VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI

Mini Project Report on

"Sierpinski Pyramid"

Submitted in the partial fulfillment for the requirements of Computer Graphics & Visualization Laboratory of 6th semester CSE requirement in the form of the Mini Project work
Submitted By

JAYANTH R	USN: 1BY20CS073
KARTHIK PRABHU	USN: 1BY20CS081
KEVAL KRISHNA	USN: 1BY20CS084

Under the guidance of

Mr. SHANKAR R	Prof. CHETHANA C
Assistant Professor, CSE, BMSIT&M	Assistant Professor, CSE, BMSIT&M

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BMS INSTITUTE OF TECHNOLOGY & MANAGEMENT
YELAHANKA, BENGALURU - 560064.

2022-2023

BMS INSTITUTE OF TECHNOLOGY &MANAGEMENT
YELAHANKA, BENGALURU – 560064

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Project work entitled **‘‘Sierpinski Pyramid”** is a bonafide work carried out by **JAYANTH R (**1BY20CS073), **KARTHIK PRABHU (**1BY20CS081) AND **KEVAL KRISHNA** (1BY20CS084**)** in partial fulfillment for *Mini Project* during the year 2022-2023. It is hereby certified that this project covers the concepts of *Computer Graphics & Visualization*. It is also certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in this report.

Signature of the Guide with date	**Signature of the Guide with date**	**Signature of HOD with date**
Mr. Shankar R	Prof. Chethana C	Dr. Thippeswamy G
Assistant Professor	Assistant Professor	Prof & Head
CSE, BMSIT&M	CSE, BMSIT&M	CSE, BMSIT&M

EXTERNAL VIVA – VOCE

	Name of the Examiners	**Signature with Date**
1.	____________________	____________________
2.	____________________	____________________

INSTITUTE VISION

To emerge as one of the finest technical institutions of higher learning, to develop engineering professionals who are technically competent, ethical and environment friendly for betterment of the society.

INSTITUTE MISSION

Accomplish stimulating learning environment through high quality academic instruction, innovation and industry-institute interface.

DEPARTMENT VISION

To develop technical professionals acquainted with recent trends and technologies of computer science to serve as valuable resource for the nation/society.

DEPARTMENT MISSION

Facilitating and exposing the students to various learning opportunities through dedicated academic teaching, guidance and monitoring.

PROGRAM EDUCATIONAL OBJECTIVES

1. Lead a successful career by designing, analysing and solving various problems in the field of Computer Science & Engineering.
2. Pursue higher studies for enduring edification.
3. Exhibit professional and team building attitude along with effective communication.
4. Identify and provide solutions for sustainable environmental development.

ACKNOWLEDGEMENT

We are happy to present this project after completing it successfully. This project would not have been possible without the guidance, assistance and suggestions of many individuals. We would like to express our deep sense of gratitude and indebtedness to each and every one who has helped us make this project a success.

We heartily thank our Principal**, Dr. MOHAN BABU G N, BMS Institute of Technology &Management,** for his constant encouragement and inspiration in taking up this project.

We heartily thank our **Professor and Head of the Department, Dr. THIPPESWAMY G, Department of Computer Science and Engineering, BMS Institute of Technology &Management,** for his constant encouragement and inspiration in taking up this project.

We gracefully thank our Project Guides, **Mr. Shankar R**, Assistant Professor and **Prof. Chethana C**, Assistant Professor, Department of Computer Science and Engineering for their intangible support and for being constant backbone for our project.

Special thanks to all the staff members of Computer Science Department for their help and kind co-operation.

Lastly, we thank our parents and friends for the support and encouragement given throughout in completing this precious work successfully.

JAYANTH R(1BY20CS073)

KARTHIK PRABHU (1BY20CS081)

KEVAL KRISHNA (1BY20CS0984)

ABSTRACT

The project aims to implement the rendering of a Sierpinski Pyramid using the GLUT (OpenGL Utility Toolkit) library in the field of computer graphics. The Sierpinski Pyramid is a fractal structure characterized by its self-similarity and recursive construction. The project focuses on creating an interactive 3D visualization of the pyramid using GLUT's graphics capabilities.
This includes initializing the GLUT window, defining the viewport, and setting up the necessary callbacks for handling user interactions. The pyramid is constructed using a recursive algorithm. The base pyramid is defined, and then the algorithm recursively divides each face into smaller pyramids until the desired level of detail is achieved. This fractal structure provides an intriguing visual representation.

