D.A.F.T.
The Book of Specifications

Florian Laine
Angela Saade
David Wu
Timothy Edward Pearson

Contents

1 Introduction

1.1 Project presentation

Ray casting is a technique used to create 2D images and maps from 3D environments in computer graphics. This process is used to create a 2D representation of the 3D environment, and can be used for applications such as video games, architectural visualization, and robotic mapping. However, creating efficient and accurate ray casting algorithms can be challenging. This paper presents the design and implementation of the DAFT Engine Library, a tool that aims to simplify the process of creating ray casting simulations and improve their performance. The library is designed to be highly easy to use and integrate into existing systems. The DAFT Engine Library is an open-source project that can be freely used and modified by anyone. The purpose of this synopsis is to provide an overview of the library's features, design, and implementation.

1.2 Group presentation

1.2.1 Timothy Edward Pearson

I am really looking forward to this project and I think that it will serve very useful in the future. Furthermore, building a 3D engine relies partially on vectors and linear algebra which we have done in maths class so it will be fun to apply these subjects to a real world situation. We intend to add some physics to the objects in our world if we have time which I am very excited about. I am eager to learn more physics application to my work at EPITA projects and better my knowledge in C programming.

1.2.2 Florian Laine

I'm really eager to start this project ! This 3D engine is a great opportunity of deeply understanding a lot of math and algorithm optimization at the same time. I introduced this project idea to my team mates and they immediately seemed to agreed. I personally am truly into the 3D processing since a few months ago and am super excited to start ! I have been doing some test on my own up to now but I am thrilled to take it to the next level by getting involved as part of a group work. I want to help as much as possible considering my few previous experiences and can feel a bit delighted about it. I obviously am very enthusiastic improving my skills and knowledge by implementing a whole new architecture and discovering new ways of solving the issues I can already have faced. Finally, this project represent for me a huge opportunity to accomplish one of my child dream I had when I was wondering how 3D graphics are working while playing some of my favorite games.

1.2.3 Angela Saade

I am excited to begin this project and see the potential it holds for me to gain a deeper understanding of 3D processing. This project is a great opportunity to learn, improve my skills by being part of this group. Even though I have had little to no experience with this subject and working in a team can be challenging sometimes, but I am determined to help and contribute as much as possible. It is a new type of project for me; building approximately everything from scratch, and it motivates me to take on this challenge.

1.2.4 David Wu

Being a novice at programming, this type of project seemed to be a real challenge for me. Indeed, I've never looked into such a project before : it includes loads of mathematical notions but also a great algorithm interest. To some extent, I am excited to begin the realization of this project, since I truly think that it can bring me some knowledge and skills that I need for my future engineer career. I know that I have teammates that I can count on, who will be able to help me if I struggle. Moreover, since I have been playing video games for several years, I have always been interested in 3D engines even if I have never had the opportunity to study the subject. I hope this project will make me discover a new exciting field of computer sciences.

2 Origin and Type of project

2.1 Project's Origins

The idea of making a 3D engine came from one of us. Following consultation, we were all up to realize this project. Since each member of our group had already manipulated 3D engine such as Unity3D in the context of the S2 project, we could see which expectations were required for such a project. Moreover, we know that 3D engines are nowadays useful in several domain, such as video game conception, movie realization, but also in scientific fields, in order to simulate complex situations in a realistic 3D environment.

Afterwards, we found that this project could be a real challenge for us, since it includes a deep reflection on how to optimize our algorithms, this reflection being one of the foundations of today's programming. Furthermore, this project requires great understanding of 3D geometrical mathematics, what really made us want to look into it to improve our skills in this domain. Indeed, even if we have already manipulated software including 3D engine, most of us had never thought about how these engines were made.

2.2 Type of Project

As said before, our goal is to make our own 3D engine. This project will be entirely made in the C programming language. As a goal, our 3D engine will be able to generate 2D images from a 3D environment. The quality of our results will rely heavily on our ability to optimize the calculations given to the computer. Indeed, it exists several ways to make 3D engine nowadays. We chose the Ray Casting process, which will be explained later in the specification book. At the end, we would like the user to be able to place objects in a 3D environment, making it as realistic as possible. Therefore, we will have to implement functions that will be presented as "tools" for the user.

