Chapter 1

INTRODUCTION

1.1 Introduction to Computer Graphics

Computer graphics started with the display of data on hardcopy plotters and Cathode Ray Tube (CRT) screens soon after the Introduction of computers themselves. It has grown to include the Creation, Storage and Manipulation of Models and Images of objects. These models come from a diverse and expanding set of fields, and include physical, mathematical, engineering, architectural and even conceptual structures, natural phenomenon and so on. Computer graphics today is largely interactive: the user controls the contents, structure, and appearance of objects and of their displayed images by using input devices, such as a keyboard, mouse, or touch-sensitive panel on the screen. Because of the close relationship between the input devices and the display, the handling of such devices is included in the study of computer graphics.

1.2 Uses of Computer Graphics:

Computer graphics is used in many different areas of industry, business, government, education, entertainment etc.

User Interfaces

Word-processing, spreadsheet and desktop-publishing programs are typical applications of such user-interface techniques.

Interactive Plotting in Business, Science and Technology

The common use of graphics is to create 2D and 3D graphs of mathematical, physical and economic functions, histograms, and bar and pie charts.

Computer Aided Drafting and Design (CAD)

In CAD, interactive graphics is used to design components and systems of mechanical, electrical and electronic devices including structures such as buildings, automobile bodies, aero planes, ship hulls etc.

Simulation and Animation for Scientific Visualization and Entertainment

Computer-produced animated movies are becoming increasing popular for scientific and engineering visualization. Cartoon characters will increasingly be modeled in the computer as 3D shape descriptions whose movements are controlled by computer commands.

2D Graphics

These editors are used to draw 2D pictures (line, rectangle, circle and ellipse) alter those with operations like cut, copy and paste. These may also support features like translation, rotation etc.

3D Graphics

These editors are used to draw 3D pictures (line, rectangle, circle and ellipse).These may also support features like translation, rotation etc.

About the project:

This project implements the movement of light source on the surfaces of the different objects.

1.3 ADVANTAGES

- Scientific visualization
- Information visualization
- Computer vision
- Image processing
- Computational geometry
- Computational topology
- Applied mathematics

1.4 Header Files

Most of our applications will be designed to access OpenGL directly through functions in three libraries. Functions in the main GL library have names that begin with the letters *gl* and are stored in a library usually referred as GL. The second is the OpenGL Utility Library (GLU). This library uses only GL functions but contains code for creating common objects and simplifying viewing.

To interface with the window system and to get input from external devices into our programs, we need at least one library. For the X window system, this library is called GLX, for windows, it is wall etc.

GLUT will use GLX and X libraries.

#include<GL/glut.h>

OR

#include<GLUT/glut.h>

GLUT is the OpenGL Utility Toolkit, a window system independent toolkit for writing OpenGL programs. It implements a simple windowing application programming interface (API) for OpenGL. GLUT makes it considerably easier to learn about and explore OpenGL programming. GLUT provides a portable API so you can write a single OpenGL program that works across all PC and workstation OS platforms.

GLUT is designed for constructing small to medium sized OpenGL programs. While GLUT is well-suited to learning OpenGL and developing simple OpenGL applications, GLUT is not a full-featured toolkit so large applications requiring sophisticated user interfaces are better off using native window system toolkits. GLUT is simple, easy, and small.

The GLUT library has C, C++ (same as C), FORTRAN, and ADA programming bindings. The GLUT source code distribution is portable to nearly all OpenGL implementations and platforms. The current version is 3.7. Additional releases of the library are not anticipated. GLUT is not open source.

Chapter 2

LITERATURE SURVEY

- ## Inspiration and source of the idea:

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.

Chapter 3

SYSTEM REQUIREMENTS

System requirements are intended to communicate in precise way, the functions that the system must provide. To reduce ambiguity, they may be written in a structured form of natural language supplemented by tables and system models.

