I. ABSTRACT

This project is related to Computer Graphics by using OpenGL. This is an interactive mini-project and a 2D animation which depicts the story '**THE FOX AND THE CROW'.** The idea of this project is to illustrate the greatness of culture and moral values, it is totally a story based theme on one of the most popular Moral bedtime stories for Kids. The user will be provided with keyboard/mouse interaction where on each selection of numbers/keys different action takes place. It involves transformation techniques and shows the appearance of nature, movement of fox, flying and singing of crow. For conversation rectangular boxes or elliptical shapes will be used. This project will be done in C/C++ Language using OpenGL. Various commands and built in functions from the standard graphics package provided in OpenGL will be used to draw figures, objects and to colour them.

II. INTRODUCTION

2.1 About Computer Graphics

The Computer Graphics is one of the most effective and commonly used methods to communicate the processed information to the user. It displays the information in the form of graphics objects such as pictures, charts, graphs and diagram instead of simple text.

In computer graphics, pictures or graphics objects are presented as a collection of discrete picture elements called pixels. The pixel is the smallest addressable screen element

Computer graphics today is largely interactive: The user controls the contents structure, and appearance of objects and their displayed images by using input devices, such as a keyboard, mouse, or touch-sensitive panel on the screen.

Computer Graphics relies on an internal model of the scene, that is, mathematical representation suitable for graphical computations. The model describes the 3D shapes, layout and materials of the scene. This 3D representation then has to be projected to compute a 2D image from a given viewpoint, this is rendering step. Rendering involves projecting the objects, handling visibility (which parts of objects are hidden) and computing their appearance and lighting interactions. Finally, for animated sequence, the motion of objects has to be specified.

2.2 About OpenGL

OpenGL (Open Graphics Library) is a standard specification defining a cross-language, cross-platform API for writing applications that produce 2D and 3D computer graphics. The interface consists of over 250 different function calls which can be used to draw complex three-dimensional scenes from simple primitives. OpenGL was developed by Silicon Graphics Inc.

OpenGL's basic operation is to accept primitives such as points, lines and polygons, and convert them into pixels. This is done by a graphics pipeline known as the OpenGL state machine.

Features of OpenGL:

Geometric Primitives allow you to construct mathematical descriptions of objects. Viewing and Modelling permits arranging objects in a 3-dimensional scene move our camera around space and select the desired vantage point for viewing the scene to be rendered.

Materials lighting OpenGL provides commands to compute the colour of any point given the properties of the material and the sources of light in the room.

Transformations: rotation, scaling, translations, perspectives in 3D, etc.

2.3 About Project

One day, a crow found a piece of cheese and he take in his beak and sit on the tree branch. A fox was walking through the forest when he saw a crow sitting on a tree with a Piece of cheese. Fox thought for a while. Soon, an idea struck him. He decided to flatter the crow and thus began to praising him. Fox said to crow, "you are a very good singer, sing a song for me!" Now, the crow was very proud and he wanted to show the fox that he could sing very Well so he opened his mouth to sing, the piece of cheese he was eating fell to the ground. The Fox laughed at the crow and picked up the cheese and went from there.

The project is implemented about the story. It is coded in c/**c++** language Using Open GL libraries and functions. The moving of fox and crow and also a cheese are based on the different keys like 1, 2, 3, 4 etc. They include additional operations like translations and Primitives. Primitives may have many other options like dots, lines and triangles. We will also

Introduce the user interactivity by providing various keyboard functions and provide the output in form of various snapshots.

III IMPLEMENTATION

Implementation is the stage where all planned activities are put into action. Before the implementation of a project, the implementers (spearheaded by the project committee or executive) should identify their strength and weaknesses (internal forces), opportunities and threats (external forces).

Implementation also includes a pseudo code.

3.1 Pseudo code

The execution of the program starts from the main function. It calls various inbuilt and user defined functions, the pseudo code is as follows:

```
Void init ()
{
    This function is used to initialize the viewing of the output.
}
Void sky ()
{
    Drawn the abstract implementation of the background.
}
Void grass ()
{
    This function is used to draw the grass.
}
Void mountain ()
{
    This function is used to draw the mountains.
 }
Void gate ()
{
    This function is used to draw the gate.
}
```

Void tree ()

{

This function is used to draw the tree.

}

Void cheese ()

{

This function is used to draw cheese.

}

Void crow ()

{

This function is used to draw crow.

}

Void fox ()

{

This function is used to draw fox.

}

Void mykeys ()

{

This function is used to provide the user interaction.

}

Void main ()

{

This function is used to call the main function.

}

3.2 Functions

The functions that are used in the program are discussed below. This section contains brief description of all headers and functions. These functions are as follows:

- **#include<GL/glut.h>**

  Use to include glut library files.

3.2.1 User defined functions used in the project

- **Display ()**

  This function is used to calls the other functions and displays the objects on the screen.

- **Crow ()**

  This function is used to draw the crow.

- **Fox ()**

  This function is used to draw the fox.

- **Cheese ()**

  This function is used to draw the cheese.

- **Crowfly ()**

  This function is used to scale the motion of crow.

- **Foxmove ()**

  This function is used to scale the motion of fox.

- **Crowsings ()**

  This function is used to build the singing objects.

- **Tree ()**

  This function is used to builds the tree objects.

- **Gate ()**

  This function is used to build the gates.

- **Mountain ()**

  This function is used to draw the mountains.

3.2.2 Standard library functions used in the project:

- **glutInit ()**

  glutInit is used to initialize the GLUT library.

- **glutInitDisplayMode()**

  glutInitDisplayMode sets the initial display mode.

- **glutInitWindowPosition, glutInitWindowSize ()**

  These functions are used to set the initial window position and size respectively.

- **glBegin ()**

  Specifies the primitive or primitives that will be created from vertices presented between glBegin () and glEnd ().

- **gllutDisplay ()**

  It is the function that displays the primitives onto the screen by calling the user defined functions.

- **glutKeyboardFunc ()**

  KeyboardFunc sets the keyboard call back for the current window.

- **glClear()**

  The clear function clears buffers to pre-set values.