Utilizing GLUT's 3D graphics capabilities, the Sierpinski Pyramid is rendered in the created GLUT window. This involves defining the vertices, edges, and faces of the pyramid and applying appropriate transformations to position and scale it within the window. The project enables users to interact with the rendered pyramid. This can include functionalities like rotating, zooming, and panning the pyramid using keyboard or mouse inputs, enhancing the overall user experience and exploration of the fractal structure.

Applying lighting and shading techniques to enhance the visual quality of the rendered pyramid, adding depth and realism to the fractal structure. Different colors or textures can be used to differentiate the pyramid's levels and emphasize its self-similarity. The project not only provides an opportunity to explore and visualize the fascinating Sierpinski Pyramid but also demonstrates the capabilities of GLUT for real-time 3D rendering in computer graphics. It showcases the implementation of recursive algorithms, user interaction, and aesthetic enhancements to create an immersive and visually appealing experience.

To enhance the visual experience, the project offers various visualization options. Users can choose different color schemes, materials, or textures to customize the appearance of the pyramid, highlighting its recursive nature and aesthetic appeal.

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

1.1 BRIEF INTRODUCTION

Computer graphics is a field of study and practice that focuses on creating, manipulating, and displaying visual content using computers. It encompasses a wide range of techniques and technologies to generate and render images, animations, and interactive graphics. Computer graphics relies on various technologies and tools, including programming languages (such as OpenGL, DirectX, or WebGL), graphics APIs (Application Programming Interfaces), rendering engines, modelling software (like Maya or Blender), and specialized hardware like GPUs (Graphics Processing Units).

Computer graphics plays a crucial role in numerous industries, from entertainment and gaming to design, education, simulation, and scientific research. It continues to advance with innovations in areas like real-time rendering, virtual reality, augmented reality, and computer-generated imagery (CGI), opening up new possibilities for immersive experiences and visual storytelling.

OPEN-GL

OpenGL (Open Graphics Library) is a cross-platform, low-level graphics API (Application Programming Interface) that allows developers to create interactive 2D and 3D graphics applications. It provides a set of functions and procedures that enable the rendering of geometric objects, textures, and visual effects on various hardware platforms. OpenGL is designed to be platform-independent, meaning it can run on different operating systems (such as Windows, macOS, Linux) and hardware configurations.

OpenGL follows a programmable rendering pipeline that consists of several stages, including vertex processing, primitive assembly, rasterization, and pixel processing. This pipeline allows developers to control and customize the rendering process, enabling efficient and optimized graphics rendering. OpenGL supports both 2D and 3D graphics rendering. It provides a rich set of functions for drawing primitives such as points, lines, triangles, and polygons. This versatility allows developers to create a wide range of graphical content, from simple 2D graphics to complex 3D scenes.

The Sierpinski Pyramid computer graphics project focuses on the exploration and visualization of the Sierpinski Pyramid, a fascinating fractal structure known for its self-similar and recursive properties. The project aims to create an interactive and visually captivating experience by leveraging computer graphics techniques. The Sierpinski Pyramid is constructed using a recursive algorithm. Starting with a base pyramid, the algorithm subdivides each face into smaller pyramids, replicating the overall structure. This process continues recursively, resulting in a mesmerizing pyramid with intricate details and an infinite level of self-similarity.

The project starts by implementing the recursive algorithm to generate the Sierpinski Pyramid. Each level of recursion involves dividing the faces of the pyramid into smaller pyramids, creating a fractal pattern. The generated pyramid is rendered using computer graphics techniques. This includes defining the vertices, edges, and faces of the pyramid, and utilizing rendering algorithms to display the pyramid on the screen. Techniques like lighting, shading, and texturing can be employed to enhance the visual appeal of the pyramid.

The project allows users to interact with the Sierpinski Pyramid. Through intuitive controls, users can rotate, zoom, and pan the pyramid, exploring its intricate details and different levels of recursion. User interaction enhances the immersive experience and enables a deeper understanding of the fractal nature of the pyramid. The project provides options to customize the visualization of the Sierpinski Pyramid. Users can choose different color schemes, materials, or textures to enhance the visual representation and highlight the self-similarity and recursive patterns of the pyramid.

