Chapter 1

INTRODUCTION

1.1 Computer Graphics

To draw a picture say, fish moving inside the water. Suddenly will get an idea to use paint, but the degree of accuracy, quality of image is not satisfied and become very sad. There is no need to worry, for every problem there will be a solution, so this problem of creating fish moving inside the water can be solved using COMPUTER GRAPHICS without any difficulties.

Computer Graphics become a powerful tool for the rapid and economical production of pictures. There is virtually no area in which Graphical displays cannot be used to some advantage so it is not surprising to find the use of CG so widespread.

Although early application in engineering & science had to rely on expensive & cumbersome equipment, advances in computer technology have made interactive computer graphics a practical tool.

Computer Graphics in a diverse area such as science, engineering, medicine, business, industry, government, art, entertainment, education and training.

Now it can be answered about computer graphics as generalized tool for drawing and creating pictures and simulates the real world situations within a small computer window.

1.2 History

William fetter was credited with coning the term Computer Graphics in 1960, to describe his work at Boeng. One of the first displays of computer animation was future world (1976), which included an animation of a human face and hand-produced by Carmull and Fred Parkle at the University of Utah.

There are several international conferences and journals where the most significant results in computer-graphics are published. Among them are the SIGGRAPH and Euro graphics conferences and the association for computing machinery (ACM) transaction on Graphics journals.

1.3 Applications of computer graphics

Nowadays Computer Graphics used in almost all the areas ranges from science, engineering, medicine, business, industry, government, art, entertainment, education and training.

1.3.1 CG in the field of CAD

Computer Aided Design methods are routinely used in the design of buildings, automobiles, aircraft, watercraft, spacecraft computers, textiles and many other applications.

1.3.2 CG in presentation Graphics

Another major application area presentation graphics used to produce illustrations for reports or generate slides. Presentation graphics is commonly used to summarize financial, statistical, mathematical, scientific data for research reports and other types of reports.2D and 3D bar chart to illustrate some mathematical or statistical report.

1.3.3 CG in computer Art

CG methods are widely used in both fine art and commercial art applications. Artists use a variety of computer methods including special purpose hardware, artist’s paintbrush program (lumena), other pain packages, desktop packages, maths packages, animation packages that provide facility for designing object motion. Ex: cartoons design is an example of computer art which uses CG.

1.3.4 Entertainment

Computer graphics methods are now commonly used in making motion pictures, music, videos, games and sounds. Sometimes graphics objects are combined with the actors and live scenes.

1.3.5 Education and Training

Computer generated models of physical financial, economic system is often as education aids. For some training application, special systems are designed. Ex: specialized system is simulator for practice sessions or training of ship captain, aircraft pilots and traffic control.

1.3.6 Image Processing

Although the methods used in CG image processing overlap, the 2 areas are concerned with fundamentally different operations. In CG a computer is used to create picture. Image processing on the other hand applies techniques to modify existing pictures such as photo scans, TV scans.

1.3.7 User Interface

It is common for software packages to provide a graphical interface. A major component of a graphical interface is a window manager that allows a user to display multiple window area. Interface also displays menus, icons for fast selection and processing.

1.4 Statement of the Problem

It is a computer version of the simple board game usually play in daily life. About the dice rolling, coins moving as per the numbers indicated by the dice. Whoever first reaches all his coins on to HOME area is the winner. Here the implementation is done for a version of two players and two coins.

1.5 Objectives

- The interactive demo of Ludo – Board Game.
- Graphical approach towards understanding the Ludo Game design.

1.6 Introduction to OpenGL

Most of the application will be designed to access OpenGL directly through functions in three libraries. Functions in the main GL (or OpenGL in windows) library have names that begin with the letters gl and are stored in a library usually referred to as GL (or OpenGL in windows). The second is the **OpenGL Utility Library** (GLU). This library uses only GL functions but contains code for creating common objects and simplifying viewing. All functions in GLU can be created from the core GL library but application programmers prefer not to write the code repeatedly. The GLU library is available in all OpenGL implementations; functions in the GLU library begin with letters glu.

To interface with the window system and to get input from external devices into the programs, need at least one more system-specific library that provides the “glue” between the window system and OpenGL. For the X window system, this library is functionality that should be expected in any modern windowing system.

Fig 2.1 shows the organization of the libraries for an X Window System environment. For this window system, GLUT will use GLX and the X libraries. The application program, however, can use only GLUT functions and thus can be recompiled with the GLUT library for other window systems.

