Cool Particle Shit in Vulkan with lights Woop Woop

**Lewis Bond**[lewis2.bond@live.uwe.ac.uk](mailto:lewis2.bond@live.uwe.ac.uk)  
Supervisor: James Huxtable

Department of Computer Science and Creative Technology

University of the West of England  
Coldharbour Lane  
Bristol BS16 1QY

*Screenshot/ image of the work (600 pixels high x 800 pixels wide .jpg)  
This image is to be used for the degree show booklet. For white backgrounds please use 1/2pt black border.  
Also submit a copy of image file separately (600 pixels high x 800 pixels wide .jpg @ 240 dpi).*

**Abstract**

Verdana, 9pt. This is a top line description of your project; not the introduction. That comes later. Here you summarise your project, giving a bird’s eye view of your application/ software/ animation/ etc. So: what is the project? What is its aim? How was this realized in practice? What were the key results/ outcomes? Don’t try to write anything new here, just repeat the main points achieved in your project. Short summaries like this are for other professionals who just want to get to the core of your work without having to read every detail.

Note: this information is not included in the word count.

**Keywords**: term, term, term

**Brief biography**

Verdana, 9pt. A 70- 100 word section about you in relation to this project: how does this project link in with your professional skills, ambitions, interests, how will this project help you in your career. (not included in word count). Add a link to your online portfolio.

**How to access the project** (not included in word count)

Verdana, 9pt. Please put down the project URL and/or details of how to access your project, and the ***URL of your final video***. If we need to see any back end / administration interface please provide us with necessary passwords and URLs to access this. Basically, what we cannot access will not be considered for marking. You can change the passwords after you receive your mark.

We also want to be able to see source code, and the best way is to download it from your site or Github. Please clearly comment code to show us what is your own and what has been used from frameworks, libraries, OSS or borrowed from elsewhere.

If there is some other method for providing access to your project you will need to provide instructions here. Also if there is anything we need to know about the work that will not be self-explanatory, then also provide brief instructions here.

**1. Introduction** 400 words

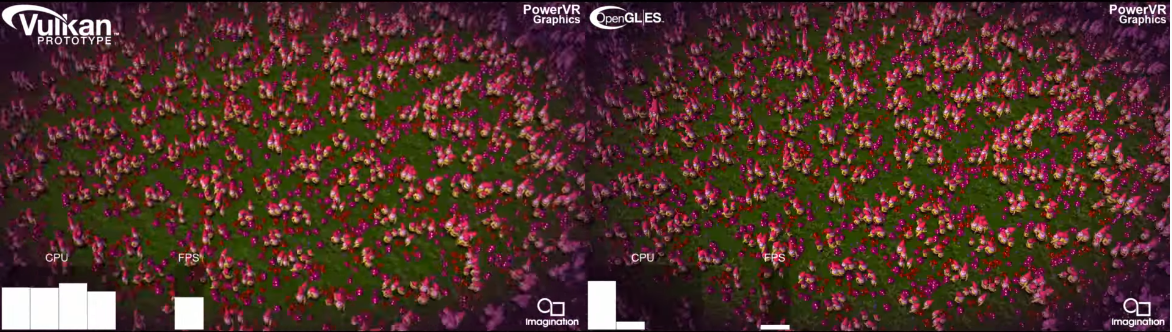
The formatting for the main text is Verdana, 9pt.; the text should be justified, except for the references section and bibliography. Sub headings are 9pt bold. Avoid writing in first person, focus on the project instead.

Briefly introduce the project within its professional context. Tell us what it is about, what problems it solved, why it is important, interesting or valuable. What was it seeking to answer, how did it arise, why was it worth investigating?

* Also bullet list the
* key deliverables and
* project objectives here.

**2. Practice** 1500 words

Make cool particle systems in Vulkan with dynamic lighting.



### Vulkan

Vulkan is a lot more explicit as a graphics application programming interface (API) than other such as DirectX 11 and OpenGL. It allows the user to have more control over the graphics processing unit (GPU) and central processing unit (CPU). Since this is the case there are a lot more systems and processes that the user must be aware of and account for when using Vulkan.