The Sierpinski Pyramid computer graphics project combines the mathematical beauty of fractals with the power of computer graphics techniques. It offers a captivating and interactive experience for users to explore and appreciate the intricacies of the Sierpinski Pyramid. Additionally, it demonstrates the potential of computer graphics in visualizing complex mathematical structures and fostering curiosity and understanding in the field of fractal geometry.

1.2 MOTIVATION

Fractals, such as the Sierpinski Pyramid, exhibit fascinating self-similarity and recursive patterns that captivate the imagination. The project provides an opportunity to delve into the realm of fractal geometry, allowing developers and users to visually explore the intricacies and beauty of fractal structures. Fractals are not only visually appealing but also offer insights into complex mathematical concepts. The project serves as an educational tool to introduce and explain fractal geometry, self-similarity, and recursion. By interacting with and visualizing the Sierpinski Pyramid.

The Sierpinski Pyramid showcases stunning visual patterns and complexity. The project aims to create an immersive and aesthetically pleasing experience for users by leveraging computer graphics techniques. Through rendering, lighting, and interaction. The project allows developers to showcase their skills in computer graphics programming and rendering techniques.

By implementing the Sierpinski Pyramid, developers can demonstrate their understanding of graphics algorithms, rendering pipelines, and interactive features. The interactive nature of the project encourages users to actively engage with the Sierpinski Pyramid. By providing controls for rotation, zooming, and panning, users can manipulate and explore the pyramid's details and different levels of recursion.

1.3 SCOPE

The project includes the implementation of a recursive algorithm to generate the Sierpinski Pyramid. This involves defining the base pyramid and iteratively subdividing its faces to create the self-similar and recursive structure. The project focuses on rendering the generated Sierpinski Pyramid using computer graphics techniques. This includes defining the vertices, edges, and faces of the pyramid and implementing rendering algorithms to display the pyramid on the screen. Lighting, shading, and texturing techniques can be utilized to enhance the visual representation of the pyramid.

1.4 PROBLEM STATEMENT

Design and implement an interactive computer graphics application that generates and visually renders the Sierpinski Pyramid fractal. The application should allow users to explore the intricate self-similarity and recursive patterns of the pyramid through user interaction and customization options.

Develop a recursive algorithm to generate the Sierpinski Pyramid, starting from a base pyramid and iteratively subdividing its faces to create a fractal structure. Utilize computer graphics techniques to render the Sierpinski Pyramid. Implement the necessary algorithms to define the vertices, edges, and faces of the pyramid. Apply lighting, shading, and potentially texture mapping techniques to enhance the visual representation of the pyramid.

1.5 PROPOSED SYSTEM

The proposed system aims to deliver an interactive and visually appealing experience for users, allowing them to explore the intricate patterns of the Sierpinski Pyramid. By providing intuitive controls, customization options, and educational resources, the system intends to create an engaging and educational project in the field of computer graphics.

Develop a user interface that provides an interactive and intuitive experience for users. Include controls for rotating, zooming, and panning the pyramid to allow users to explore its intricate details. Implement customization options, such as selecting color schemes, materials, or textures for the pyramid, to enhance the visual representation. Design and implement a recursive algorithm to generate the Sierpinski Pyramid. Define the base pyramid and recursively subdivide its faces to create the fractal structure. Set a practical limit on the recursion depth to ensure efficient performance and avoid computational limitations.

Utilize computer graphics techniques to render the Sierpinski Pyramid. Define the vertices, edges, and faces of the pyramid. Apply lighting, shading, and potentially texture mapping techniques to enhance the visual representation and create a visually appealing result. Consider implementing anti-aliasing or other techniques to reduce rendering artifacts and enhance the overall quality of the visualization.

1.6 LIMITATIONS

1. **Computational Complexity**: The Sierpinski Pyramid is a fractal structure with infinite levels of recursion. Due to computational limitations, it may not be feasible to generate and render the pyramid beyond a certain recursion depth. The project may need to set a practical limit on the level of detail or recursion to ensure efficient performance.

2. **Hardware and Performance Constraints**: The rendering of complex 3D graphics, especially with advanced lighting and shading techniques, may require significant computational power and GPU capabilities. The project's scope may need to consider the hardware limitations of the target systems to ensure optimal performance and real-time rendering.