- **glClearColor()**

  The glClearColor function specifies clear values for the colour buffers.

- **glMatrixMode()**

  This function specifies which matrix is the current matrix.

- **glTranslatef()**

This function multiplies the current matrix by a translation matrix.

3.2.2 Code of the project:

```
void print_text(char *str, GLint x, GLint y)
{
	GLint i;
	glRasterPos2d (x,y) ;
	for (i=0;str[i]!='\0';i++)
		glutBitmapCharacter (GLUT_BITMAP_TIMES_ROMAN_24, str[i]) ;
}
void nature(void)                                          //key2
{
	glClear(GL_COLOR_BUFFER_BIT);
	sky();
	water();
	grass();
	grass1();
	mountain();
	gate();
	gate1();
	gate2();
	tree();
	tree1();
	tree2();
	cheese1();
	glFlush();
key=3;
}
void crowfly()//for key3                                   //key3
{
	float x=0,y=0;
	for(int i=0;i<200;i++)
	{
		glClear(GL_COLOR_BUFFER_BIT);
	sky();
	grass();
	mountain();
	water();
	gate();
	gate1();
	gate2();
	tree();
	grass1();
	tree1();
	tree2();
	cheese1();
			glPushMatrix();
			glTranslatef(x,y,0.0);
			crow();//for key3
			x-=.500;
			y-=.30;
			glPopMatrix();
			glFlush();
```

```
        }
key=4;
}
void crow1()//cheese in mouth moving towards the branch
{
        glBegin(GL_TRIANGLES);//cheese
        glColor3f(1.0,0.9,0.0);
        glVertex2i(157,30);
        glVertex2i(153,33);
        glVertex2i(157,35);
        glEnd();
        glBegin(GL_LINES);// cheese outline
        glColor3f(0.1,0.0,0.0);
        glVertex2i(153,33);
        glVertex2i(157,35);
        glEnd();
        glBegin(GL_LINES);
        glColor3f(0.1,0.0,0.0);
        glVertex2i(153,33);
        glVertex2i(157,30);
        glEnd();
        glBegin(GL_LINES);
        glColor3f(0.1,0.0,0.0);
        glVertex2i(157,30);
        glVertex2i(157,35);
        glEnd();
        glBegin(GL_TRIANGLES); //mouth
        glColor3f(0.1,0.0,0.0);
        glVertex2f(157,32.5);
        glVertex2i(160,34);
        glVertex2i(160,31);
        glEnd();
        glBegin(GL_POLYGON); //head
        glColor3f(0.7,0.8,1.0);
        glVertex2i(160,34);
        glVertex2i(162,35);
        glVertex2i(164,35);
        glVertex2i(166,34);
        glVertex2f(166,32.5);
        glVertex2i(166,31);
        glVertex2i(164,30);
        glVertex2i(164,30);
        glVertex2i(160,31);
        glEnd();
        glBegin(GL_POLYGON); //eye
        glColor3f(0.1,0.0,0.0);
        glVertex2i(162,32.5);
        glVertex2i(164,32.5);
        glVertex2i(164,33);
        glVertex2i(162,33);
        glEnd();
```

```
	glBegin(GL_POLYGON); //body
	glColor3f(0.1,0.0,0.0);
	glVertex2i(166,34);
	glVertex2i(176,38);
	glVertex2i(174,34);
	glVertex2i(186,38);
	glVertex2i(184,34);
	glVertex2i(192,34);
	glVertex2i(188,32);
	glVertex2i(200,27);
	glVertex2i(188,29);
	glVertex2i(192,27);
	glVertex2i(184,31);
	glVertex2i(186,27);
	glVertex2i(174,31);
	glVertex2i(176,27);
	glVertex2i(166,31);
	glEnd();
	glBegin(GL_POLYGON); //leg1
	glColor3f(0.1,0.0,0.0);
	glVertex2i(169,30);
	glVertex2i(169,27);
	glVertex2i(168,25);
	glVertex2i(171,25);
	glVertex2i(170,27);
	glVertex2i(170,30);
	glEnd();
	glBegin(GL_POLYGON); //leg2
	glColor3f(0.1,0.0,0.0);
	glVertex2f(170.5,26);
	glVertex2f(172.5,26);
	glVertex2i(172,29);
	glVertex2i(171,29);
	glVertex2i(171,29);
	glEnd();
	glFlush();
}
void crowfly1()                                          //key4
{
	float x=0,y=0;
	for(int i=0;i<300;i++)
	{
		glClear(GL_COLOR_BUFFER_BIT);
	sky();
	grass();
	mountain();
	water();
	gate();
	gate1();
	gate2();
	tree();
```

```
        grass1();
        tree1();
        tree2();
                        glPushMatrix();
                        glTranslatef(x,y,0.0);
                        crow1();
                        x-=0.45;
                        y+=0.15;
                        glPopMatrix();

                glFlush();
        }
        crow2();
key=5;
}
//--------------------key5------------------
void text()//fox flattering
{
        glBegin(GL_POLYGON);
        glColor3f(1.0,0.9,0.9);
        glVertex2i(80,50);
        glVertex2i(130,50);
        glVertex2i(130,62);
        glVertex2i(80,62);
        glEnd();
        glBegin(GL_TRIANGLES);
        glVertex2i(84,50);
        glVertex2i(84,47);
        glVertex2i(87,50);
        glEnd();
        glColor3f(1.0,0.0,0.0);
        print_text1("You are a very good singer.",81,58);
        print_text1("Sing a song for me!",81,54);
        glFlush();
}
void flower()
{
