CHAPTER 1

INTRODUCTION

1.1 Computer Graphics

Computer graphics is a sub-field of computer science and is concerned with digitally synthesizing and manipulating visual content. Although the term refers to three-dimensional computer graphics, it also encompasses two-dimensional graphics and image processing. Computer graphics is often differentiated from field of visualization, although two have some similarities. Graphics are visual presentation on some surface like wall, canvas, computer screen. Graphics often combine text, illustration and colour.

Computer graphics started with the display of data on hardcopy plotters and cathode ray tube (CRT) screens soon after the introduction of computers. These models come from a diverse and expanding set of fields, and include physical, mathematical, engineering, architectural and even conceptual structures. 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 keyboard, mouse or touch-sensitive panel on screen.Figure1.1 shows the library organization of OpenGL.

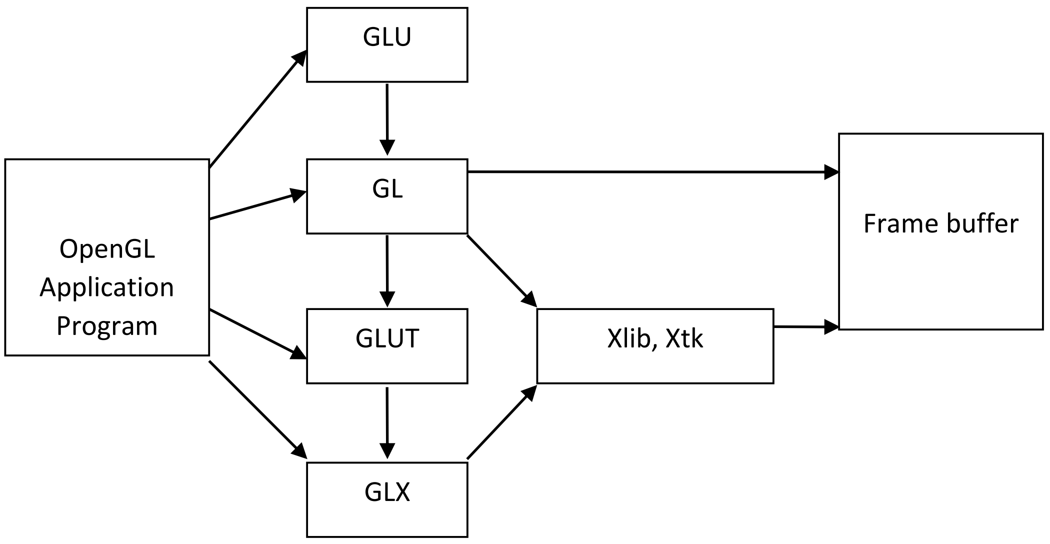


Figure 1.1: OpenGL Library Organization

Computer graphics is no more a rarity. Even people who do not use computers in their daily work encounter computer graphics in television commercials and as cinematic special effects. Computer graphics is a part of all user interfaces and is indispensable for visualizing objects.

Graphical interfaces have replaced textual interfaces as the standard means for user-computer interaction. Graphics has also become a key technology for communication ideas, data and trends in most areas of commerce, science, engineering and education. Much of the task of creating effective graphic communication lies in modelling the objects whose image we want to produce.

1.2 OpenGL

Open Graphics Library (OpenGL) is a cross-language, cross-platform application programming interface (API) for rendering 2D and 3D vector graphics. The API is typically used to interact with a graphics processing unit (GPU), to achieve hardware-accelerated rendering.

Silicon Graphics Inc., (SGI) started developing OpenGL in 1991 and released it in January 1992; applications use it extensively in the fields of computer-aided design (CAD), virtual reality, scientific visualization, information visualization, flight simulation, and video games. Since 2006 OpenGL has been managed by the non-profit technology consortium Khronos Group.

One aspect of OpenGL that suits it so well for use in computer graphics course is its device independence or portability. A student can develop and run program on any available computer. OpenGL offers rich and highly usable API for 2D graphics and image manipulation, but its real power emerges with 3D graphics.

OpenGL provides a powerful but primitive set of rendering commands, and all higher-level drawing must be done in terms of these commands. Also, OpenGL programs have to use the underlying mechanisms of the windowing system. A number of libraries exist to simplify the programming tasks, including the following:

The OpenGL Utility Library(GLU)contains several routines that use lower-level OpenGL commands to perform such tasks as setting up matrices for specific viewing orientations and projections, performing polygon tessellation, and rendering surfaces. The library is provided as part of every OpenGL implementation.

Function names in the OpenGL basic library are prefixed with gl, and each component word within a function name has its first letter capitalized **glBegin, glClear, glCopyPixels, glPolygonMode.** Component words within a constant name are written in capital letters, and the underscore (_) is used as a separator between all component words in the name. GL_2D, GL_RGB, GL_CCW, GL_POLYGON, GL_AMBIENT_AND_DIFFUSE OpenGL data-types GLbyte, GLshort, GLint, GLfloat, GLdouble, Glboolean.

The **OpenGL Utility Library (GLU)** contains several routines that use lower-level OpenGL commands to perform such tasks as setting up matrices for specific viewing orientations and projections, performing polygon tessellation, and rendering surfaces. The library is provided as part of every OpenGL implementation.

