VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“Jnana Sangama”, Belagavi – 590 018

A

Mini Project Report

on

“RUBIKS CUBE”

Submitted in partial fulfillment of Computer Graphics Laboratory with Mini project 17CSL68 in Computer Science and Engineering for the Academic Year 2019-2020

Submitted by

GAGANA ANANDA

1GA17CS055

Under the Guidance of

Mrs. BhagyaShri R Hanji	**Mrs. Ashwini**
Assistant Professor	Assistant Professor

GLOBAL ACADEMY OF TECHNOLOGY

Department of Computer Science and Engineering

Rajarajeshwarinagar, Bengaluru - 560 098

2019 – 2020

GLOBAL ACADEMY OF TECHNOLOGY

Department of Computer Science and Engineering

CERTIFICATE

Certified that the VI Semester Mini Project in Computer Graphics Laboratory with Mini project Entitled **"RUBIKS CUBE"** carried out by **Gagana Ananda**, bearing **USN 1GA17CS055** is submitted in partial fulfillment for the award of the **Bachelor of Engineering** in Computer Science and Engineering from **Visvesvaraya Technological University, Belagavi** during the year 2019-2020.The Computer Graphics with Mini project report has been approved as it satisfies the academic requirements in respect of the mini project work prescribed for the said degree.

Dr. Bhagyashri R Hanji Assistant Professor, Dept of CSE, GAT, Bengaluru.	**Mrs. Haseeba Yaseen** Assistant Professor, Dept of CSE, GAT, Bengaluru.	**Dr. N Guruprasad** Professor & Head, Dept of CSE, GAT, Bengaluru.

Name of the Examiner — Signature with date

1. ____________________ ____________________

2. ____________________ ____________________

Acknowledgement

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant encouragement and guidance crowned our efforts with success.

I consider myself proud, to be part of **Global Academy of Technology** family, the institution which stood by our way in endeavors.

I express my deep and sincere thanks to our principal **Dr. N. Rana Pratap Reddy** for his support.

I would be grateful to **Dr. N Guruprasad,** Professor and HOD, Dept Of CSE who is source of inspiration and of invaluable help in channelizing my efforts in right direction.

I wish to thank my internal guides **Mrs. Bhagya Shri R Hanji** , Assistant Professor, Dept of CSE and **Mrs. Ashwini,** Assistant Professor, Dept of CSE for guiding and correcting various documents of mine with attention and care. They have taken lot of pain to go through the document and make necessary corrections as and when needed.

I would like to thank the faculty members and supporting staff of the Department of CSE, GAT for providing all the support for completing the Project work.

Finally, I am grateful to my parents and friends for their unconditional support and help during the course of my Project work.

GAGANA ANANDA
1GA17CS055

ABSTRACT

This project was developed on the basis of a popular game on most Sony Ericsson mobiles – Puzzle Slider.

The project improvised on this basic idea to convert a normal 2 Dimensional game into a graphical 3 Dimensional game.

The original game is a 2 Dimensional game that involves swapping of any 2 tiles and arranging the tiles in the correct sequence. We modified this idea to include a blank position. This now increases level of difficulty as only the tiles adjacent to the blank position can now be translated, unlike the Puzzle Slider game.

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION

1.1 INTRODUCTION TO COMPUTER GRAPHICS

Computer Graphics is concerned with all aspects of producing pictures or images using a computer. Graphics provides one of the most natural means of communicating within a computer, since our highly developed 2D and 3D pattern-recognition abilities allow us to perceive and process pictorial data rapidly and effectively. Interactive computer graphics is the most important means of producing pictures since the invention of photography and television.

Applications of Computer Graphics

1. Display of information
2. Design
3. Simulation and animation
4. User interfaces

The Graphics Architecture

Graphics Architecture can be made up of seven components:

1. Display processors
2. Pipeline architectures
3. The graphics pipeline
4. Vertex processing
5. Clipping and primitive assembly
6. Rasterization
7. Fragment processing

