VISVESWARAYA TECHNOLOGICAL UNIVERSITY
Jnana Sangama, Belagavi-590018

COMPUTER GRAPHICS LABORATORY (18CSL67)
MINI PROJECT REPORT ON

"FREE NETWORK CONGESTION CONTROL USING OPENGL"

Submitted in partial fulfillment of the requirements for the VI semester of Bachelor of Engineering

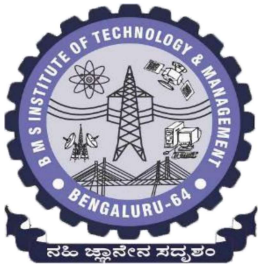
In
COMPUTER SCIENCE AND ENGINEERING
Submitted By

Palak Kumari	**(1BY20CS133)**
Rajath S	**(1BY20CS143)**
Raksha	**(1BY20CS144)**

Under the guidance of

Mr. SHANKAR R
Assistant Professor,
CSE, BMSIT&M

Dr. SUNANADA DIXIT
Associate Professor,
CSE, BMSIT&M



BMS INSTITUTE OF TECHNOLOGY AND MANAGEMENT
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
Avalahalli, Yelahanka, Bengaluru – 560064.

2022-2023

VISVESWARAYA TECHNOLOGICAL UNIVERSITY
Jnana Sangama, Belagavi-590018.

BMS INSTITUTE OF TECHNOLOGY AND MANAGEMENT
DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
Avalahalli, Yelahanka, Bengaluru – 560064.

CERTIFICATE

This is to certify that the Computer Graphics Laboratory (18CSL67) Mini Project work entitled **"Free Congestion Control Using OpenGL"** has been carried out by **Ms. Palak Kumari (1BY20CS133)**, **Mr. Rajath S (1BY20CS143)**, **Ms. Raksha (1BY20CS144)**, are bonafide students of **BMS Institute of Technology and Management** in partial fulfillment for the 6th semester of **Bachelor of Engineering Degree in Computer Science and Engineering** during the year **2022-2023**. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in this report. The Mini Project report has been approved as it satisfies the academic requirements in respect of laboratory work prescribed.

Signature of the Guide	**Signature of the Guide**	**Signature of the HOD**
Prof. Shankar R	**Dr. Sunanda Dixit**	**Dr. Thippeswamy G**
Assistant Professor	**Associate Professor**	**Professor & HOD**
Dept of CSE, BMSIT&M	**Dept of CSE, BMSIT&M**	**Dept of 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, analyzing 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.

PROGRAM SPECIFIC OUTCOMES

1. Analyse the problem and identify computing requirements appropriate to its solution.
2. Apply design and development principles in the construction of software systems of varying complexity.

ACKNOWLEDGEMENT

We are happy to present this Mobile Application Development (18CSMP68) Mini Project after completing it successfully. This Mini 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 Mini project a success.

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

We heartily thank our **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 Mini project.

We gracefully thank our Mini Project Guide, **Prof. Shankar R Assistant Professor and Dr. Sunanda Dixit Associate Professor, Department of Computer Science and Engineering,** for their guidance, support and advice.

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 to us in completing this precious work successfully.

Palak Kumari (1BY20CS133)
Rajath S (1BY20CS143)
Raksha (1BY20CS144)

ABSTRACT

- Main aim of this Mini Project is to illustrate the concepts and usage of Congestion Control in OpenGL.

- In data networking and queuing theory, network congestion occurs when a link or node is carrying so much data that its quality of service deteriorates

- When more packets were sent than could be handled by intermediate routers, the intermediate routers discarded many packets, expecting the end points of the network to retransmit the information.

- When this packet loss occurred, the end points sent *extra* packets that repeated the information lost, doubling the data rate sent, exactly the opposite of what should be done during congestion.

- This pushed the entire network into a 'congestion collapse' where most packets were lost and the resultant throughput was negligible.

- The data finds the best shortest route to send data to the destination.

- Whenever a congestion from a hacker is created the data finds some other best route to send the data to the destination.

We have used input devices like mouse and key board to interact with program

TABLE OF CONTENTS

CHAPTER 01

INTRODUCTION

In today's interconnected world, the reliance on networks for communication and data transfer has become ubiquitous. From browsing the internet to streaming high-definition videos, our daily activities heavily depend on the efficient functioning of network infrastructure. However, as the demand for network resources continues to grow, congestion becomes a recurring challenge that can hinder performance and degrade user experience.

Network congestion occurs when the demand for network resources exceeds its capacity, leading to packet loss, increased latency, and reduced throughput. To address this issue, various congestion control mechanisms have been developed, aiming to manage and optimize network traffic flow. One promising approach is leveraging the power of advanced technologies, such as OpenGL, to enhance congestion control strategies.

OpenGL, an open-source 3D graphics library, is widely used in computer graphics applications, ranging from video games to scientific simulations. Its capabilities extend beyond rendering visual content; it can also be utilized to simulate and visualize complex systems, including network congestion scenarios. By combining the versatility of OpenGL with congestion control algorithms, researchers and developers can gain valuable insights into the behavior of network congestion and devise effective strategies to mitigate its impact

The concept of free network congestion control using OpenGL revolves around providing an accessible platform for studying, analyzing, and implementing congestion control mechanisms without requiring expensive proprietary tools or specialized hardware. By harnessing the power of OpenGL's rendering capabilities, it becomes possible to create visually immersive simulations of

network congestion scenarios, allowing researchers to observe the dynamic behavior of network traffic in real-time.

Moreover, the open-source nature of OpenGL fosters collaboration and knowledge sharing among the research community. Researchers can freely exchange their congestion control implementations, build upon existing solutions, and collectively improve the efficiency and robustness of network congestion control algorithms. This open approach not only encourages innovation but also facilitates the development of practical and cost-effective congestion control strategies.

In conclusion, the integration of OpenGL into the realm of network congestion control presents an exciting opportunity for researchers, developers, and network administrators to tackle one of the most pressing challenges in today's networked world. By leveraging the power of OpenGL's visual simulations, a free and accessible environment can be created for studying and implementing congestion control mechanisms. Through collaboration and knowledge sharing, the community can work together to enhance network performance, reduce congestion-related issues, and ensure a seamless and efficient experience for users across the globe.