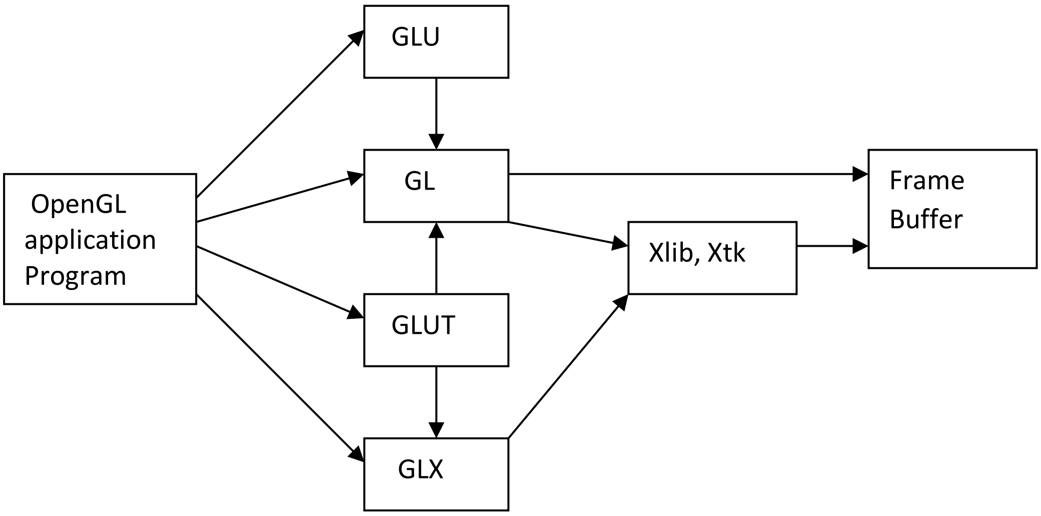


Fig 1.1 Library organization of OpenGL

Chapter 2

SYSTEM ANALYSIS AND PROCEDURE TO PLAY

2.1 System Analysis

- At the start of the game, the player's four pieces are placed in the start area of their colour.
- Players take it in turn to throw a single die. A player must first throw a six to be able to move a piece from the starting area onto the starting square.
- In each subsequent turn the player moves a piece forward 1 to 6 squares as indicated by the die.
- When a player throws a 6 the player may bring a new piece onto the starting square, or may choose to move a piece already in play. Any throw of a six results in another turn.
- If a player cannot make a valid move they must pass the die to the next player.
- If a player's piece lands on a square containing an opponent's piece, the opponent's piece is captured and returns to the starting area.
- A piece may not land on a square that already contains a piece of the same colour (unless playing doubling rules).
- Once a piece has completed a circuit of the board it moves up the home column of its own colour. The player must throw the exact number to advance to the home square.
- The winner is the first player to get all four of their pieces onto the home square.

2.2 Procedure to Play

- A ludo board is normally a square marked with a cross. Each arm of the cross is divided into three columns, with the columns divided into usually six squares.
- The centre of the cross is the finishing square which is often divided into four colored triangles. Each colored triangle is combined with a colored middle column appears as an arrow pointing to the finish.
- The shaft of each arrow is a player's "home column" and is five squares long. To the left of each home column, one square from the edge of the board is a starting square, also colored.
- During game play a piece moves from its starting square, clockwise around the perimeter of the board, and up the player's home column to the finishing square.

- In the space to the left of each arm is a circle or square to hold a player's pieces before they are allowed into play. There are no resting squares, but the colored home column may only be entered by its own player's tokens.
- The special areas on the board are typically brightly colored with yellow, green, red, and blue. Each player uses cardboard or plastic tokens of matching color.

2.3 Hardware and Software requirements:

2.3.1 Hardware Requirements:

- Processor of 2.2G Hz or higher speed
- 20MB Hard Disk Space
- 1GB RAM
- Keyboard
- Mouse

2.3.2 Software Requirement:

- Microsoft Visual Studio 2010
- Windows 8/10 Operating System
- MS-Office
- Graphics package available in Microsoft Visual Studio 2010

Chapter 3

DESIGN AND IMPLEMENTATION

3.1 Graphic functions

3.1.1 void glBegin(glEnum mode);

Initiates a new primitive of type mode and starts the collection of vertices. Values of mode include GL_POINTS, GL_LINES and GL_POLYGON.

3.1.2 void glEnd( );

Terminates a list of vertices.

3.1.3 void glColor3f[i f d] (TYPE r, TYPE g, TYPE b);

Sets the present RGB colors. Valid types are int (i), float (f) and double (d). The maximum and minimum values of the floating-point types are 1.0 and 0.0, respectively.

3.1.4 void glClearColor(GLclampf r,GLclampf g,GLclampf b,GLclampf a);

Sets the present RGBA clear color used when clearing the color buffer. Variables of GLclampf are floating-point numbers between 0.0 and 1.0.

3.1.5 int glutCreateWindow(char *title);

Creates a window on the display. The string title can be used to label the window. The return value provides a reference to the window that can be used where there are multiple windows.

3.1.6 void glutInitWindowSize(int width, int height);

Specifies the initial height and width of the window in pixels.

3.1.7 void glutInitWindowPosition(int x, int y);

Specifies the initial position of the top-left corner of the window in pixels.

3.1.8 void glutInitDisplayMode(unsigned int mode);

Request a display with the properties in mode. The value of mode is determined by the logical OR of operation including the color model (GLUT_RGB, GLUT_INDEX) and buffering (GLUT_SINGLE, GLUT_DOUBLE);

3.1.9 void glFlush( );

Forces any buffered any OpenGL commands to execute.

3.1.10 void glutInit (int argc, char **argv);

Initializes GLUT. The arguments from main are passed in and can be used by the application.

3.1.11 void glutMainLoop( );

Cause the program to enter an event processing loop. It should be the last statement in main.

3.1.12 void glutDisplayFunc(void (*func) (void));

Registers the display function func that is executed when the window needs to be redrawn.

3.1.13 gluOrtho2D(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top);

Defines a two-dimensional viewing rectangle in the plane Z=0;

3.1.14 void glutBitmapCharacter(void *font, int char);

Renders the character with ASCII code char at the current raster position using the raster font given by font. Fonts include GLUT_BITMAP_TIMES_ROMAN_10 and GLUT_BITMAP_TIMES_ROMAN_8_Y_13. The raster position is incremented by the width of the character.

3.1.15 void glClear(GL_COLOR_BUFFER_BIT);

To make the screen solid and white.

3.1.16 void MouseFunc(myMouse);

It is used for the implementation of mouse interface.

Passing the control to

void myMouse(int button,int state,int x,int y);

3.1.17 void KeyboardFunc(key);

It is used for the implementation of keyboard interface.

Passing control to

void key(unsigned char key,int x,int y);

3.1.18 void translate[fd](TYPE x,TYPE y,TYPE z);

Alters the current matrix by displacement of (x,y,z).Type is either GLfloat or GLdouble.