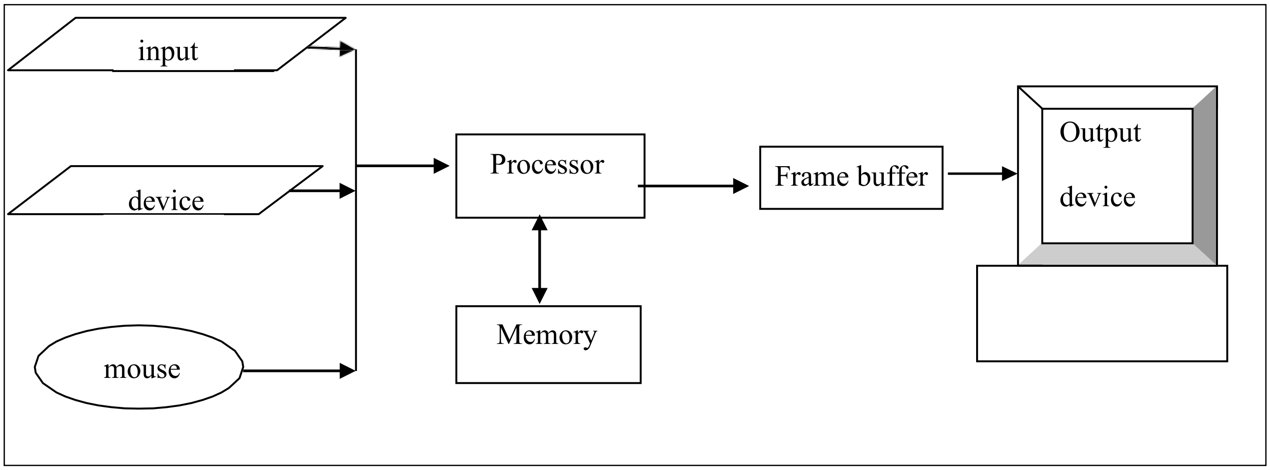

Figure 1.1: Components of Graphics Architecture and their working

1.2 INTRODUCTION TO OPENGL

OpenGL is software used to implement computer graphics. The structure of OpenGL is similar to that of most modern APIs including Java 3D and DirectX. OpenGL is easy to learn, compared with other.

APIs are nevertheless powerful. It supports the simple 2D and 3D programs. It also supports the advanced rendering techniques. OpenGL API explains following 3 components

1. Graphics functions
2. Graphics pipeline and state machines
3. The OpenGL interfaces

There are so many polygon types in OpenGL like triangles, quadrilaterals, strips and fans. There are 2 control functions, which will explain OpenGL through,

