RAMAIAH
Institute of Technology

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CSE08: COMPUTER GRAPHICS

TERM: Jan-May 2020

OPENGL PROJECT

Submitted to
Dr. S Seema
(Professor)

Submitted By
Darshan R Konnur
1MS17CS026

Rubrics for Assignment 2:

Assessment	Excellent (10)	Very Good (6-8)	Satisfactory (3-5)
Assignment 2 (OpenGL)	A complex 3D model with all graphics concepts applied.	A complex or simple 3D model with minimal features like colour, transformation and camera.	A simple scene with minimal effects

Signature of the Guide

ACKNOWLEDGEMENT

I, Darshan R Konnur, student of Computer Science Engineering of 3rd year in Ramaiah Institute of Technology, Bengaluru, it is my privilege to express my sincerest regards to my project guide, Dr. S Seema, for the valuable inputs, able guidance, encouragement, whole-hearted cooperation and constructive criticism throughout the duration of my project. I deeply express my sincere thanks to our Head of Department Dr. Anita Kanavalli for encouraging and allowing us to have this amazing course in our curriculum. I would like to express my deepest appreciation to all those who provided me the possibility to complete this report.

TABLE OF CONTENTS

Chapter No. **Title** **Page No.**

1. INTRODUCTION

1.1 Purpose

The objective of this project is to demonstrate the working of the transformers through animation. The robot is transformed to a vehicle by repositioning the body parts of robot. First, we can see a robot in the desert which will automatically be transformed to a vehicle and moves on the road track. The whole project is done using OpenGL.

1.2 Scope

OpenGL has always been an interesting and most popular cross platform graphics APIs, It will remain among popular graphics APIs even though Khronos developers have introduced next generation low level Vulkan APIs, OpenGL always best APIs for small applications where you have reasonable performance (FPS). Only in case of performance challenges you can move to Vulkan APIs which are much lower level and give better control on GPU. Otherwise OpenGL is always in demand.

1.3 Definitions and abbreviations

- **Projection:** In computer graphics, there are two common camera projections used.
- **Perspective:** A perspective view is geometrically constructed by taking a scene in 3D and placing an observer at point O. The 2D perspective scene is built by placing a plane (e.g. a sheet of paper) where the 2D scene is to be rendered in front of point O, perpendicular to the viewing direction. For each point P in the 3D scene a PO line is drawn, passing by O and P. The intersection point S between this PO line and the plane is the perspective projection of that point. By projecting all points P of the scene, you get a perspective view [1].
- **Orthographic:** In an orthographic projection, you have a viewing direction but not a viewing point O. The line is then drawn through point P so that it is parallel to the viewing direction. The intersection S between the line and the plane is the orthographic projection of the point P. By projecting all points P of the scene, you get the orthographic view.
- **Rendering:** The process of computationally generating a 2D image from 3D geometry.

1.4 Overview

I was motivated to for this project after I watched the movie ‘Transformers’. I always wonder if I have a small machine which can help me on all my works and that was transformers. So, this project aims at creating a scene on transformer.

2. OVERALL DESCRIPTION

2.1 Project perspective

The Project perspectives are mentioned below. Hardware interfaces and Software interfaces which are aimed to be rendered are as follows.

2.1.1 Hardware interfaces

OpenGL (Open Graphics Library) 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) began developing OpenGL in 1991 and released it on June 30, 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.

2.1.2 Software interfaces

Since OpenGL is just built in a “glut” library, so we need an IDE to import those libraries and to build the project. Hence, I chose Microsoft Visual Studio Code 2017 as an IDE for the project.

2.2 Project functions

The primary aim/function of the project is to render an animation video for the theme I have opted for.

2.3 Constraints

A major hurdle during the design of the project was displaying the movement of the elements. Since OpenGL contains a large number of API’s and built in functions, we were able to clear this hurdle.

