ABSTRACT

Open GL is a hardware- and system dependent interface. An Open GL-application will work on every platform, as long as there is an installed implementation, Because it is system independent, there are no functions to create windows etc., but there are helper functions for each platform. A very useful thing is GLUT.

The graphics project developed helps in demonstration of The Leaky Bucket Algorithm.The graph is a mathematical representation of the algorithm .The code was developed from scratch.

The code for the calculation of points along the graph have been written in C.Along with a few graphics function we produce a nice colorful image with clarity which goes about implementing the package effectively without harming the originality of the package and also enhancing the visual appeal to the user.

The main objective of the package is to implement the algorithm for different inputs.We have also designed a pictorial representation the algorithm. These functions are implemented through various functions of OpenGL like primitives,Stings, Translation …etc each are explained in detail further .

TABLE OF CONTENTS

INTRODUCTION

The leaky bucket algorithm is a method of controlling the rate at which data is transmitted between a source and a destination in a network. In this project, we demonstrate the leaky bucket algorithm using OpenGL, a graphics rendering API. We visualize the algorithm using a graphical representation of a bucket that fills with water (data) and leaks out at a constant rate (transmission rate). The user can adjust the rate at which the bucket fills and the rate at which it leaks to observe how the algorithm limits the amount of data that can be transmitted at any given time. This project provides a unique and interactive way to understand the leaky bucket algorithm and its importance in network traffic management.

The implementation of the leaky bucket algorithm in OpenGL involves creating a graphical scene that represents the network traffic flow. The scene includes a bucket that fills up with water, which represents incoming data packets. The water level in the bucket gradually increases until it reaches a threshold level. At this point, the leaky bucket algorithm kicks in, and water starts to leak out of the bucket at a constant rate, representing the transmission rate of the network.

The user can interact with the scene by adjusting the rate at which the bucket fills and the rate at which it leaks. By changing these parameters, the user can observe how the algorithm limits the flow of data in the network, ensuring that the transmission rate does not exceed a certain level, preventing network congestion and potential data loss.

PROBLEM STATEMENT

The Leaky Bucket is an algorithm used in packet switched networks and telecommunications networks.The project helps in easy understanding of how packets are dealt with in a network.Input is the incoming packets in the form of a binary array.Given unit time interval, we have values to represent packets,1:incoming 0:no packet .A graph is then plotted showing conforming and non-conforming packets.We also have an animated bucket to show packets that travel through the network without hassles.Text strings help differentiate between conforming and non-conforming packets.

Design and develop a graphical simulation of the leaky bucket algorithm using OpenGL. The simulation should provide an interactive and visual representation of the algorithm's concept, including a graphical bucket that fills up with water (data packets) and leaks out at a constant rate (transmission rate). The simulation should allow users to adjust the fill rate and the leak rate to observe how the algorithm limits the flow of data in the network, preventing network congestion and potential data loss. The simulation should be user-friendly and easy to use, providing a fun and engaging way for students and network professionals to learn about network traffic management and flow control. The simulation should be implemented in a cross-platform manner, supporting multiple operating systems and hardware configurations. Finally, the simulation should be well-documented and properly tested to ensure its reliability and effectiveness.

OBJECTIVES

The specific objectives of this project can be outlined as follows:

1. To design and develop a graphical scene using OpenGL that represents the leaky bucket algorithm and its concepts.

2. To create a user-friendly interface that allows users to interact with the simulation by adjusting the fill rate and the leak rate of the bucket.

3. To provide a visual representation of the algorithm's workings, including how it limits the flow of data in the network, preventing network congestion and potential data loss.

4. To provide a fun and engaging learning experience for students and network professionals who are interested in understanding network traffic management and flow control.

5. To implement the simulation in a cross-platform manner, ensuring that it can run on different operating systems and hardware configurations.

6. To properly document and test the simulation to ensure its reliability and effectiveness.

Overall, the objective of this project is to use graphics and simulation to provide a unique and immersive learning experience for a complex technical concept, such as the leaky bucket algorithm.

