from.



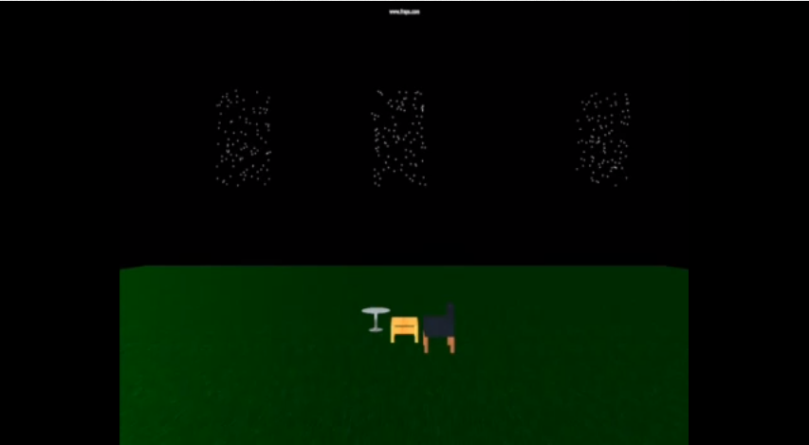


## Implementation

The scene that I choose to implement provided many interesting challenges. Using the framework that was built up during the practical sessions I created a small scene to begin working from and use as a base to build up from. This base scene read in multiple pieces of geometry from a JSON file and created furniture that sat on a plane. The scene made use of lighting effects and texturing for the geometry, the textures make use of mip-maps, scaling and anisotropic filtering. A screen shot is provided below.



After I had the base scene implemented I began working on the fireworks. I found an interesting tutorial which made use of particle effects to make 2D fireworks (source http://r3dux.org). The way in which the fireworks are created was by making a class of fireworks which each have a set number of particles. Each firework has a position, speed, point where it will explode and colour value associated with it. The fireworks also have an initialise, move and explode method. When a firework is initialised its position, initial movement speeds and the point at which it will explode are set. When the firework is being drawn to the screen a check is performed to check if a firework has exploded or not. If the firework has not exploded the move method is called which will continue to move the firework upwards. If a firework has exploded the explode method is called which iterates through each particle in the firework and increases its x, y and z position based upon a speed which is calculated randomly using the rand function to determine the direction to fire the particle in. fireworks are also faded out using the alpha value of the colour and then reinitialised when this value reaches zero. To create the trail effect for the fireworks I made use of the accumulation buffer, this allows me to take the last frame and store it in the accumulation buffer and then when I go to render the next frame I can copy the contents of the accumulation buffer into the colour buffer and then draw the particles again at their new position. A Screen shot of the fireworks with the base scene is provided below.



The way in which the fireworks explode are very square. One way I looked into trying to fix this was by surrounding the firework in a bounding sphere and culling particles that were outside of the sphere. I was unable to get this working properly as I could not figure out the maths involved to create a bounding sphere around my fireworks and due to time running short I stopped working on it to continue adding more features.

I began working on the park scene next. I created the geometry for a park bench and street light which are read in from a JSON file. There is a spot light which is attached to the bulb of the street light which illuminates the park bench. The scene also makes use of a skybox to create a background for the scene which includes the trees and houses that I planned on implementing in my design.



The next major task was to try and implement fire, I should have tried to do this nearer the begging but I left it quite late and was unable to get anything working. I researched multiple ways to do this and the most common response was to make use of a particle engine. I briefly tried to rewrite my fireworks to create a fire effect but I couldn’t figure out how to create it properly without my fireworks breaking as well. Another way I looked into was by using point sprites but again I couldn’t figure out how to create the effect from the point sprites example that was provided and decided to stop trying and attempt to combine my current scenes into one.

When combining both my scenes I discovered that my fireworks would not work properly with my skybox, I am not entirely sure why this was the case but I could either get my skybox to render or the fireworks to render but not both at the same time. It was at this point that I decided to slightly change my idea for my scene.