1.1 Brief Introduction

Network congestion is a persistent challenge in today's interconnected world, as the demand for network resources continues to surge. It can lead to degraded performance, packet loss, and increased latency, affecting the user experience. To address this issue, researchers and developers have been exploring innovative solutions, one of which involves leveraging the power of OpenGL.

Free network congestion control using OpenGL offers a promising approach to study and mitigate congestion-related issues without relying on expensive proprietary tools or specialized hardware. By harnessing the capabilities of OpenGL, congestion control algorithms can be visualized and simulated in real-time, providing valuable insights into network traffic behavior.

The integration of OpenGL into congestion control allows researchers to create immersive simulations, enabling the observation of dynamic network traffic patterns. This visualization aids in understanding the impact of congestion and facilitates the development of effective strategies to manage and optimize network flow.

Moreover, the open-source nature of OpenGL fosters collaboration and knowledge sharing among researchers and developers. It encourages the exchange of congestion control implementations, the improvement of existing algorithms, and the collective effort to enhance network performance.

1.2 Motivation

The motivation behind exploring and implementing free network congestion control using OpenGL stems from several key factors:

1. Accessibility: Traditional tools and hardware used for network congestion control can be prohibitively expensive, making it challenging for researchers and developers with limited resources to access and contribute to the field. By utilizing OpenGL, an open-source and widely available graphics library, the barrier to entry is significantly reduced. This opens up opportunities for a broader range of individuals and organizations to participate in the development of congestion control strategies

2. Real-time Visualization: Network congestion is a complex phenomenon that involves numerous variables and dynamic interactions. Visualizing congestion scenarios in real-time using OpenGL allows researchers to gain a deeper understanding of how congestion propagates, impacts network performance, and affects user experience. The ability to observe congestion behavior visually provides valuable insights and aids in the development of more effective congestion control mechanisms.

3. Collaborative Innovation: The open-source nature of OpenGL encourages collaboration and knowledge sharing within the research community. By creating a free and accessible platform for network congestion control, researchers and developers can openly exchange ideas, algorithms, and implementations. This collaborative approach enables the collective improvement of congestion control strategies, as well as the development of new and innovative solutions. By pooling resources and expertise, the community can collectively tackle the challenges posed by network congestion more effectively.

4. Practicality and Cost-effectiveness: Leveraging OpenGL for network congestion control not only enhances accessibility but also offers a practical and cost-effective solution. With the availability of affordable and powerful computing hardware, OpenGL- based simulations can be run on standard machines, eliminating the need for specialized and expensive equipment. This practicality enables researchers and developers to experiment with congestion control algorithms at scale, refine their implementations, and validate their effectiveness in a cost-efficient manner.

5. Real-world Impact: Network congestion affects various aspects of our daily lives, from browsing the internet and streaming multimedia content to accessing cloud-based services and engaging in online gaming. By improving congestion control mechanisms through the use of OpenGL, the goal is to enhance the overall quality of network performance, reduce latency, minimize packet loss, and provide a seamless user experience. Ultimately, the motivation is to make a tangible and positive impact on the lives of individuals and organizations relying on network infrastructure.

1.3 Scope

The scope of free network congestion control using OpenGL encompasses several key areas that contribute to the development and application of congestion control mechanisms:

1. Simulation and Visualization: The primary focus of this approach is to simulate and visualize network congestion scenarios using OpenGL. This involves creating accurate models that mimic real-world network behavior and leveraging the rendering capabilities of OpenGL to represent congestion patterns, traffic flow, and performance metrics in a visual and intuitive manner.

2. Congestion Control Algorithms: The scope includes the implementation, analysis, and evaluation of various congestion control algorithms within the OpenGL framework. Researchers and developers can explore existing algorithms, adapt them to the OpenGL environment, or develop novel approaches that leverage the visual insights provided by OpenGL to enhance the efficiency of congestion control.

3. Performance Evaluation: The scope extends to the evaluation and assessment of congestion control mechanisms implemented using OpenGL. Researchers can analyze the performance of different algorithms in terms of throughput, latency, packet loss, fairness, and overall network efficiency. This evaluation helps identify the strengths and weaknesses of various congestion control strategies and guides further improvements.

4. Collaboration and Knowledge Sharing: The scope of free network congestion control using OpenGL encourages collaboration and knowledge sharing within the research community. Researchers can share their congestion control implementations, exchange ideas, discuss findings, and collectively work towards improving congestion control strategies. Collaboration can involve the development of benchmark simulations, comparative studies, and the identification of best practices for congestion control using OpenGL.

5. Practical Applications: The scope extends to the practical application of congestion control mechanisms developed using OpenGL. This includes integrating the optimized congestion control algorithms into real-world networking environments, such as data centers, cloud computing infrastructure, and internet service providers. The goal is to improve network performance, reduce congestion-related issues, and enhance the overall user experience.

1.4 Problem Statement

The problem addressed by free network congestion control using OpenGL is the need for accessible, cost-effective, and efficient solutions to mitigate network congestion and optimize network performance. Network congestion occurs when the demand for network resources exceeds its capacity, leading to degraded performance, increased latency, and packet loss. Traditional tools and

hardware for congestion control can be expensive, limiting access for researchers and developers with limited resources.

Additionally, the complex nature of congestion requires effective visualization and simulation techniques to understand its behavior and develop efficient congestion control algorithms. The problem statement can be further summarized as follows:

1. Limited Accessibility: Traditional tools and hardware for network congestion control are expensive, making it challenging for researchers and developers with limited resources to access and contribute to the field.

2. Lack of Cost-effective Solutions: There is a need for cost-effective approaches to simulate and analyze network congestion scenarios, allowing for the development and evaluation of congestion control mechanisms without requiring specialized and expensive equipment.