3.1.19 void glPushMatrix(); void glPopMatrix();

Pushes to and pops from the matrix stack corresponding to current matrix mode.

3.1.20 void glLoadMatrix[fd](TYPE *m);

Loads the 16 element array of TYPE GLfloat or GLdouble as a current matrix.

3.2 Game Design

The main functions used in the game design are:

CHECK_COND ();

Checking condition for 2 players and initializing there corresponding position. At the win condition DRAW(); PLY1WIN(); PLY2WIN(); functions are called to end the game and restart.

DICE ();

Rolling of the dice and displaying the dice.

DISPLAY ();

Display's board, coins position and moves the coin of the player as per the condition checked in CHECK_COND(); function, changing the chances from player 1 to player 2 alternatively. Displays the game playing instructions and rules of playing game. Capturing of the one player coin to the other player coin and return the hit coin to the start position. The capturing coin can enter the home column. The player moving all his coins through this home region is considered the winner. If none of the player coins are hit then the game is drawn.

MYMOUSE ();

Select's the players coin, roll the dice for the player, increments the coins position of the selected players coin. Traversing from one screen of detail to other, Restart of the game. The control is passed to the DISPLAY(); function sending the selected coin data which result in the traversing and display of that particular coin.

The game moves are explained in the flow charts. Explaining the step by step execution of main and game play.

3.3 SOURCE CODE

```
VOID DISPLAY_ABOUT(VOID)
{
  CHAR *ABOUT="THIS MINI-PROJECT IS BASED ON A VERY ANCIENT GAME LUDO,USING OPENGL.\N\NINSTRUCTIONS :\NINPUT CAN BE PROVIDED EITHER FROM MOUSE OR FROM KEYBOARD.\NFOR MOUSE INTERACTION RIGHT-CLICK ON THE SCREEN AND SELECT THE REQUIRED OPTION.\NKEYBOARD AND MOUSE INTERFACES ARE EXPLAINED IN INSTRUCTIONS SECTION.\N\N\N\N\N\N";
ENTER=0;
  INT I;
  FLOAT X=-1.8,Y=0.9,Z=0;
  IF(IN_ABOUT!=0)
  {
  IN_ABOUT=1;
  GLCOLOR3F(1,0,0);
  GLRASTERPOS3F(X,Y,Z);
  FOR(I=0;ABOUT[I]!='\0';I++)
  {
              IF(ABOUT[I]=='\N')
              {   Y-=0.08;
                          GLRASTERPOS3F(X,Y,Z);
              }
              ELSE
                          GLUTBITMAPCHARACTER(FONT,ABOUT[I]);
            }
 GLPOPMATRIX();
GLFLUSH();
  }
  }
VOID DISPLAY(VOID)
{
  GLCLEAR(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
  DISPLAY_ABOUT();
}
VOID DISPLAY_ABOUT_GAME(VOID)
{
  CHAR *ABOUT_GAME="RULES:\N\N\NAT THE START OF THE GAME, THE PLAYER'S FOUR PIECES ARE PLACED IN THE START AREA OF THEIR COLOUR.\N\NPLAYERS TAKE IT IN TURN TO THROW A SINGLE DIE. A PLAYER MUST FIRST THROW A SIX TO BE ABLE TO MOVE A PIECE FROM THE STARTING AREA\NONTO THE STARTING SQUARE.IN EACH SUBSEQUENT TURN THE
```

```
PLAYER MOVES A PIECE FORWARD 1 TO 6 SQUARES AS INDICATED BY THE DIE.WHEN
A\NPLAYER THROWS A 6 THE PLAYER MAY BRING A NEW PIECE ONTO THE STARTING SQUARE,
OR MAY CHOOSE TO MOVE A PIECE ALREADY IN PLAY.ANY\NTHROW OF A SIX RESULTS IN
ANOTHER TURN.\N\NIF A PLAYER CANNOT MAKE A VALID MOVE THEY MUST PASS THE DIE TO
THE NEXT PLAYER.\N\NIF A PLAYER'S PIECE LANDS ON A SQUARE CONTAINING AN
OPPONENT'S PIECE, THE OPPONENT'S PIECE IS CAPTURED AND RETURNS TO THE
STARTING\NAREA.A PIECE MAY NOT LAND ON A SQUARE THAT ALREADY CONTAINS A PIECE
OF THE SAME COLOUR(UNLESS PLAYING DOUBLING RULES).\N\NONCE A PIECE HAS