.

OpenGL Utility Toolkit (GLUT)

Provides a library of functions for interacting with any screen-windowing system. The GLUT library functions are prefixed with glut contains methods for describing and rendering quadric curves and surfaces. GLUT is an interface to other device-specific window systems, so the programs will be device-independent.

1.3 About Project

Our project basically includes all the simple and basic functions that we have studied and have implemented it in the project. The mouse function and the menu options together help us to scroll through the phases of agriculture.

The menu includes all five phases of agriculture and the final exit. These menu option can be used to toggle between the phases and go through the entire project with all the depiction in detail. Each phase has its own scenes and different function used for the creation of all the object.

The flow is very simple and easy to understand and use. Many of the keyboard functions are used in almost each phase of the project to depict the required scene by pressing the special keys from the keyboard like the up arrow key for the crop to grow, the down arrow key for the rain and the end button for exit.

The still scenes in each of the phase is done using the basic OpenGL functions like the GL_POLYGON, GL_LINE_LOOP, GL_TRIANGLES, etc,.

The movement of the vehicles and the birds are done using the basic Translation function, similarly the rotation of the windmill is also implemented using rotation function.

In the harvesting and collecting phase scaling function is used to depict the collection of cotton after the harvest for transportation. We have also used some of the algorithms learnt in the subject like the circle drawing algorithm.

CHAPTER 2

REQUIREMENT SPECIFICATIONS

The basic purpose of software requirement specification (SRS) is to bridge the communication between the parties involved in the development project. SRS is the medium through which the user's needs are accurately specified; indeed, SRS forms the basis of software development. Another important purpose of developing an SRS is helping the users understand their own needs.

Now we will be discussing the requirement analysis of the project. This report gives the description of the roles of users, the functional overviews of the project, input and output characteristics and also the hardware and software for the project.

2.1 Hardware Requirements

- Processor-Intel or AMD(Advance Micro Devices)
- RAM-512MB(minimum)
- Hard Disk -1MB(minimum)
- Mouse
- Keyboard
- Monitor

2.2 Software Requirements

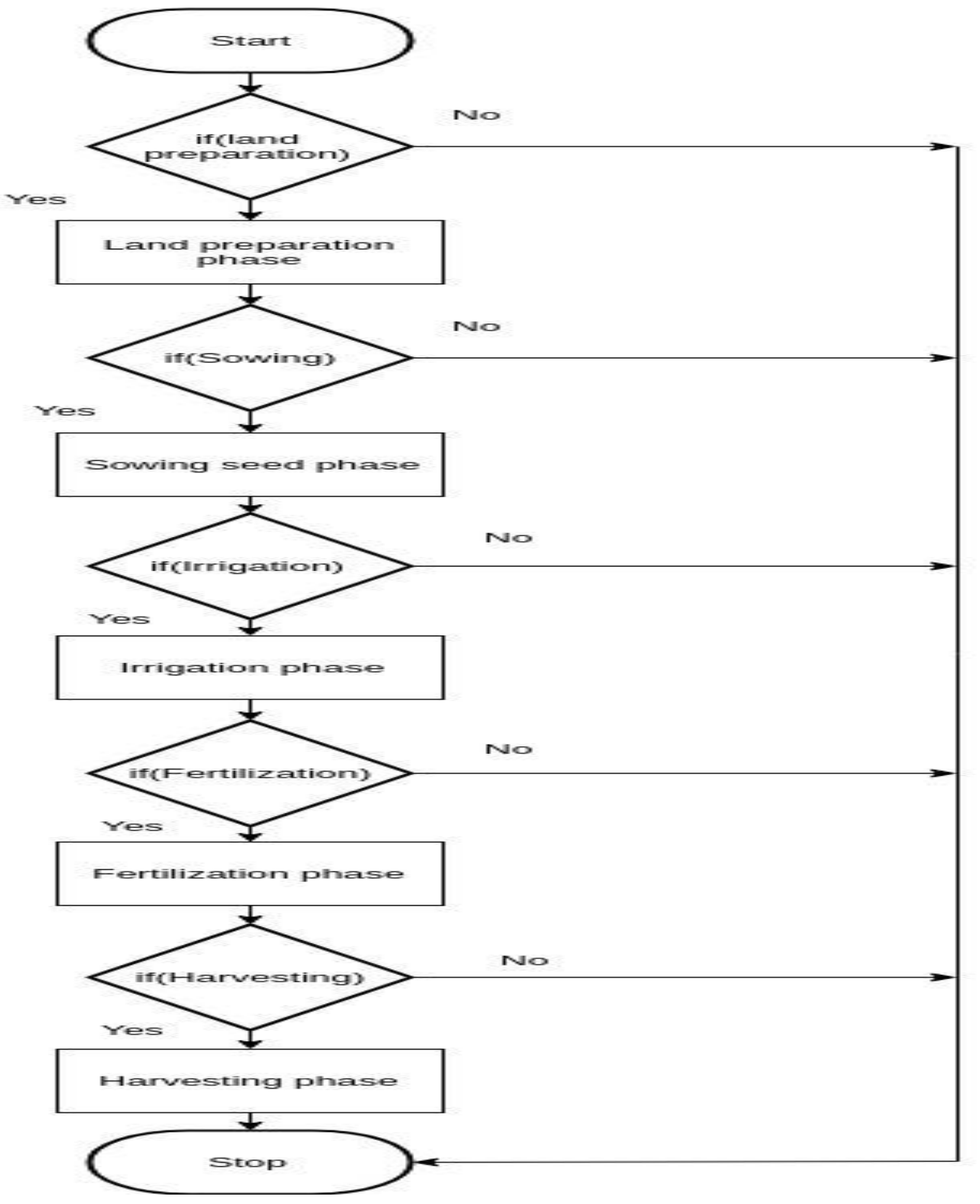
- C++ language compliers.
- OpenGL libraries.
- WINDOWS or LINUX based platform OS.
- gedit.