3.1 HARDWARE REQUIREMENTS

The physical components required are:

- Processor - Pentium Pro
- Memory - 128MB RAM
- 40GB Hard Disk Drive

3.2 SOFTWARE REQUIREMENTS

The software used in building this program are as specified:-

- Operating system – LINUX(UBUNTU 10.4)
- Tools : Eclipse
- Graphics Library – glut.h
- OpenGL 2.0

Chapter 4

IMPLEMENTATION

- **file_gen** is used to check whether high score file already exists. If it does not, it will create. If it does, it will read data from existing file.
- **render_header** to display the player's name along with a running timer. The timer gets its values from the clock class, which maintains the time elapsed since the game started in hour, minutes and seconds.
- **hiscore_check** to determine the position of the player in the highscores list. If he completes the game in a time lesser than the top 10 players, it implements an insertion sort to insert the player's name and time into the existing high score list.
- **write_file** is used to write a text file of the current player class. The player class contains details of the player name and the time he took to complete the game. All these details, along with the pre-existing records of the highscores are arranged in ascending order of time and are written into a text file.
- **sub** is the display function for the sub window created to display the highscores. This window is activated when the user chooses the option "highscores" from the menu obtained on right click of the mouse.

Mouse events:

When mouse event occurs, the ASCII code for the corresponding coordinates that generate the event and the location of mouse are returned. Mouse callback function is

glutMouseFunc (mouse);

Void mouse (int btn, int state, int x, int y) {

if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN)

begin = x;

}

Menu Entry:

GLUT provides one additional feature, pop_up menus, which we can use with the mouse to create sophisticated interactive application

glutCreateMenu ();

glutAddMenuEntry ();

glutAttachMenu (GLUT_RIGHT_BUTTON);

4.1 DESIGN:

The flow chart describes how the path travels.

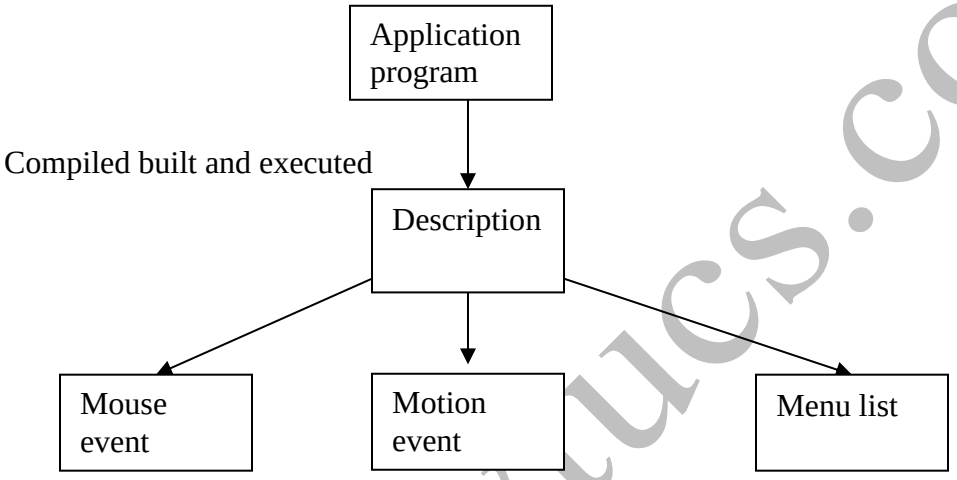


Fig 4.1: Data flow diagram.

- Figure 4.1 under Implementation Chapter explains the Data Flow Diagram.

  This includes the flow from application program to handling different events like Mouse event, Motion event and the Menu list.

4.2 ALGORITHM:

STEP 1: Define vertices for 27 cubes which are used to compose one whole cube known as "Rubiks cube".

STEP 2: Define colors for each cube of the Rubiks cube to distinguish one cube from the other and as well as color for the speed meter, which is used to control the speed of rotation.

Output( ) function:

STEP 3: This function is used to display the message using the Commands glutBitmapCharacter( ) and glRasterpos*( ).

Polygon( ) function:

STEP 4: Draw a polygon via list of vertices with the line of 3 pixels wide.

Colorcube( ) function:

STEP 5: Map vertices to faces. Hence the use of colorcube function has done 27 times with different prefixes for all the 27 cubes amongst which one with no color, 6 with one color, 12 with two colors and 8 with three colors on it.