:

HARDWARE / SOFTWARE Requirements

Hardware Requirements

- Platform used: UBUNTU
- Technology used: OpenGL Libraries such has OpenGL Utility library, OpenGL Utility toolkit
- Language: C

Software Specification

- Pentium 4
- 40 GB hard disk
- 128 MB RAM
- VGA Color Monitor
- Input devices-key board, mouse

Literature Survey

Literature survey for the implementation of the leaky bucket algorithm in OpenGL reveals several related works that have been done in the past. Some of the notable ones are:

1. "Visualization of network flow control mechanisms" by Klemetson, M., & Klemetson, S. (2007): This paper proposes a graphical simulation of various network flow control mechanisms, including the leaky bucket algorithm. The simulation provides a visual representation of how each mechanism operates and its effect on network traffic flow.

2. "Simulation of Leaky Bucket Algorithm for Congestion Control in Networks" by Bhushan, V., & Singh, R. (2014): This paper presents a simulation of the leaky bucket algorithm using MATLAB. The simulation provides a visual representation of the algorithm's operation and its impact on network traffic flow.

3. "Modeling and Simulation of the Leaky Bucket Algorithm in a Wireless Environment" by Al-Rousan, M., & Al-Qudah, M. (2017): This paper proposes a simulation of the leaky bucket algorithm in a wireless network environment. The simulation provides a visual representation of how the algorithm works in a wireless network setting and its impact on network performance.

4. "A simulation study of the Leaky Bucket algorithm for network congestion control" by Alshahrani, M. A., & Alqarni, A. S. (2019): This paper presents a simulation study of the leaky bucket algorithm for network congestion control using the NS3 simulator. The simulation provides a visual representation of the algorithm's effect on network traffic flow and its impact on network performance.

METHODOLOGY

The methodology for implementing the leaky bucket algorithm in OpenGL can be divided into several stages:

1. Design the graphical scene: The first step is to design the graphical scene that represents the leaky bucket algorithm. The scene should include a bucket that fills up with water (data packets) and leaks out at a constant rate (transmission rate). The scene should also include a control panel that allows users to adjust the fill rate and the leak rate of the bucket.

2. Implement the graphical scene using OpenGL: The next step is to implement the graphical scene using OpenGL. This involves writing code to create the bucket, fill it with water, and animate the water level as it leaks out of the bucket at a constant rate.

3. Implement the leaky bucket algorithm logic: The next step is to implement the leaky bucket algorithm logic. This involves writing code to regulate the rate at which data is transmitted between the source and the destination in the network. The algorithm should ensure that the rate of data transmission does not exceed a certain level, preventing network congestion and potential data loss.

4. Integrate the graphical scene and the algorithm logic: The next step is to integrate the graphical scene and the algorithm logic. This involves writing code to ensure that the animation of the water level in the bucket corresponds to the algorithm's operation.

5. Test the simulation: The final step is to test the simulation. This involves running the simulation with different values of the fill rate and the leak rate and observing how the algorithm regulates the flow of data in the network. The simulation should be tested on different hardware and operating system configurations to ensure its reliability and effectiveness.

Overall, the methodology for implementing the leaky bucket algorithm in OpenGL involves designing a graphical scene, implementing the scene using OpenGL, implementing the leaky bucket algorithm logic, integrating the scene and the algorithm logic, and testing the simulation to ensure its reliability and effectiveness.

IMPLEMENTATION

The implementation of the leaky bucket algorithm in OpenGL can be divided into several steps:

1. Setting up the OpenGL environment: This involves setting up the OpenGL environment by including the necessary header files and libraries and creating the OpenGL window.
2. Designing the graphical scene: The next step is to design the graphical scene that represents the leaky bucket algorithm. This involves creating a bucket object and animating the water level as it fills up and leaks out of the bucket at a constant rate.
3. Implementing the leaky bucket algorithm logic: The next step is to implement the leaky bucket algorithm logic. This involves writing code to regulate the rate at which data is transmitted between the source and the destination in the network. The algorithm should ensure that the rate of data transmission does not exceed a certain level, preventing network congestion and potential data loss.
4. Integrating the graphical scene and the algorithm logic: The next step is to integrate the graphical scene and the algorithm logic. This involves writing code to ensure that the animation of the water level in the bucket corresponds to the algorithm's operation.
5. Testing the simulation: The final step is to test the simulation. This involves running the simulation with different values of the fill rate and the leak rate and observing how the algorithm regulates the flow of data in the network. The simulation should be tested on different hardware and operating system configurations to ensure its reliability and effectiveness.

