**Chapter** **1**

**INTRODUCTION**

  This report contains implementation of “**SIMULATION OF COLLEGE BUS**” using a set of OpenGL functions. The project is a 2D view of a college bus which is moving in a virtual world. We are using the keyboard to control the entire motion of the bus. The objects are drawn by using GLUT functions. This project has been developed using Ubuntu14.10 with OpenGL package.

**1.1 Introduction to Computer Graphics**

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 [1]. It has the added advantage that, with the computer, we can make pictures not only of concrete real world objects but also of abstract, synthetic objects, such as mathematical surfaces and of data that have no inherent geometry, such as survey results.

Computer graphics started with the display of data on hardcopy plotters and cathode ray tube 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 phenomena, and so on. Computer graphics today is largely interactive. The user controls the contents, structure, and appearance of the objects and of their displayed images by using input devices, such as keyboard, mouse, or touch-screen. Due to close relationships between the input devices and the display, the handling of such devices is included in the study of computer graphics. The advantages of the interactive graphics are many in number.

Graphics provides one of the most natural means of communicating with a computer, since our highly developed 2D and 3D patter-recognition abilities allow us to perceive and process data rapidly and efficiently. In many design, implementation, and construction processes today, the information pictures can give is virtually indispensable. Scientific visualization became an important field in the 1980s when the scientists and engineers realized that they could not interpret the prodigious quantities of data produced in supercomputer runs without summarizing the data and highlighting trends and phenomena in various kinds of graphical representations.

**1.2 OpenGL Interface**

OpenGL is an application program interface (API) offering various functions to implement primitives, models and images. This offers functions to create and manipulate render lighting, colouring, viewing the models. OpenGL offers different coordinate system and frames. OpenGL offers translation, rotation and scaling of objects.

Most of our applications will be designed to access OpenGL directly through functions in three libraries. They are:

1. Main GL: Library has names that begin with the letter gl and are stored in a library usually referred to as GL.

1. OpenGL Utility Library (GLU): This library uses only GL functions but contains code for creating common objects and simplifying viewing.
2. OpenGL Utility Toolkit(GLUT): This provides the minimum functionality that should be accepted in any modern windowing system.

**1.3 OpenGL Overview**

* OpenGL (Open Graphics Library) is the interface between a graphic program and graphics hardware. *It is streamlined*. In other words, it provides low-level functionality.For example, all objects are built from points, lines and convex polygons.Higher level objects like cubes are implemented as six four-sided polygons.
* OpenGL supports features like 3-dimensions, lighting, anti-aliasing, shadows, textures, depth effects, etc.
* *It is system-independent*. It does not assume anything about hardware or operatingsystem and is only concerned with efficiently rendering mathematically described scenes. As a result, it does not provide any windowing capabilities.
* *It is a state machine*. At any moment during the execution of a program there is acurrent model transformation
* *It is a rendering pipeline*. The rendering pipeline consists of the following steps:
* Defines objects mathematically.
* Arranges objects in space relative to a viewpoint.
* Calculates the color of the objects.
* Rasterizes the objects.

Graphics provides one of the most natural means of communicating with a computer, since our highly developed 2D and 3D pattern-recognition abilities allow us to perceive and process pictorial data rapidly and efficiently. Interactive computer graphics is the most important means of producing pictures since the invention of photography and television. It has the added advantage that, with the computer, we can make pictures not only of concrete real world objects but also of abstract, synthetic objects, such as mathematical surfaces and of data that have no inherent geometry, such as survey results.[2]

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. OpenGL was developed by silicon graphics Inc. (SGI) in 1992 and is widely used in CAD, virtual reality, scientific visualization, information visualization and flight simulation. It is also used in video games.

**OpenGL serves two main purpose:**

* To hide the complexities of interfacing with different 3D accelerators, by presenting programmer with a single, uniform API
* To hide the differing capabilities of hardware platforms, by requiring that all Implementations support the full openGL, feature set.