   glLineWidth(1.0);
        glBegin(GL_POLYGON);//5 with color
        glVertex2f(450.0/2,100.0/2);
        glVertex2f(450.0/2,70.0/2);
        glVertex2f(430.0/2,50.0/2);
        glVertex2f(425.0/2,75.0/2);
        glEnd();

        glBegin(GL_POLYGON);//6 with color
        glVertex2f(450.0/2,100.0/2);
        glVertex2f(425.0/2,75.0/2);
        glVertex2f(400.0/2,80.0/2);
   glVertex2f(425.0/2,100.0/2);
        glEnd();
```

```
        glBegin(GL_POLYGON);//7
        glVertex2f(450.0/2,100.0/2);
        glVertex2f(425.0/2,100.0/2);
        glVertex2f(400.0/2,120.0/2);
glVertex2f(425.0/2,125.0/2);
        glEnd();

        glBegin(GL_POLYGON);//8
        glVertex2f(450.0/2,100.0/2);
        glVertex2f(425.0/2,125.0/2);
        glVertex2f(435.0/2,150.0/2);
glVertex2f(450.0/2,125.0/2);
        glEnd();

        glBegin(GL_POLYGON);//1
        glVertex2f(450.0/2,100.0/2);
        glVertex2f(450.0/2,125.0/2);
        glVertex2f(470.0/2,150.0/2);
glVertex2f(475.0/2,125.0/2);
        glEnd();

        glBegin(GL_POLYGON);//2
        glVertex2f(450.0/2,100.0/2);
        glVertex2f(470.0/2,125.0/2);
        glVertex2f(500.0/2,120.0/2);
glVertex2f(480.0/2,100.0/2);
        glEnd();

        glBegin(GL_POLYGON);//3
        glVertex2f(450.0/2,100.0/2);
        glVertex2f(480.0/2,105.0/2);
        glVertex2f(500.0/2,75.0/2);
glVertex2f(470.0/2,78.0/2);
        glEnd();

        glBegin(GL_POLYGON);//4
        glVertex2f(450.0/2,100.0/2);
        glVertex2f(475.0/2,75.0/2);
        glVertex2f(480.0/2,50.0/2);
glVertex2f(450.0/2,70.0/2);
        glEnd();

        //border
        glColor3f(0.0,0.0,0.0);
        glLineWidth(2.0);
        glBegin(GL_LINE_LOOP);//8
        glVertex2f(450.0/2,100.0/2);
        glVertex2f(450.0/2,70.0/2);
        glVertex2f(430.0/2,50.0/2);
        glVertex2f(425.0/2,75.0/2);
```

```
        glEnd();

        glBegin(GL_LINE_LOOP);//7
        glVertex2f(450.0/2,100.0/2);
        glVertex2f(425.0/2,75.0/2);
        glVertex2f(400.0/2,80.0/2);
    glVertex2f(425.0/2,100.0/2);
        glEnd();

        glBegin(GL_LINE_LOOP);//6
        glVertex2f(450.0/2,100.0/2);
        glVertex2f(425.0/2,100.0/2);
        glVertex2f(400.0/2,120.0/2);
    glVertex2f(425.0/2,125.0/2);
        glEnd();

        glBegin(GL_LINE_LOOP);//5
        glVertex2f(450.0/2,100.0/2);
        glVertex2f(425.0/2,125.0/2);
        glVertex2f(435.0/2,150.0/2);
    glVertex2f(450.0/2,125.0/2);
        glEnd();

        glBegin(GL_LINE_LOOP);//1
        glVertex2f(450.0/2,100.0/2);
        glVertex2f(450.0/2,125.0/2);
        glVertex2f(470.0/2,150.0/2);
    glVertex2f(475.0/2,125.0/2);
        glEnd();

        glBegin(GL_LINE_LOOP);//2
        glVertex2f(450.0/2,100.0/2);
        glVertex2f(475.0/2,125.0/2);
        glVertex2f(500.0/2,120.0/2);
    glVertex2f(480.0/2,100.0/2);
        glEnd();

        glBegin(GL_LINE_LOOP);//3
        glVertex2f(450.0/2,100.0/2);
        glVertex2f(483.0/2,100.0/2);
        glVertex2f(500.0/2,75.0/2);
    glVertex2f(470.0/2,79.0/2);
        glVertex2f(450.0/2,100.0/2);
        glEnd();

    glBegin(GL_LINE_LOOP);//4
        glVertex2f(450.0/2,100.0/2);
        glVertex2f(475.0/2,75.0/2);
        glVertex2f(480.0/2,50.0/2);
    glVertex2f(450.0/2,70.0/2);
        glVertex2f(450.0/2,100.0/2);
```

```
        glEnd();


glFlush();
        }

//to draw flowers of vase
void bouquet()
{
glPushMatrix();//5
        glTranslatef(-190.0/2,120.0/2,0.0);
        glColor3f(1.0,0.0,0.0);
        flower();
  glPopMatrix();
        glPushMatrix();//2
        glTranslatef(-190.0/2,170.0/2,0.0);
        glColor3f(0.0,1.0,0.0);
        flower();
  glPopMatrix();
        glColor3f(0.0,0.0,1.0);//6
  glPushMatrix();
  glColor3f(1.0,0.0,1.0);
        glTranslatef(-250.0/2,130.0/2,0.0);
  flower();
  glPopMatrix();

        glPushMatrix();//1
  glColor3f(1.0,1.0,0.0);
        glTranslatef(-250.0/2,180.0/2,0.0);
  flower();
  glPopMatrix();

        glColor3f(1.0,0.5,0.0);//4
        glPushMatrix();
  glTranslatef(-135.0/2,138.0/2,0.0);
  flower();
  glPopMatrix();

        glColor3f(1.0,0.0,0.5);//3
        glPushMatrix();
        glTranslatef(-130.0/2,180.0/2,0.0);
  flower();
  glPopMatrix();

        glFlush();
}
void vase()//flickering
{
        bouquet();
        int x=15.0;
        for(int i=0;i<100;i++)//80 to 100
```

```
	{
		glClear(GL_COLOR_BUFFER_BIT);
		bouquet();

		glFlush();
		glutSwapBuffers();
	}
	glFlush();