## Scene Design Part 2

### Background

The idea for my second scene came from sitting looking at images on the internet. I came across an optical illusion which got the user to stare at the negative of an image for 15 seconds and then it would display the same picture but in black and white, to the user though the image would appear in colour even though it was actually black and white. I realised that by making use of post processing effects it may be possible to replicate this illusion.

### Design

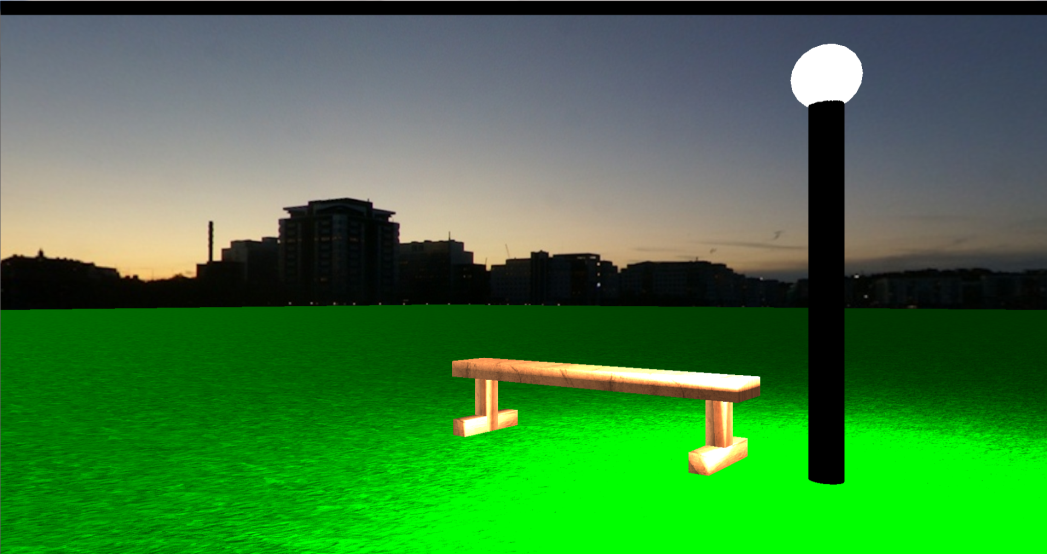
I decided that what I would attempt to do is to keep both my scenes that I currently had separate from each other and allow the user to switch which scene they are currently viewing. When the user switches to either scene it will be shown in its negative and will stay that way for 15 seconds before switching to a grey scale image.