1. Interaction with window system
2. Aspect ratio and view ports

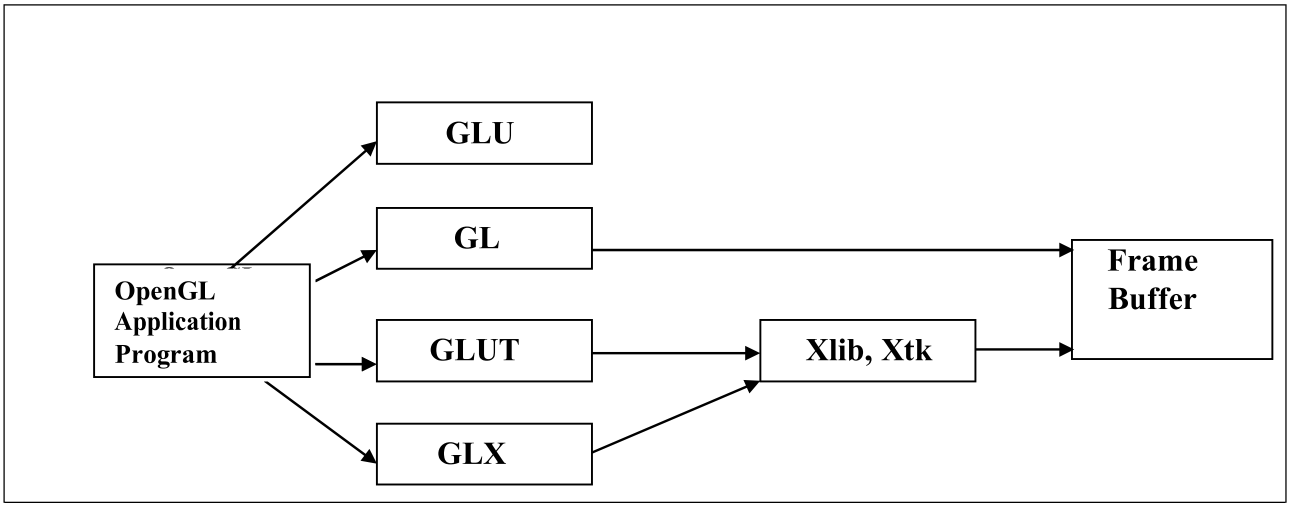

Figure 1.2: OpenGL Library organization

Most implementations of OpenGL have a similar order of operations, a series of processing stages called the OpenGL rendering pipeline. This ordering, as shown in Figure 1.2, is not a strict rule of how OpenGL is implemented but provides a reliable guide for predicting what OpenGL will do. The following diagram shows the assembly line approach; which OpenGL takes to process data. Geometric data (vertices, lines, and polygons) follow the path through the row of boxes that includes evaluators and per-vertex operations, while pixel data (pixels, images, and bitmaps) are treated differently for part of the process. Both types of data undergo the same final steps before the final pixel data is written into the frame buffer.

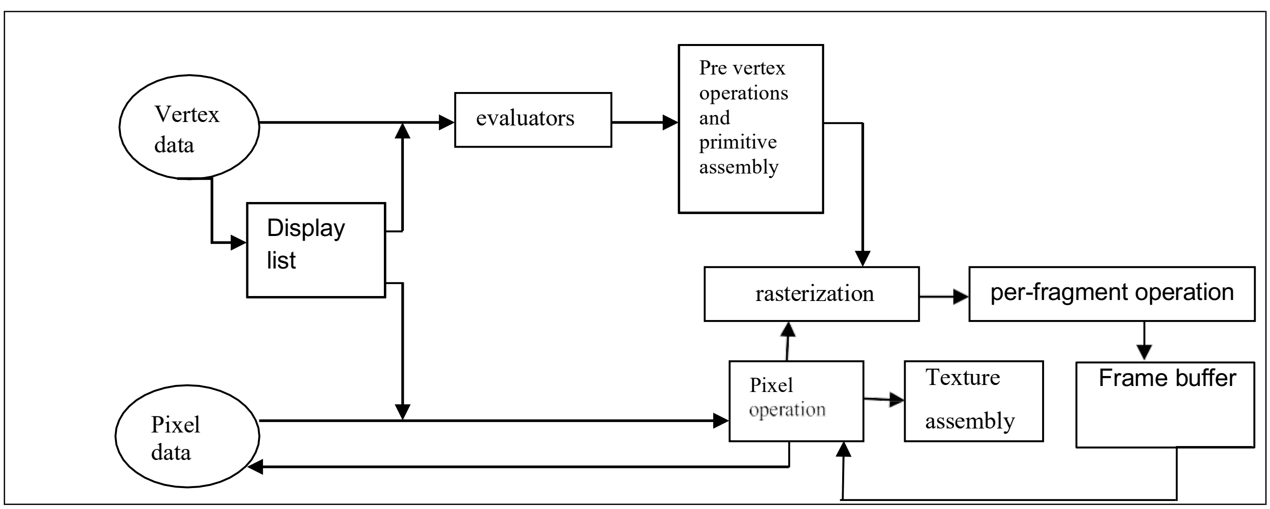

Figure 1.3: OpenGL Order of Operation

CHAPTER 2

REQUIREMENTS SPECIFICATION

2.1 SOFTWARE REQUIREMENTS

- Operating system – Windows 10
- Code::Blocks 17.12
- OPENGL library files – GL, GLU, GLUT
- Language used is C/C++

2.2 HARDWARE REQUIREMENTS

- Processor – Intel i5 7^{th} Gen
- Memory – 8GB RAM
- 1TB Hard Disk Drive
- Mouse or other pointing device
- Keyboard
- Display device

CHAPTER 3

SYSTEM DEFINITION

3.1 PROJECT DESCRIPTION

OpenGL is software which provides a graphical interface. It is an interface between the application program and the graphics hardware.

The goal of the Project is to Create a system which generates simulation of the Rubiks cube which interact to command given by the user. This Project Consists of many user defined functions such as controlling the rubiks cube by using keyboard keys. The user can also interact with program through mouse, keyboard functions.

3.2 USER DEFINED FUNCTIONS

- **void polygon ():** This function will show the structure of the cube.
- **void colorcube ():** This function will show the structure of the color of the cube.
- **void speedmeter ():** This function will show the speed of rotation of the cube.
- **void transpose ():** This function is used to define the transpose for all the six faces.
- **void topc():** This function is used to assign the values when the operation is rotation of the top face.
- **void frontc ():** This function is used to assign the values when the operation is rotation of the front face.
- **void rightc ():** This function is used to assign the values when the operation is rotation of the right face
- **void leftc ():** This function is used to assign the values when the operation is rotation of the left face.