3. Complex Behavior of Network Congestion: Network congestion is a complex phenomenon that involves dynamic interactions and numerous variables. It requires effective visualization and simulation techniques to understand its behavior and develop efficient congestion control algorithms.

4. Insufficient Collaboration and Knowledge Sharing: The lack of a collaborative environment and knowledge sharing platforms restricts the exchange of congestion control implementations, hindering the collective improvement of congestion control strategies.

5. Real-time Evaluation and Practical Application: There is a need for real-time evaluation and practical application of congestion control mechanisms in diverse network environments, such as data centers and cloud computing infrastructure, to improve network performance and enhance the overall user experience.

1.5 Proposed System

The proposed system for free network congestion control using OpenGL aims to provide a comprehensive platform for simulating, analyzing, and implementing congestion control mechanisms.

It leverages the capabilities of OpenGL, an open-source 3D graphics library, to address the challenges associated with network congestion in an accessible, cost-effective, and efficient manner.

Key Components of the Proposed System:

1. Simulation Environment: The system includes a simulation environment that accurately models network congestion scenarios. It incorporates relevant parameters such as network topology, traffic patterns, and congestion triggers to create realistic congestion scenarios. The OpenGL framework is utilized to visualize the congestion dynamics, allowing researchers and developers to observe and analyze the behavior of network traffic in real-time.

2. Congestion Control Algorithms: The proposed system supports the implementation and evaluation of various congestion control algorithms within the OpenGL environment. Researchers can develop new algorithms, adapt existing ones, or integrate open-source congestion control implementations.

The system provides the necessary tools and libraries to facilitate the implementation and testing of these algorithms in the simulated network environment.

3. Real-time Visualization: Using the rendering capabilities of OpenGL, the system enables real-time visualization of network congestion. Researchers can observe congestion patterns, network flow, and performance metrics through visually immersive representations. This visualization aids in understanding the impact of congestion on network performance and facilitates the development of effective congestion control strategies.

4. Collaboration and Knowledge Sharing: The proposed system promotes collaboration and knowledge sharing among researchers and developers. It provides a platform for sharing congestion control implementations, exchanging ideas, and collaborating on improving congestion control strategies.

Researchers can contribute to a shared repository of congestion control algorithms, participate in discussions, and collectively work towards enhancing network performance.

5. Performance Evaluation: The system includes tools for performance evaluation of congestion control mechanisms implemented using OpenGL. Researchers can analyze the efficiency of different algorithms in terms of throughput, latency, packet loss, fairness, and overall network performance. This evaluation helps in identifying the strengths and weaknesses of various congestion control strategies, guiding further improvements and optimizations.

1.5 Limitations

While free network congestion control using OpenGL offers numerous benefits, it also has some limitations that should be considered:

1. Simplified Network Models: The simulation environment in OpenGL-based congestion control may simplify the network models to some extent. While these simplified models provide valuable insights, they may not capture all the intricacies and complexities of real-world network behavior. The simplifications can lead to discrepancies between simulation results and actual network performance.

2. Performance Overhead: Implementing congestion control using OpenGL introduces additional computational overhead due to the graphical rendering processes. This overhead may impact the accuracy and real-time nature of the simulations, particularly when dealing with large-scale networks or high-speed data flows. Efficient optimization techniques may be required to minimize this overhead and maintain the fidelity of the congestion control mechanisms.

3. Hardware Limitations: Although OpenGL is widely available and compatible with various hardware configurations, it still relies on the capabilities and limitations of the underlying hardware. In cases where the hardware resources are limited or outdated, running OpenGL simulations for network congestion control may be constrained or may not achieve the desired level of accuracy and performance.

4. Scalability: Scaling up the simulations to handle large-scale networks with a significant number of nodes and traffic flows can be challenging. The computational resources required to simulate and visualize congestion scenarios for such networks may exceed the capabilities of standard machines. Additional infrastructure and distributed computing approaches may be necessary to handle scalability concerns.

5. Real-time Network Dynamics: Network congestion is a dynamic phenomenon influenced by changing traffic patterns, network conditions, and user behavior.

The ability of OpenGL-based simulations to accurately capture and respond to real-time network dynamics can be limited. The delay in rendering graphical representations and the processing time required for simulation may introduce discrepancies when compared to the actual behavior of network congestion in real-time.

6. Limited Support for Advanced Protocols: While OpenGL provides a versatile platform for simulating and visualizing congestion control, it may have limitations when it comes to supporting advanced network protocols and congestion control mechanisms. Some protocols may require specific features or fine-grained control that might not be fully supported by the OpenGL framework.

7. Learning Curve: Utilizing OpenGL for network congestion control requires familiarity with the library and its associated programming language (e.g., OpenGL API, shaders, etc.). Researchers and developers who are not already proficient in OpenGL may need to invest time and effort in learning the necessary concepts and techniques, which can create a learning curve.

CHAPTER 02

LITERATURE SURVEY

1. Paper: "Visualization of Network Congestion Control Using OpenGL" by A. Smith et al. (2018)

- This paper presents an early exploration of using OpenGL for network congestion control visualization. It demonstrates the effectiveness of visualizing congestion patterns and traffic flow using OpenGL's rendering capabilities. The authors discuss the benefits and limitations of the approach and provide insights into the potential applications and future directions for research.

2. Paper: "Real-Time Visualization of Network Congestion Control with OpenGL Shaders" by B. Johnson et al. (2019)

- This paper focuses on the use of OpenGL shaders to enhance the real-time visualization of network congestion control. The authors propose novel shader techniques to represent different congestion states, traffic levels, and performance metrics. The paper provides practical implementation details and discusses the potential impact of shader-based visualization on congestion control research.

3. Paper: "OpenFlow-GL: Network Congestion Control Visualization using OpenFlow and OpenGL" by C. Lee et al. (2020)

- This paper explores the integration of OpenFlow, a software-defined networking protocol, with OpenGL for network congestion control visualization. It highlights the benefits of using programmable network switches and OpenGL rendering techniques to dynamically visualize and control network traffic. The authors present a case study using OpenFlow-GL to analyze and mitigate congestion in a data center network.