## Implementation Part 2

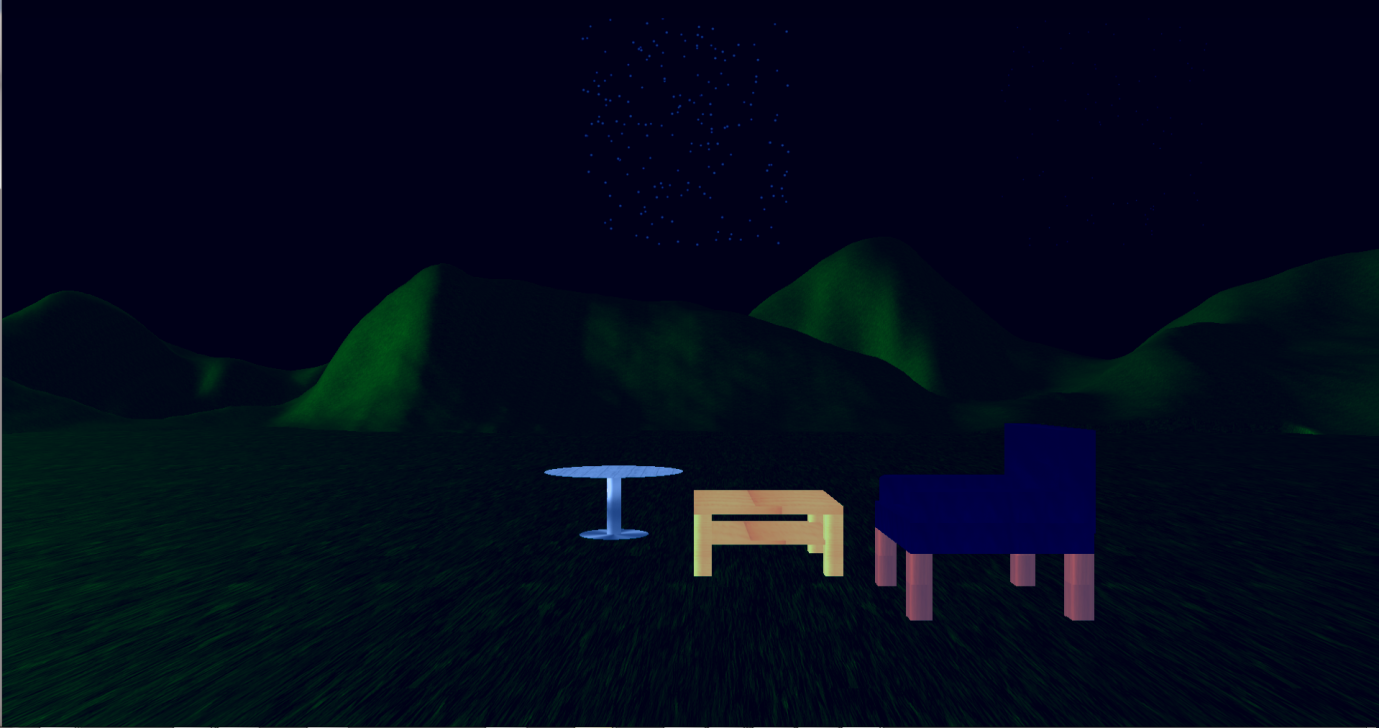
In order to do this I had to implement post processing effects on my scenes. The first post process I added was grey scale which was one of the effects that was given to us. Once I had this working on my scenes I researched how to create a negative of an image. This turned out to be relatively simple and only involved writing a shader called negative.frag which took in the original colour of the image and does (1.0 – colour). Images of the park scene are shown below as a negative and a grey scale.



I then set up a timer which is started when the user selects a scene and then switches between the effects after 15 seconds. Once this was working properly I then added more post process effects which my user can switch between by pressing the keys y,u,I,o and p . I added in a sepia effect which involved making a small change to the greyscale shader and a night time affect which gives the scene the impression that it is late at night with a dark blue colour applied. This was created by taking the original colour of the image and subtracting the sepia value. Screen shots of the effects are shown below with no effect shown for comparison.

No EffectNight EffectSepia Effect

The final part of my original scene left to add was the terrain effects. Making use of the create terrain method that was given in the practical lessons I added a height map to my firework scene JSON file and used it to create a small mountain range within my scene. Screenshots from 2 different cameras are shown below.



The final requirement for the coursework was to have multiple cameras in the scene. I have 2 different cameras in each scene, all of which are first person cameras which the user can control using the wasd keys and q,e, left shift and space to rotate and move up and down.

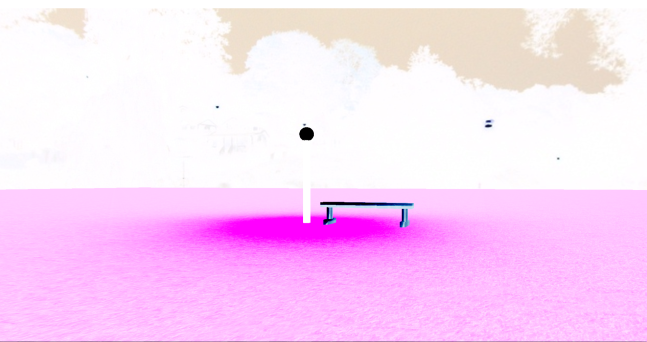
As I was able to switch between scenes I created a third scene which displays a cube with a texture with the instructions for the application displayed on it. This is the scene that is first shown when the application is run.