}

void moral()                                          //default
{
	glClear(GL_COLOR_BUFFER_BIT);
	glColor3f(1.0,1.0,0.6);
	glBegin(GL_POLYGON);
	glVertex2i(0,0);
	glVertex2i(300,0);
	glVertex2i(300,200);
	glVertex2i(0,200);
	glEnd();
	glColor3f(1.0,0.0,0.0);
     vase();
    glColor3f(1.0,0.39,0.23);
	print_text("**\"MORAL\"**",115,70);
	print_text("\"ONE SHOULDN'T BELIEVE IN FLATTERERS\"",80,60);

	glColor3f(0.6,0.0,0.0);
	glBegin(GL_POLYGON);
		glVertex2f(208,60);//700/11
		glVertex2f(214,60);//right
		glVertex2f(216,63);
		glVertex2f(206,63);
	glEnd();
	glColor3f(1.0,0.5,0.0);
	glBegin(GL_POLYGON);
		glVertex2f(211,63);
		glVertex2f(213,64);
		glVertex2f(211,65);
		glVertex2f(209,64);
	glEnd();
glColor3f(0.6,0.0,0.0);
	glBegin(GL_POLYGON);//left
		glVertex2f(68,60);
		glVertex2f(74,60);
		glVertex2f(76,63);
		glVertex2f(66,63);
	glEnd();
	glColor3f(1.0,0.5,0.0);
	glBegin(GL_POLYGON);
		glVertex2f(71,63);
		glVertex2f(73,64);
```

```
                glVertex2f(71,65);
                glVertex2f(69,64);
        glEnd();
        glFlush();
}

void mykeys(unsigned char ke,int x,int y)
{
        //while(1)
//{
    if(ke=='1'||key==1)
    {
         topic();
         ke=0;
        //else
     for(int w=0;w<2000000000;w++);
     }
                if(ke=='2'||key==2)
        {               nature();
        //else
                  ke=0;
              for(int w=0;w<2000000000;w++);
         }
                if(ke=='3'||key==3)
        {                       crowfly();
                ke=0;//else
          for(int w=0;w<2000000000;w++);
         }
                if(ke=='4'||key==4)
         {                      crowfly1();
                ke=0;// else
           for(int w=0;w<2000000000;w++);
         }
                        if(ke=='5'||key==5)
            {                           foxmove();
                ke=0;//                 else
             for(int w=0;w<2000000000;w++);
             }
                                if(ke=='6'||key==6)
                {       crowsings();
                        ke=0;//                 else
                 for(int w=0;w<2000000000;w++);
                 }
                                        if(ke=='7'||key==7)
                    {   foxmove1();
                        ke=0;//                         else
                     for(int w=0;w<2000000000;w++);
                     }
                                                if(ke=='8'||key==8)
                        { crowfly2();
                          ke=0;
```

```
                                for(int w=0;w<2000000000;w++);
                                }
                                    if(ke=='0'||key==0)
                                                                exit(0);
                                                else
                                {       moral();
                                for(int w=0;w<2000000000;w++);
                                }
        }
void main(int argc,char **argv)
{
        printf("************ Events are as follows ***************\n");
        printf("------------ Team members name ------------------------------------ \n");
        printf("------------ Num 1 : Project Topic --------------------------------- \n");
        printf("------------ Num 2 : Nature (Background) --------------------------- \n");
        printf("------------ Num 3 : Crow fly and come near Cheese ----------------- \n");
        printf("------------ Num 4 : Crow sit on tree ------------------------------ \n");
        printf("------------ Num 5 : Fox come near Tree ---------------------------- \n");
        printf("------------ Num 6 : Crow Drops Cheese and starts singing ---------- \n");
        printf("------------ Num 7 : Fox takes cheese and run from there ----------- \n");
        printf("------------ Num 8 : Crow fly with vain ---------------------------- \n");
        printf("------------ Num 9 : Moral of the Story ---------------------------- \n");
        printf("------------ After finishing, press Num 0 to EXIT ------------------ \n");
        glutInit(&argc,argv);
        glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
        glutInitWindowSize(2000,2000);
        glutInitWindowPosition(0,0);
        glutCreateWindow("THE FOX AND THE CROW");
        init();
        glutKeyboardFunc(mykeys);
        glutDisplayFunc(display);
        glutMainLoop();
}
```