CHAPTER 3

SYSTEM DESIGN

3.1 FLOWCHART

A flowchart is a common type of chart that represents an algorithms or process showing the steps as boxes of various kinds and their order by connecting these with arrows.



3.1: Flowchart for the project

CHAPTER 4

IMPLEMENTATION

The implementation stage of the model involves the following phases.

- Implementation of OpenGL built in function.
- User defined function Implementation.

4.1 OpenGL Functions

4.1.1 Specifying Simple Geometry

void glBegin (glEnum mode)

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

void glEnd()

Terminates a list of vertices.

4.1.2 Attributes

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 floating–point numbers between 0.0 and 1.0.

void glPointSize(GLfloat size)

Sets the point size attribute in pixels.

4.1.3 Working with the window

void glFlush()

Forces any buffered OpenGL commands to execute.

void glutInit(int argc, char **argv)

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

int glutCreateWindow(char *title)

Creates a window on the display.The string title can be used to label the window.

void glutInitDisplayMode(unsigned int mode)

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

void glutInitWindowSize(int width,int height)

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

void glutInitWindowPosition(int x,int y)

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

void glutMainLoop()

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

void glutDisplayFunc(void (*func)(void))

Registers the display function func that is executed after the current callback returns.

void glutPostRedisplay()

Requests that the display callback be executed after the current callback returns.

4.1.4 Interactions

void glutMouseFunc(void *f(int button, int state, int x, int y))

Registers the keyboard callback function f. The callback function returns the button, the state of the button after the event (GLUT_UP,GLUT_DOWN), and the position of the mouse relative to the top left corner of the window.

void glutKeyboardFunc(void *f(char key, int width, int height)

Registers the keyboard callback function f. The callback function returns the ASCII code of the key pressed and the position of the mouse.

4.1.5 Enabling features

void glEnable(GLenum feature)

Enable an openGL feature. Feature that can be enabled include GL_DEPTH_TEST, GL_LIGHTING, GL_TEXTURE_ID , GL_TEXTURE_2D , GL_TEXTURE_3D, GL_LINE_SMOOTH, GL_POLYGON_SMOOTH, GL_POINTS_SMOOTH, GL_BLEND, GL_LINE_STIPPLE, GL_PLOYGON_STIPPLE, GL_NORMALIZE.

void glDisbale(GLenum feature)

Disable an openGL feature.

4.1.6 Transformations

void glMatrixMode(GLenum mode)

Specifies which matrix will be affected by subsequent transformations. Mode can be GL_MODELVIEW, GL_PROJECTION or GL_TEXTURE.

void glLoadIdentity()

Sets the current transformation matrix to an identity matrix.

void glPushMatrix() & void glPopMatrix()

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

void glRotate[fd](TYPE angle, TYPE dx, TYPE dy, TYPE dz)

Alters the current matrix by a rotation of angle degrees about the axis(dx, dy, dz).

void glTranslate[fd](TYPE x, TYPE y, TYPE z)

Alters the current matrix by a displacement of(x, y, z).

void glScale[fd](TYPE sx, TYPE sy, TYPE sz)

Alters the current matrix by a scaling of(sx, sy, sz).

4.1.7 Viewing

void glOrtho(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble near, GLdouble far)

Defines an orthographic viewing volume with all parameters measured from the centre of projection plane.

4.2 User defined functions

We have used many user defined functions for the convenience in translating and using other basic functions.

1. **void GoMenu(int)**

The function defines the menu options in our project.

There are 6 menus in our project:

1. Preparation of land

2. Seeds Sowing

3. Irrigation of land

4. Fertilization

5. Harvesting

6. Exit

2. **void DrawCircle(float cx ,float cy, float z, float r, int num_segment)**

It is one of the sub function called under function called clouds. It is used for the creation of clouds and other circular objects required in the project. It takes four parameters the x position,

y position, z position and the radius of the circle to be drawn .

3. **void draw()**

The function is called to draw the land of the field. The tiling of land is one of the important phase which requires the proper description of the change in the land and the soil.

4. **void clouds()**

It is used to draw the clouds that are used in the irrigation phase. This function is called during the irrigation phase where the clouds gather together and rain.