RESULTS AND DISCUSSIONS

Menus help navigate through the project giving the user privilege of choice.An exit option terminates the algorithm demo.I,the packet size,is 3 and L,the bucket depth,is 9.Right click to initiate the project once the binary array is fed.

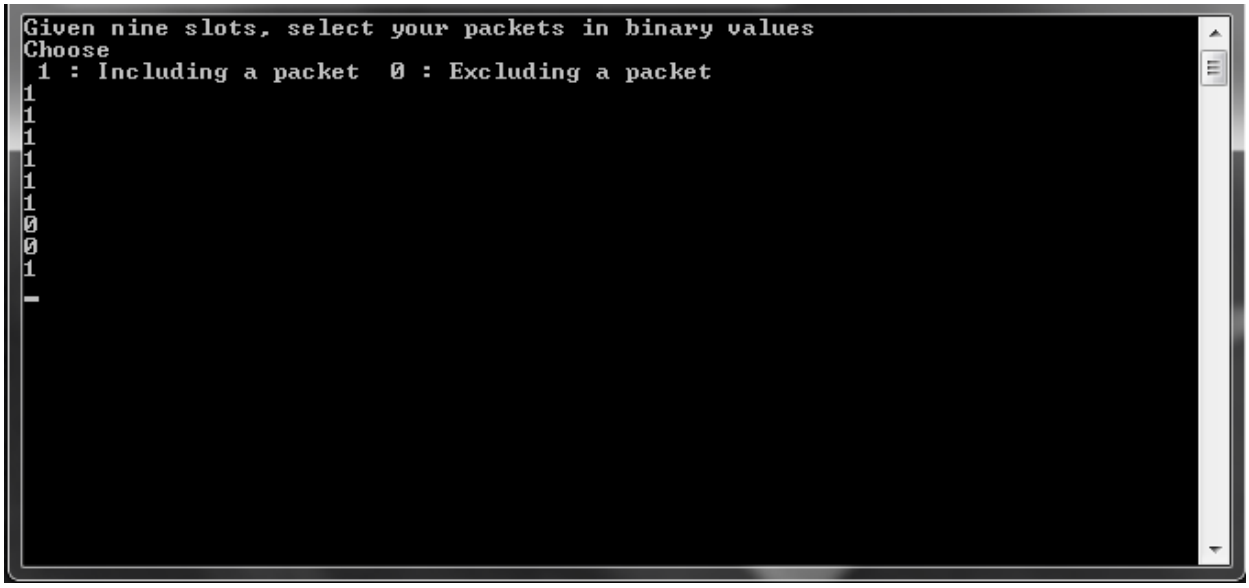

Fig 6.1: Input of packets via0's and 1's

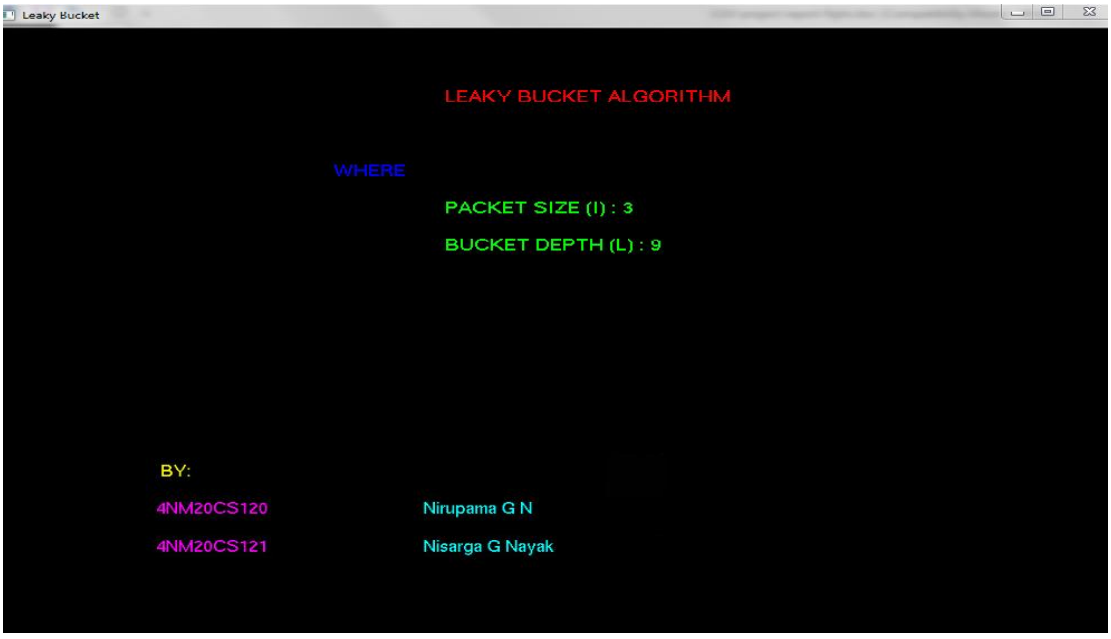

Fig 6.2: Initial screen: A cover page giving us firsthand information about the algorithm.