- **void backc ():** This function is used to assign the values when the operation is rotation of the back face
- **void bottomc ():** This function is used to assign the values when the operation is rotation of the bottom face
- **void spincube():** This function is an idle callback which rotates the cube accordingly , if rotation is one and inverse is zero then rotate the top face of the cube by 90 degrees in the clockwise direction and if the inverse is one then rotate the same face by 90 degrees in the anti-clockwise direction . The same procedure is implemented for the other faces of the cube with different rotation values .
- **void motion ():** This function is used to rotate the cube about the selective axis .
- **void mouse ():** This function is used to allow the user to give input through mouse buttons.
- **void keyboard ():** This function is used to allow the user to rotate the faces of the cube using keyboard keys.
- **void myreshape ():** This function is used to define a viewport and set the matrix to projection and modelview matrics
- **void mymenu ():** This function is used to Define the actions corresponding to each entry in the menu.

3.3 THE FLOW OF THE PROGRAM

- The interaction between the windowing system and OPENGL is initiated. We initialize the window size and window position.
- Display function is called where the functions and operations for the Rubiks cube are defined.
- Using the left mouse button the cube can be rotated along the required axis.
- Using the right mouse button menus can be viewed, by selecting the options from the menu the required face of the cube can be rotated either clockwise or anti-clockwise.
- The rotation of faces of the cube can also be done using keys from the keyboard.
- Press the key ‘a’ to rotate the top face of the cube in the clockwise direction.
- Press the key ‘q’ to rotate the top face of the cube in the anti-clockwise direction.
- Press the key ‘s’ to rotate the right face of the cube in the clockwise direction.
- Press the key ‘w’ to rotate the right face of the cube in the anti-clockwise direction.
- Press the key ‘d’ to rotate the front face of the cube in the clockwise direction.
- Press the key ‘e’ to rotate the front face of the cube in the anti-clockwise direction.
- Press the key ‘f’ to rotate the left face of the cube in the clockwise direction.

CHAPTER 4
IMPLEMENTATION

4.1 SOURCE CODE

This is a snippet of the program is implemented using various openGL functions.

```
int main(int argc, char** argv)
{
   glutInit(&argc, argv);
   glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
   glutInitWindowSize(500, 500);
   glutCreateWindow("RUBIK'S CUBE");
   glutReshapeFunc(myreshape);
   glutIdleFunc(spincube);
   glutMouseFunc(mouse);
   glutMotionFunc(motion);
   glutCreateMenu(mymenu);
   glutAddMenuEntry("Top              :a", 1);
   glutAddMenuEntry("Top Inverted     :q", 2);
   glutAddMenuEntry("Right            :s", 3);
   glutAddMenuEntry("Right Inverted   :w", 4);
   glutAddMenuEntry("Front            :d", 5);
   glutAddMenuEntry("Front Inverted   :e", 6);
   glutAddMenuEntry("Left             :f", 7);
   glutAddMenuEntry("Left Inverted    :r", 8);
   glutAddMenuEntry("Back             :g", 9);
   glutAddMenuEntry("Back Inverted    :t", 10);
   glutAddMenuEntry("Bottom           :h", 11);
   glutAddMenuEntry("Bottom Inverted :y", 12);
   glutAddMenuEntry("Exit", 13);
   glutAttachMenu(GLUT_RIGHT_BUTTON);
   glutKeyboardFunc(keyboard);
   glutDisplayFunc(display);
   glEnable(GL_DEPTH_TEST);
   glutMainLoop();
   //return 0;
}
```

CHAPTER 4

TESTING AND RESULTS

5.1 DIFFERENT TYPES OF TESTING

1. Unit Testing

Individual components are tested to ensure that they operate correctly. Each component is tested independently, without other system components.

2. Module Testing

A module is a collection of dependent components such as a object class, an abstract Data type or some looser collection of procedures and functions. A module related Components, so can be tested without other system modules.

3. System Testing

This is concerned with finding errors that result from unanticipated interaction between Sub-system interface problems.

4. Acceptance Testing

The system is tested with data supplied by the system customer rather than simulated test data.

5.2 TEST CASES

The test cases provided here test the most important features of the project.