COMPLETED A CIRCUIT OF THE BOARD IT MOVES UP THE HOME COLUMN OF ITS OWN COLOUR.
THE PLAYER MUST THROW THE EXACT\NNUMBER TO ADVANCE TO THE HOME SQUARE.THE
WINNER IS THE FIRST PLAYER TO GET ALL FOUR OF THEIR PIECES ONTO THE HOME SQUARE.";
  INT I;
  FLOAT X=-1.8,Y=0.9,Z=0;
  IN_ABOUT=1;
  GLPUSHMATRIX();
  GLLOADIDENTITY();
  GLCOLOR3F(1,0,0);
  GLRASTERPOS3F(X,Y,Z);
  FOR(I=0;ABOUT_GAME[I]!='\0';I++)
  {
              IF(ABOUT_GAME[I]=='\N')
              {   Y-=0.08;
                          GLRASTERPOS3F(X,Y,Z);
              }
              ELSE
                GLUTBITMAPCHARACTER(FONT,ABOUT_GAME[I]);
  }
 GLPOPMATRIX();
GLFLUSH();
}
VOID DISPLAY_INTER_GUIDE(VOID)
{
  CHAR *GUI="KEYBOARD OPTIONS:\NQ-TO QUIT.\N\NMOUSE OPTIONS:\NLEFT BUTTON- TO
SELECT THE COIN,\NCLICK ON THE COIN OF PLAYER TO\NROLL DICE & TO MOVE.\NRIGHT
BUTTON- TO RESTART.";
  FLOAT X=1.1,Y=0.4,Z=0;
  INT I;
  GLCOLOR3F(0.4,0.5,0.3);
  GLBEGIN(GL_LINE_LOOP);
    GLVERTEX2F(1.08,0.5);
```

```
              GLVERTEX2F(1.92,0.5);
              GLVERTEX2F(1.92,-0.2);
              GLVERTEX2F(1.08,-0.2);
  GLEND();
  IN_GUIDE=1;
  GLCOLOR3F(1,0,0);
  GLRASTERPOS3F(X,Y,Z);
  FOR(I=0;GUI[I]!='\0';I++)
  {
                IF(GUI[I]=='\N')
                {             Y-=0.08;
                GLRASTERPOS3F(X,Y,Z);
  }
                ELSE
    GLUTBITMAPCHARACTER(FONT,GUI[I]);

  GLFLUSH();
  }
VOID ENTER_DISPLAY2()
{

              CHAR *ABOUT="PRESS  ENTER\N";
  VOID *EN=GLUT_BITMAP_TIMES_ROMAN_24;
  FLOAT X=0.8,Y=-0.8,Z=0;
  INT I;
  ENTER=1;
              GLCLEAR(GL_COLOR_BUFFER_BIT);
              GLLINEWIDTH(9);
              GLCOLOR3F(0,0,0);
//            GLCOLOR3F(0.545,0.6434,0.743);
              GLBEGIN(GL_LINES);
                GLVERTEX2F(-1.8,0.8);
                GLVERTEX2F(-1.8,0.4);
                GLVERTEX2F(-1.8,0.4); //L
                GLVERTEX2F(-1.6,0.4);
                GLEND();
      GLFLUSH();
  FOR(I=0;ABOUT[I]!='\0';I++)
  {
                IF(ABOUT[I]=='\N')
```

```
            {   WAIT();
                    Y-=0.08;
                    GLRASTERPOS3F(X,Y,Z);
            }
            ELSE IF(ABOUT[I]=='\T')
            {WAIT();
                    X-=0.2;
                    GLRASTERPOS3F(X,Y,Z);
            }
            ELSE
            {WAIT();
              GLUTBITMAPCHARACTER(EN,ABOUT[I]);
            }
  }
SQUARE();
  }
VOID OPTIONS(INT ID)
{
 SWITCH(ID)
 {
  CASE 1:
            GLCLEAR(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
            IN_ABOUT=1;
            DISPLAY_ABOUT();
            BREAK;
  CASE 2:
            GLCLEAR(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
            DISPLAY_ABOUT_GAME();
     BREAK;
  CASE 4:
            GLCLEAR(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
                  GLCLEARCOLOR(0.6,0.7,0,0);
            DISPLAY_ENTER1();
            BREAK;
  CASE 5:
            EXIT(0);
 }
 }

INT MAIN(INT ARGC, CHAR **ARGV)
```

```
{
  GLUTINIT(&ARGC, ARGV);
           ENTER=1;
  GLUTINITDISPLAYMODE(GLUT_SINGLE|GLUT_RGB|GLUT_DEPTH);
  INIT();
  GLUTCREATEMENU(OPTIONS);
  GLUTADDMENUENTRY("ABOUT THE PROJECT",1);
  GLUTADDMENUENTRY("RULES OF THE GAME",2);
  GLUTADDMENUENTRY("GAME MODE OR RESTART",4);
  GLUTADDMENUENTRY("QUIT",5);
  GLUTMOUSEFUNC(MYMOUSE);
  GLUTKEYBOARDFUNC(KEY);
  GLUTATTACHMENU(GLUT_RIGHT_BUTTON);
 GLUTRESHAPEFUNC(MYRESHAPE);
  GLUTDISPLAYFUNC(DISPLAY);
  FONT=GLUT_BITMAP_9_BY_15;//GLUT_BITMAP_HELVETICA_12;
  GLUTMAINLOOP();
RETURN 0;
}
```