OpenGL has historically been influential on the development of 3D accelerator, promoting a base level of functionality that is now common in consumer level hardware:

* Rasterized points, lines and polygons are basic primitives.
* A transform and lighting pipeline.
* Z buffering.
* Texture Mapping.
* Alpha
* Blending.

**Chapter 2**

**SYSTEM REQUIREMENTS SPECIFICATION**

**2.1 Hardware Requirements**

* Microprocessor: **1.0 GHz** and above CPU based on either AMD or INTEL Microprocessor Architecture.
* Main memory: **2 GB RAM**
* Hard disk: **40GB**
* Hard disk sped in RPM: **5400RPM**
* Keyboard: **QWERTY** keyboard
* Monitor: **1027 x 768** display resolution

**2.2 Software Requirements**

* + Programming language – C using OpenGL
  + Operating system – Linux operating system
  + Compiler – C Compiler
  + Graphics library – <GL/glut.h >
  + OpenGL 2.0

**2.3 Functional Requirements**

**OpenGL APIs:**

If we want to have a control on the flow of program and if we want to interact with the window system, then we use OpenGL API’S. Vertices are represented in the same manner internally, whether they are specified as two-dimensional or three-dimensional entities, everything that we do are here will be equally valid in three dimensions. Although OpenGL is easy to learn, compared with other APIs, it is nevertheless powerful. It supports the simple three dimensional programs and also supports the advanced rendering techniques.

**GL/glut.h:**

We use a readily available library called the OpenGL Utility Toolkit (GLUT), which provides the minimum functionality that should be expected in any modern windowing system.[3]

The application program uses only GLUT functions and can be recompiled with the GLUT library for other window system. OpenGL makes a heavy use of macros to increase code readability and avoid the use of magic numbers. In most implementation, one of the include lines.

**Chapter 3**

**ABOUT THE PROJECT**

**3.1 Overview**

This Project is on “Simulation of college bus” which is a demo of implementation of Computer Graphics using *OpenGL* *Functions.* It is a User interactive program where in the User can view the required displayby making use of the input device like Keyboard. The project demonstrates the simulation of a college bus. The project consists of a virtual world with numerous elements like a bus, the bus-stop, a student, the college etc. The main objective of this project is to drive the bus to the bus-stop, pick up the student and drop her to the college.

Initially, the bus is driven to the bus-stop where the student is waiting, on reaching the bus-stop the student boards the bus. The bus is then driven to the respective college. On reaching the college, the student gets down the bus and then it is driven to the parking lot.

We are mainly using keyboard as interface to perform the necessary operations in the project with respect to controlling the movement of the bus.

**3.2 User interface**

The whole project is controlled by using the UPPER ARROW KEY. This key controls the entire movement of the bus and enables to move the bus in the appropriate direction.

**3.3 Objective**

* The aim of project is to demonstrate a 2D view and various use of opengl/glut function.
* As Linux doesn’t provide graphics editor, it should be designed in such a way that it provides a very useful graph implementation interface.
* It should be easy to understand, user interactive interface.
* Providing human interaction through Mouse and keyboard.

**Chapter 4**

**IMPLEMENTATION**

**4.1 Existing System**

Existing system for a graphics is the TC++. 3D graphics package being designed should be easy to use and understand. It should provide various options such as free hand drawing, line drawing, polygon drawing, filled polygons, flood fill, translation, rotation, scaling, clipping etc. Even though these properties were supported, it was difficult to render 2D graphics cannot be very difficult to get a 3 Dimensional object [4]. Even the effects like lighting, shading cannot be provided. So we go for Microsoft Visual Studio software.

