UFCFHQ-45-3 Comprehensive Creative Technologies Project Proposal	
Student Name:	Liam Neale
Student Number:	18014341
Project Title:	Direct and Indirect Lighting using Raytracing

Description

Ray tracing is getting more and more popular as the capability of graphics hardware increases allowing for higher quality games whilst using more expensive methods to achieve this. There are many reasons for using ray casting in the computer graphics pipeline (Eric Haines 2020 basics of ray tracing) but one of them is to do with lighting. One reason rays are used other previous methods is to calculate the lighting of a scene when there are a lot of lights in a scene. This is known as Global Illumination and the methods for calculating the lighting are known as direct and indirect lighting as Scratch Pixel 2.0 explains in more detail.

I propose a DirectX12 implemented API to produce real time direct lighting with many light sources in the scene whilst using the ray tracing pipeline. Then I will need to denoise the resulting image of the ray tracing utilising one of the algorithms by (NvideaGameWorks (2018) Ray Tracing and Denoising) but this is a stretch goal.

A similar example of what I want to achieve is known as the Cornell box (Henrik Wan Jensen (2005)). Figure 2 is an example of this where you can test different methods of lighting and see the output in a scene. I will be testing something like this however I will use many light sources in my scene.

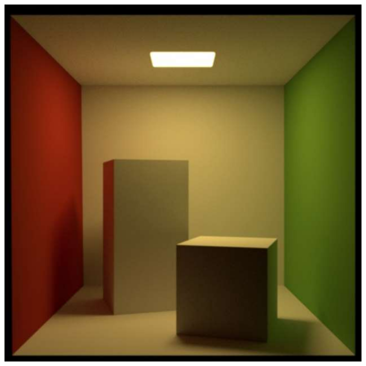


Figure 1

Research and background

The initial methods for calculating light in a scene was screen space reflection but as Josh Van Dogen (2020) explains the major drawback is you can’t reflect offscreen objects such as “using a mirror to look round a corner”. Shadow Mapping has aliasing problems and Surface Acne as explained in the Shadow Mapping PDF. precompiled light maps, image based light maps but with this method dynamic light failed to bounce or illuminate beyond the area it hit and (Eric Haines 2020 Denoising for Ray Tracing 4:38) explains the peter pan effect.

Nvidia have created their own API called DXR which uses RTX and DirectX 12 to render with the ray tracing pipeline. In Martin Stich (2018) blog you can see a Star Wars demo of real time reflections using RTX. I will be using their examples to help me along the way of possibly building my own API for Direct Lighting.

A more modern approach for calculating real-time direct lighting is called RESTIR which contains a reservoir of weighted random samples (Pavlos S. Efraimidis (2015)) and no complex data structures. RESTIR uses Monte Carlo integration to achieve direct lighting in a scene (Cline, Talbot and Egbert).

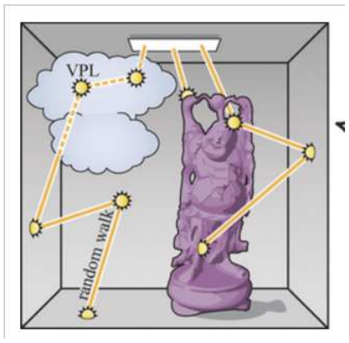
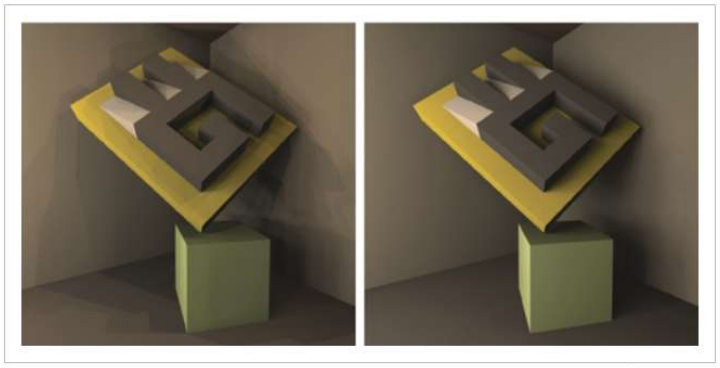