Speedmeter( ) function:

STEP 6: This function is used to define vertices for a speedmeter, which is used to control the speed of rotation.

Display( ) function:

STEP 7: Clear the frame buffers and the z-buffer.

STEP 8: Invoke the function speedmeter and output.

STEP 9: Using the variables rotation and inverse the rotation for the faces are defined, where if rotation is one and the inverse flag is zero then the top face of the cube will be rotated in the clockwise direction and incase if rotation is one and also is the inverse flag then the top face of the cube will be rotated in the anti-clockwise direction.

STEP 10: The same procedure that has been specified in the step 9 is adopted with different values for the rotation, such as rotation with the value two is implemented for right three for front, four for left, five for back and six for

bottom rotations respectively with the implementation of inverse variable being same.

STEP 11: Invoke the output function and swap the buffers.

Transpose( ) function:

STEP 12: This function is used to define the transpose for all the six faces.

Topc( ) function:

STEP 13: This function is used to assign the values when the operation is rotation of the top face.

Frontc( ) function:

STEP 14: This function is used to assign the values when the operation is rotation of the front face.

Rightc( ) function:

STEP 15: This function is used to assign the values when the operation is rotation of the right face.

Leftc( ) function:

STEP 16: This function is used to assign the values when the operation is rotation of the left face.

Backc( ) function:

STEP 17: This function is used to assign the values when the operation is rotation of the back face.

Bottomc( ) function:

STEP 18: This function is used to assign the values when the operation is rotation of the bottom face.

Spincube( ) function:

STEP 19: This is 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 inverse is one then rotate the same face by 90 degree in the anti-clockwise direction.

STEP 20: The same procedure is implemented for the other faces of the cube with different rotation values.

STEP 21: The cube is redisplayed after the rotation.

Motion( ) function:

STEP 22: This is used to rotate the cube about the selected axis.

Mouse( ) function:

STEP 23: Mouse callback function. This allows user to give input through mouse buttons.

Keyboard( ) function:

STEP 24: use the keys 'a','s','d','f','g','h' to rotate the faces of the accordingly in the clockwise direction and the keys 'q','w','e','r','t','y' to rotate in the anti-clockwise direction.

STEP 25: use the keys '1','2','4','5','6','8','9' are used to move the viewer along the axis.

STEP 26: use the keys 'm' and 'n' to alter the value of the rotation meter which controls the speed of the rotation.

STEP 27: use the key 'o' for the automatic solving of the cube.

Myreshape( ) function:

STEP 28: Define a viewport and set the matrix to projection and modelview matrices.

Mymenu( ) function:

STEP 29: Define the actions corresponding to each entry in the menu.

Main( ) function:

STEP 30: Both double and z-buffer is enabled. Invoke the start function.

STEP 31: Add entries to the menu and link the menu to the right mouse button.