**4.2 Proposed System**

To achieve three dimensional effects, open GL software is proposed. It is software which provides a graphical interface. It is an interface between application program and graphics hardware. The advantages are:

1. Open GL is designed as a streamlined.

2. It’s a hardware independent interface i.e it can be implemented on many different hardware platforms.

3. With Open GL we can draw a small set of geometric primitives such as points, lines and polygons etc.

4. It provides double buffering which is vital in providing transformations.

5. It is event driven software.

6. It provides call back functions.

**4.3 User Defined Functions**

* **wheel1():** This function is used to draw the wheels on one side of the car. Here in this function we draw circle by using glutSolidTorus() that renders a solid torus (doughnut) respectively centered at the modeling coordinates origin whose axis is aligned with the Z axis [5].
* **wheel2():** This function is used to draw the wheels on the other side of the car.
* **polygon():** This function is used to draw the body of the bus using glBegin(GL\_POLYGON).
* **colorcube():** This function is used to color the different portions of the bus using glColor3ub().
* **bus\_stop():** This routine is used to create the bus-stop along with the associated elements such as the roof and the two chairs at the bus-stop using glBegin(GL\_POLYGON).
* **road2():** This user-defined function is used to create the road along with the visible stripes.
* **text():** This function is used to display and position the various associated texts on the screen such as the Bus-Stop label using the GL function glutBitmapCharacter().
* **text1():** This function is used to display and position the various associated texts on the screen such as the JSSATE label using the GL function glutBitmapCharacter().
* **text2(),text3(),text4d(),text5d():** These routines perform the same function as the above two text display routines using the GL function glutBitmapCharacter().
* **tree(),tree1(),tree12():** These functions are used to create the tress along with the leaves using the gl Function glutSolidCone().
* **woman():** This routine will create the woman along with her dress,hair and various features who is standing at the bus-stop waiting for the bus to arrive.The main gl functions that are used are glutSolidTorus(),glBegin(GL\_LINES)and glBegin(GL\_POLYGON).
* **man():** This routine will create the man along with his dress,beard,eyes and various features who is standing at the bus-stop waiting for the bus to arrive.The main gl functions that are used are glutSolidTorus(),glBegin(GL\_LINES)and glBegin(GL\_POLYGON).
* **lamppost(), lamppost1(), lamppost2(), lamppost4():** These will create the lamppost using the gl functions glutSolidCube(),glutSolidCone() and glutSolidSphere().
* **wheel1d(), wheel2d():** These functions are used to draw the wheels of the car which are displayed in the second screen of the program.
* **polygond():** This function is used to draw the body of the bus using glBegin(GL\_POLYGON) which is displayed in the second screen.
* **colorcubed():** This function is used to color the different portions of the bus which appears in the second screen.
* **womand():** This routine is used to draw the women who gets down from the bus in the second part of the program.
* **road2d():** This user-defined function is used to create the road along with the visible stripes in the second part of the program.
* **textd(), text1d(), text2d(), text3d():**These functions are used to display and position the various associated texts on the second screen such as the Bus-stop label and JSSATE College label using the GL function glutBitmapCharacter().
* **buildingd():**This function is used to draw the buildings in the second screen of the program.
* **wall():** This routine is used to draw the walls in the second part of the output screen using glBegin(GL\_LINE\_LOOP) and glBegin(GL\_POLYGON).
* **gated():** This function is used to draw the college gate using glBegin(GL\_POLYGON).
* **treed():** This routine is used to draw the trees in the second part of the output screen.
* **shrubd(), shrub1d(), shrub2d(), shrub3d(), shrub4d():**  These functions are used to draw the shrubs in the second part of the program by mainly using glutSolidSphere().
* **stopd():** This function is used to draw the the STOP sign board using glutSolidCube(), glutSolidSphere() and glutSolidTorus().
* **intro():** This function is used to draw the various color cubes and designs in the introduction screen using glBegin(GL\_POLYGON) and glColor3ub().
* **texti():** This routine is used to display the various associated text on the introduction screen.
* **mouse():** This is used to define the mouse function along with the various associated controls which are handled by the mouse in the particular program.
* **bus\_move(),bus\_moved():**These routines are used to control the various movements of the bus both in the first and second part of the program.
* **SpecialKeyFunc():** This routine defines the keys which control the movement of the bus.
* **display():** This routine defines the associated display function.
* **myreshape():** This routine defines the user-defined reshape function.
* **main ():** The main function is used for creating the window for display. Here, we also call the various user-defined functions. The callback functions, i.e., mouse callback, keyboard callback, display callback are written in main. The callback functions registered in main() are glutDisplayFunc(display), glutSpecialFunc(SpecialKeyFunc), glutMouseFunc(mouse)etc.