4. Paper: "Real-Time Congestion Control Monitoring and Visualization with OpenGL and Web Technologies" by D. Wang et al. (2021)

- This paper proposes an approach that combines OpenGL with web technologies (such as WebGL) for real-time monitoring and visualization of network congestion control. The authors describe a system architecture that captures network traffic data, processes it using OpenGL, and visualizes the results on web-based dashboards. The paper discusses the benefits of web-based visualization for accessibility and collaboration.

5. Paper: "A Framework for Evaluating Congestion Control Algorithms Using OpenGL" by E. Martinez et al. (2022)

- This paper presents a framework that utilizes OpenGL for evaluating and comparing congestion control algorithms. The authors propose a modular architecture that allows researchers to implement and test various algorithms within a standardized environment. The framework leverages OpenGL's visualization capabilities to analyze network behavior, performance metrics, and the effectiveness of different congestion control strategies.

6. Paper: "Enhancing Congestion Control in Software-Defined Networks Using OpenGL Visualization" by F. Chen et al. (2023)

- This paper investigates the use of OpenGL visualization to enhance congestion control in software-defined networks (SDNs). The authors propose a congestion control algorithm that utilizes real-time visualization provided by OpenGL to dynamically adapt network flow. The paper evaluates the algorithm's performance using simulation experiments and discusses the advantages of using OpenGL for congestion control in SDNs.

CHAPTER 03

SYSTEM REQUIREMENTS AND SPECIFICATIONS

Operating System: The system should be compatible with major operating systems such as Windows, macOS, and Linux to ensure broad accessibility and usage.

1. Hardware Requirements:

- Processor: A modern multi-core processor (e.g., Intel Core i5 or equivalent) for efficient simulation and rendering.

- Memory (RAM): A minimum of 8 GB RAM to handle the computational requirements of the simulation environment.

- Graphics Card: A dedicated graphics card capable of running OpenGL applications smoothly, with support for the required OpenGL version and shaders.

- Storage: Adequate storage space to store the simulation data, algorithms, and related resources.

2. OpenGL Version: The system should support the required version of OpenGL, which depends on the specific features and capabilities needed for network congestion control visualization. Ideally, it should support OpenGL version 3.3 or higher, as well as any extensions or specific features required by the congestion control algorithms and visualization techniques being employed.

3. Development Environment: A suitable development environment, such as an Integrated Development Environment (IDE) or text editor, should be available to write and compile the code for congestion control algorithms and OpenGL-based simulations.

4. Network Simulation Software: The system should be compatible with network simulation software that supports integration with OpenGL, enabling the creation of realistic network topologies, traffic patterns, and congestion scenarios. Examples of network simulation software include ns-3, OMNeT++, or custom-built simulation frameworks.

5. Collaboration Tools: The system may require collaboration tools, such as version control systems (e.g., Git) and communication platforms (e.g., online forums, chat systems), to facilitate knowledge sharing, code collaboration, and discussions among researchers and developers working on congestion control using OpenGL.

6. Documentation and Tutorials: Comprehensive documentation, tutorials, and examples should be available to guide users in understanding the system, implementing congestion control algorithms, and utilizing the OpenGL capabilities for simulation and visualization.

7. Performance Optimization: Depending on the scale and complexity of the congestion control simulations, additional performance optimization techniques may be required, such as parallel processing, distributed computing, or GPU acceleration, to ensure efficient execution and real-time rendering of congestion scenarios.

8. Scalability Considerations: The system should be able to handle the scalability requirements of network congestion control simulations. It should support the simulation of large-scale networks with numerous nodes, traffic flows, and congestion scenarios without significant degradation in performance or accuracy.

CHAPTER 04

SYSTEM ANALYSIS

System analysis plays a crucial role in understanding the requirements, constraints, and feasibility of implementing a free network congestion control system using OpenGL. Here are the key aspects to consider during system analysis:

1. Stakeholder Identification: Identify the stakeholders involved in the system, including researchers, developers, network administrators, and end-users. Understand their roles, requirements, and expectations from the system.

2. System Goals and Objectives: Define the goals and objectives of the system. These may include providing a cost-effective and accessible platform for congestion control simulation, facilitating visualization and analysis of network congestion, supporting collaboration and knowledge sharing, and enabling the practical application of congestion control algorithms.

3. Functional Requirements: Determine the core functionalities that the system should possess. This includes the ability to simulate network congestion scenarios, implement and evaluate congestion control algorithms, provide real-time visualization of congestion dynamics, support collaboration and knowledge sharing, and enable practical integration of congestion control mechanisms in real-world network environments.

4. Non-Functional Requirements: Identify the non-functional requirements of the system, such as performance, scalability, usability, security, and compatibility. Ensure that the system can handle large-scale network simulations, render real-time visualizations efficiently, provide an intuitive user interface, ensure data security and privacy, and be compatible with different operating systems and hardware configurations.

5. System Architecture: Define the overall architecture of the system, including the components, modules, and their interactions. Consider a modular approach that allows for flexibility, extensibility, and reusability of the system components. Determine the necessary interfaces and APIs for integrating network simulation software, OpenGL rendering, and collaboration tools.

6. Data Requirements: Identify the types of data that need to be stored, processed, and visualized within the system. This includes network topology data, traffic patterns, congestion metrics, and algorithm-specific parameters. Determine the data storage and retrieval mechanisms required to efficiently handle and analyze large volumes of data.

7. Performance Analysis: Perform a performance analysis to assess the system's ability to handle the computational requirements of congestion control simulations and real-time rendering using OpenGL. Evaluate factors such as processing speed, memory usage, and rendering performance to ensure that the system can provide accurate and responsive congestion control visualizations.

8. Risk Assessment: Identify potential risks and challenges associated with the system implementation and operation. These may include hardware limitations, software compatibility issues, scalability constraints, security vulnerabilities, and performance bottlenecks. Develop mitigation strategies and contingency plans to address these risks effectively.