Using Vulkan may not have been necessary, as the scale of the project doesn’t require to use the aspects of Vulkan that would make a significant difference compared to other APIs. Though , if this project were to be taken further, that is when the performance difference would be noticeable. As shown if figure 1 where frames per sec (FPS) is a lot higher using Vulkan than compared to OpenGL.

Understanding the different elements and aspects to Vulkan was a challenge. Once the project was up and running though, and a cube was being rendered onto the screen. A challenge of the project was getting the particles into the scene.

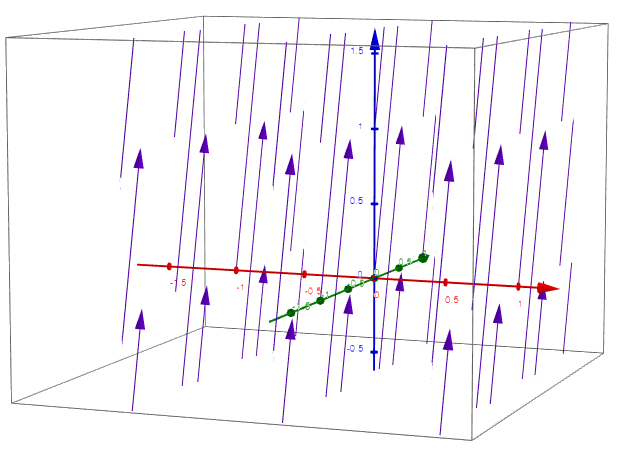
Instancing was used at first as a method of getting multiple particles into the scene. Instancing was used instead of multiple models as it more efficient. If the particles were multiple models, then the performance would of dropped as creating thousands of the same model over and over again becomes inefficient. It is performing the same process repeatedly just for the same object. Calls to the GPU are expensive so the aim to limit the amount of these is necessary for better performance. By using instancing, the ‘vkCmdDraw’ command is only being used once instead of once for each object. This also saves on function calls such as ‘vkCmdBindPipeline’ and ‘vkCmdBindVertexBuffer’ as these are all needed when creating and object and displaying to the screen. Having to call these commands thousands of times a frame would be impractical, compared to using the draw command once and setting how many instances the GPU needs to draw.

* Insert data from instancing to non-instancing

After a discussion with a supervisor, instancing was the right idea but wrong execution. A sphere model was instanced, which still was a significant number of vertices (points in 3D space) and would be improved by using billboarded sprites. A sprite is a 2D image, that is the graphical representation of an object and billboarding is a technique of making it also face the camera. Since it is 2D then only 4 vertices are required instead of how many there are in the sphere model. A geometry shader was implemented and replaced instancing as the chosen method. A vertex was passed to the geometry shader, this acts as the sprite’s origin, and 4 vertices are calculated from that point.

* Insert images from models to sprites

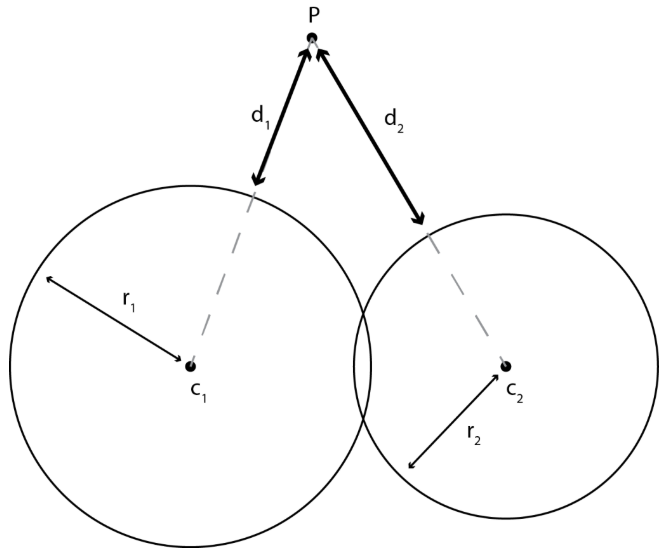
### Signed Distance Field/Function

The particles were meant to flow and move about in a way that looks natural. A way of doing this was using flow fields or vector fields. A flow field is a series of vectors assigned to a point in space as shown in fig. A vector field was implemented to make the particles travel in different directions.