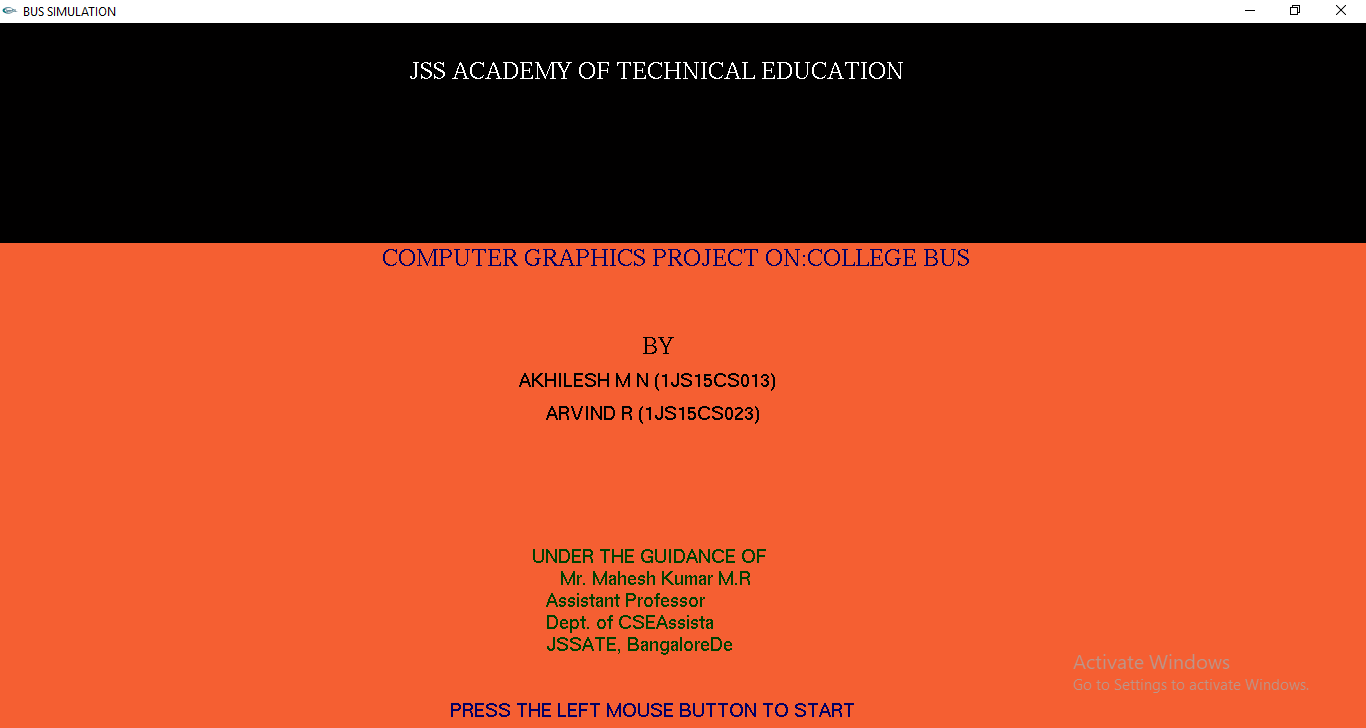
**Chapter 5**

**OpenGL Functions**

* **glColor3f (float, float, float):**This function will set the current drawing color.
* **glClear():**Takes a single argument that is the bitwise OR of several values indicatingwhich buffer is to be cleared.
* **glClearColor ():**Specifies the red, green, blue, and alpha values used by **glClear** to clear the color buffers.
* **GlLoadIdentity( ):**The current matrix with the identity matrix.
* **glMatrixMode(mode):**Sets the current matrix mode,modecan be **GL\_MODELVIEW,GL\_PROJECTION or GL\_TEXTURE.**
* **void glutInit (int \*argc, char\*\*argv):**Initializes GLUT, the arguments from main are passed in and can be used by the application.
* **void glutInitDisplayMode (unsigned int mode):**Requests a display with the properties inmode. The value of mode is determined by the logical OR of options including the color model and buffering.
* **void glutInitWindowSize (int width, int height):**Specifies the initial position of the top-left corner of the window in pixels.
* **glutInitCreateWindow (char \*title):**A window on the display. The string title can be usedto label the window. The return value provides references to the window that can be used when there are multiple windows.
* **void glutDisplayFunc (void (\*func) (void)):**Register the display function func that is executed when the window needs to be redrawn.
* **glutPostReDisplay ( ) :**It requests that the display call back be executed after the current call back returns.
* **void glutMainLoop ():**Causes the program to enter an event-processing loop. It should be the last statement in main function.
* **glTranslatef(GLfloat x,GLfloat y,GLfloat z):**This is used to multiply the current matrix by a translation matrix.The parameters x,y and z specify the coordinates of the translation vector.
* **glBegin(GL\_TRIANGLES):** This treats each triplet of vertices as an independent triangle.
* **glutSolidSphere(GLdouble radius,GLint slices, GLint stacks):** This function renders a sphere centered at the modeling coordinates origin of the specified radius. The sphere is subdivided around the Z axis into slices and along the Z axis into stacks.