IV. SCREENSHOTS

A snapshot is the state of the system at a particular point in time. It can refer to the actual copy of a state of a system or to a capability provide by systems.

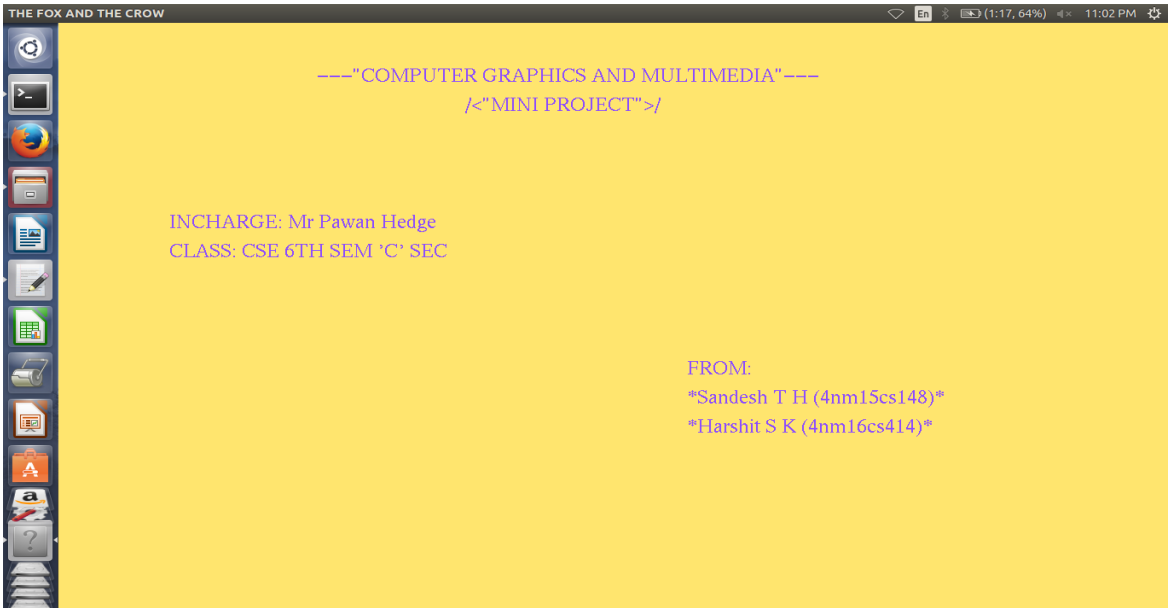

Fig 1 : Initial Output, Team members

Fig 2 : Key 1 : Project Topic

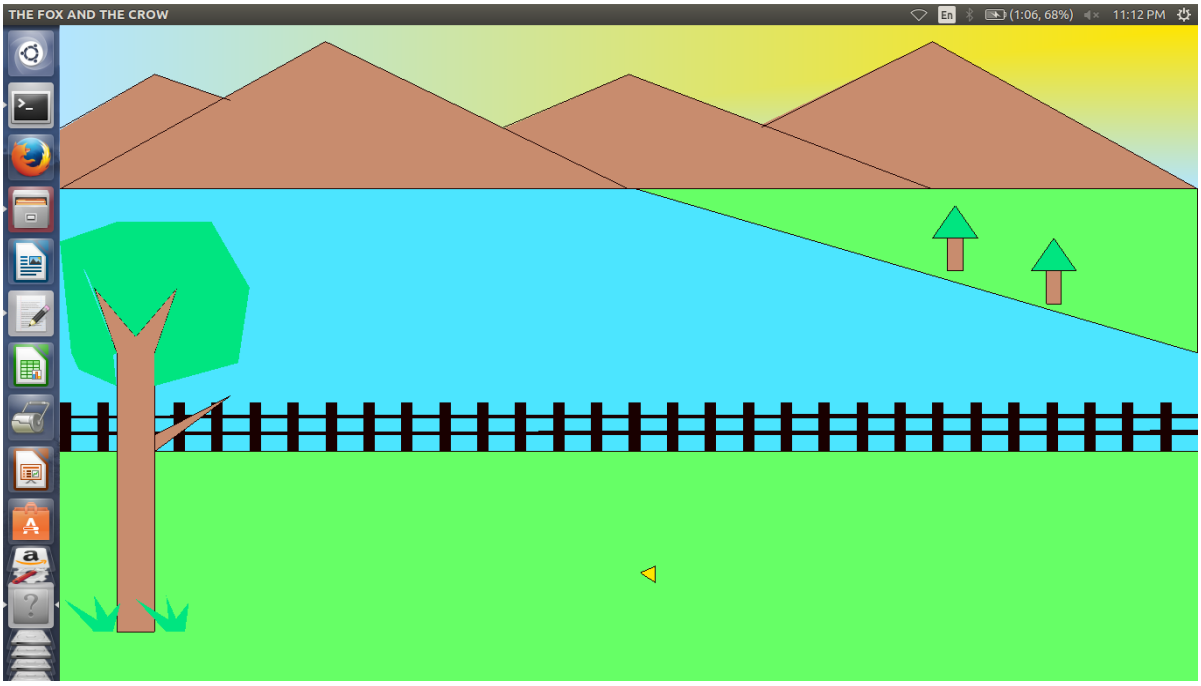

Fig 3 : Key 2 : Nature (Background)