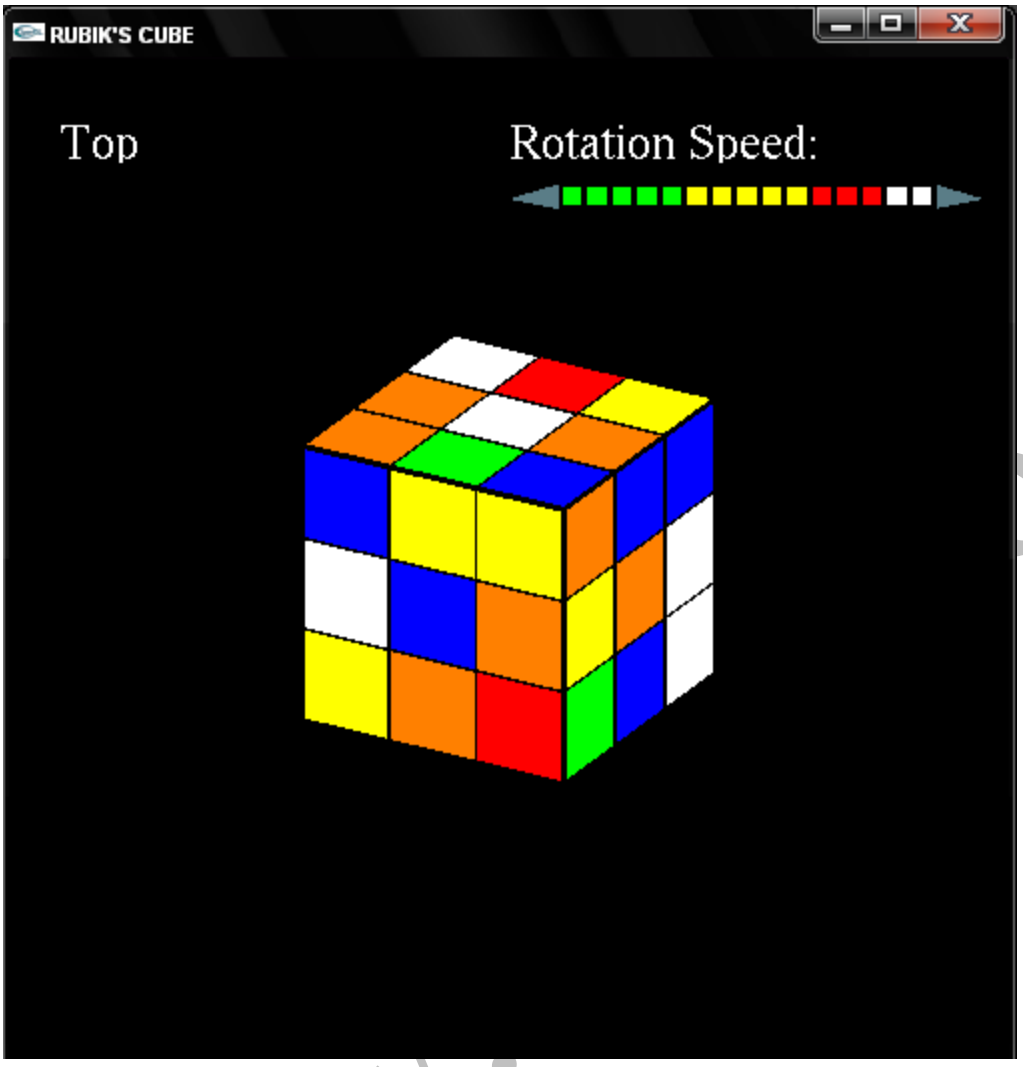
STEP 32: Stop.