3.4 FLOW CHART

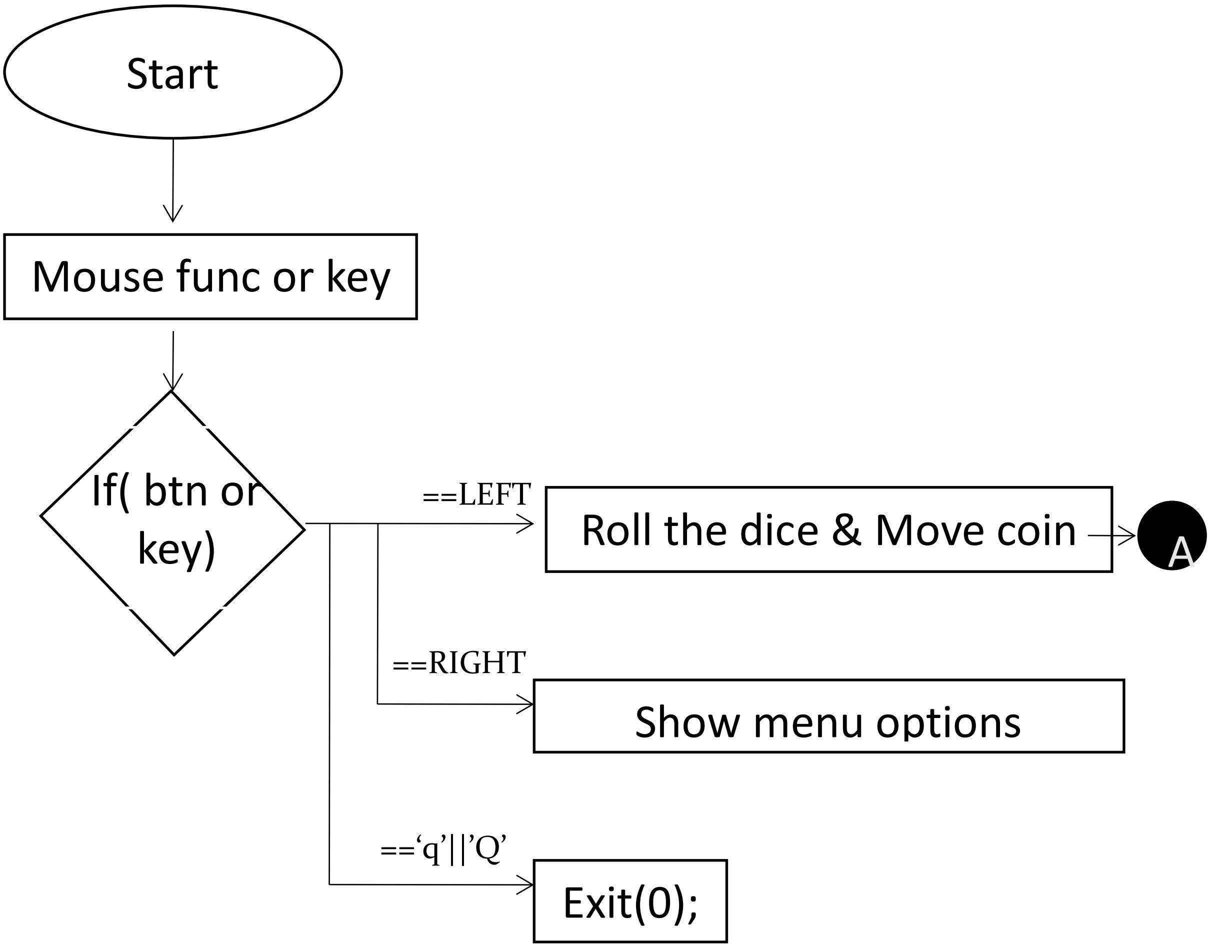
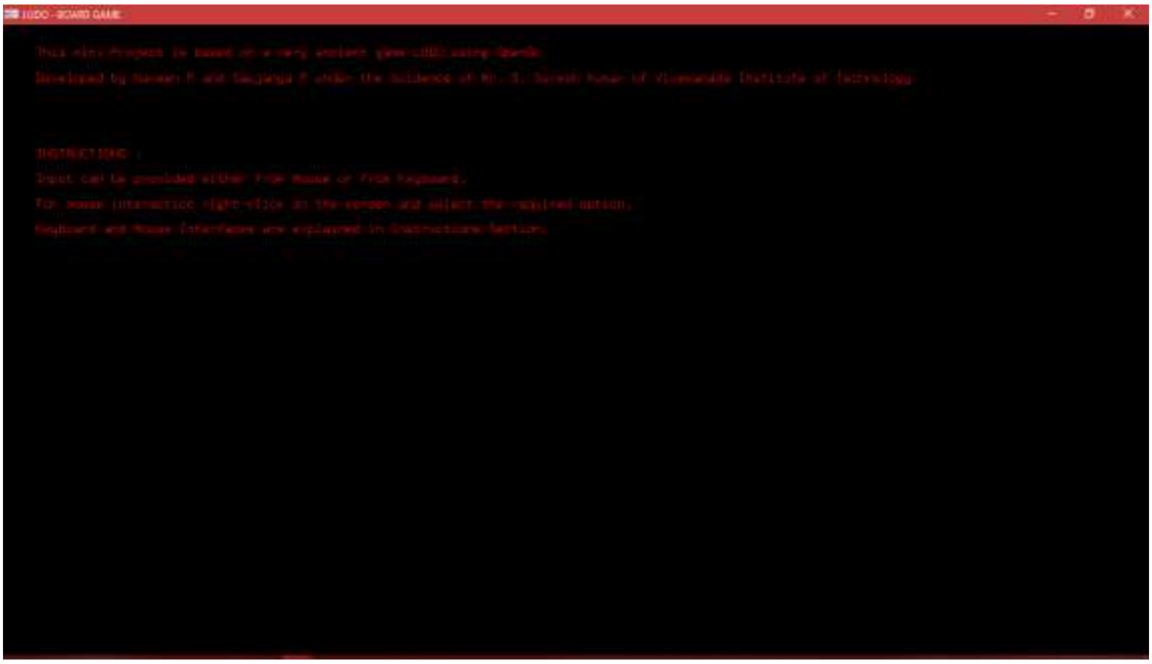


Fig 3.1: Flow chart for main()