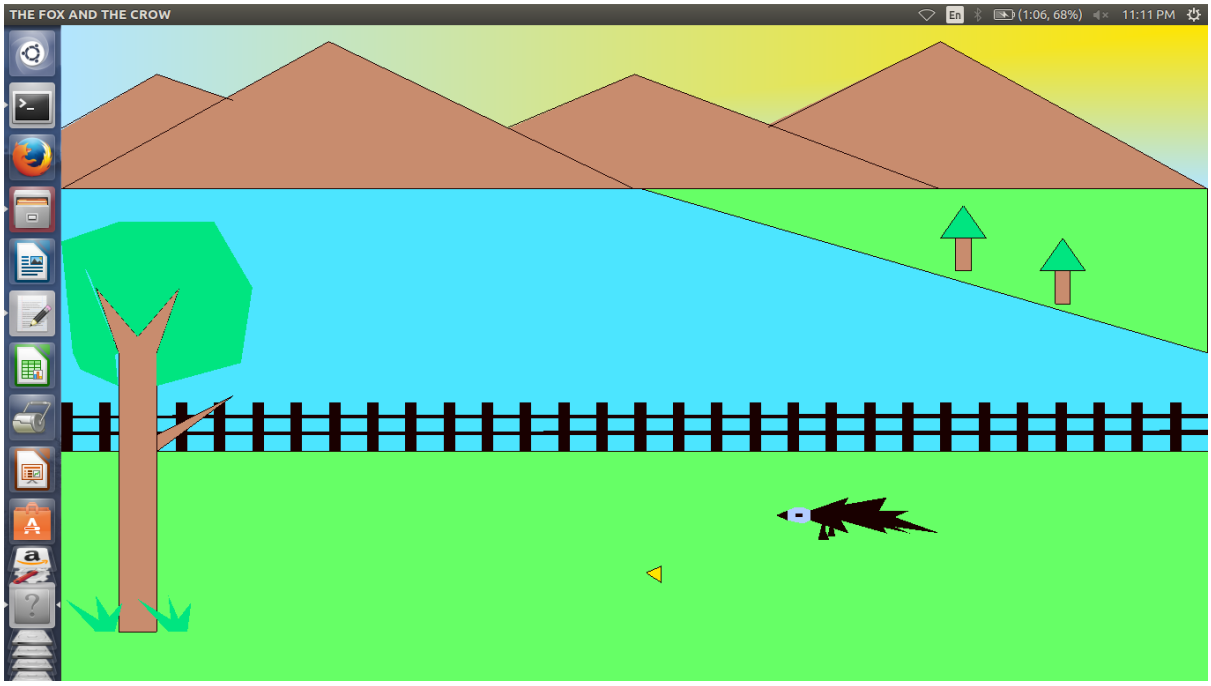

Fig 4 : Key 3 : Crow fly and come near Cheese

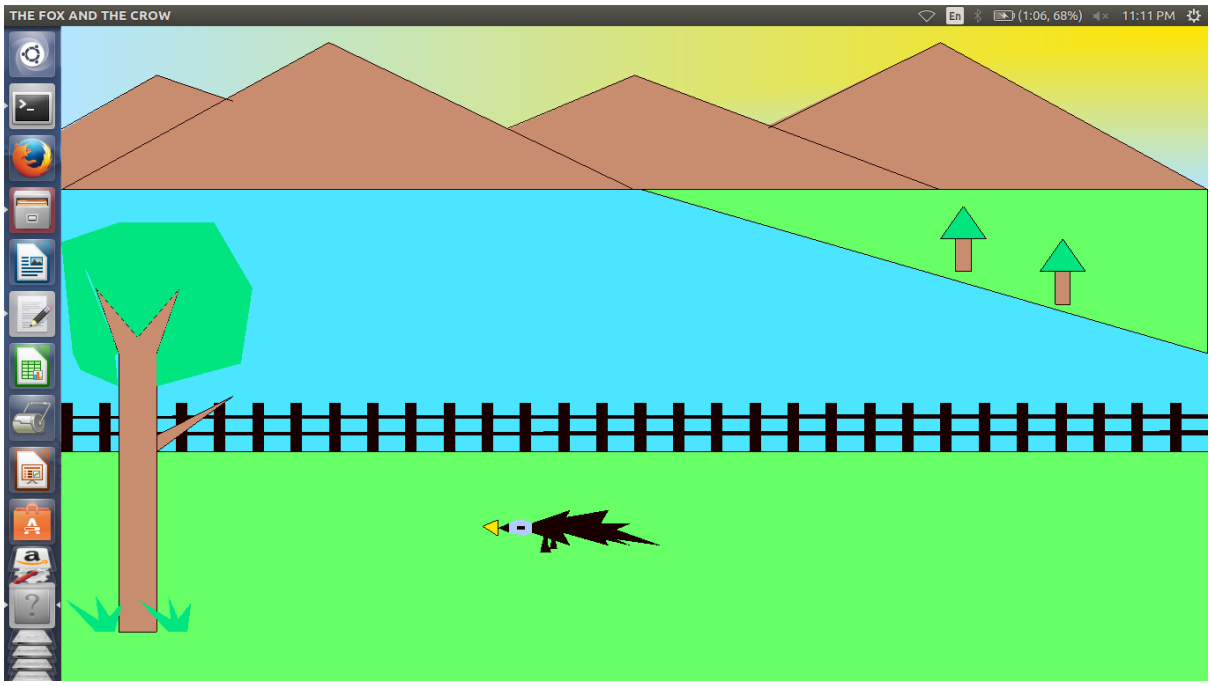

Fig 5 : Key 4 : Crow sit on tree

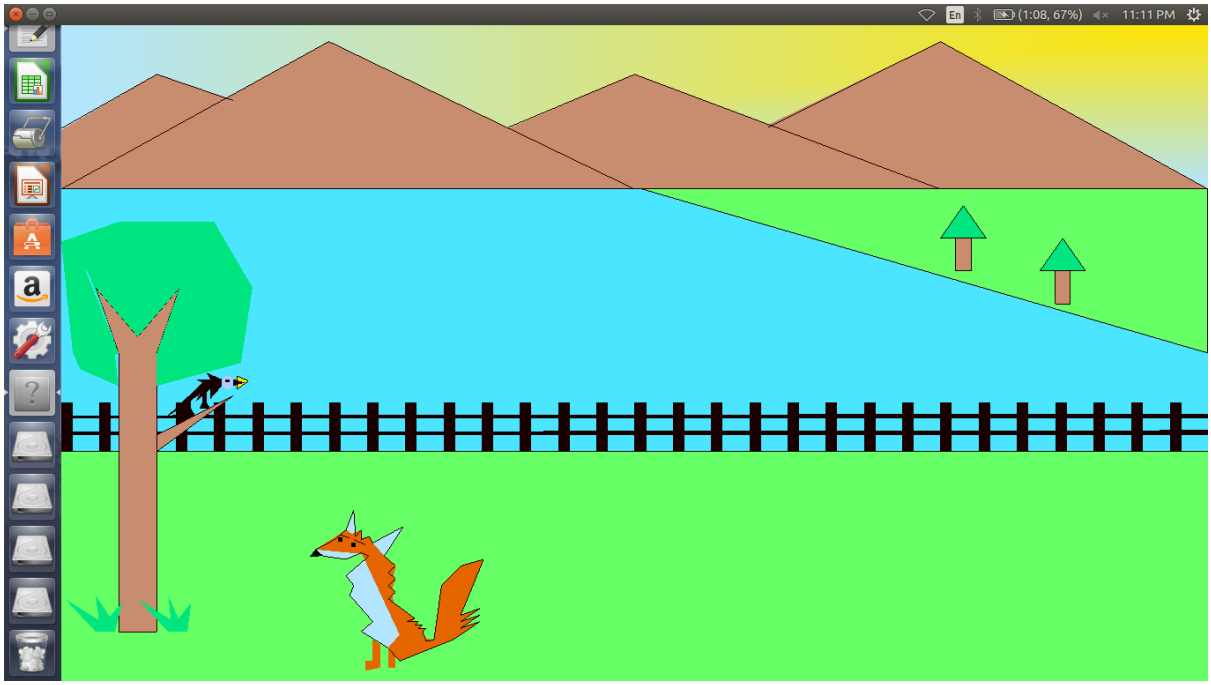

Fig 6 : Key 5 : Fox come near Tree

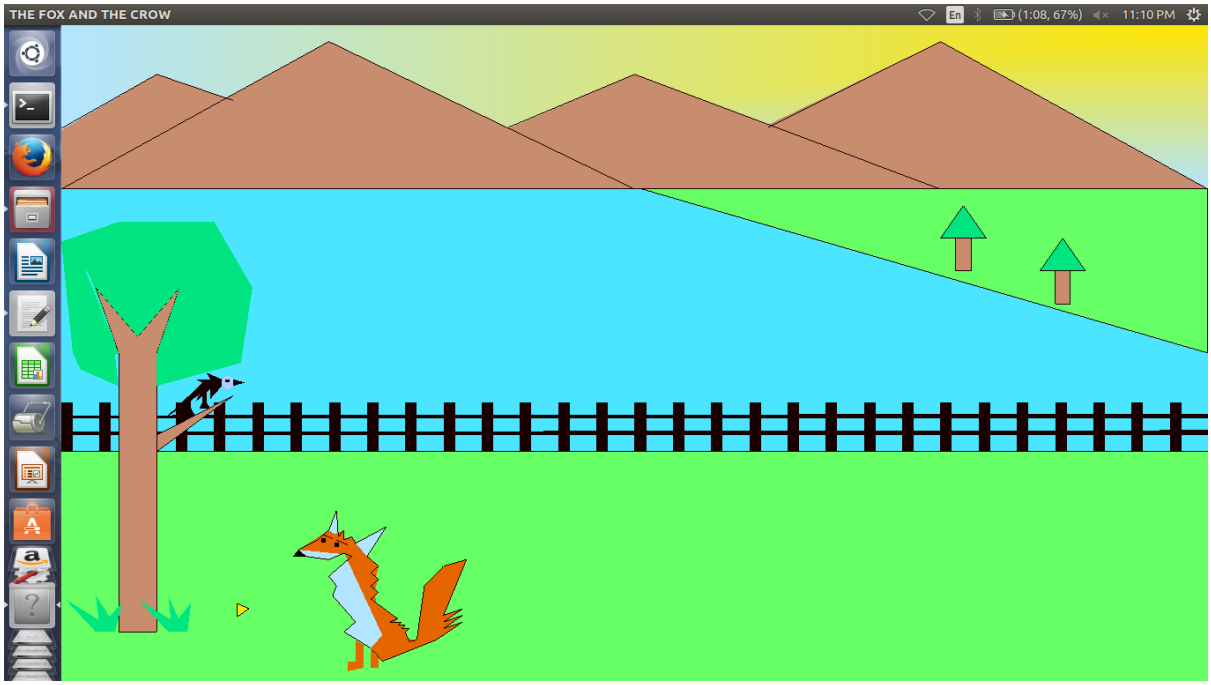

Fig 7 : Key 6 : Crow Drops Cheese and starts singing

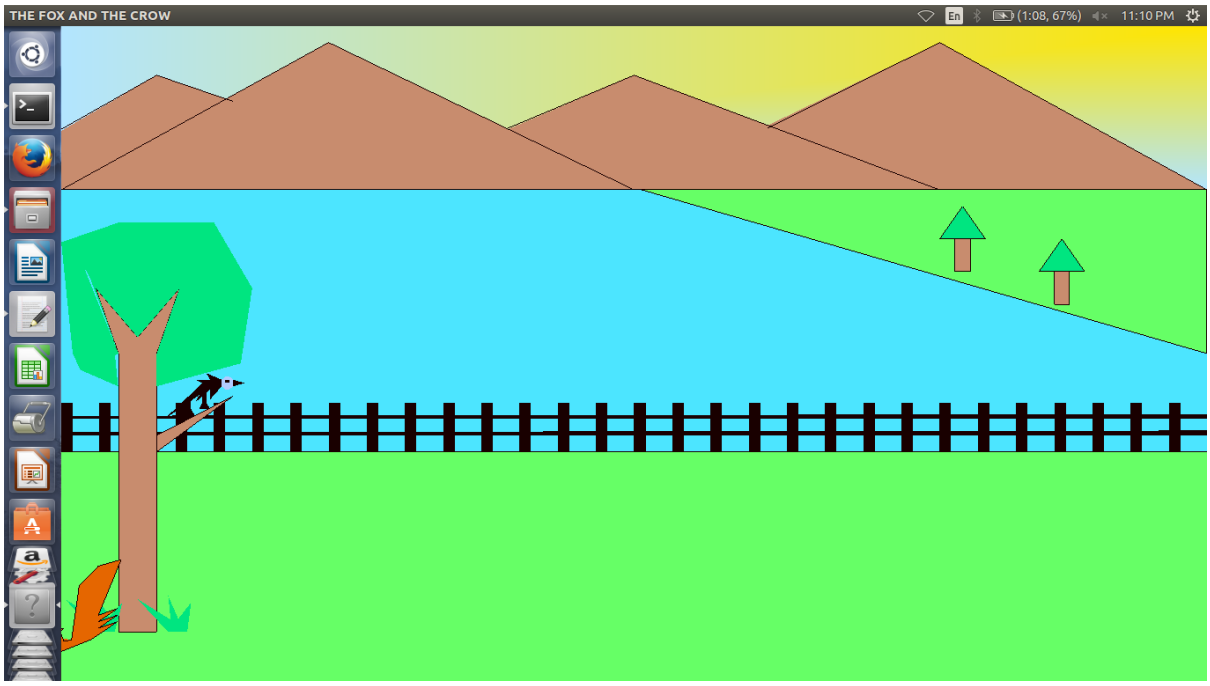

Fig 8 : Key 7 : Fox takes cheese and run from there

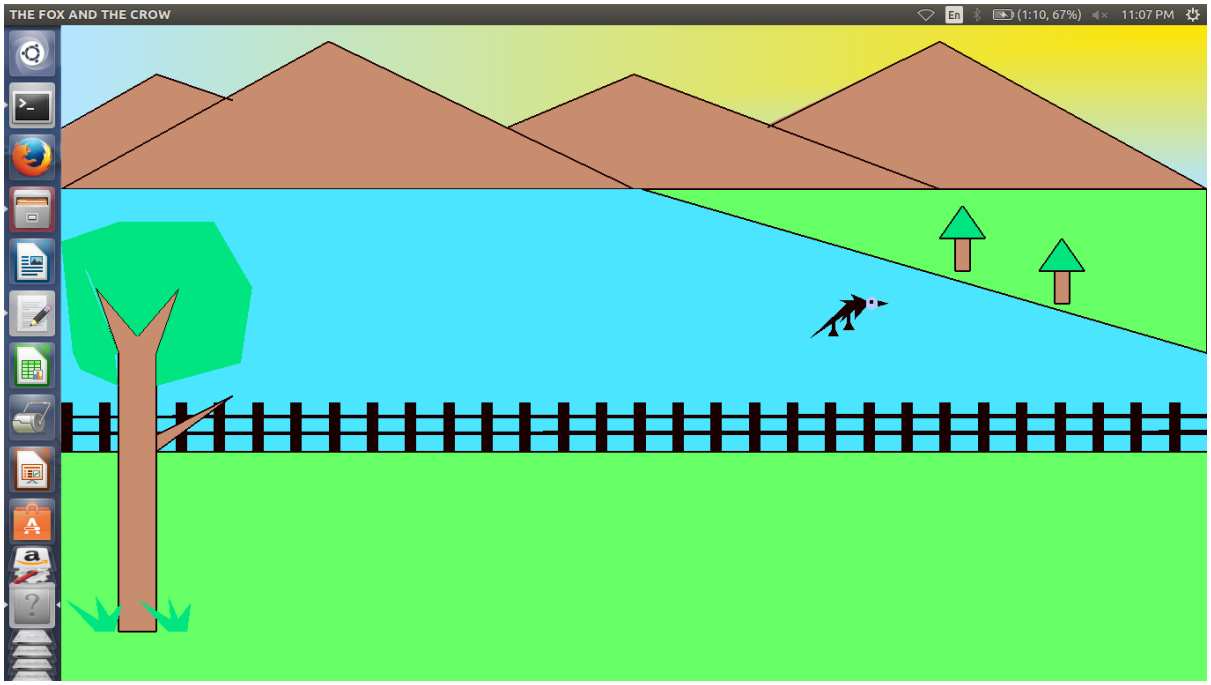

Fig 9 : Key 8 : Crow fly with vain

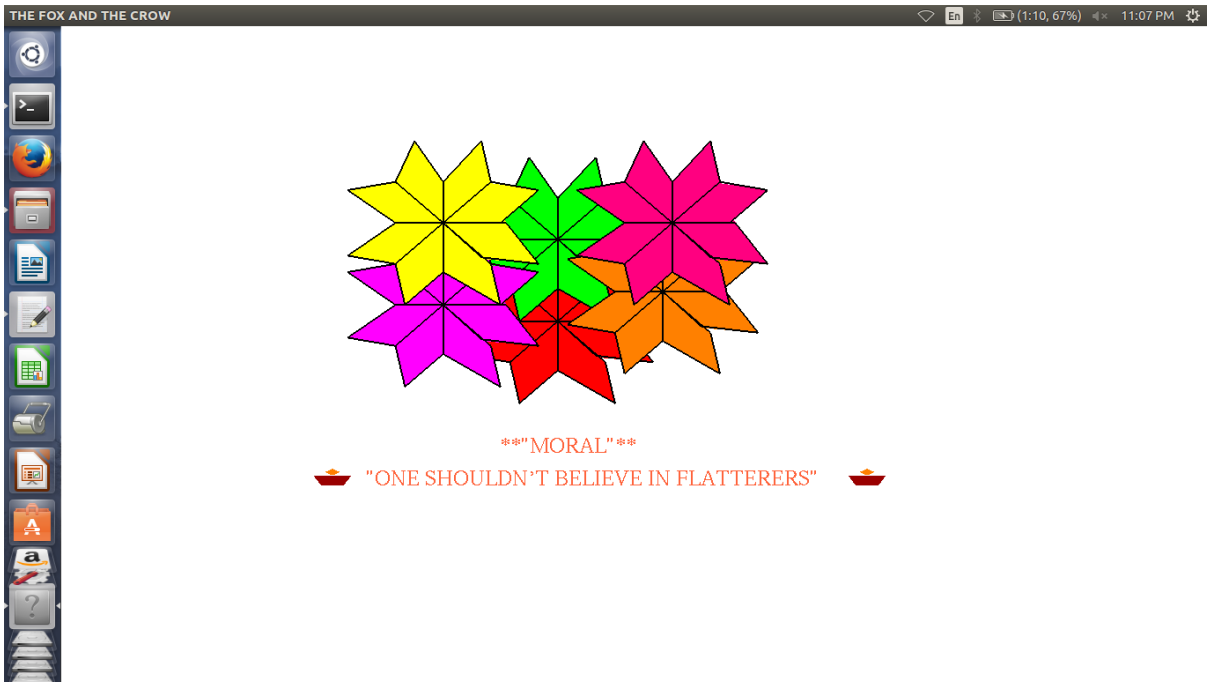

Fig 10: Key 9 : Moral of the Story

V. CONCLUSION

The project has involved the design of story “The Fox and The Crow”. It is implemented using some in-built functions which are provided in the standard graphics package. Some of the functions are mountain, tree, sky etc. We have shown the view of story which tells one should not be vain.