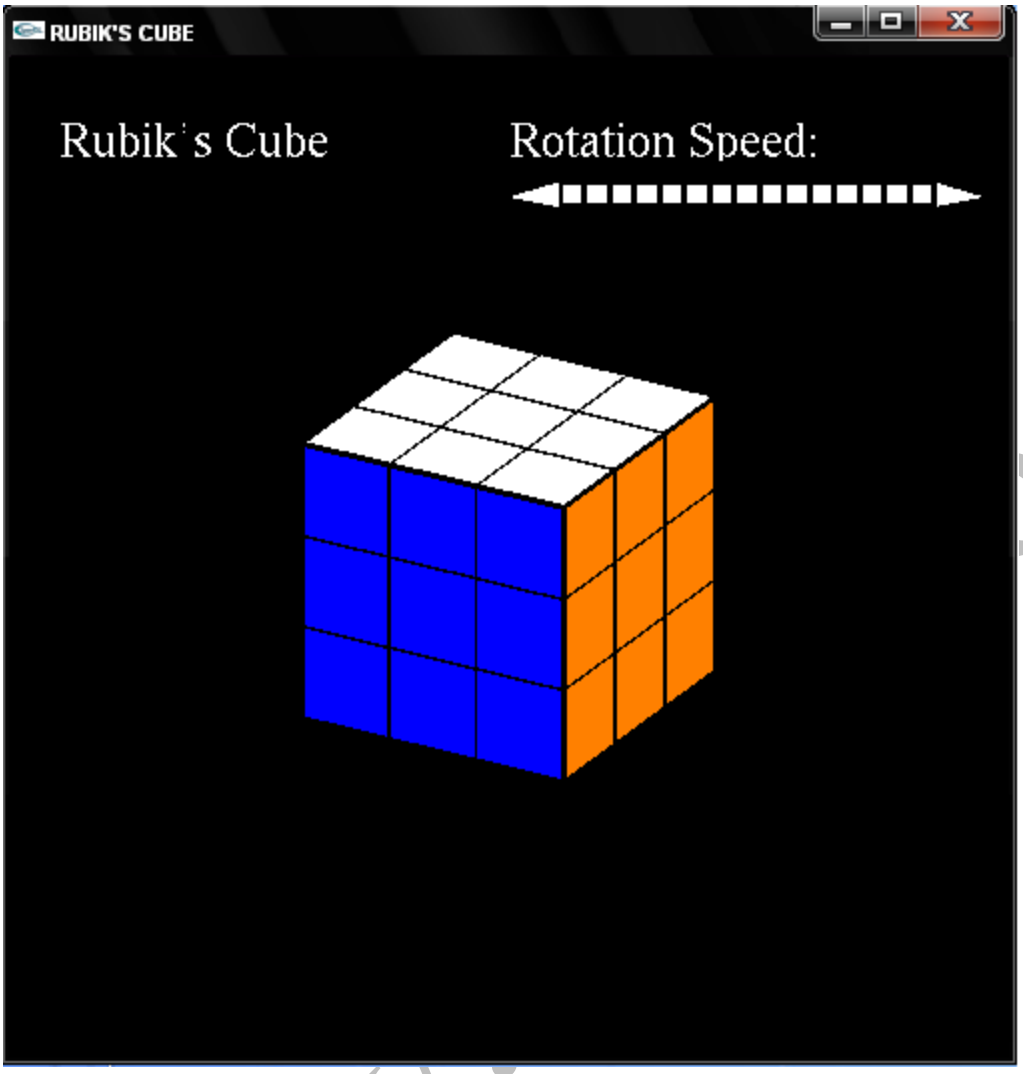
4.3 FLOW OF THE PROGRAM:

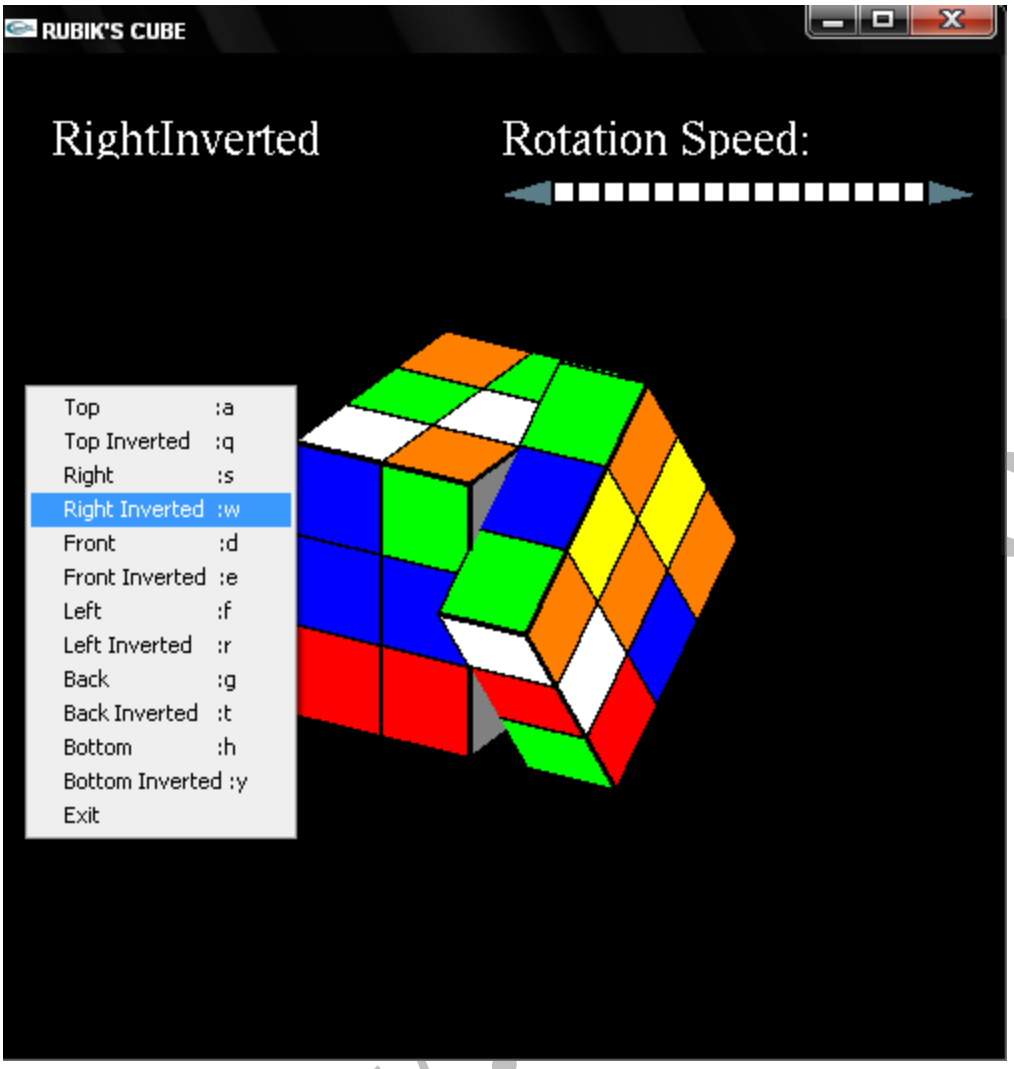
1. The interaction between the windowing system and OPENGL is initiated. We initialize the window size and window position.
2. Display function is called where the functions and operations for the Rubiks cube are defined.
3. Using the left mouse button the cube can be rotated along the required axis.
4. 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.
5. The rotation of faces of the cube can also be done using keys from the keyboard.
6. Press the key ‘a’ to rotate the top face of the cube in the clockwise direction.
7. Press the key ‘q’ to rotate the top face of the cube in the anti-clockwise direction.
8. Press the key ‘s’ to rotate the right face of the cube in the clockwise direction.
9. Press the key ‘w’ to rotate the right face of the cube in the anti-clockwise direction.
10. Press the key ‘d’ to rotate the front face of the cube in the clockwise direction.
11. Press the key ‘e’ to rotate the front face of the cube in the anti-clockwise direction.
12. Press the key ‘f’ to rotate the left face of the cube in the clockwise direction.