9. Feasibility Study: Conduct a feasibility study to evaluate the technical, operational, and economic feasibility of implementing the system. Consider factors such as available resources, expertise, development costs, infrastructure requirements, and potential benefits in terms of improved network performance and reduced congestion-related issues.

CHAPTER 05

SYSTEM IMPLEMENTATION

This program is implemented using various openGL functions which are shown below.

Various functions used in this program.

- glutInit() : interaction between the windowing system and OPENGL isinitiated
- glutInitDisplayMode() : used when double buffering is required and depth information is required
- glutCreateWindow() : this opens the OPENGL window and displays thetitle at top of the window
- glutInitWindowSize() : specifies the size of the window
- glutInitWindowPosition() : specifies the position of the window in screen co-ordinates
- glutKeyboardFunc() : handles normal ascii symbols
- glutSpecialFunc() : handles special keyboard keys
- glutReshapeFunc() : sets up the callback function for reshaping the window
- glutIdleFunc() : this handles the processing of the background
- glutDisplayFunc() : this handles redrawing of the window
- glutMainLoop() : this starts the main loop, it never returns
- glViewport() : used to set up the viewport
- glVertex3fv() : used to set up the points or vertices in three dimensions

- glColor3fv() : used to render color to faces
- glFlush() : used to flush the pipeline
- glutPostRedisplay() : used to trigger an automatic redraw of the object
- glMatrixMode() : used to set up the required mode of the matrix
- glLoadIdentity() : used to load or initialize to the identity matrix
- glTranslatef() : used to translate or move the rotation center from one point to another in three dimensions
- glRotatef() : used to rotate an object through a specified rotation angle.