Also, our program will aim at returning the most realistic images as possible, (processing at 30 frames per second at minimum) which is equivalent to generate a smooth 3D video stream in real time. Finally, we want our project to include a whole API part, in order to make it easy handling for those who want to generate simple 3D videos. Indeed, we aim to make it easy for users to understand the different features implemented, in order to deliver the best possible user experience.

3 State of the Art

3.1 Simulate ray

Ray casting and ray tracing are both techniques used in computer graphics to simulate the behavior of light, but they are based on different principles and are used for different purposes.
Ray casting is a simpler and more efficient technique used for rendering 3D scenes from a 2D perspective. It works by casting rays from the camera, or viewpoint, into the scene and determining the closest object that the ray intersects. The color and other properties of that object are then used to determine the color of the pixel in the final image. Ray casting is mainly used for creating 3D games and real-time applications, it is fast and efficient but less accurate.
Ray tracing, on the other hand, is a more complex and computationally expensive technique used for creating highly realistic images. It works by tracing the path of light as it bounces off of and passes through objects in a scene. This allows for accurate simulations of reflections, refractions, and shadows, as well as global illumination and ambient occlusion. Ray tracing is mainly used for offline rendering, animation, and architectural visualization, it is accurate but takes more time to render.
In summary, ray casting is a fast, efficient method for creating 3D images from a 2D perspective, while ray tracing is a more accurate and computationally expensive method for creating highly realistic images.

3.1.1 Ray Tracing

Ray tracing is a technique used in computer graphics to generate images by simulating the behavior of light. It works by tracing the path of light as it travels through a 3D scene, taking into account the properties of the objects in the scene and how the light interacts with them.
The process of ray tracing starts by casting a ray from the virtual camera, or viewpoint, into the scene. The ray is tested for intersection with any objects in the scene, if an intersection is found, the algorithm calculates the color of the pixel based on the surface properties of the object, such as its texture, reflectivity, refractive index and so on.
It also takes into account other lighting conditions in the scene, such as the position and intensity of light sources, as well as the properties of the surrounding environment. This allows for the simulation of realistic reflections, refractions, and shadows, as well as global illumination and ambient occlusion.
Ray tracing is a more computationally intensive technique than other rendering methods, such as rasterization, but it can produce highly realistic images with accurate lighting and shading. It is mainly used in offline rendering, animation, and architectural visualization, and increasingly used in real-time applications such as video games and virtual reality.
Ray tracing is used in a variety of fields and industries. Some examples include:

- Film and animation: Many high-budget films and animated movies use ray tracing to create photorealistic visual effects.
- Gaming: Some video games, such as the popular game Minecraft RTX, use ray tracing to create more realistic lighting and shadows.
- Architecture and design: Architects and designers use ray tracing to create realistic visualizations of buildings and other structures, to help clients understand what the final product will look like.
- Automotive design: Automotive companies use ray tracing to visualize and test the aerodynamics of car designs.
- Medicine: Medical researchers use ray tracing to simulate the behavior of light in the human eye and other biological systems.
- Scientific visualization: Researchers in fields such as astrophysics and atmospheric science use ray tracing to create accurate visualizations of complex data.
- Virtual Reality: Ray tracing is used in Virtual Reality to create realistic and immersive experiences.

These are just a few examples, but ray tracing is used in many other fields as well.

3.1.2 Ray Casting

Ray casting is a technique used in computer graphics to generate images by tracing rays from the viewpoint or camera into the scene. It is used to determine which object in the scene is visible to the viewer and to calculate the color of each pixel.
The process of ray casting starts by casting a ray from the virtual camera, or viewpoint, into the scene. The ray is tested for intersection with any objects in the scene, if an intersection is found, the algorithm calculates the color of the pixel based on the surface properties of the object, such as its texture, reflectivity, refractive index and so on.
Unlike Ray tracing, Ray casting does not take into account more complex lighting conditions and interactions such as reflections, refractions, and global illumination. Therefore, the images generated by ray casting are generally less realistic and detailed than those generated by ray tracing.
Ray casting is a simpler and more efficient technique than ray tracing, which makes it suitable for real-time applications such as video games and interactive simulations. It is used to create 3D games and real-time applications, it is fast and efficient but less accurate than ray tracing.
Ray casting is used in a variety of fields and industries, some examples include:

- Gaming: Ray casting is widely used in video games to quickly and efficiently render 3D environments and characters. It is particularly useful for first-person shooters and other fast-paced games, as it can quickly determine which objects are visible to the player and calculate the appropriate colors and textures.
- Virtual Reality: Ray casting is used in VR and AR applications to quickly render 3D scenes and objects, allowing for smooth and responsive interactions.
- Medical imaging: Ray casting is used in medical imaging to create 3D visualizations of internal organs, bones, and other structures, making it easier to diagnose and treat various medical conditions.
- Robotics: Ray casting is used in robotics to map and navigate environments, allowing robots to safely and efficiently move around in the real world.
- Industrial design: Ray casting is used in industrial design to create 3D visualizations of products and machinery, allowing designers to test and refine their designs before they are built.
- Computer-aided design: Ray casting is used in CAD programs to quickly render 3D models and visualizing them.

These are just a few examples, but ray casting is used in many other fields as well.

3.2 Rasterization

Rasterization is a technique used in computer graphics to convert vector graphics and 3D models into a 2D image representation. It works by taking a 3D scene and dividing it into a series of small, rectangular pixels, called fragments or samples. The properties of each fragment, such as its color, depth, and texture, are determined by processing the scene's geometric data, and the final image is created by assembling the colored fragments on a 2D surface.
The process of rasterization starts by transforming the 3D coordinates of each vertex in the scene into 2D screen coordinates. Then, the fragments are generated by filling the area of the triangle formed by the three vertices of the 3D object. The fragments generated in this way are called "primitives". The color and other properties of each primitive are determined by applying textures, lighting, and shading.
Rasterization is a fast and efficient rendering technique, which makes it suitable for real-time applications such as video games, interactive simulations, and virtual reality. It is widely used in computer graphics because it is relatively simple and can be implemented on a wide variety of hardware platforms.
It's important to note that rasterization is not as accurate as ray tracing or ray casting, it is more suitable for real-time applications because of its speed, but images generated using rasterization are less realistic and detailed.

4 Goal and interest of the project

Creating a ray casting engine will allow us to delve deeper into the mathematical and computational principles of 3D graphics. We will gain a comprehensive understanding of how to simulate the behavior of rays in a 3D space, including how to determine the intersection of rays with 3D objects. This knowledge will be invaluable in understanding the fundamental principles of 3D graphics, which can be applied in a wide range of fields. Additionally, the process of creating a ray casting engine will give us hands-on experience in implementing and optimizing the algorithms used in 3D graphics, further strengthening our understanding of the underlying principles.

The project we are embarking on will be an exciting opportunity to not only create a dynamic 3D rendering engine, but also to develop our programming skills in the process. One of the key components of the project will be the implementation of the ray casting algorithm, which is used to generate the images we see on screen. This will require a deep understanding of the mathematics and physics behind the algorithm, as well as the ability to optimize it for real-time performance.
Additionally, we will be implementing backface culling and spatial partitioning techniques, which are essential for improving the performance and efficiency of the engine. These techniques will help us to optimize the rendering process by reducing the number of unnecessary calculations and improving the overall performance of the engine. Overall, this project will be a challenging and rewarding experience that will help us to become better programmers and gain a deeper understanding of 3D graphics.

As a group, we are highly interested in game development and creating a ray casting engine is a perfect opportunity for us to gain hands-on experience in real-time 3D game development. Through the process of creating this engine, we will gain a deeper understanding of the intricacies and challenges of game development, and will be able to apply this knowledge to our future endeavors. In addition to gaining valuable technical skills, we will also have the opportunity to develop our problem-solving and teamwork skills as we work together to create a functional engine.
Furthermore, once the engine is completed, we will have the potential to use it to create our own simple game. This would be an exciting opportunity to apply our knowledge and skills in a practical setting, and to see the fruits of our labor come to life in the form of a fully-functioning game. Additionally, this will give us a chance to test and optimize the engine in real-world scenarios, and identify any potential areas for improvement. Overall, creating a ray casting engine is not only a great learning opportunity, but it also has the potential to be the starting point of our own game development journey.