A simple flow field was used at first, it takes the position of the particle and uses this to get the velocity which will then be added to the position of the particle, the equation is shown in fig. ‘V’ is the output velocity and ‘p’ is the position of the particle.

This produced this effect as shown in fig. The equation was slightly altered, and this gave a drastic effect to how the particles acted as shown in fig. The equation for this is shown in fig, and all that was changed was the z component of the vector being divided.

Though this method produced effective results on making the particles swirl, look fluid and dynamic. After another discussion with the supervisor another method was suggested, signed distance functions. A signed distance function (SDF) takes in a position and outputs the shortest distance from that point to the object as shown in fig.



The SDF that was used was for a triangle, so it would be given as position and then work out the shortest distance to that position.

FUNCTION Distance PARAM Vector3 Position

Implementing the SDF was a challenge to work out, as since there was a lot of particles and with more complex shapes, there are a lot of triangles. The performance dropped significantly having to calculate the shortest distance for 10,000 particles from 3,000 triangles. The solution for this was to put the calculations for the shortest point on a separate thread. This worked as it improved performance significantly.

* insert comparisons between threading and not threading

The results of the threading the signed distance field were as shown in fig.

* insert pictures of the signed distance field working with models

### Lighting

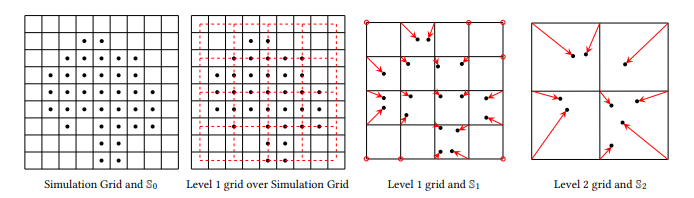
A lot of particles systems all have in can act as light sources such as fire, a common trick is to use a single point light as the light source. [get pic from unity] This is to save on computation time as it is a real problem in the industry to calculate accurate lighting efficiently. This project is implementing the lighting by having each particle act as a light source.

The lighting implemented at first was done on the CPU side. The equations for the lighting being implemented are shown in fig.

Having the lighting on the CPU side is slow, so to increase performance, threading was used again for the light calculations and this was helpful.

* Insert comparison of thread and non-threaded approach

### Lighting Grid Hierarchy

The Lighting Grid Hierarchy (LGH), is method developed for rendering real time lighting. It builds grids around all the lights and calculate the average colour and position of the lights within that grid space. It then does this for the how many levels of the hierarchy the user requires and with each level, increases size of each box within the grid as shown in fig. This is an effective method of optimising the lighting and allows for different levels of detail with the lighting. The more accurate lighting the user wants can accessed by traversing the hierarchy. It advances on the work of [insert previous smart guy work here] and [insert previous smart guy work here]. [smart guy] proposed the many lights problem which is having a lot of lights in the scene producing light and efficiently calculating that. The LGH mentions solutions for calculating shadows and global illumination but this project isn’t incorporating those aspects.

The LGH was implemented by using a vector of light structs. The vector containing all the lights would be passed in and the vector within the LGH is resized to be 64. The average position and colour of the lights within in each section of the grid is calculated from the numbers of lights within each section. Some sections will contain no lights so that is set to 0.

The issue with this was getting the accuracy of the lights as shown in fig, the effects produced were not expected. This was down to the LGH only having 1 level in the hierarchy and having it average the positions being to large.

State the main outcome(s) of the project. To help you do this, you can insert quotes, graphs, screenshots, diagrams, short code snippets etc. if useful and important.

<p>   
For short code examples please use Courier Regular, 9pt. <br /> Larger code examples go in an appendix. Highlight code sections in colour if necessary.

</p>

Reflect: How has the project developed *after* the research phase, and in hindsight how successful was its developmental phase? How did the research impact on the practice and vice-versa? How did user/ peer/ tutor feedback impact on it?

Evaluate: How successfully were problems encountered along the way being dealt with? Give us a few specific examples, not the whole story. These examples could be conceptual, technical, practical, legal (think copyright) or ethical (think user studies). Tell us how problems were overcome, or at least how every method was exhausted to arrive at the best possible solution.