Table 5.2.1: Test Case

Sl No	Test Input	Expected Results	Observed Results	Remarks
1.	a	Rotate top face in clockwise direction	Moves Successfully	Pass
2.	s	Rotate right face in clockwise direction	Moves Successfully	Pass
3.	d	Rotate front face in clockwise direction	Moves Successfully	Pass
4.	f	Rotate left face in clockwise direction	Moves Successfully	Pass
5.	g	Rotate back face in clockwise direction	Moves Successfully	Pass
6.	h	Rotate bottom face in clockwise direction	Moves Successfully	Pass
7.	q	Rotate top face in anti-clockwise direction	Moves Successfully	Pass
8.	w	Rotate right face in anti-clockwise direction	Moves Successfully	Pass
9.	e	Rotate front face in anti-clockwise direction	Moves Successfully	Pass
10.	r	Rotate left face in anti-clockwise direction	Moves Successfully	Pass
11.	t	Rotate back face in anti-clockwise direction	Moves Successfully	Pass
12.	y	Rotate bottom face in anti-clockwise direction	Moves Successfully	Pass
13.	m	Increase speed of rotation	Moves Successfully	Pass
14.	n	Decrease speed of rotation	Moves Successfully	Pass
15.	o	Automatic solving	Moves Successfully	Pass