The graphics system allows the user to decline pictures that includes variety of transformations. This is an interactive project which has user friendly interaction given through keyboard. Thus this project meets the basic requirements successfully and is flexible in all respects to one and all.

We aim to build our project and use suitable translations functions to move the objects. We have also introduced the user interactivity by providing various keyboard functions. The working of this project is graphically shown in the snapshots. In future this project can be made more attractive by adding audio clips. As the project has been developed using the graphics package, there is a lot of scope for implementing additional features. This project is still in its primitive stages. It can be further enhanced by making it 3D ant hill using advanced OpenGL functions.

VI. REFERENCES

Books

[1] Edward Angel, 2009, Interactive Computer Graphics. A Top-Down Approach Using OpenGL, 5th edition, Pearson education.

[2] F.S.Hill, 2001, Computer Graphics Using OpenGL, 2nd edition, Pearson Education.

[3] James D Foley, Andres Van Dam, Steven K Feiner, John F Hughes, `Computer Graphics, Addison-Wesley 1997.

[4] Donald Hearn and Pauline Baker: Computer Graphics- OpenGL Version, 2nd Edition, Pearson Education, 2003.

Websites

- http://www.opengl.org/resources/code/samples/redbook
- http://www.talisman.org/opengl-1.1/Reference.html
- http://www.codecolony.de/opengl.html