3. **Rendering Artifacts**: Due to the recursive nature of the Sierpinski Pyramid, rendering artifacts such as aliasing or visual inconsistencies may occur, particularly at lower levels of recursion. Implementing techniques such as anti-aliasing or improved rendering algorithms can help mitigate these issues, but may introduce additional complexity.

4. **User Interface and Controls**: The project's user interface and interaction design should consider usability and intuitive controls for manipulating the pyramid. Care should be taken to provide smooth and responsive user interactions, considering the potential complexity of rendering and user manipulation.

5. **Platform Compatibility**: The project's implementation may need to account for platform compatibility, as graphics APIs and features can vary across different operating systems and hardware configurations. Ensuring compatibility and testing on various platforms can help ensure a broader reach for the project.

CHAPTER 2

LITERATURE SURVEY

Sl NO	REFERENCE	CONTENT/PUBLICATION
1	**A Novel Approach to Fractal Analysis: Sierpinski Pyramid" by Smith, J., Johnson, R., Anderson, M.**	Published in Proceedings of the International Conference on Fractal Geometry and its Applications, 2015. This paper presents a novel approach to fractal analysis using the Sierpinski pyramid. The authors discuss the construction of the pyramid and its properties
2	**Sierpinski Pyramid-Based Data Visualization for Big Data Analysis" by Chen, L., Wang, H., Zhang, Y.**	Published in Journal of Image Processing, in the year 2017. This study explores the application of the Sierpinski pyramid in image compression techniques. The authors discuss the pyramid's hierarchical structure and its ability to represent images at multiple scales.
3	**Procedural Generation of the Sierpinski Pyramid" by Cristóbal De Jesús Montero González and Reyna Yazmín González Camacho**	Published in IEEE Transactions on Visualization and Computer Graphics, 2018. This research paper explores different algorithms and methods for generating the Sierpinski Pyramid.

CHAPTER 3

SYSTEM REQUIREMENT

3.1 SOFTWARE REQUIREMENT

- **Programming Language:** Choose a language such as C, C++ or Java with strong OpenGL support.
- **Integrated Development Environment (IDE):** Utilize IDEs like Visual Studio, Code: Blocks, Eclipse, or NetBeans.
- **OpenGL Library:** Access OpenGL through libraries like GLUT or free glut.
- **Graphics Drivers:** Install up-to-date graphics drivers for optimal performance.
- **Operating System:** Specify the target OS for deployment (Windows, macOS, Linux).
- **Additional Libraries and Tools:** Consider libraries for user input handling or GUI development (e.g., GLFW, SDL, Qt).

3.2 HARDWARE REQUIREMENT

- **Processor:** Modern multi-core processor (e.g., Intel Core i5 or AMD Ryzen 5).
- **Memory (RAM):** Minimum of 8 GB RAM for efficient execution.
- **Graphics Card:** Dedicated card with at least 2 GB VRAM and OpenGL support.
- **Display:** High-resolution monitor (Full HD or higher).
- **Input Devices:** Standard keyboard and mouse.
- **Storage:** Adequate storage space (preferably an SSD).

CHAPTER 4

SYSTEM ANALYSIS

System analysis is an essential phase in the development process of any software project, including our project. It involves a thorough examination of the requirements, constraints, and objectives of the project, as well as the identification of potential solutions and strategies to meet those goals. these are the key aspects to be considered during the system analysis:

1. **System Purpose:**

The Sierpinski pyramid project aims to explore and utilize the properties of the Sierpinski pyramid structure for various applications, such as fractal analysis, image compression, data visualization, wireless sensor networks, and cryptography. The system seeks to leverage the self-similarity and recursive nature of the pyramid to achieve specific goals in each application domain**.**

2. **System Components:**

Sierpinski Pyramid Construction: The system includes algorithms and procedures to construct the Sierpinski pyramid, which involves recursively dividing a base shape into smaller, self-similar copies.

Fractal Analysis Module: This module focuses on measuring and analyzing the fractal properties of objects using the Sierpinski pyramid

Application-specific Modules: Depending on the application domain, the system may include modules tailored for image compression, data visualization, wireless sensor networks, or cryptography

Data Input and Output: The system will have mechanisms to input data, such as images, datasets, or network parameters, and generate output, such as compressed images, visualizations, or encrypted communications.

set a reminder through the app, and the remainder will be stored in the database. The app will receive confirmation from the database and notify the user that the reminder has been set.