## Scene Screen Shots

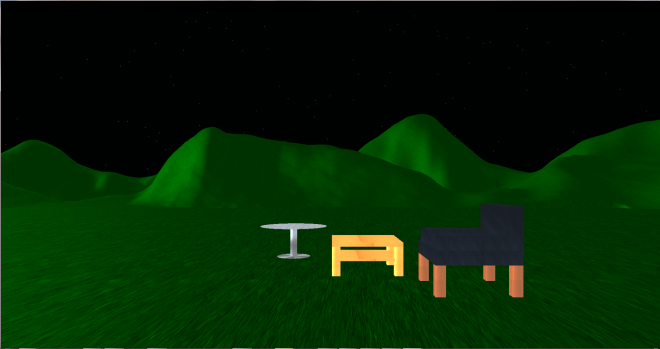
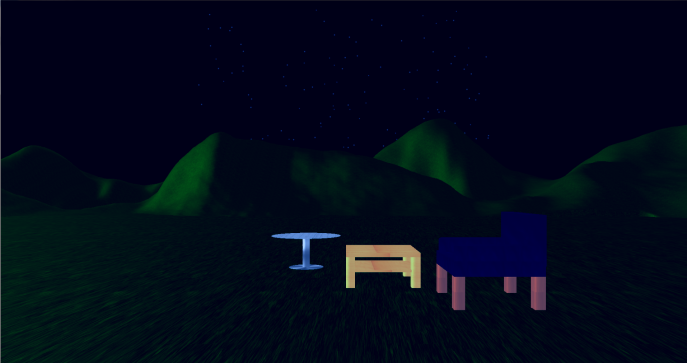
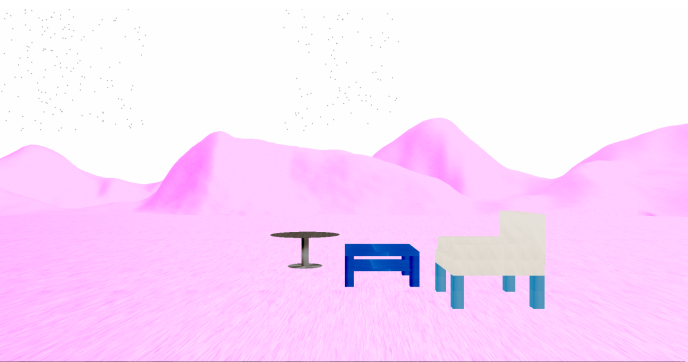
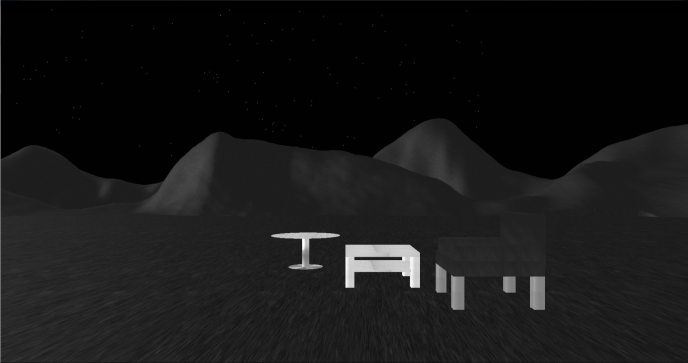
### Park Scene Camera 1