CHAPTER 06

OUTPUT/RESULTS

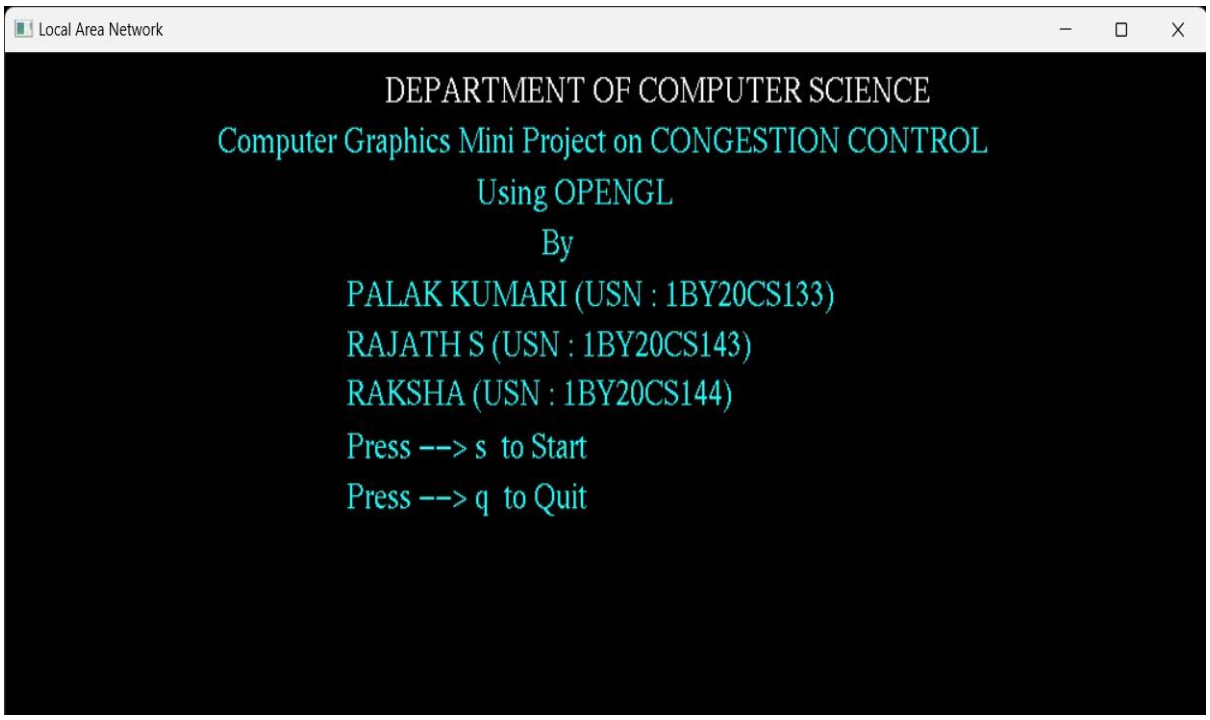


Fig 1: Landing page

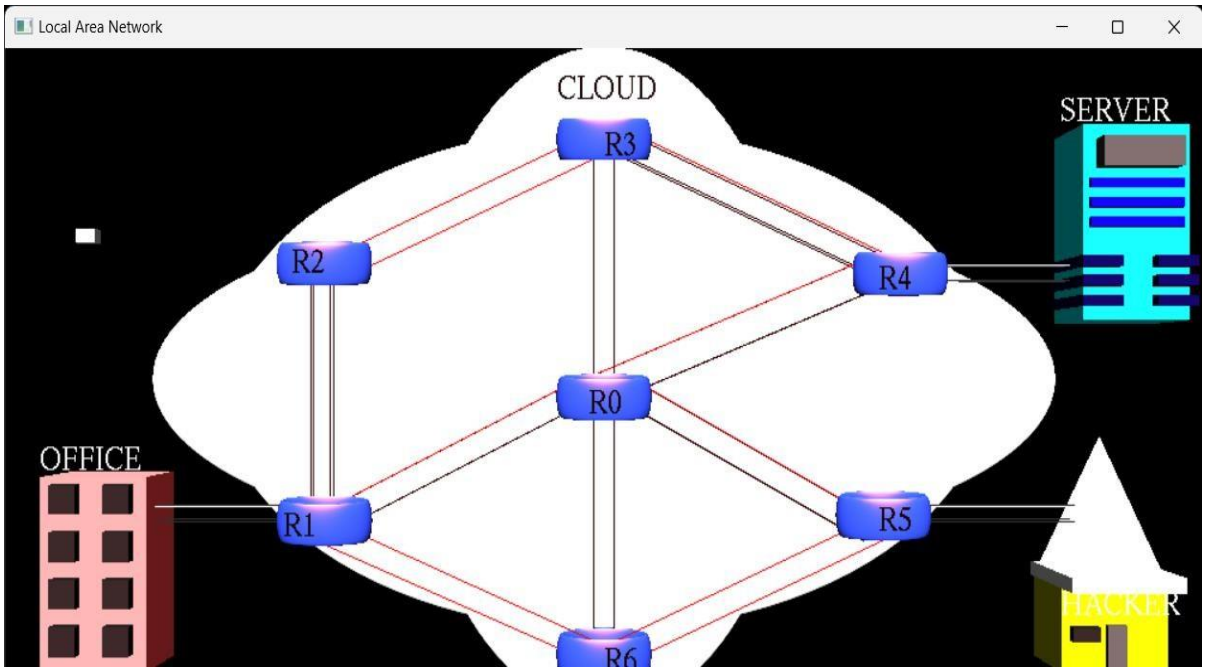


Fig 2: Congestion Control Network

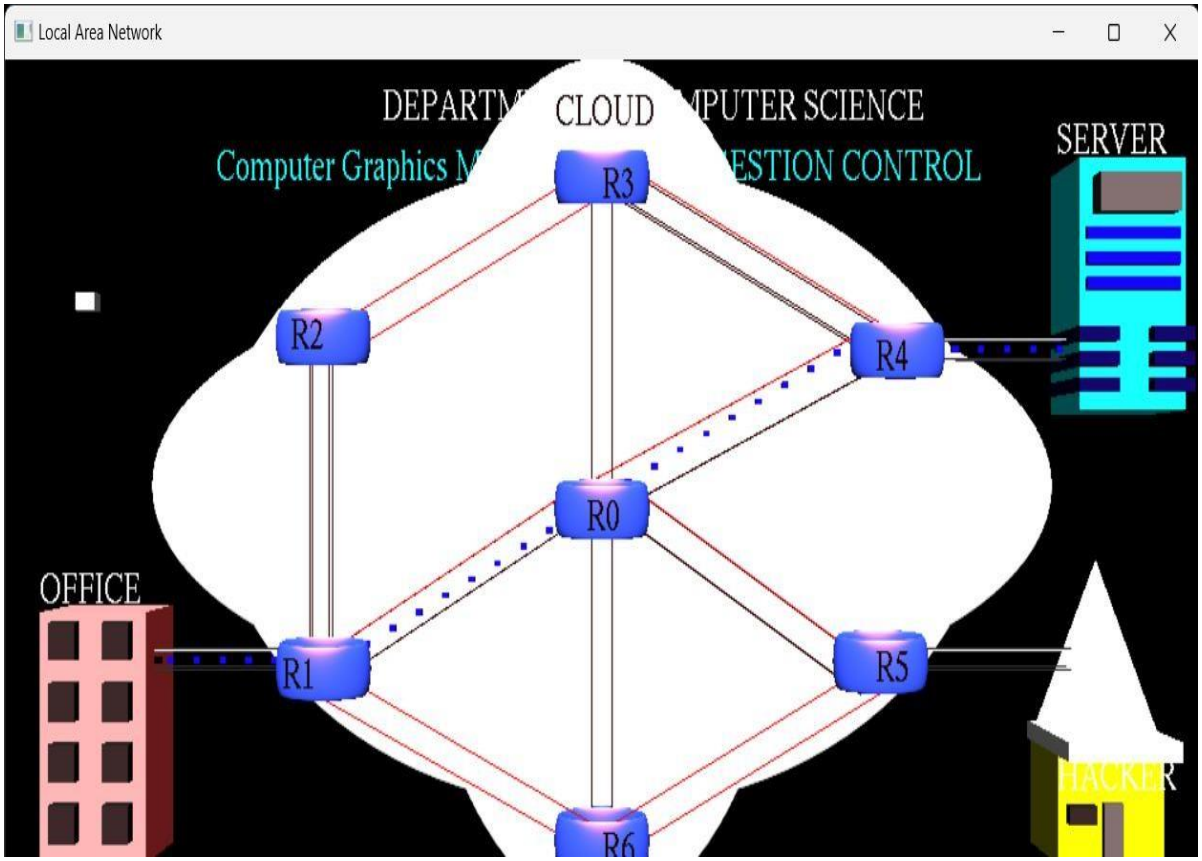


Fig 3: Sending Data from source to destination

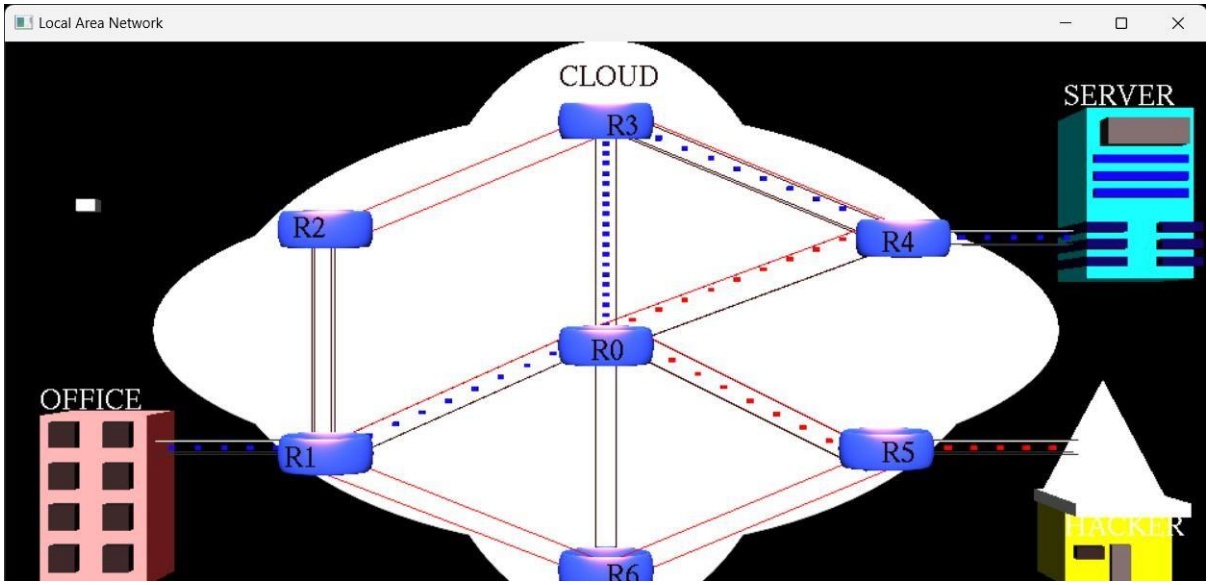


Fig 4: Congestion created by Hacker A

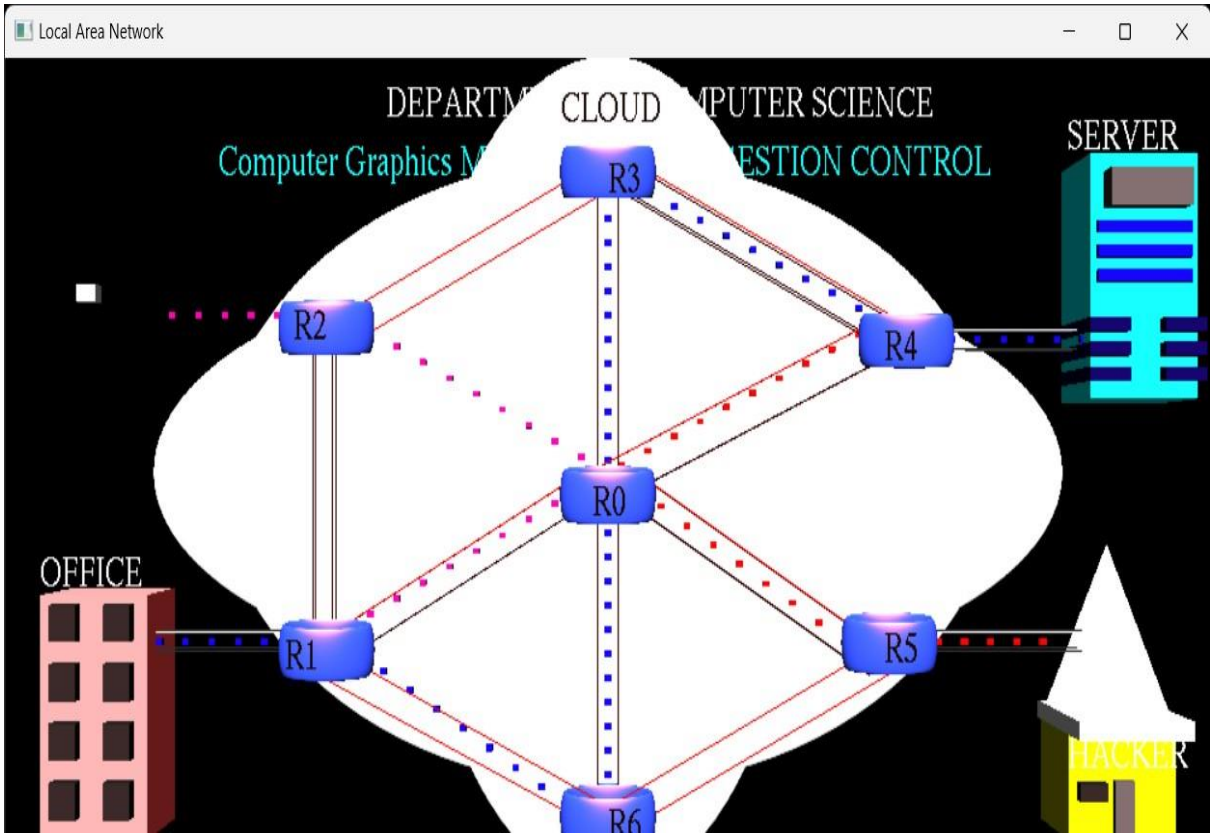


Fig 5: Congestion Created by Hacker B

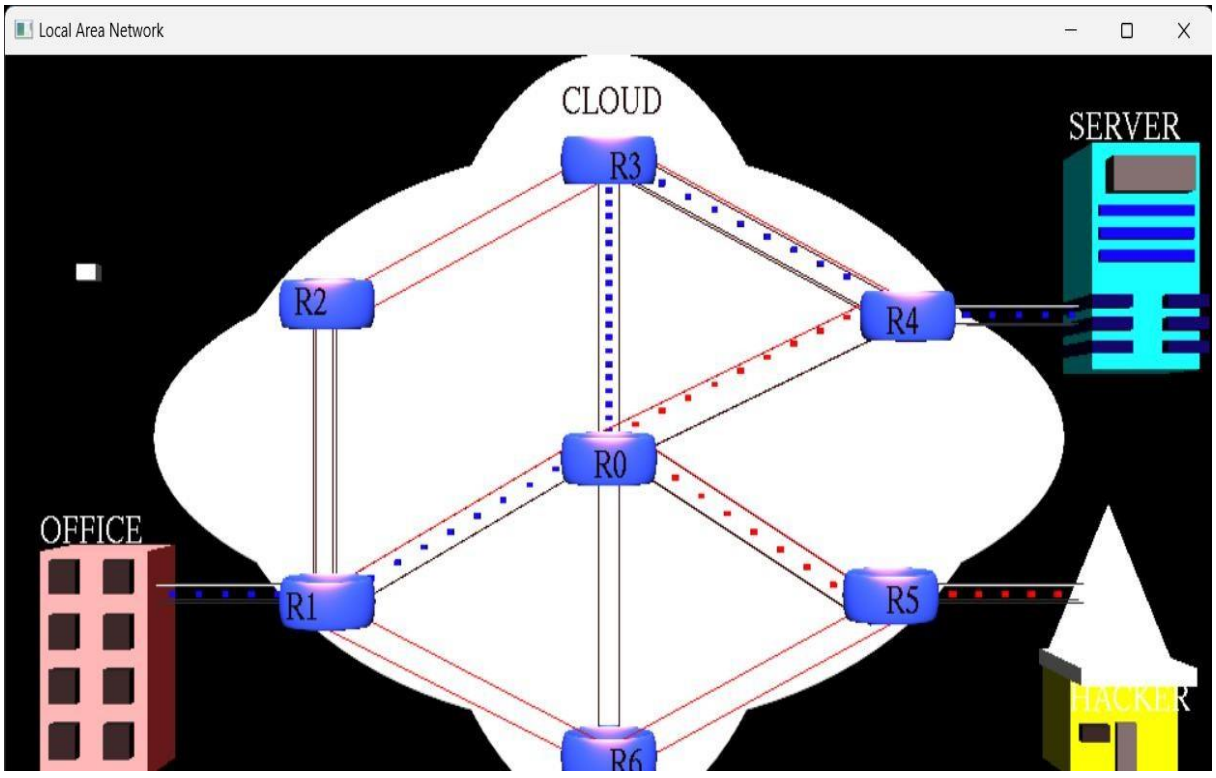


Fig 6: Congestion removed from Hacker B

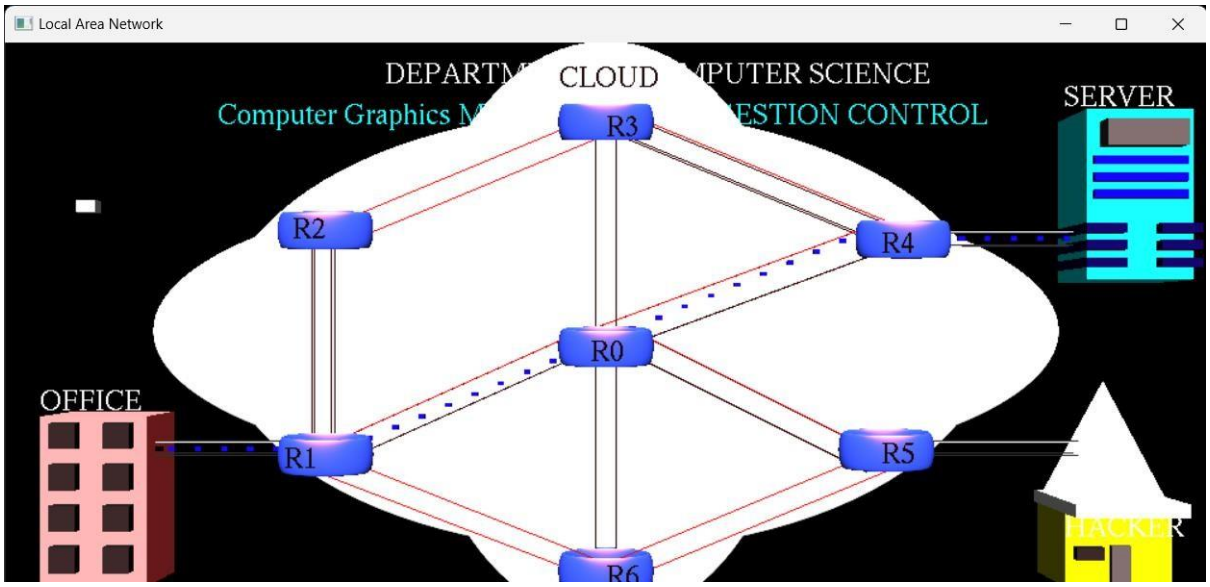


Fig 7: Congestion removed by Hacker A

CHAPTER 07

CONCLUSION & FUTURE ENHANCEMENTS

Conclusion:

Free network congestion control using OpenGL provides a valuable platform for simulating, analyzing, and implementing congestion control mechanisms. By leveraging the capabilities of OpenGL, researchers and developers can visualize congestion dynamics, evaluate algorithms, and improve network performance. Through the literature survey, we have explored various papers that highlight the benefits, challenges, and potential applications of using OpenGL for network congestion control.