* **glBegin(GL\_TRIANGLE\_FAN):** This function draws a connected group of triangles. One triangle is defined for each vertex presented after the first two vertices. Vertices 1, n + 1, and n + 2 define triangle n. N - 2 triangles are drawn.
* **glBegin(GL\_QUAD\_STRIP):** Draws a connected group of quadrilaterals. One quadrilateral is defined for each pair of vertices presented after the first pair. Vertices 2 n - 1, 2 n , 2 n + 2 , and 2 n + 1 define quadrilateral n. N2 - 1quadrilaterals are drawn. Note that the order in which vertices are used to construct a quadrilateral from strip data is different from that used with independent data.
* **glRotatef(GLfloat angle,GLfloat x,GLfloat y,GLfloat z):** glRotate produces a rotation of *angle* degrees around the vector x y z . The current matrix is multiplied by a rotation matrix with the product replacing the current matrix.[6]
* **glBegin(GL\_QUADS):** Treats each group of four vertices as an independent quadrilateral.Vertices 4n - 3,4n - 2, 4n - 1,and 4n define quadrilateral n.N4 quadrilaterals are drawn.
* **glMaterialfv():** glMaterial assigns values to material parameters. There are two matched sets of material parameters. One, the front-facing set, is used to shade points, lines, bitmaps, and all polygons (when two-sided lighting is disabled), or just front-facing polygons (when two-sided lighting is enabled). The other set, back-facing, is used to shade back-facing polygons only when two-sided lighting is enabled.[7]
* **glVertex3f():** glVertex commands are used within glBegin/glEnd pairs to specify point, line, and polygon vertices. The current color, normal, texture coordinates, and fog coordinate are associated with the vertex when glVertex is called.
* **glBegin(GL\_POLYGON):** Draws a single, convex polygon. Vertices 1 through N define this polygon.

**Chapter 6**

**RESULTS AND DISCUSSIONS**

****

**Fig 5.1:** Introduction Screen.

The above figure shows the introduction screen. When the program is run the opening screen is the introduction screen.