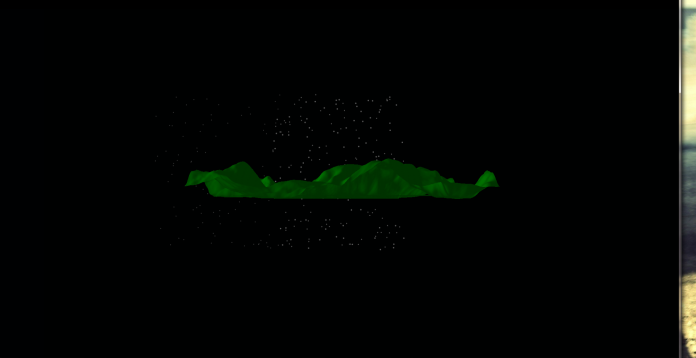
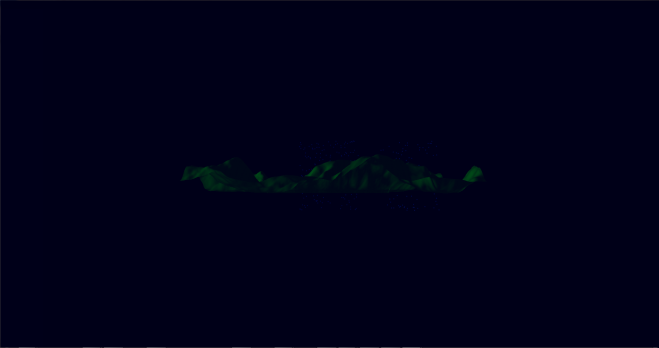
### Park Scene Camera 2



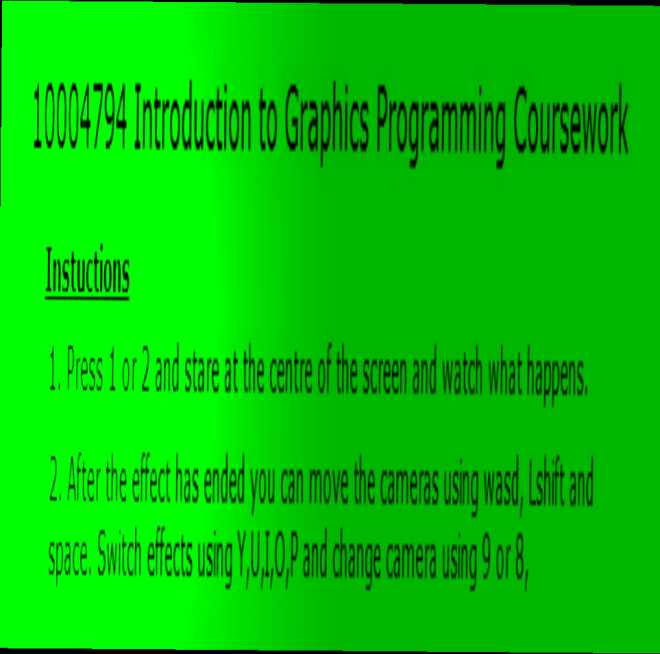
### Firework Scene Camera 1



### Firework Scene Camera 2



### Instruction Scene



## Evaluation

Overall my scene is not as close to my original idea as I would have liked it to be. The main feature that my scene is missing from my original design is the fire. I never left myself enough time to properly research and implement it in a way that worked with the rest of my framework however if I had spent more time than I had on it I would have not got any of my other features implemented such as the skybox, post process effects and terrain. However the scene that I have is fairly close to my second idea. I was able to almost recreate the optical illusion and it does work if you stare at the screen until it changes to the greyscale effect and it does briefly appear in colour, the effect appears to work better on the park bench scene compared to the firework scene which may be due to the fireworks as all the examples I found online only worked on static images.