Figure 3

In Figure 3 it shows an example of a point light in the scene. There are rays shot out of it at random angles to determine different paths which determine how the point light affects the scene. Each ray bounce at a vertex will create a virtual point light (VPL) at that position. The information that is stored is the incident direction, surface normal and a reference to the scattering function.

According to Carsten Dachsbache (2013) using just this technique when you have a camera in the scene looking at a small part of the scene can lead to bad effects where most of the VPL's created could be out of the cameras view. One method to prevent this is to ignore VPLs which are not in view of the camera where all VPLs created have equal contribution to the image on the camera. Figure 4 shows an example of the normal VPL creation (left) and rejection VPL creation on the right. However, one downside to rejection VPL creation is that many VPLs may need to be created if the scene is complex before one is accepted but Dachsbache talks about replacing the VPL algorithm with a Metrophis Hasting Sampler to solve this.



Dachsbache continues to talk about way to produce VPLs inside the image by using Bi-direction VPLs are when you shoot rays from the camera point of view. However, these rays need to be connected to a light source to form complete light transport paths.

Objectives

Probably 3-4 objectives for each.

Project objectives

What is your project intending to achieve?

- A 3D scene using directx12 that is using direct lighting
- Uses the RESTIR algorithm
- Extension is to implement a denoiser

Research objectives

What do you want/need to find out? What area are you exploring or discovering?

- Research into implementing RESTIR for real-time global illumination
- Research into is and why raytracing replacing older rendering techniques
- Research into Monte Carlo integration:
    - VPL Method

Learning Objectives

What do you intend to learn from this project?

- Learn Directx12 render pipeline
- Learn the best debug techniques in Directx12
- Learn the RESTIR algorithm for calculating direct lighting
- Learn hardware limitations with ray count per frame

Methods, techniques, tools and processes

To manage my projects progress I will be using a software called Jira. I used this at my placement and is widely used in industry. This works by having goals and these goals will be completed by doing many small tasks. Each goal will have a deadline and some goals may depend on others. This then allows me to apply my time in the correct order.

The way I will complete these tasks is by adding them into weekly sprints. This allows me to regularly reflect on what I have achieved so far and gives me a break down using graphs to predict completion date. This will flag any problems or delays I have had or will have in the future which will allow me to manage my time appropriately.

I will use Git for version control to prevent loss of work as swell as preventing me from breaking my own code and not being able to revert to a working state. I will keep this in sync with Jira as there is a built-in functionality that allows you to track tasks with certain commits/branches inside Git.

To achieve this, I will be using a Visual Studio 2019 writing in c++ with the extension libraries of Directx12. GPU raytracing is not supported on all graphics cards so I will need to have the appropriate hardware for this project.

I will integrate the Restir algorithm (Restir 2020) for calculating direct lighting in the scene. Using a bi-directional light transport algorithm but firstly shooting rays from the light sources and storing the local environment of each ray bounce e.g., position, normal, incident direction. This is to calculate the outgoing illumination scattered from each vertex hit by a ray which is known as VPLs (Carsten Dachsbache (2013)). I will start off with the most basic VPL algorithm but can advance onto more complex ones.