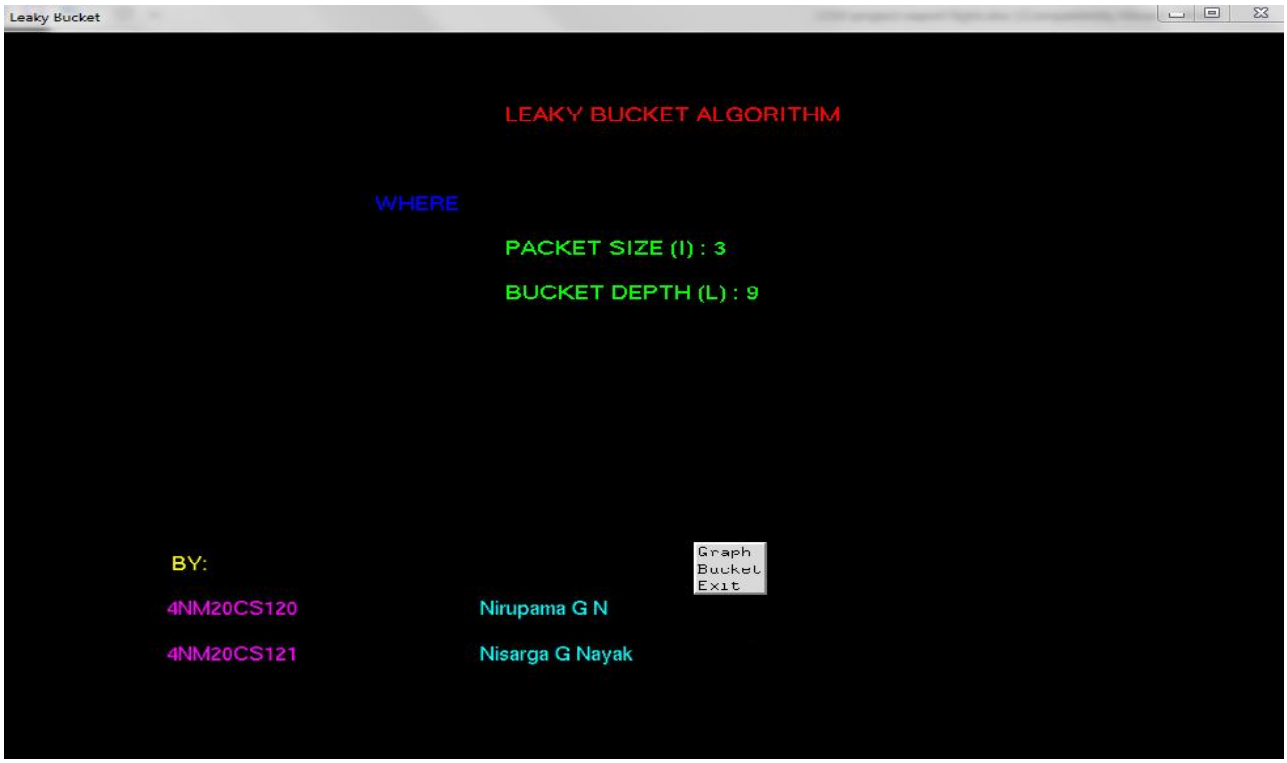

Fig 6.3:Drop down menu options.