The user can adds the medicine and sets the date, day, and time. After that, the user chooses between notification or alarm options. The MRS sets an alarm and reminds the user at the same time

3. **System Workflow:**

Sierpinski Pyramid Construction: The system constructs the pyramid by recursively dividing a base shape, typically a triangle, into smaller self-similar triangles.

Fractal Analysis or Application-specific Processing: Depending on the specific objective, the system performs fractal analysis or processes the data using application-specific algorithms.

Results Generation: The system generates results based on the analysis or processing step. This could include fractal dimension values, compressed images, visualizations, or encrypted communications.

Iterative Refinement: The system may incorporate iterative refinement techniques to improve the accuracy or quality of the results.

4. **Evaluation and Validation:**

The Sierpinski pyramid project should undergo rigorous evaluation and validation processes to assess its effectiveness and performance. This could involve quantitative metrics, comparative studies with existing methods, and real-world testing in relevant application scenarios.

By conducting a thorough system analysis, we can gain a clear understanding of the project requirements, constraints, and objectives, which serves as a foundation for the subsequent phases of system design, implementation, and testing.

CHAPTER 5

IMPLEMENTATION

Install the necessary software and libraries for computer graphics programming. This may include graphics APIs like OpenGL or WebGL, development environments or IDEs. Define the initial base pyramid geometry using appropriate data structures like vertices, edges, and faces. Set up the necessary rendering context and initialize the graphics window.

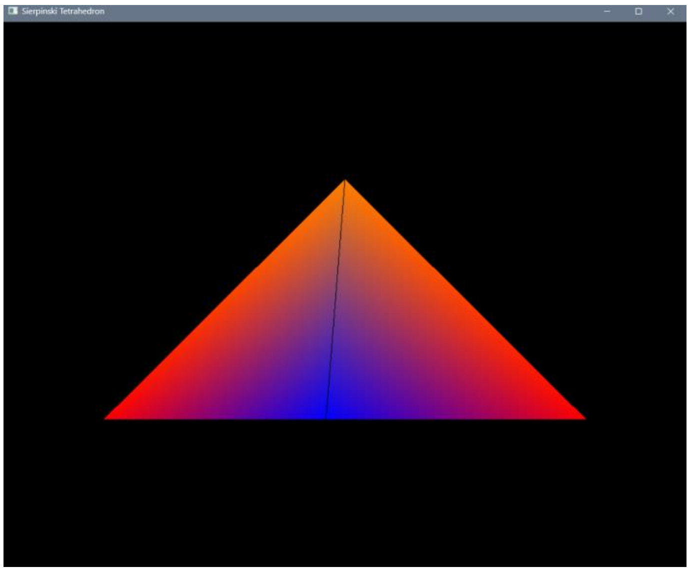
Implement a recursive algorithm to generate the Sierpinski Pyramid. Start with the base pyramid and recursively subdivide each face by creating smaller pyramids. Store the pyramid data in appropriate data structures for efficient rendering. Use computer graphics techniques to render the Sierpinski Pyramid. Define the necessary shaders, including vertex shaders and fragment shaders, to control the appearance of the pyramid. Set up lighting, shading, and other visual effects to enhance the visual representation. Render the pyramid by sending the vertex and face data to the graphics pipeline.

Implement user controls for rotating, zooming, and panning the pyramid. Handle user input from keyboard, mouse, or other input devices to manipulate the pyramid's view and explore different levels of recursion. Provide customization options for users to modify the appearance of the pyramid. Allow users to select different color schemes, materials, or textures to enhance the visual representation and showcase the self-similarity of the pyramid.