5. **void man()**

The function is used to call in the man that is the farmer during the sowing and fertilization phase.

We have include sub function for the man () so as to make it convenient or the movement of the hands and legs during the phases.

All sun functions are written using

glBegin(GL_POLYGON);

glEnd();

Some of the sub functions are listed below:

void rightHand()- to draw the right hand of the farmer.

void leftHand()- to draw the left hand of the farmer.

void neck()- to draw the neck.

void torso()- to draw body of the farmer.

void stripes()- to draw the srips on the dress.

void buttons()- to draw the button on the dress.

void pants()- to draw the outline of the leg.

void movLftLeg()- to a movable part of the left leg for translation effects.

void movRgtLeg()- to a movable part of the right leg for translation effects.

void pocket() and void pocketR()- to draw both pockets on the pants.

void square()- middle design on the dress which is the normal square.

void face()- the outline of the phase of the framer.

void hat()- to the draw the hat on the framers head.

void mustaque() and void eyes()- to draw all the necessary features on the face of the framer.

void fingerRyt() and void fingerLft()- to draw palm for the hands.

void movHand()- a movable part of the hand which is required for the translation of the seeds during sowing seeds and fertilization.

void Basket()- to draw the basket used by the farmer during sowing seeds and irrigation.

void rytLeg() and void leftLeg()- to draw foot of the leg.

void Handlecap()- the tip of the fertilizers.

void handle()- the object used to spray the fertilizer in the fertilization phase.

6. void mouseFunction(int button,int state ,int x,int y)

Its is to enable the mouse function for the menu to appear. Right click on the mouse provides us the required menu in the program.

7. void renderBitmapStringb(float x,float y,const char *string)

It is used to print characters on the console.

8. void renderBitmapStrings(float x, flaot y, const char *string)

Its is used to print the character on the console.

9. void stilldraw()

It is used to draw all the common and the static scenes. During all the five phases there are some common and static scenes involved which are drawn in this function and called in every phase of the project.

10. void grow(int ki)

It called when the arrow up key is pressed for the crops to grow. In the irrigation phase after the rain the crops should be grown, to change the size of the crop this function is called .

11. void cotton()

It is used to draw cotton crop grown at the end of the fertilization phase.

12. void harvestingcar()

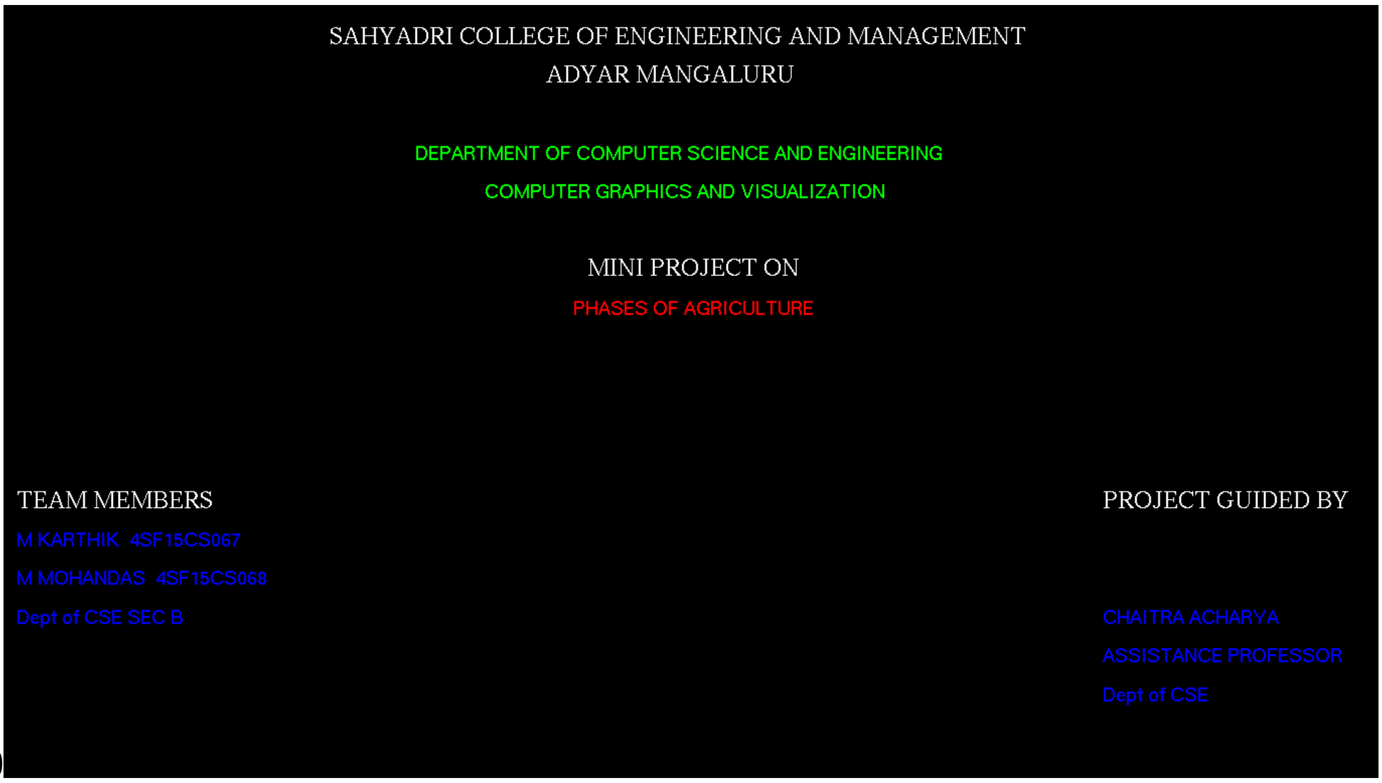
Two harvesting cars that are used to harvest the crop at the end of all the phases.