Risks and issues

Risk	Mitigation	Contingency
Maths is to complex	Brush up on maths early on	Use math libraries
Don't have the correct hardware	Research what the minimum requirements are before project	Buy/rent correct hardware
Run out of time	User Jira for time management	Only do direct lighting not denoiser

Specialist resources and support required

None

Sources and references

Author, A. (2009) *A Book About Student Projects*. Publisher.
Author, B (2008) 'Journal Article', *Digital Media Journal*, 13, pp 13-23
University of the West of England (2009) *UWE Library Services:Study skills - The Harvard System* [Online] Available from http://www.uwe.ac.uk/library/resources/general/iskillzone/referencing/harvardreferencing/ [18 September 2009]

Josh Van Dogen (2020) "Screen Space Reflections in Blightbound"
https://www.gamedeveloper.com/disciplines/screen-space-reflections-in-blightbound

Nvidia (2016) Hybrid Frustum Traced Shadows
https://developer.nvidia.com/hybrid-frustum-traced-shadows-0

Shadow Mapping
https://research.ncl.ac.uk/game/mastersdegree/graphicsforgames/shadowmapping/Tutorial%2014%20-%20Shadow%20Mapping.pdf

Scratch Pixel 2.0 () Global Illumination and path Tracing
https://www.scratchapixel.com/lessons/3d-basic-rendering/global-illumination-path-tracing
Part 2
https://www.scratchapixel.com/lessons/3d-basic-rendering/global-illumination-path-tracing/global-illumination-path-tracing-practical-implementation

Nvidia DXR12 API Examples and Tutorials

https://developer.nvidia.com/rtx/raytracing/dxr/dx12-raytracing-tutorial-part-1

Scratch Pixel 2.0 () An Overview of the Ray-Tracing Rendering Technique https://www.scratchapixel.com/lessons/3d-basic-rendering/ray-tracing-overview/light-transport-ray-tracing-whitted

Martin Stich (2018) Introduction to Nvidia RTX and DirectX Ray Tracing https://developer.nvidia.com/blog/introduction-nvidia-rtx-directx-ray-tracing/

Matt Pharr, Wenzel Jakob, and Greg Humphreys (2004-2021)Physically Based Rendering (PBR) https://www.pbr-book.org/3ed-2018/Light_Transport_I_Surface_Reflection/Direct_Lighting
Figure 1 https://www.scratchapixel.com/lessons/3d-basic-rendering/rendering-3d-scene-overview/introduction-light-transport

Raimond Tunnel, Janu's Jaggo, Margus Luik, Study IT , Global Illumination https://cglearn.codelight.eu/pub/computer-graphics/global-illumination#

Restir, BENEDIKT BITTERLI, Dartmouth College CHRIS WYMAN, NVIDIA MATT PHARR, NVIDIA PETER SHIRLEY, NVIDIA AARON LEFOHN, NVIDIA WOJCIECH JAROSZ, Dartmouth College (2020)
Spatiotemporal reservoir resampling for real-time ray tracing with dynamic direct lighting
https://research.nvidia.com/sites/default/files/pubs/2020-07_Spatiotemporal-reservoir-resampling/ReSTIR.pdf

Henrik Wan Jensen (2005) Cornell Box AND Figure 1
http://graphics.ucsd.edu/~henrik/images/cbox.html

Eric Haines (2020) basics of ray tracing
https://youtu.be/gBPNO6ruevk

Eric Haines (2020) rendering equation
https://youtu.be/AODo_RjJoUA

Eric Haines Denoising for Ray Tracing (2020)
https://youtu.be/6O2B9BZiZjQ

NvideaGameWorks (2018) Ray Tracing and Denoising
https://www.youtube.com/watch?v=7uPLAC5uB8c

Pavlos S. Efraimidis (2015) Weighted Random Sampling over Data Streams
https://arxiv.org/pdf/1012.0256.pdf

Nvidia(2020) Spatiotemporal Importance Resampling for many Light Ray tracing
https://www.youtube.com/watch?v=HiSexy6eoy8