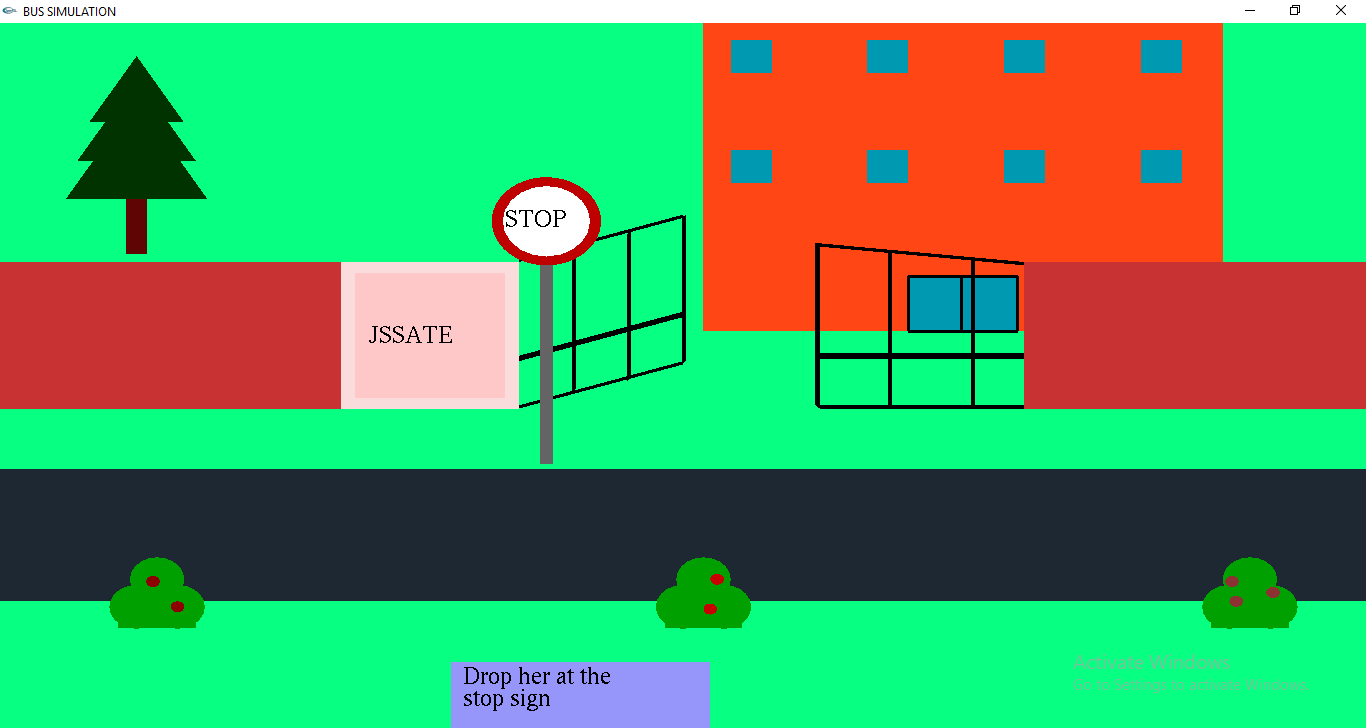
**Fig 5.2:** Bus stop view

The above figure shows the girl waiting as the bus arrives at the bus stop.