13. void scene()

It is called in the beginning of the project that displays the name and other required information on the screen. After this function is called all the other require function to the next phase of the project is called depending in the suitable condition.

CHAPTER 5

RESULT

0
SAHYADRI COLLEGE OF ENGINEERING AND MANAGEMENT
ADYAR MANGALURU

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
COMPUTER GRAPHICS AND VISUALIZATION

MINI PROJECT ON
PHASES OF AGRICULTURE

TEAM MEMBERS
M KARTHIK 4SF15CS067
M MOHANDAS 4SF15CS068
Dept of CSE SEC B

PROJECT GUIDED BY
CHAITRA ACHARYA
ASSISTANCE PROFESSOR
Dept of CSE

Figure 5.1: Introduction Page

.

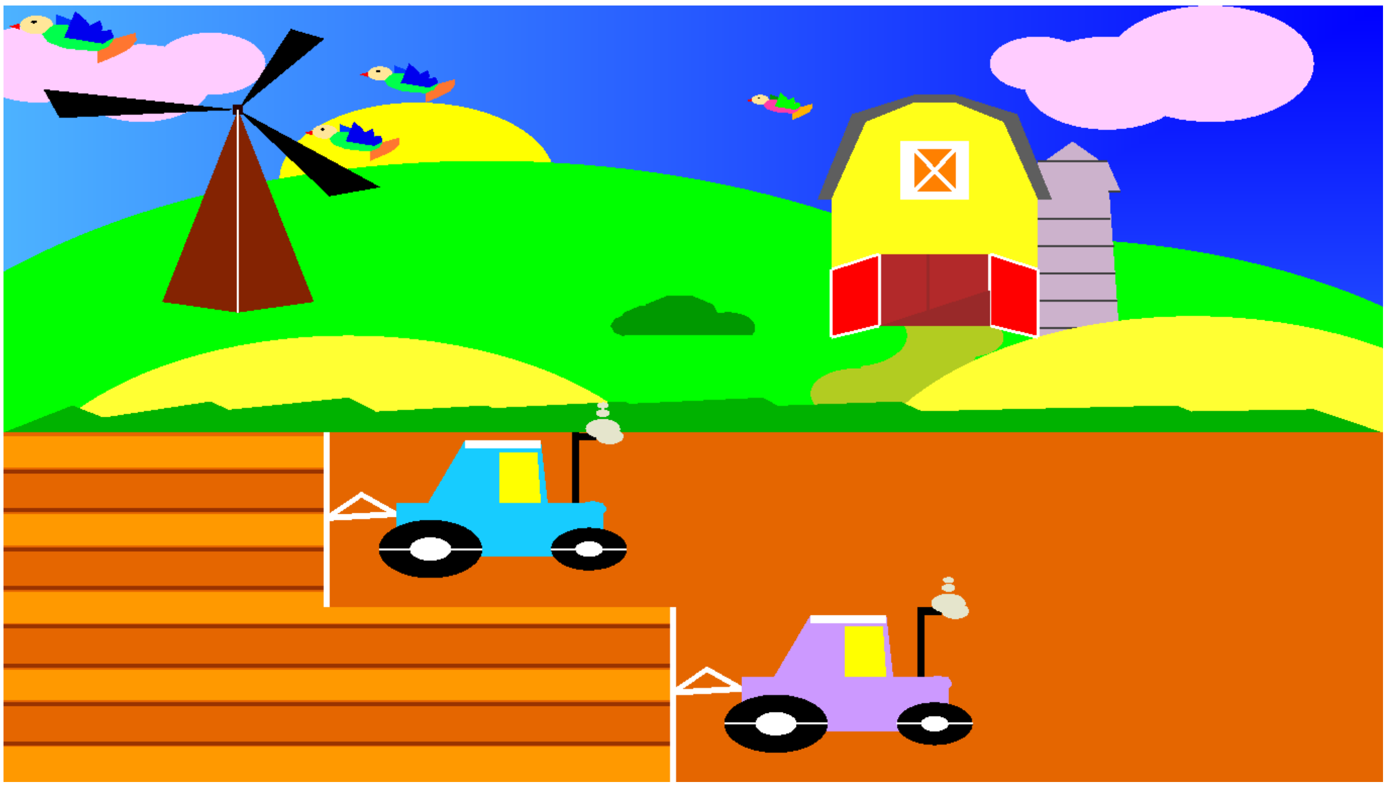


Figure 5.2: Preparation of Land .