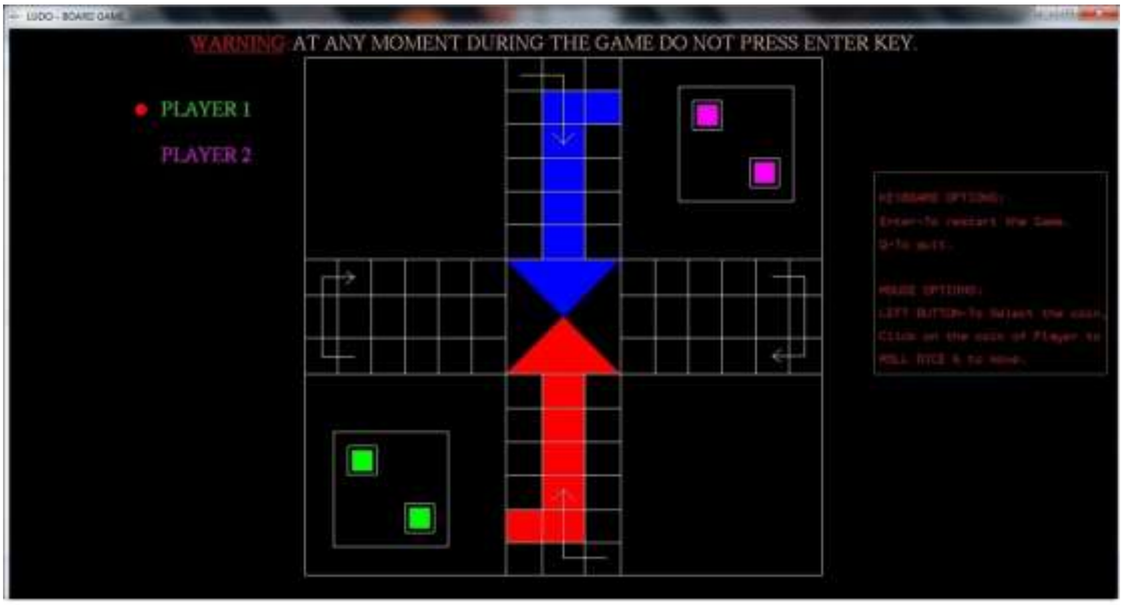
3.5 RESULTS



```
This mini Project is based on a very ancient game LUDO using OpenGL
Developed by Naveen P and Saujanya P under the Guidance of Mr. S. Suresh Kumar of Vivekananda Institute of Technology


INSTRUCTIONS :
Input can be provided either from mouse or from keyboard.
For mouse interaction right-click on the screen and select the required option.
Keyboard and Mouse Interfaces are explained in Instructions Section.
```

Fig 3.2: Initial window of Game



WARNING AT ANY MOMENT DURING THE GAME DO NOT PRESS ENTER KEY.

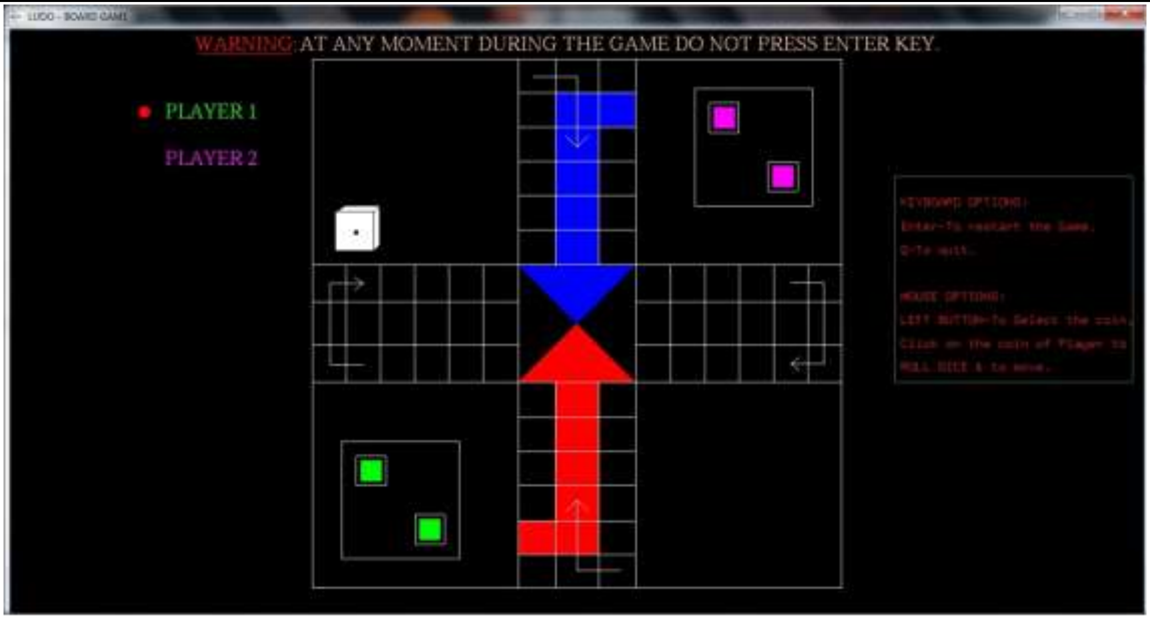
● PLAYER 1

PLAYER 2

```
KEYBOARD OPTIONS:
Enter-To restart the Game.
Q-To quit.

MOUSE OPTIONS:
LEFT BUTTON-To Select the coin.
Click on the coin of Player to
ROLL DICE & to move.
```

Fig 3.3: Loaded Game



LUDO - BOARD GAME

WARNING: AT ANY MOMENT DURING THE GAME DO NOT PRESS ENTER KEY.