3. SPECIFIC REQUIREMENTS

3.1 External interface requirements

3.1.1. User interfaces

- **OpenGL:** OpenGL is the industry's most widely used, supported and best documented 2D/3D graphics API making it inexpensive & easy to obtain information on implementing OpenGL in hardware and software. There are numerous books, tutorials, online coding examples, coding seminars, and classes that document the API, Extensions, Utility Libraries, and Platform Specific Implementations.
- OpenGL is the premier environment for developing portable, interactive 2D and 3D graphics applications. Since its introduction in 1992, OpenGL has become the industry's most widely used and supported 2D and 3D graphics application programming interface (API), bringing thousands of applications to a wide variety of computer platforms. OpenGL fosters innovation and speeds application development by incorporating a broad set of rendering, texture mapping, special effects, and other powerful visualization functions. Developers can leverage the power of OpenGL across all popular desktop and workstation platforms, ensuring wide application deployment.
- Any visual computing application requiring maximum performance-from 3D animation to CAD to visual simulation-can exploit high-quality, high-performance OpenGL capabilities. These capabilities allow developers in diverse markets such as broadcasting, CAD/CAM/CAE, entertainment, medical imaging, and virtual reality to produce and display incredibly compelling 2D and 3D graphics. [3].

3.1.2 Hardware interfaces

- **Minimum:**
  - Windows 8/10, 64 bits (PC or Mac computers using Boot Camp).
  - Any CPU (Intel i5/ i7/ Xeon recommended).
  - Any GPU that is compatible with OpenGL 3.2. (integrated graphics cards Intel HD 4000 or above).
  - Small projects (under 100 images at 14 MP): 4 GB RAM, 10 GB HDD Free Space.
  - Medium projects (between 100 and 500 images at 14 MP): 8 GB RAM, 20 GB HDD Free Space.
  - Large projects (between 500 and 2000 images at 14 MP): 16 GB RAM, 40 GB HDD Free Space.
  - Very Large projects (over 2000 images at 14 MP): 32 GB RAM, 80 GB HDD Free Space.

● **Recommended:**
- Windows 8/10, 64 bits.
- CPU quad-core or hexa-core Intel i7/Intel i9/Thread Ripper/Xeon/.
- GeForce GTX GPU compatible with OpenGL 3.2 and 2 GB RAM.
- Hard disk: SSD.
- Small projects (under 100 images at 14 MP): 8 GB RAM, 15 GB SSD Free Space.
- Medium projects (between 100 and 500 images at 14 MP): 16GB RAM, 30 GB SSD Free Space.
- Large projects (over 500 images at 14 MP): 32 GB RAM, 60 GB SSD Free Space.
- Very Large projects (over 2000 images at 14 MP): 64 GB RAM, 120 GB SSD Free Space.

3.1.3 Software interfaces

OpenGL, it runs on every major operating system:

- Windows 10, 8.1 and 7, XP, Vista
- macOS 10.12+
- Linux

3.2 Performance requirements

- Fast and smooth rendering of 3D models built
- Renders to be clear and compact
- Prior knowledge on rendering engines.
- Lighting should be proper to constitute photorealism.
- Rendering techniques are essential for smooth and time saving and high-quality rendering.