CHAPTER 6

SNAPSHOTS

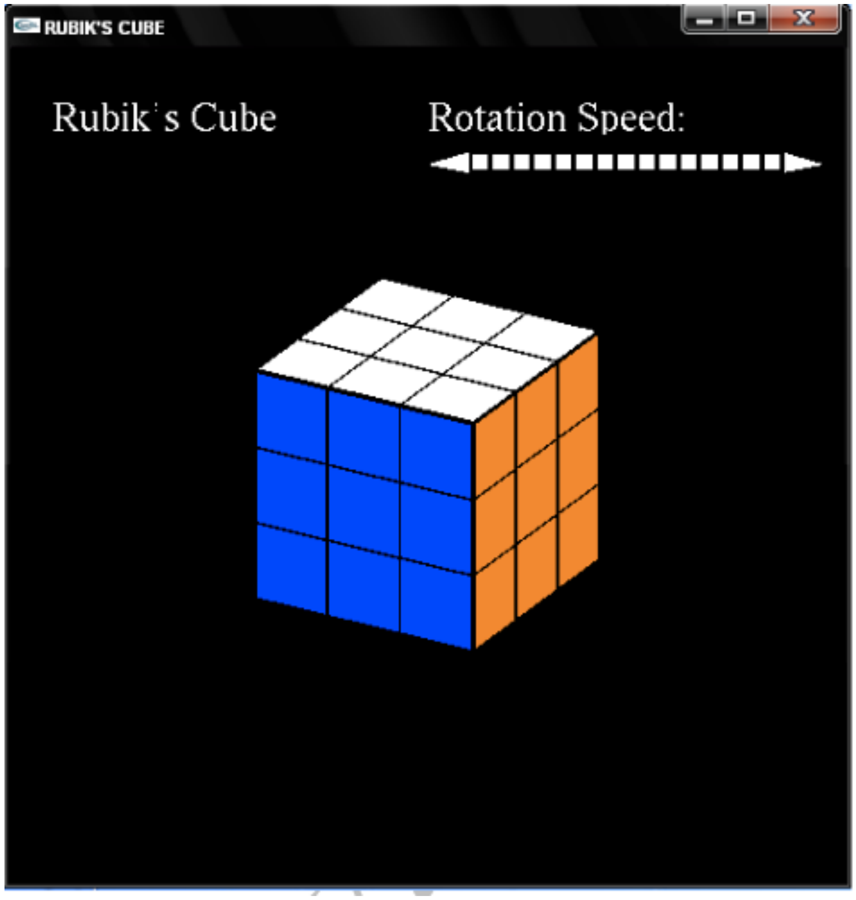

Fig 6.1: Initial view of Rubiks cube

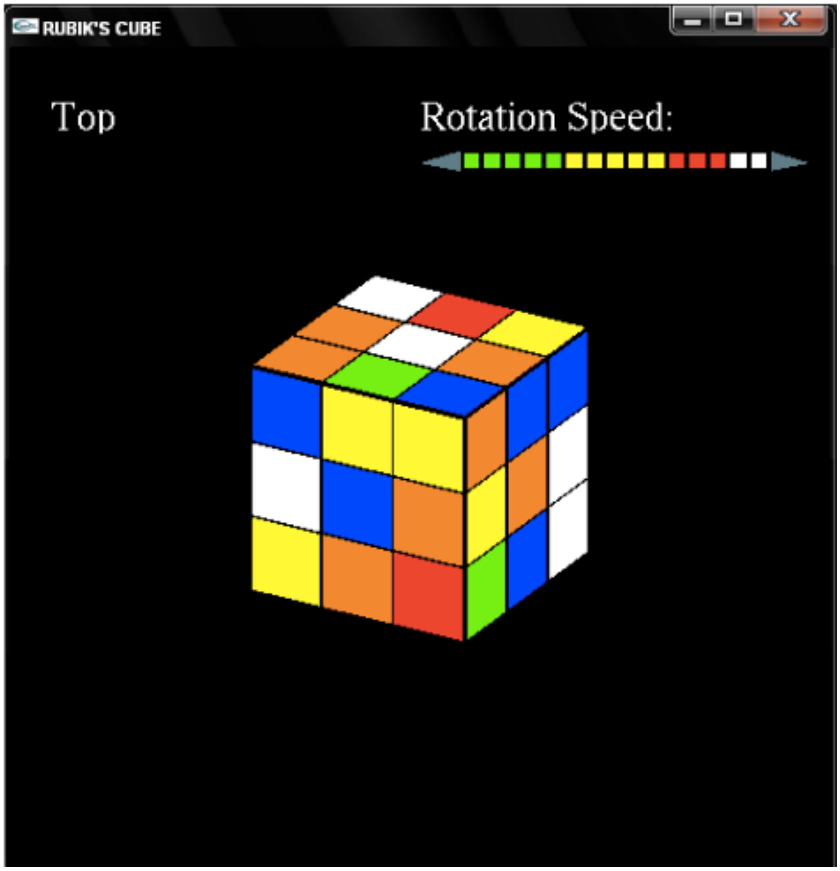

Fig 6.2 Suffled Rubiks cube

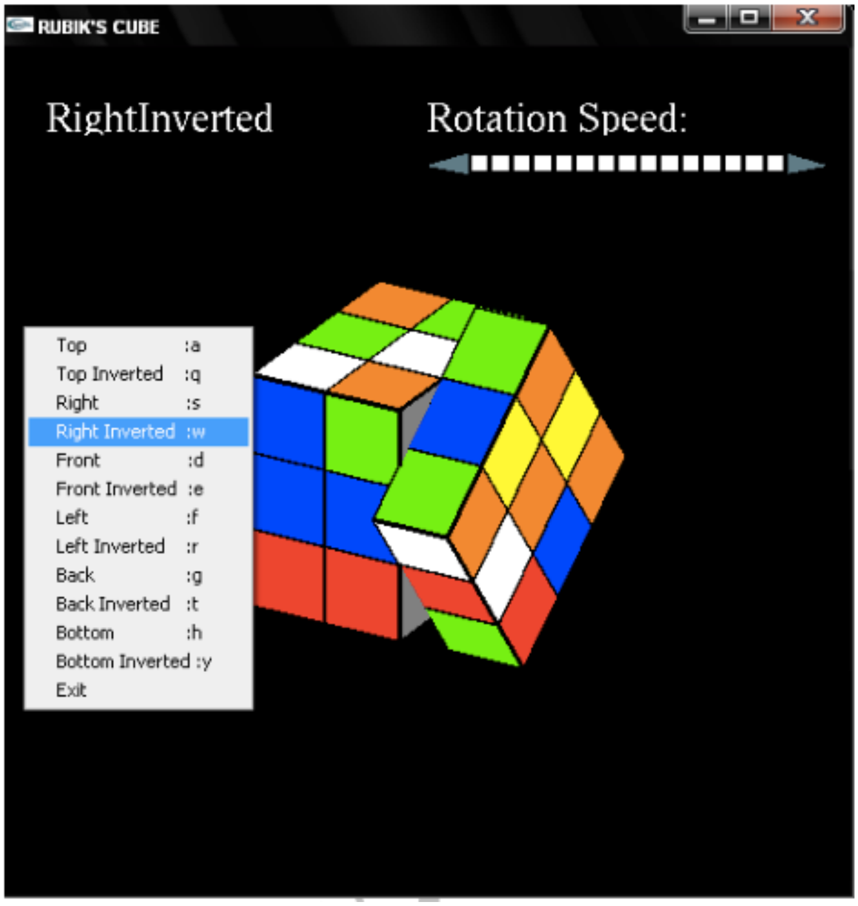

Fig 6.3 Right-Inversion using menu

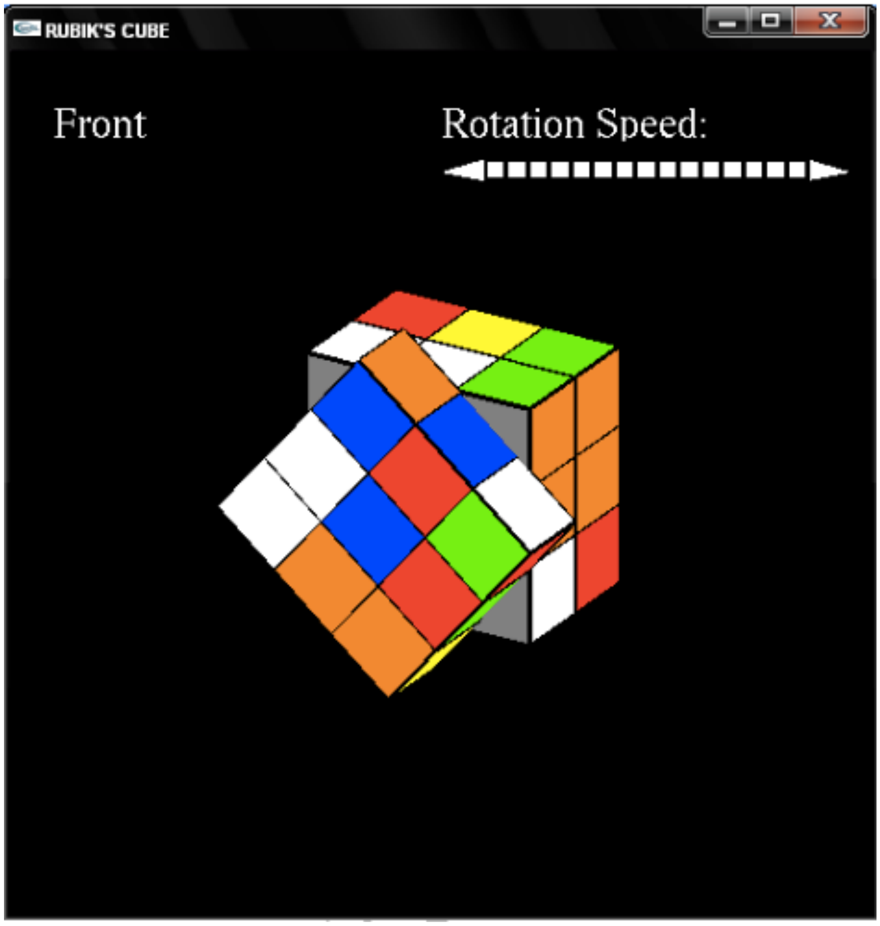

Fig 6.4 Front face rotation

CONCLUSION

This project has been successful in demonstrating rubiks cube using a variety of features and options present in OpenGL. This project combines the richness of the graphics library and programming skills. The development of this project was very helpful to demonstrate the working of rubiks cube. Designing this project, gave a good learning experience. It helped a lot to learn about computer graphics and design of graphical interfaces. This project thought how the inbuilt and user defined functions can be integrated to produce an efficient working code. The development of the mini project has given a good exposure to OpenGL by which some of the techniques which help in development of animated pictures and gaming were learnt.

The user-friendly interface allows the user to interact with it very effectively. So, I conclude on note that this project has given me a great exposure to the OpenGL and computer graphics. This is very reliable graphics package supporting various primitive objects like polygon, line loops, etc. Also, color selection, menu and mouse-based interface are included. Transformations like translation, rotation, scaling is also provided.

BIBLIOGRAPHY

References

[1]. Donald Hearn & Pauline Baker: Computer Graphics with OpenGL Version,3rd / 4th Edition, Pearson Education,2011

[2]. Edward Angel: Interactive Computer Graphics- A Top Down approach with OpenGL, 5th edition. Pearson Education, 2008

[3]. https://open.gl/

[4]. https://en.wikipedia.org/wiki/FreeGLUT

[5]. http://jerome.jouvie.free.fr/OpenGl/Lessons/Lesson3.php