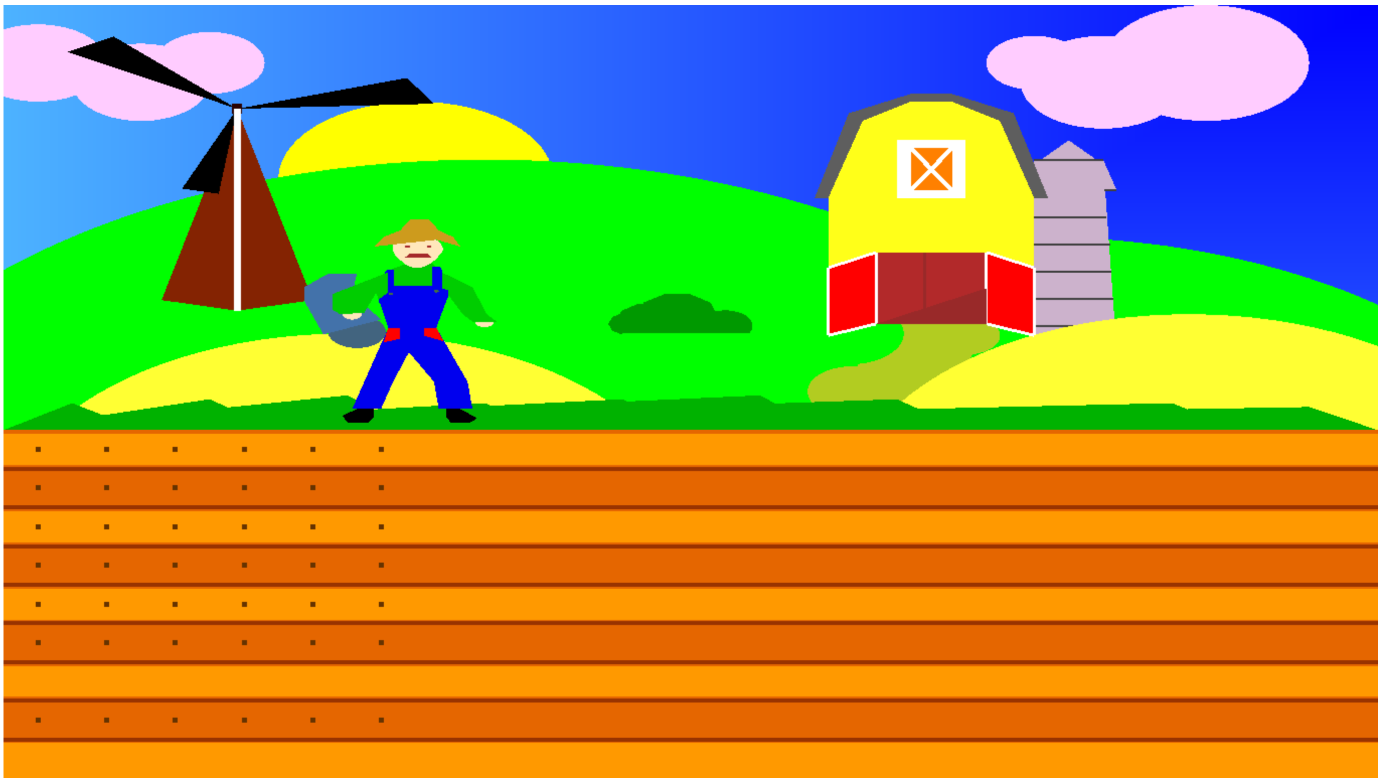


Figure 5.3: Seeds Sowing by farmer.