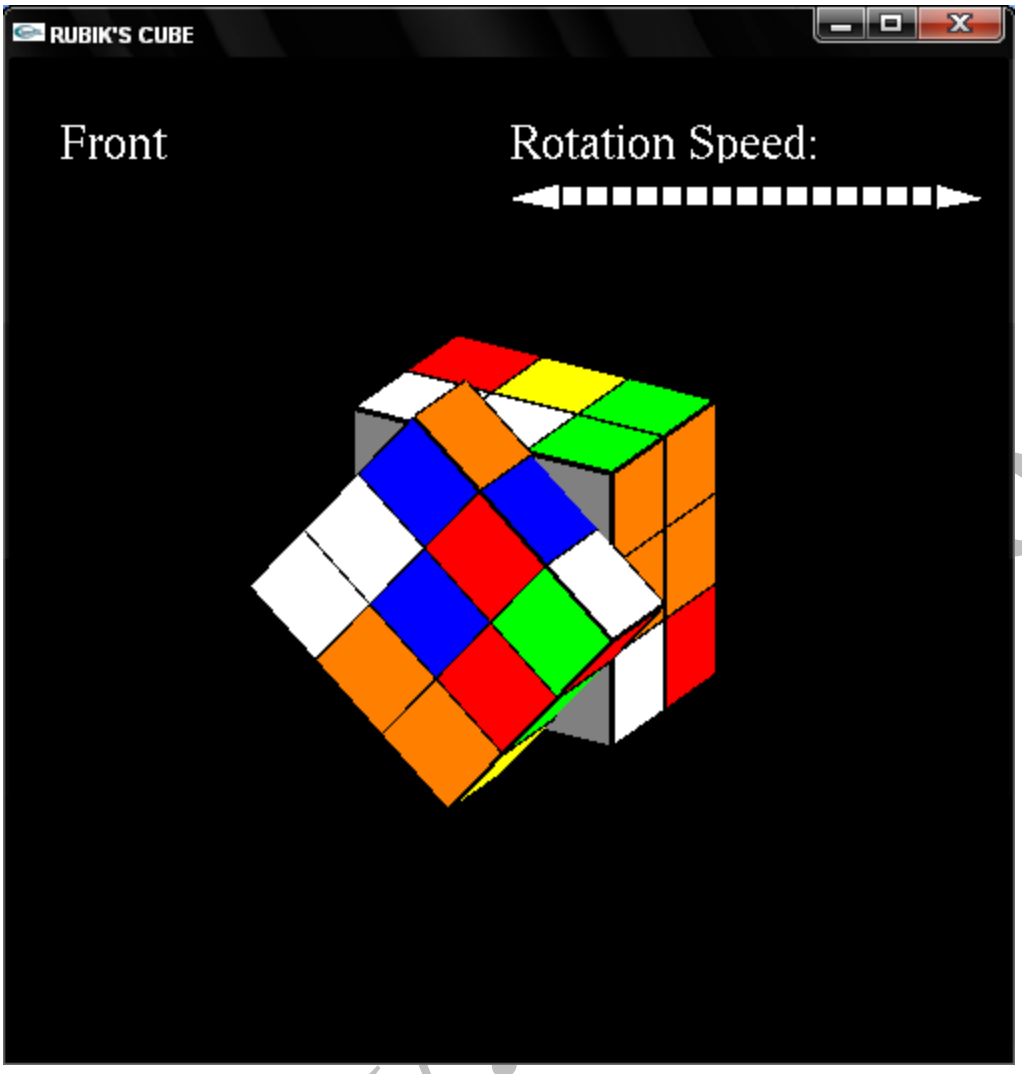
13. Press the key ‘r’ to rotate the left face of the cube in the anti-clockwise direction.
14. Press the key ‘g’ to rotate the back face of the cube in the clockwise direction.
15. Press the key ‘t’ to rotate the back face of the cube in the anti-clockwise direction.
16. Press the key ‘h’ to rotate the bottom face of the cube in the clockwise direction.
17. Press the key ‘y’ to rotate the bottom face of the cube in the anti-clockwise direction.
19. Press the key ‘o’ for the automatic solving.
20. Press the keys ‘m’ and ‘n’ to control the speed of rotation.
21. Use the keys ‘1’,’2’,’4’,’5’,’6’,’8’,’9’ to rotate the cube along different axes.

Screen Shots






RUBIK'S CUBE
RightInverted
Rotation Speed:
Top :a
Top Inverted :q
Right :s
Right Inverted :w
Front :d
Front Inverted :e
Left :f
Left Inverted :r
Back :g
Back Inverted :t
Bottom :h
Bottom Inverted :y
Exit



Chapter 5
CONCLUSION & FUTURE IDEAS

We successfully implemented the Puzzle Slider game with tiles and frame.
The extra features include Player's name and a clock that starts at the beginning of the game. Also, a high score list – **Hall Of Fame** has been included.
To make the high score list permanent, a text file is running at the back end.

5.1 FUTURE IDEAS

- Implement texture mapping to map images onto the tiles instead of the current Roman Numerals.

- Adding cheat codes like most commercial games, so that the player can enter them and the fully solved puzzle is displayed. This can also be used to view the effects when a player makes it to the **Hall of Fame**.

- Give the option of allowing the user to put any image of his choice onto the tiles. This again involves Texture mapping and image processing.

- Implementing different levels with increasing levels of difficulty.