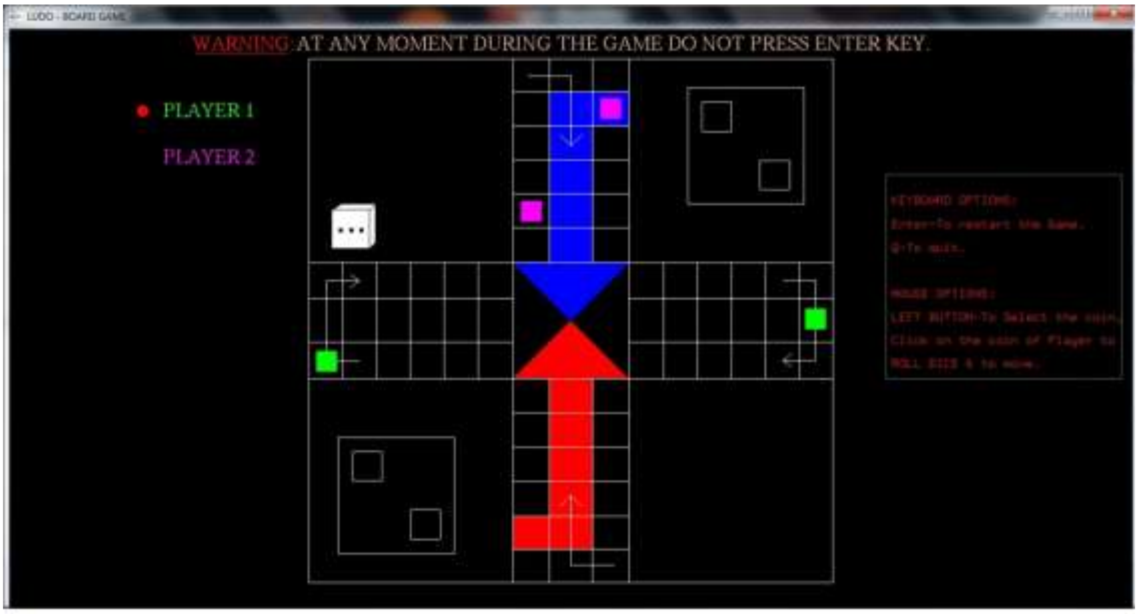
● PLAYER 1

PLAYER 2

KEYBOARD OPTIONS:
Enter-To restart the Game.
Q-To quit.

MOUSE OPTIONS:
LEFT BUTTON-To Select the coin,
Click on the coin of Player to
ROLL DICE & to move.

Fig 3.4: Player1's Coin move out



LUDO - BOARD GAME

WARNING: AT ANY MOMENT DURING THE GAME DO NOT PRESS ENTER KEY.

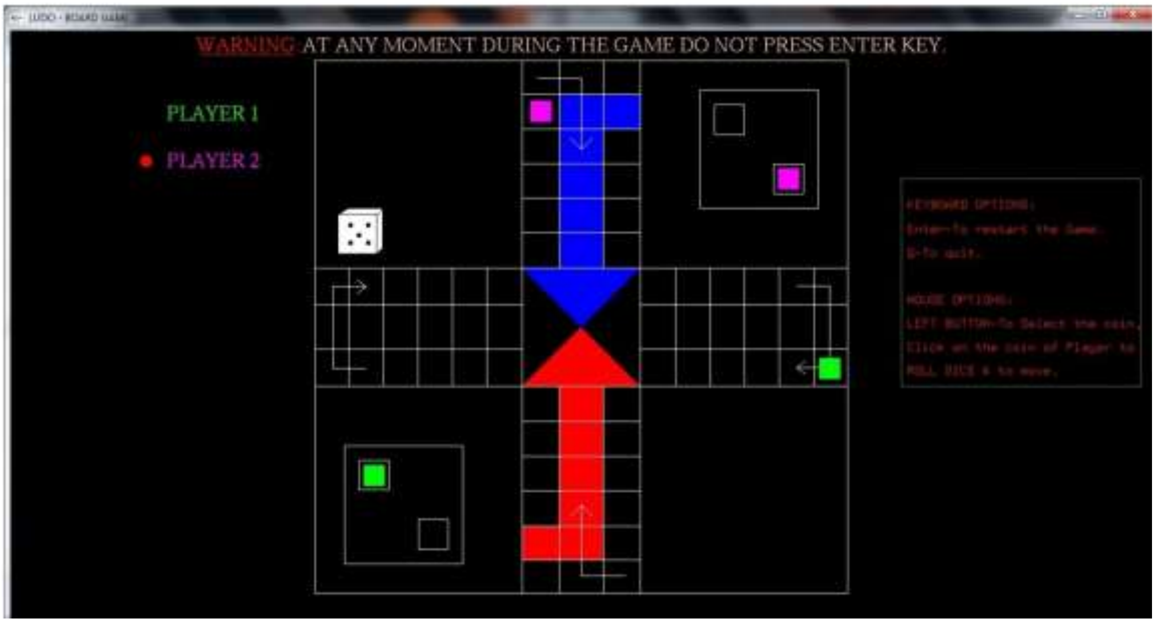
● PLAYER 1

PLAYER 2

KEYBOARD OPTIONS:
Enter-To restart the Game.
Q-To quit.

MOUSE OPTIONS:
LEFT BUTTON-To Select the coin,
Click on the coin of Player to
ROLL DICE & to move.