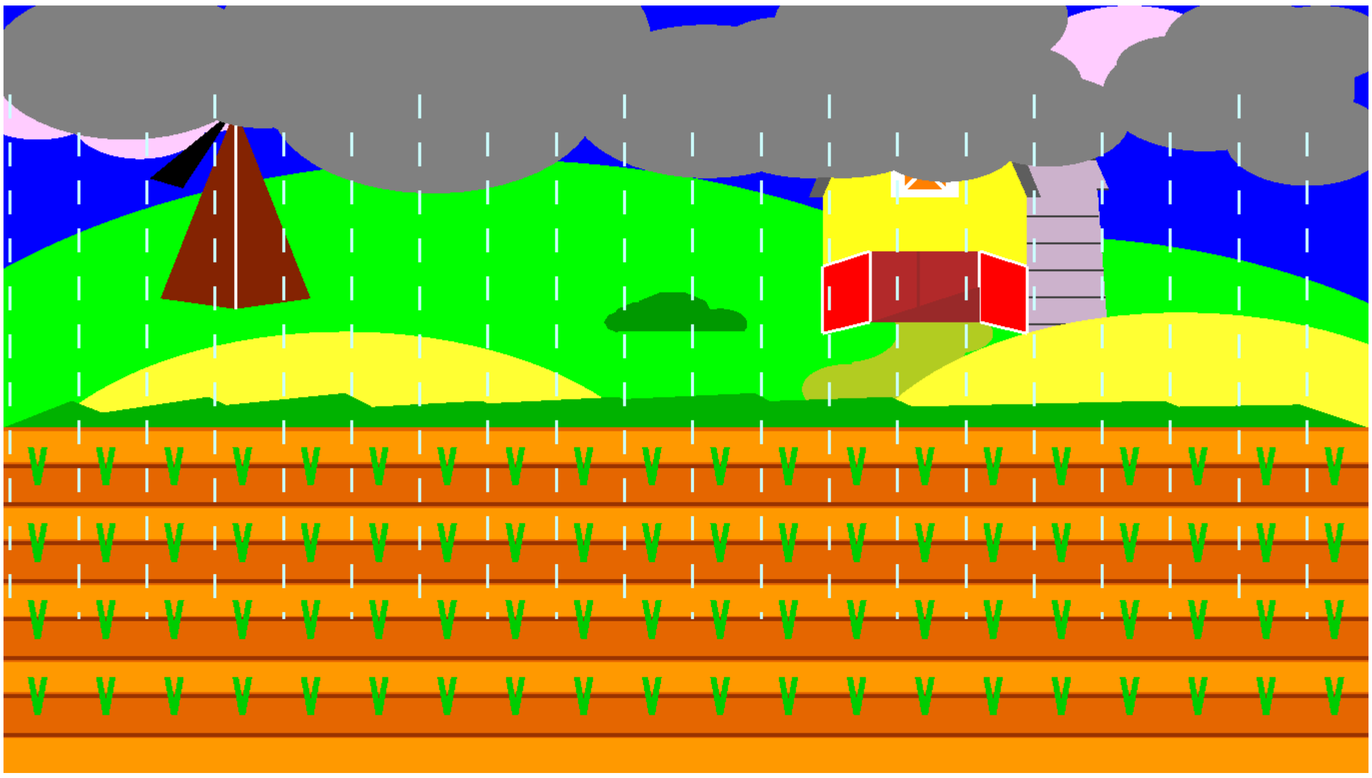


Figure 5.4: Irrigation of land.

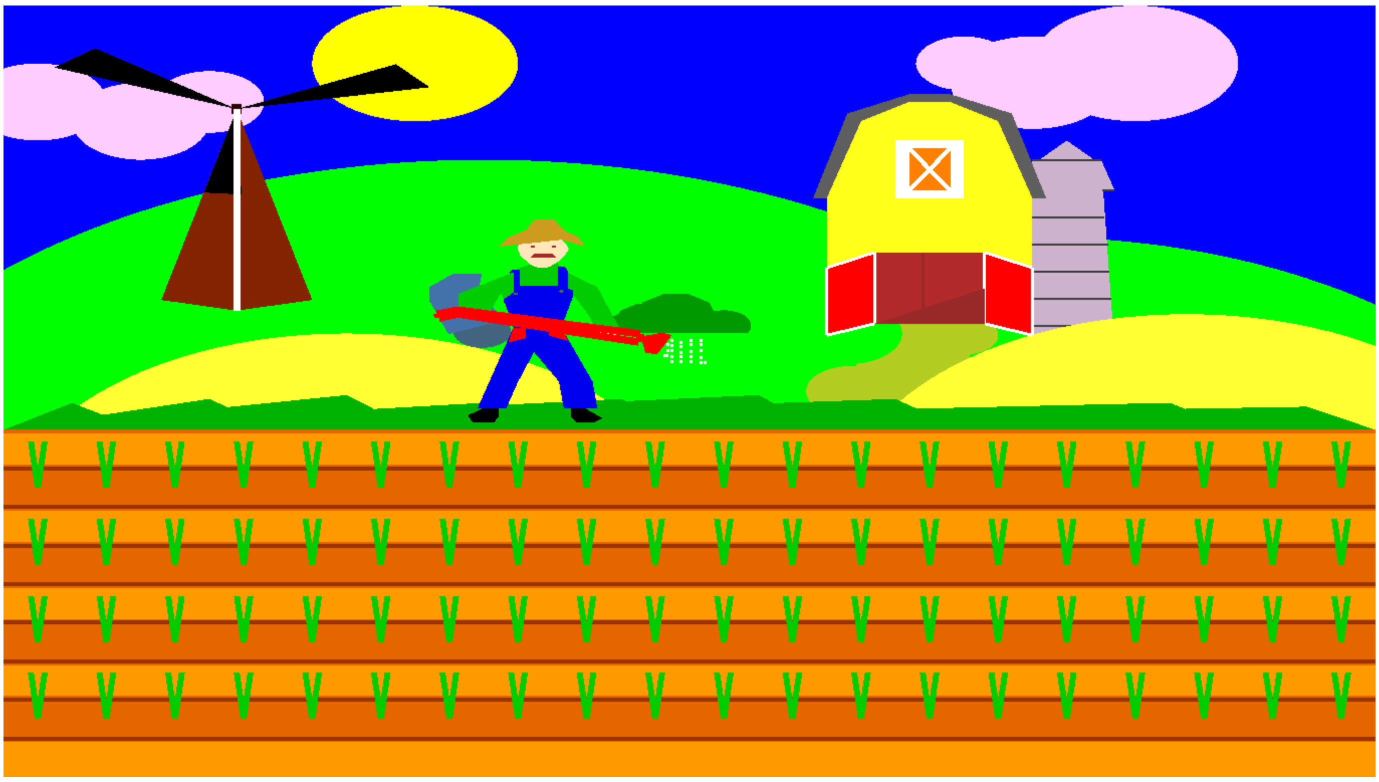


Figure 5.5: Fertilization of farm by farmer.