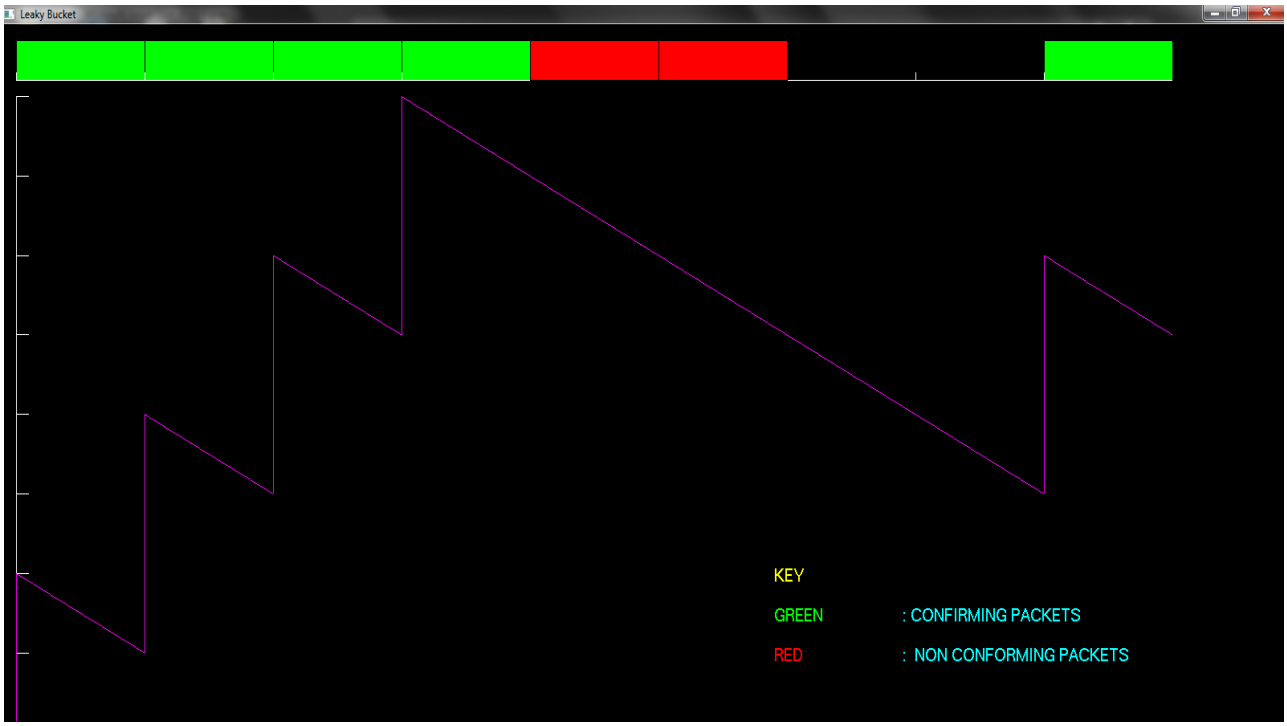

Fig 6.4:Plotting of the graph

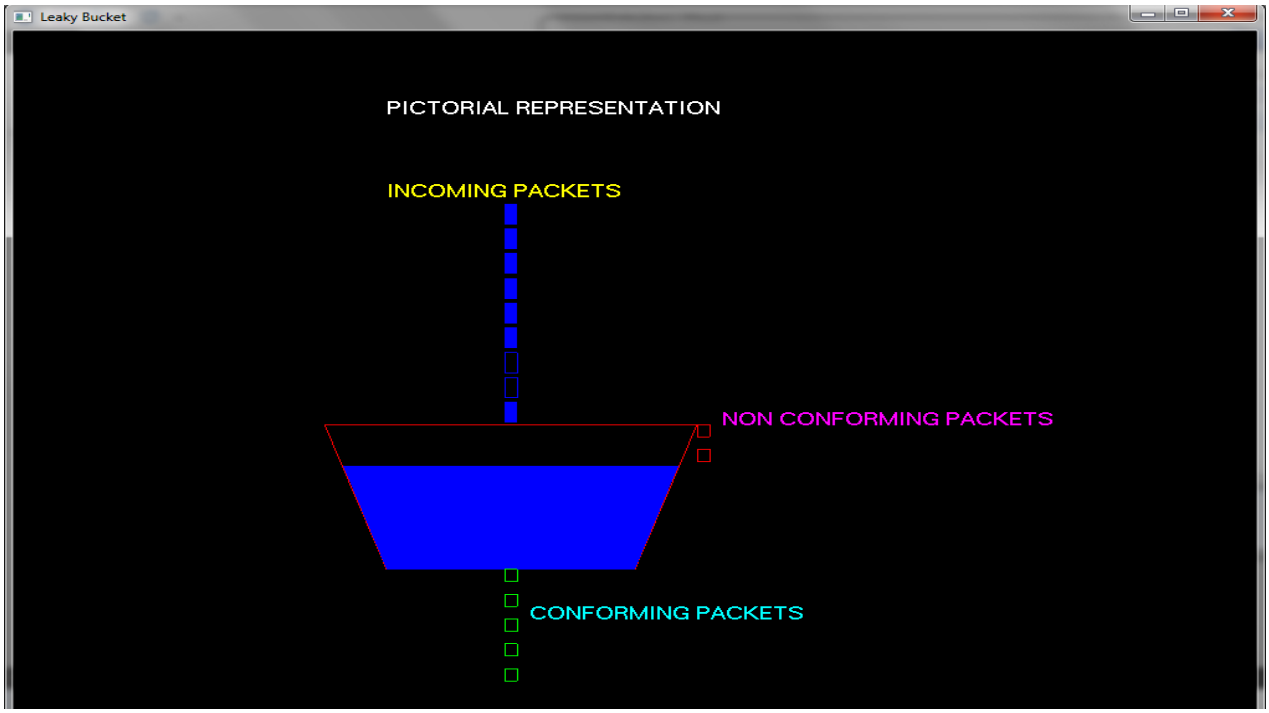

Fig 6.5:Bucket showing pictorial representation of the same(algorithm).

Fig 6.6: Exit

CONCLUSION AND FUTURE SCOPE

During the course of designing this project I have understood various concepts of OpenGL clearly.It was a challenging task as the topic is a mixture of OpenGL programming and computer networks . I have tried to my best potential to incorporate all the basic level requirements of a normal graphics package .

Owing to certain constraints I could not incorporate all the tasks that a graphics package should perform. However it meets the basic requirements of the user successfully. Since it's user friendly it enables the user to interact effectively and easily.

Special attention has been provided to generation of the graph, menus and deciding conforming and non-conforming packets.

This project has been designed using OpenGL libraries which works on the Ubuntu platform. The same can be done using other languages like Visual Studio and better graphical interfaces.Other improvements can be

- Higher languages like JAVA can be used to provide a new look to the project.
- We can also show the concept using bigger screen with smaller units to represent more number of packets.

REFERENCES

1) Edward Angel, “Interactive Computer Graphics”,5th edition, Pearson Education,2005

2) “Computer Graphics”, Addison-Wesley 1997 James D Foley, Andries Van Dam, Steven

3) http://basic4gl.wikispaces.com

4) http://www.wikipedia.com

5) Communication Networks –Fundamental Concepts and Key Architectures - Alberto Leon-Garcia and Indra Widjaja, 2nd Edition,