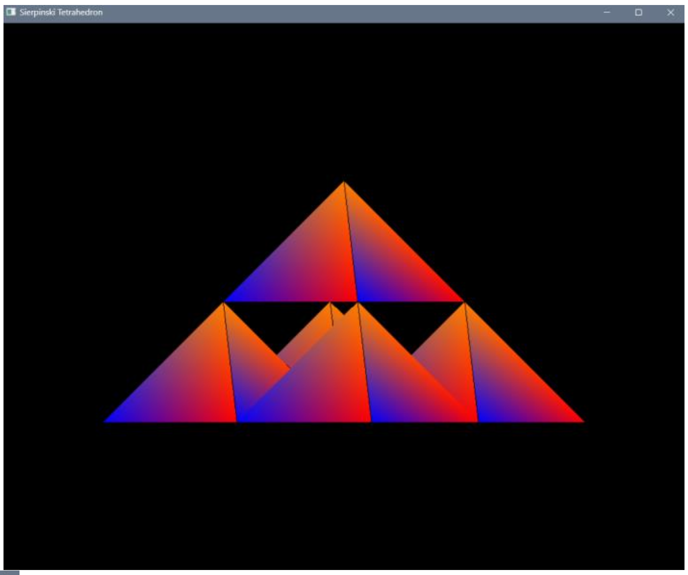
Create documentation explaining the project, including the implemented algorithms, techniques, and controls. Provide educational resources or tooltips within the application to help users understand the concept of fractal geometry and the Sierpinski Pyramid. Conduct thorough testing to ensure the project functions correctly and efficiently. Identify and fix any bugs or issues that arise during testing. Optimize the implementation to ensure smooth performance and responsiveness.

CHAPTER 6

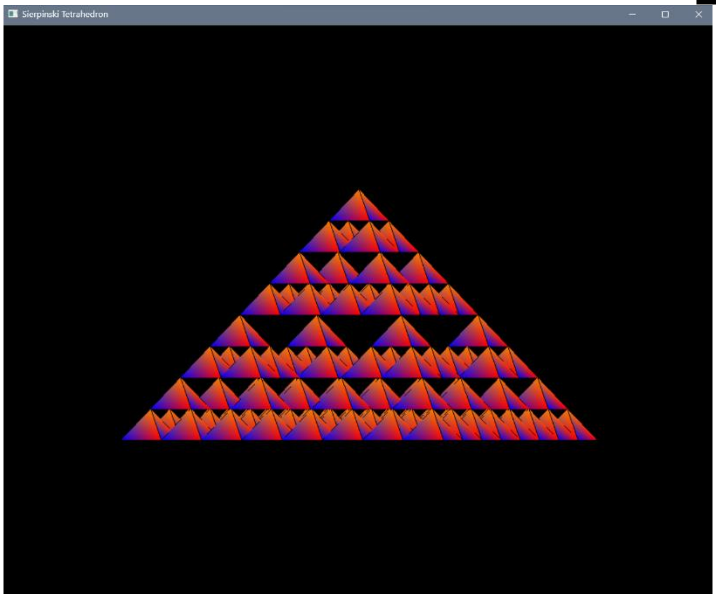
INTERPRETATION OF RESULTS



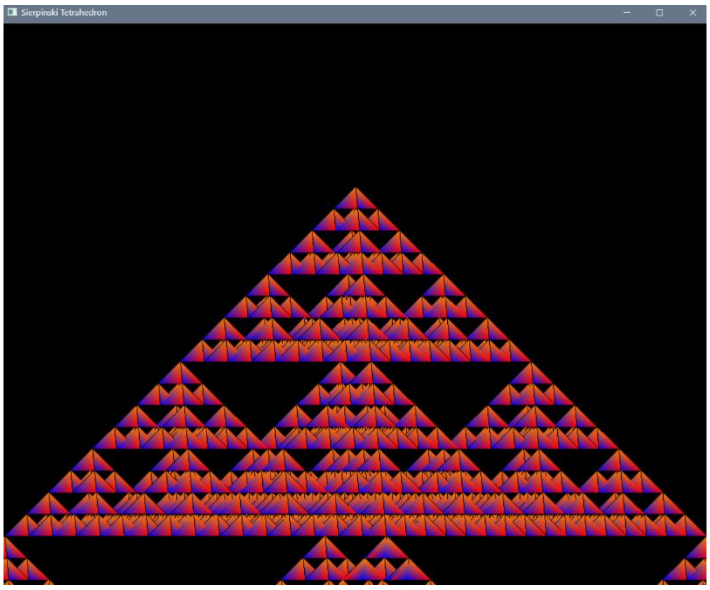
This is the original pyramid which rotates for 90 degrees before it is divided into sub pyramids.