All of my images for the textures and skybox are images that were found on Google. The original source for the optical illusion is http://imgur.com/r/gifs/VDuAa

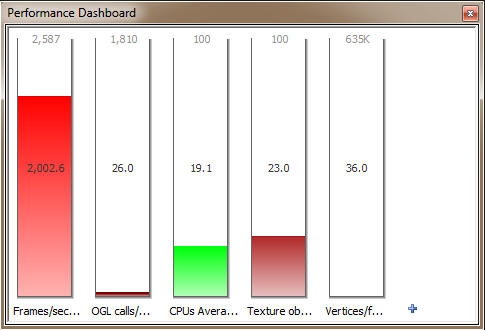
## Performance Analysis

Here I am going to compare the differences in performance of my three scenes which my user is able to switch between. The performance metrics will be from gDebugger which calculates the frames/Sec, OGL Calls/Sec, CPU Average Utilization, texture objects and vertices per frame.

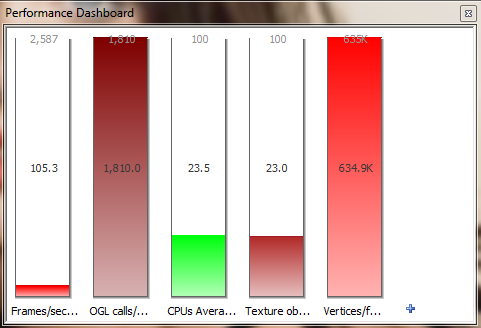
The machine that I am using to run the application on uses an NVidia gtx 560TI and an 3.4 GHz Intel i7 CPU which could be considered a fairly high performance machine. I would expect each scene to perform very well on a machine like this and don’t expect to see any major performance issues.

I have specifically set up my application so that all scenes, textures and effects are set up in the initialise and stored in memory so that when I am switching between them it is very fast and there is no loading involved. However as a result of this my application takes a considerable amount of time to load before the user can interact with the application.

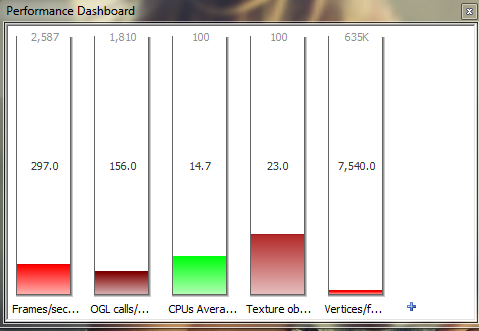
### Instruction Scene



### Firework Scene



### Park Bench Scene



### Comparison

Comparing the three scenes to each the performance is as I expected. We can see that the instruction scene had the highest frame rate with an fps of almost 2000. This is unsurprising as the scene only contains a single cube which has 36 vertices per frame to process compared to the 634k that the firework scene is processing each frame; however the firework scene is still managing over 100 fps with the park bench scene getting almost 300 fps these scores are okay for the scenes and I would expect them to be able to run with at least 30fps on a much slower machine. It would be possible though to increase the performance by making use of techniques such as frustum culling as this would mean that each frame less vertices would be being processed as only objects in view would be rendered and as we can see from comparing the images above the less vertices in a frame the higher the frame rate.

I ran the performance for all the different post process effects that I have applied on my scenes and for each of the different cameras that I have. There was no real change between any of the metrics due to the different effects.

## Conclusion

In conclusion I was able to create a 3d scene using OpenGL and the framework that was built up from the practical sessions. My scenes where all read in from JSON files and contained multiple pieces of geometry which were all textured and made use of mip-maps and anisotropic filtering. Each scene contained multiple cameras and had multiple light effects such as spot lights. My scenes also had multiple post process effects applied such as grey scale, sepia, negative and a night time effect that the user can switch between by pressing keys. I made use of additional rendering techniques within my scene which included skyboxes, terrain and particle effects which were used to create the fireworks. Overall I am fairly happy with the implementation of my scenes however I feel if I had managed my time more efficiently and started researching ideas earlier than I did I may have been able to get fire working properly and my original scene working as planned although I am happy with the optical illusion that I created instead through the use of the post processing effects.