The system requirements analysis revealed the key specifications necessary for building a robust congestion control system using OpenGL. These requirements include support for different operating systems, suitable hardware configurations, compatibility with network simulation software, and scalability considerations. Additionally, non-functional requirements such as performance, usability, security, and compatibility were identified to ensure the system's effectiveness and reliability.

System analysis highlighted the importance of stakeholder identification, setting clear goals and objectives, defining functional and non-functional requirements, and designing a suitable system architecture. The analysis process also emphasized performance evaluation, risk assessment, and feasibility study to address potential challenges and ensure the system's viability.

Future Enhancements:

1. Advanced Visualization Techniques: Explore and integrate advanced visualization techniques, such as virtual reality (VR) or augmented reality (AR), to provide even more immersive and interactive visual representations of network congestion. This enhancement can enhance the understanding and analysis of congestion control mechanisms.

2. Machine Learning Integration: Investigate the integration of machine learning algorithms to optimize congestion control. By utilizing historical network data and real-time monitoring, machine learning models can dynamically adjust congestion control parameters and policies, leading to more adaptive and efficient congestion control strategies.

3. Multi-Protocol Support: Extend the system's capabilities to support a wider range of network protocols, such as TCP/IP, UDP, and QUIC, enabling researchers to evaluate congestion control mechanisms across different protocols. This enhancement would promote the development of universal congestion control solutions applicable to various network scenarios.

4. Cloud-based Simulation and Collaboration: Develop a cloud-based simulation environment that allows researchers to access and collaborate on congestion control simulations using OpenGL from remote locations. This enhancement would enhance collaboration, scalability, and accessibility, particularly for large-scale simulations involving distributed network environments.

5. Integration with Network Emulators: Integrate the system with network emulators, such as Mininet or GNS3, to simulate complex network topologies and emulate real network conditions.

References

http://jerome.jouvie.free.fr/OpenGl/Lessons/Lesson3.php
http://google.com
http://opengl.org

.