**3. Discussion of outcomes** 900

Reflect: What significance does this completed project have in a wider professional context? What can other professionals draw from this project? What does it improve on? Critically analyse what has been achieved and if necessary, suggest different future approaches.

You need some references here to link the project convincingly to its professional context. If you are new to UWE Bristol Harvard referencing please read the [introduction to referencing](https://www1.uwe.ac.uk/students/studysupport/studyskills/referencing/introduction.aspx). Click here for [how to refer to (cite) a work in your text](https://www1.uwe.ac.uk/students/studysupport/studyskills/referencing/uwebristolharvard/howtociteawork.aspx). Also here for [General advice on how to format quotations in your assignment](https://www1.uwe.ac.uk/students/studysupport/studyskills/readingandwriting/writing/formattingyourwork.aspx) (includes when to use quotation marks). Follow guidance here on how to reference the following: [books](https://www1.uwe.ac.uk/students/studysupport/studyskills/referencing/uwebristolharvard.aspx#books),  [journal articles](https://www1.uwe.ac.uk/students/studysupport/studyskills/referencing/uwebristolharvard.aspx#journalarticles), [films](https://www1.uwe.ac.uk/students/studysupport/studyskills/referencing/uwebristolharvard.aspx#films), [television programmes](https://www1.uwe.ac.uk/students/studysupport/studyskills/referencing/uwebristolharvard.aspx#television), [images and illustrations](https://www1.uwe.ac.uk/students/studysupport/studyskills/referencing/uwebristolharvard.aspx#imagesandillustrations), [official publications](https://www1.uwe.ac.uk/students/studysupport/studyskills/referencing/uwebristolharvard.aspx#officialpublications), [web pages](https://www1.uwe.ac.uk/students/studysupport/studyskills/referencing/uwebristolharvard.aspx#webpages).

Reflect: how and to what extent does the final stage of your practice connect with what it set out to do. You want to underline the achievements of the project here, but at the same time not hide any obvious omissions/ problems.

Reflect: What makes this project *original* in a professional sense; e.g. have you discovered any new methods as you went along? New models that might help others understand processes better? Perhaps you’ve developed a good protocol for implementing things? Streamlined some process?

**4. Conclusion and recommendations** 200

This section should not contain any new information. Here conclusions are drawn from the research and practical work that was completed. Does this project have a longer-term future beyond UWE and if so what might it look like?

Think about the impact it could be achieved in the wider field. How might it benefit users, professionals, society? Perhaps make some recommendations for further work.

**5. References** (=not included in word count – these are the sources you are actually quoting in this report; in alphabetical order)

Allan A, Schneider B and Miller C 2015 *Journal Name* **37** 074203

Bertrand J and Banes C 2009 *Journal Name* **23** 544

**Bibliography** (=not included in word count – these are other items you have read around this topic; in alphabetical order)  
Author, A. (2009) *A Book About Student Projects*. Location Publisher.

Author, B (2008) ‘Journal Article’, *Digital Media Journal*, Vol 1/13, pp 13-23

University of the West of England (2009) *UWE Library Services:Study skills - The Harvard System* [Online] Available from <https://www1.uwe.ac.uk/students/studysupport/studyskills/referencing/uwebristolharvard.aspx>

[last access: 23 September 2009]

<https://www.eteknix.com/id-software-chose-vulkan-directx-12/>

<https://youtu.be/P_I8an8jXuM>

[**https://youtu.be/xZQGgNEHTu8**](https://youtu.be/xZQGgNEHTu8)

<http://www.cemyuksel.com/research/lgh/lgh-high.pdf>

<http://www.cemyuksel.com/research/lgh/real-time_rendering_with_lgh_i3d2019.pdf>

**Appendix A: Project Log** (not included in word count)

**Appendix B: Project Timeline** (not included in word count)

**Appendix C: Title title title** (not included in word count)  
If necessary, insert further mateirals such as: a list of interview questions, any larger tables, evidence of design development, longer code snippets or other relevant materials in here or in further Appendixes. This does not come under the wordcount. Nevertheless, only insert useful materials here, please don’t just bulk this report up. Your main text should be able to stand on its own, without relying on information contained in appendixes!