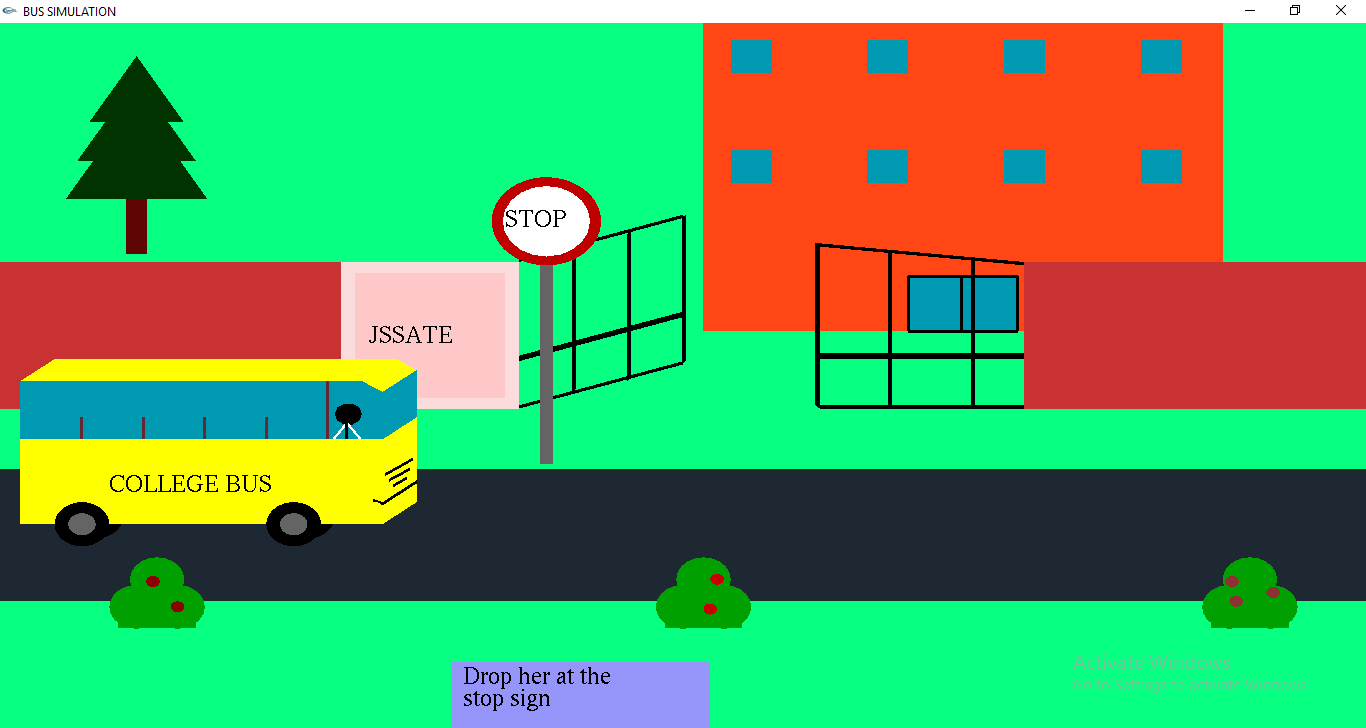
**Fig 5.3:** Arrival of bus

The above figure shows that the girl has boarded the bus and the bus is moving towards the end of the screen as the right button is pressed.



**Fig 5.4:** College view

The above figure is the next screen which depicts the girl’s college. The bus has not yet arrived to the scene.



**Fig 5.5:** Bus reaching college

The above figure shows that the bus is about to reach the college and the girl is ready to get down along with the student.



**Fig 5.6:** Dropping students

The above figure shows both the girl and the student get down near the college and the bus departs from the scene.



**Fig 5.7:** Motion of student

The above figure shows the student walking to the college and is ready to attend classes. The girl is still waiting at the stop sign.

**Chapter 7**

**CONCLUSION AND FUTURE SCOPE**

**7.1 Conclusion**

The project, so described is successfully implemented using many built in functions available in GLUT library in OpenGL package. The Waterfall model was adopted for the software process of the project as the requirements were clearly defined/ frozen at the start of the project. The project has under gone the fundamental steps of requirements generation (SRS), software design, coding and validation. Open Graphics Library (OpenGL) API functions were used extensively for implementing the drawing objects like Polygons, Points and Lines. After the completion of the static model, the animation effects were implemented using the translation *function*.

**7.2 Future Enhancements**

Working on this project has been a rewarding experience and will enable us to design more complex graphics projects in future. This project was bit time consuming, but gave lots of knowledge, skills to us. Extensive testing was carried out before the submission of the project. The project was also completed in the stipulated time frame.

**REFERENCES**

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

**[2]** OpenGL Programming Guide (Addison-Wesley Publishing Company)

**[3]** Computer Graphics – Principals and Practice (Foley, Van Dam, Fenier and Hughes)

**[4]** www.codecademy.com/opengl-example/torus

**[5]** www.stackoverflow.com/glut-libraries/font

**[6]** [www.lighthouse3d.com/examples-motio](http://www.lighthouse3d.com/examples-motio)n

**[7]** www.opengl.org/reference -programs