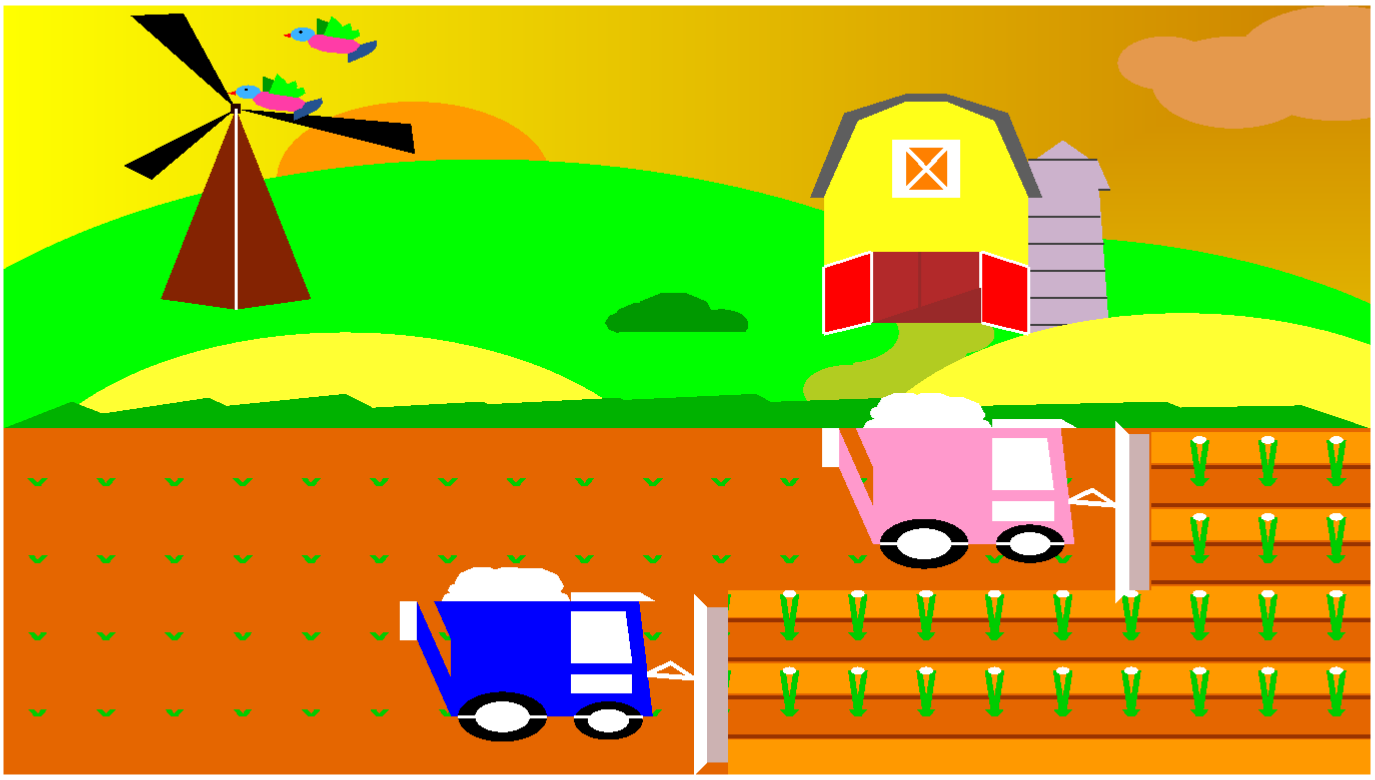


Figure 5.6: Harvesting cotton .

CHAPTER 6

CONCLUSION

Our project **"Phases of agriculture"** has helped us in understanding about computer graphics and OpenGL basic functions which can be used to manipulate the data and provide some animations and various concepts and methodologies used in computer graphics. The project illustrates the working of the phases of agriculture and the working of the farmer in the field during different seasons of the year and different times of the day. It helped us learn and understand more about the subject practically.

The project is user friendly and has features which can be easily understood by any user. It demonstrates the OpenGL applications. The program has been written in C language with OpenGL libraries. The project is used to demonstrate the uses of computer graphics and OpenGL functions in a very informative and colourful way.

REFERENCES

[1].www.geeksforgeeks.com/insertion-Sort/

[2] .www.opengl.org

[3] Interactive Computer Graphics A Top-Down Approach with OpenGL -Edward Angel, 5th Edition, Addison-Wesley, 2008.

[4] James D Foley, Andries Van Dam, Steven K Feiner, John F Hughes, **Computer Graphics**, Pearson Education 1997.

[5] OpenGL Super Bible: Comprehensive Tutorial and Reference (5th Edition).

[6] F.S. Hill Jr**.: Computer Graphics using OpenGL**, 3rd Edition, PHI, 2009.