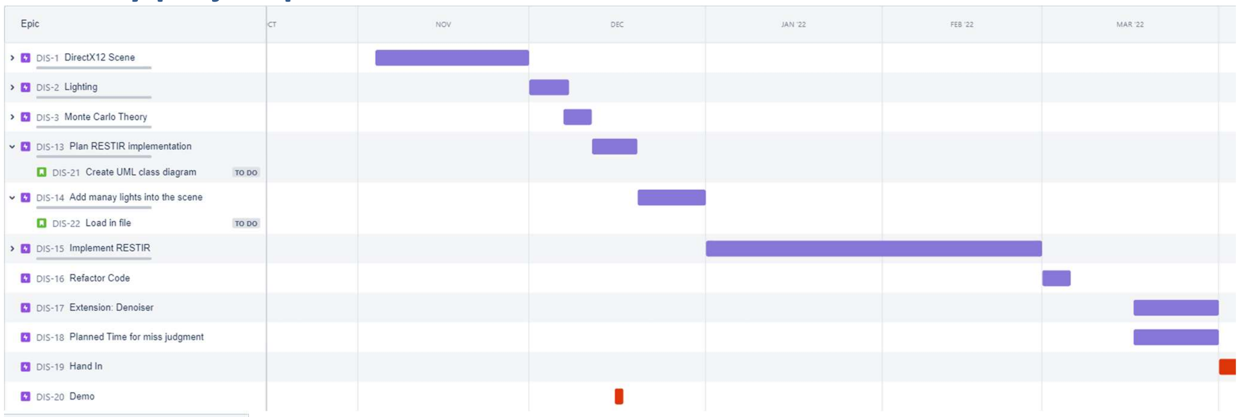
David Cline Justin Talbot Parris Egbert () Energy Redistribution Path Tracing And Figure 3 https://dl.acm.org/doi/pdf/10.1145/1186822.1073330

Kajiya (1986) The Rendering Equation
https://dl.acm.org/doi/pdf/10.1145/15886.15902
Eric Veach Leonidas J. Guibas (1997) Metropolis Light Transport
https://graphics.stanford.edu/papers/metro/metro.pdf

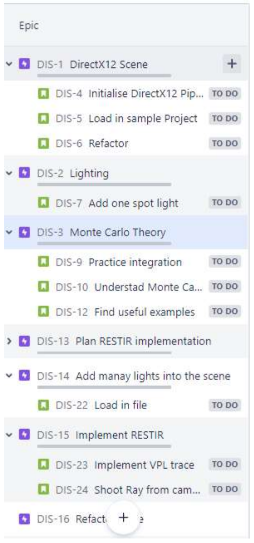
Figure 2 https://dl.acm.org/doi/pdf/10.1145/1186822.1073330

Carsten Dachsbache (2013) Scalable Realistic Rendering with Many-Light Methods
https://onlinelibrary.wiley.com/doi/10.1111/cgf.12256

Monthly project plan



Epic
DIS-1 DirectX12 Scene
DIS-2 Lighting
DIS-3 Monte Carlo Theory
DIS-13 Plan RESTIR implementation
DIS-21 Create UML class diagram TO DO
DIS-14 Add manay lights into the scene
DIS-22 Load in file TO DO
DIS-15 Implement RESTIR
DIS-16 Refactor Code
DIS-17 Extension: Denoiser
DIS-18 Planned Time for miss judgment
DIS-19 Hand In
DIS-20 Demo



Epic
DIS-1 DirectX12 Scene
DIS-4 Initialise DirectX12 Pip... TO DO
DIS-5 Load in sample Project TO DO
DIS-6 Refactor TO DO
DIS-2 Lighting
DIS-7 Add one spot light TO DO
DIS-3 Monte Carlo Theory
DIS-9 Practice integration TO DO
DIS-10 Understad Monte Ca... TO DO
DIS-12 Find useful examples TO DO
DIS-13 Plan RESTIR implementation
DIS-14 Add manay lights into the scene
DIS-22 Load in file TO DO
DIS-15 Implement RESTIR
DIS-23 Implement VPL trace TO DO
DIS-24 Shoot Ray from cam... TO DO
DIS-16 Refact e