Finally, creating a 3D engine is challenging, we will need to debug and optimize the code, and troubleshoot any issues that arise. This will help us develop problem-solving skills and gain experience working as a team.

5 Task distribution

5.1 Website

The website is closely monitoring the progress of the project and is integral to the success of the project. Therefore, in the initial stages of development, we will focus on the architecture and overall aesthetic of the website. Our main objective is to create a website that is visually appealing, easy to navigate and can adapt to changes as the project evolves. This aspect of the project is crucial for the website to be able to keep up with the progress of the project and to provide an easy way for people to stay informed about the progress. The website's architecture must be suitable for the purpose, easy to update and maintain, and scalable to accommodate future changes. By focusing on these elements in the early stages of development, we can ensure that the website is able to evolve with the project and provide valuable information to the audience.

5.2 Raycasting

It is evident that raycasting is the foundation of this project and it is imperative that it is implemented swiftly. Our goal is to produce a functional basic raycasting algorithm that generates output in a timely manner as a "proof of concept." This will provide a solid foundation to build upon and demonstrate the feasibility of the project. Therefore, our immediate focus should be on developing a basic raycasting algorithm that is functional and can be used as a starting point for further development.

5.3 Optimization

Optimization will play a vital role in the success of the project once the raycasting component is fully developed. As a real-time engine, each frame must be processed within a few milliseconds to ensure smooth performance. However, it is essential to wait until the raycasting component is fully functional and stable before beginning optimization efforts. This is because optimization should not be implemented until the core functionality of the project is established and any potential issues with the raycasting have been addressed. By waiting to tackle optimization as a separate step, we can ensure that the optimization efforts are focused on fine-tuning a stable and functional system, rather than trying to fix fundamental issues with the core functionality.

5.4 OBJ files

A .obj file is a simple text file format for representing 3D geometry. Manipulating .obj files is an important task for our project. We will mainly read the file, extract the relevant information which is basically coordinates written in different formats. This data will be then organized into a usable format; identfying texture coordinates, vertices and edges of the object. Once the data has been organized the next step is to create objects and textures from the collected information.

5.5 Physics

To add more to this project, we will add physics to the objects allowing them to interact with one another. This will mainly include objects bouncing off one another after being placed in the air. This will require us to manage the position and rotation of objects matching the frame rate with the speed that the objects move to make it as realistic as possible.

5.6 Utils

A utility file is needed to perform each of the procedures such as finding the collision point of a ray with a mesh and then finding the reflected direction that the ray must continue in. Furthermore, we will need some basic matrix and vector functions in order to find the normals of triangles in a mesh. in terms of architecture, we need need various structures so that we can store and manage and reference objects and raycasts in our world.

5.7 Textures

Rendering textures in the 3D environment is one of our goals. We will mainly take the retrieved and organized data from the .obj file and render textures and objects in the 3D environment.

6 Division of tasks

6.1 Task sharing

Tasks	Florian	Angela	David	Timothy
Website				
Ray Casting				
Optimisation				
OBJ Files				
Physics				
Utils				
Textures				

Leader **Substitute**

6.2 Advancement in view of defenses

Advancement in percentage			
Tasks	Defense 1	Defense 2	Defense 3
Website	30%	60%	100%
Raycasting	50%	70%	100%
Optimisation	30%	50%	100%
OBJ files	80%	100%	100%
Physics	0%	40%	100%
Utils	40%	60%	100%
Textures	0%	60%	100%

7 Conclusion

In conclusion, our project aims to create a 3D engine that utilizes ray casting to create realistic and detailed images and animations. Ray casting allows us to define the correct colors for the objects in our virtual world and represents a simple and efficient way to create 2D images and maps from 3D environments. By the final defense, we aim to have a fully functional 3D engine that can be used for a wide range of applications such as video games, architectural visualization, and robotic mapping. We believe that the use of ray casting in our engine will be an efficient and powerful tool for creating realistic images and animations, and we look forward to showcasing the capabilities of our engine.