Fig 3.5: Both players coin in the game

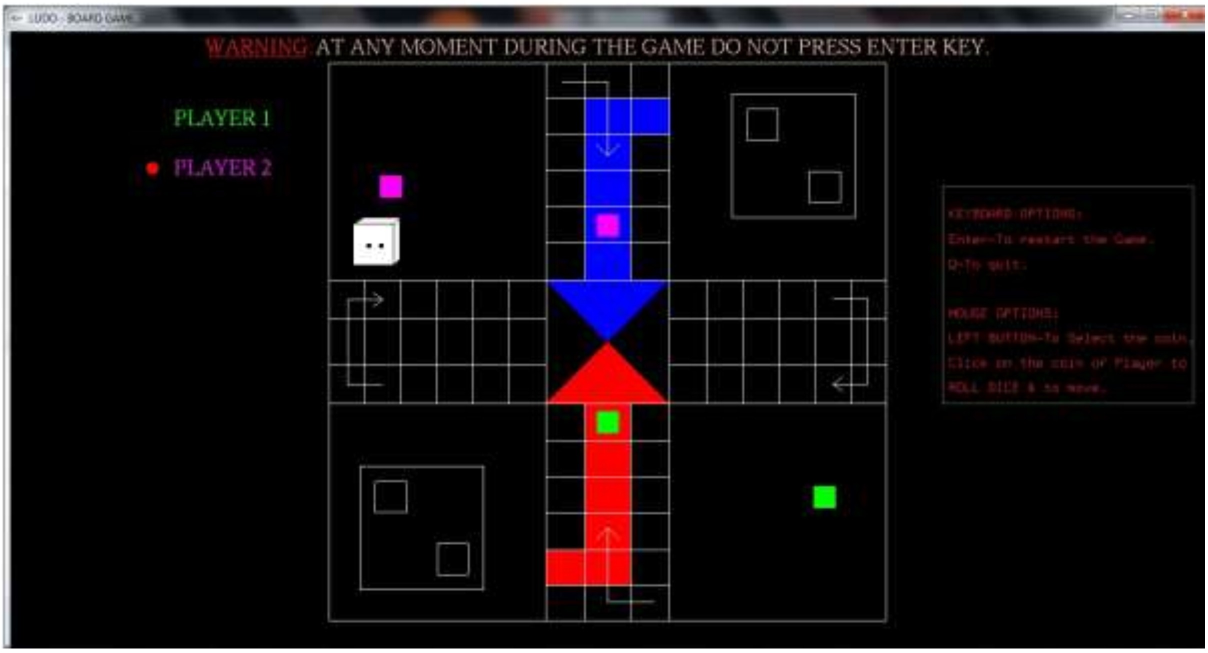


WARNING AT ANY MOMENT DURING THE GAME DO NOT PRESS ENTER KEY.

KEYBOARD OPTIONS:
Enter-To restart the Game.
Q-To quit.

MOUSE OPTIONS:
LEFT BUTTON-To Select the coin.
Click on the coin of Player to
ROLL DICE & to move.

Fig 3.6: Both players coin after capturing returned back

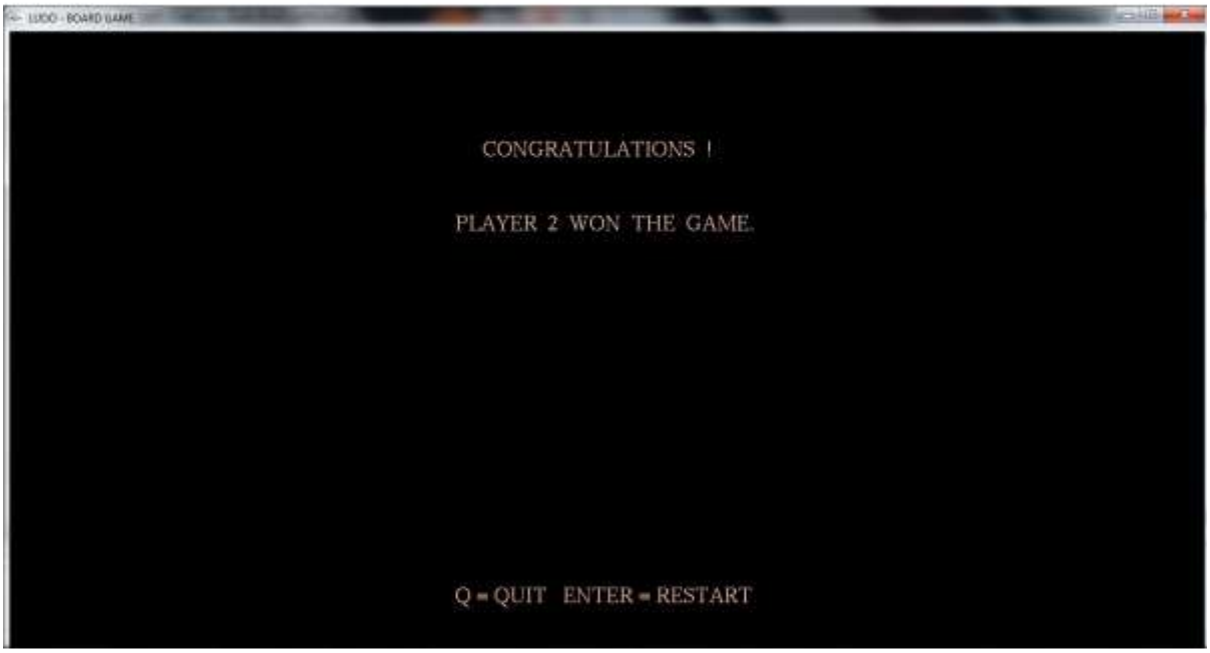


WARNING AT ANY MOMENT DURING THE GAME DO NOT PRESS ENTER KEY.

KEYBOARD OPTIONS:
Enter-To restart the Game.
Q-To quit.

MOUSE OPTIONS:
LEFT BUTTON-To Select the coin.
Click on the coin of Player to
ROLL DICE & to move.

Fig 3.7: Player2 about to win on next move



CONGRATULATIONS !

PLAYER 2 WON THE GAME.

Q = QUIT ENTER = RESTART

Fig 3.8: Player2 won the game

CHAPTER 4

CONCLUSION

It is a computer version of the simple board game usually play in daily life. About the dice rolling, coins moving as per the numbers indicated by the dice. Whoever first reaches all his coins on to HOME area is the winner. Here the implementation is done for a version of two players and two-coinsThe new interactive way of playing this game on the computer is displayed.The demo is made more interactive with a keyboard and mouse interaction module in the program.

Future Enhancement:

Even though demo designed is enriched with many options, it is a two dimensional demo, in future it can be re designed with 3D animation and sound effects. By using this demo further for four coins and four players can be implemented.

BIBLIOGRAPHY

[1] Edward Angel-**Interactive Computer Graphics: A Top-Down Approach using OpenGL** Fifth Edition, Published by Pearson Education, 2009

[2] The OpenGL Programming Guide, 5th Edition. The Official guide to learning OpenGL Version 2.1 by OpenGL Architecture Review Board.

[3] en.wikipedia.org/ludo_board_game.html