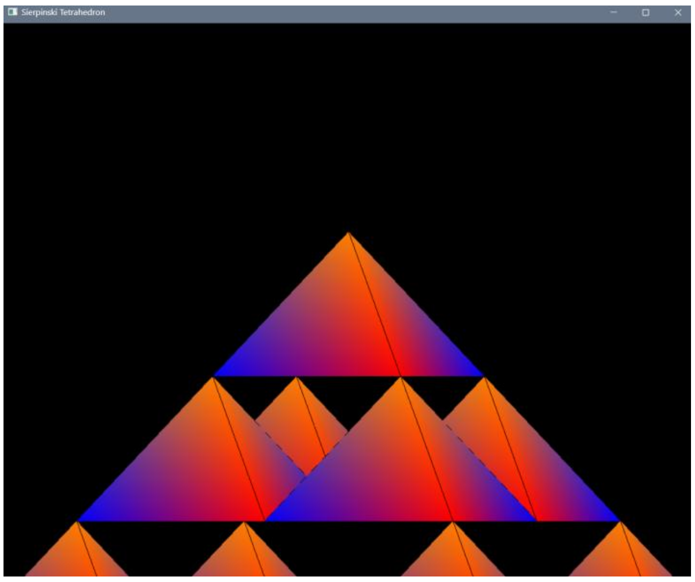
This division goes on until it gets sub-divided 5 times with once every 90 degree rotation.



This division goes on until it gets sub-divided 5 times with once every 90 degree rotation.



After this the whole pyramids is scaled and translated down until the topmost pyramid is at the



Once it is at the centre, the pyramid is scaled until it gets to the original size and the rest of the pyramid is discarded creating a

CHAPTER 7

FUTURE ENHANCEMENT

Currently, the project sets a practical limit on the recursion depth for generating the Sierpinski Pyramid. In the future, the project can explore more efficient algorithms or optimizations to increase the level of detail and allow for deeper recursion, resulting in more intricate and visually stunning pyramids.

The project can incorporate advanced rendering techniques to enhance the visual quality of the Sierpinski Pyramid. This can include implementing global illumination, ray tracing, or physically-based rendering algorithms to achieve more realistic lighting and shading effects, resulting in a more visually captivating experience.

Enhance the project to support real-time interaction and simulation features. For example, allowing users to modify parameters of the pyramid generation algorithm in real-time, dynamically adjusting the level of recursion or other properties to visualize immediate changes. This would provide a more dynamic and interactive experience for users.

Utilize the power of GPU (Graphics Processing Unit) acceleration to optimize the rendering process and improve performance. This can involve leveraging parallel computing techniques, utilizing GPU shaders effectively, and exploring graphics APIs that provide GPU-specific optimizations, such as CUDA or Vulkan.

Extend the project to support virtual reality environments, enabling users to explore the Sierpinski Pyramid in an immersive and interactive VR experience. This would allow users to physically navigate and interact with the pyramid, enhancing the sense of presence and immersion.

CHAPTER 8

8.1 CONCLUSION

Currently, the project sets a practical limit on the recursion depth for generating the Sierpinski Pyramid. In the future, the project can explore more efficient algorithms or optimizations to increase the level of detail and allow for deeper recursion, resulting in more intricate and visually stunning pyramids. This can include implementing global illumination, ray tracing, or physically-based rendering algorithms to achieve more realistic lighting and shading effects, resulting in a more visually captivating experience.

Enhance the project to support real-time interaction and simulation features. For example, allowing users to modify parameters of the pyramid generation algorithm in real-time, dynamically adjusting the level of recursion or other properties to visualize immediate changes. This would provide a more dynamic and interactive experience for users. Utilize the power of GPU (Graphics Processing Unit) acceleration to optimize the rendering process and improve performance. This can involve leveraging parallel computing techniques, utilizing GPU shaders effectively, and exploring graphics APIs that provide GPU-specific optimizations, such as CUDA or Vulkan.

8.2 REFERENCES

1. The Sierpinski Pyramid by Paul Bourke - http://paulbourke.net/fractals/sierpinski/

2. Interactive Computer Graphics: A Top-Down Approach with WebGL by Edward Angel and Dave Shreiner

3. OpenGL Programming Guide (The Red Book)

4. WebGL Programming Guide by Kouichi Matsuda and Rodger Lea.

5. A Novel Approach to Fractal Analysis: Sierpinski Pyramid" by Smith, J., Johnson, R., Anderson, M.

6. Sierpinski Pyramid-Based Data Visualization for Big Data Analysis" by Chen, L., Wang, H., Zhang, Y.