4. BASIC METHODOLOGY

4.1 Rendered images

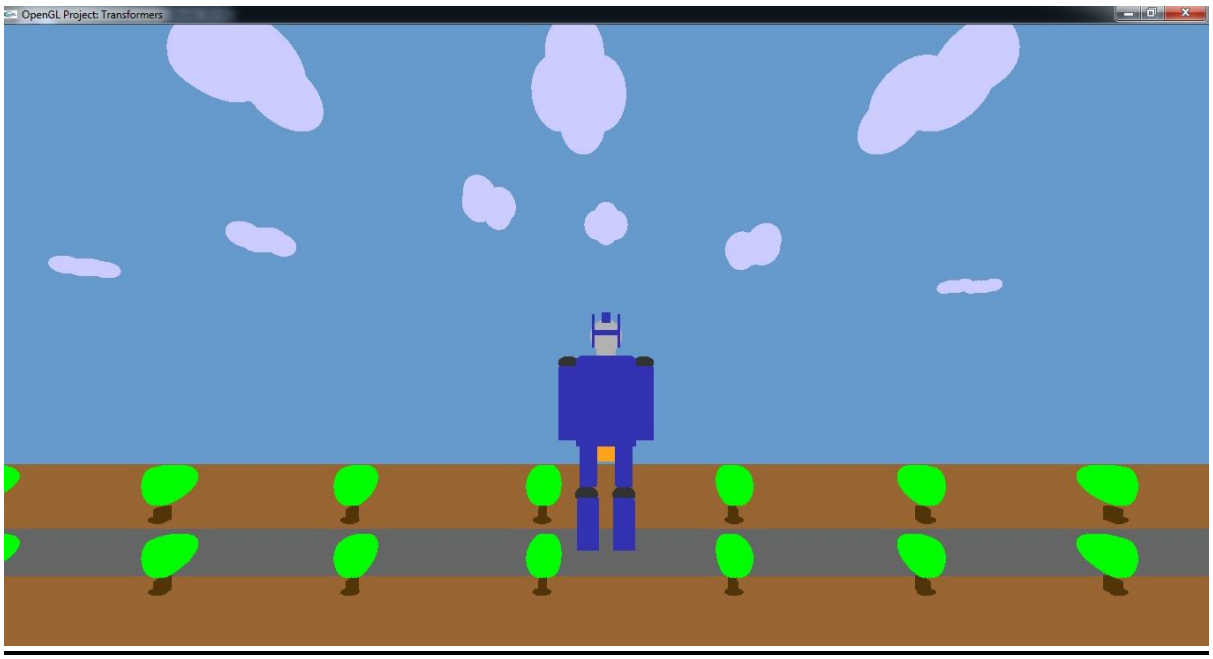

Fig 1. Initial Screen

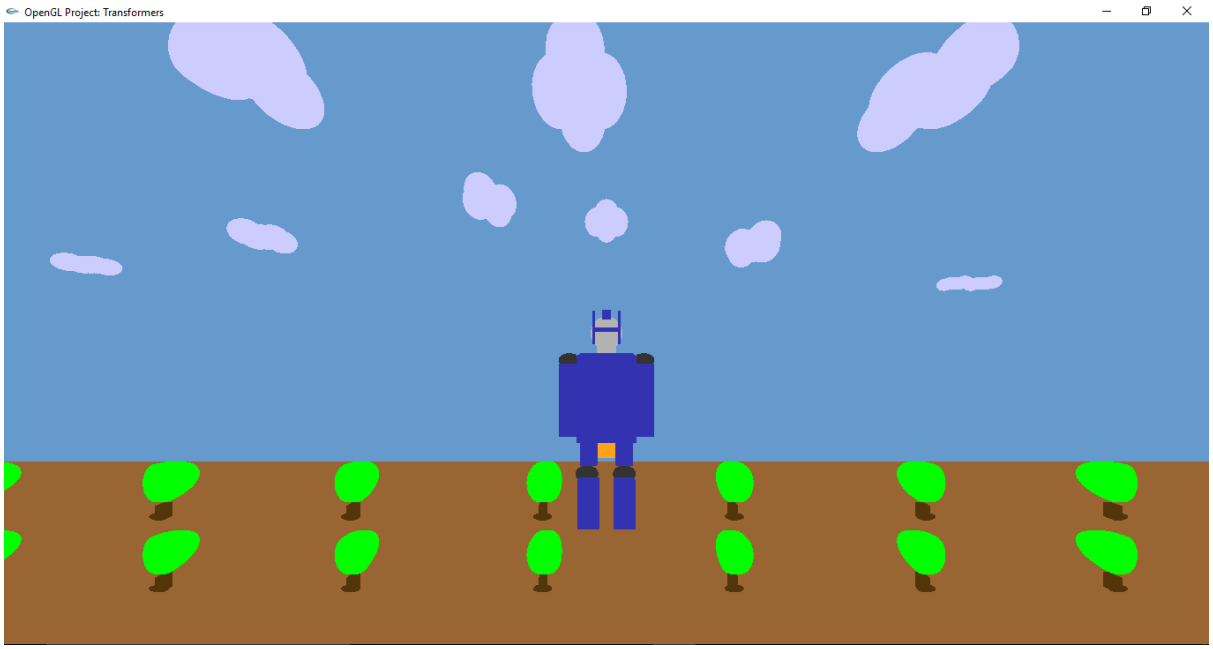

Fig 2. The robot starts to transform

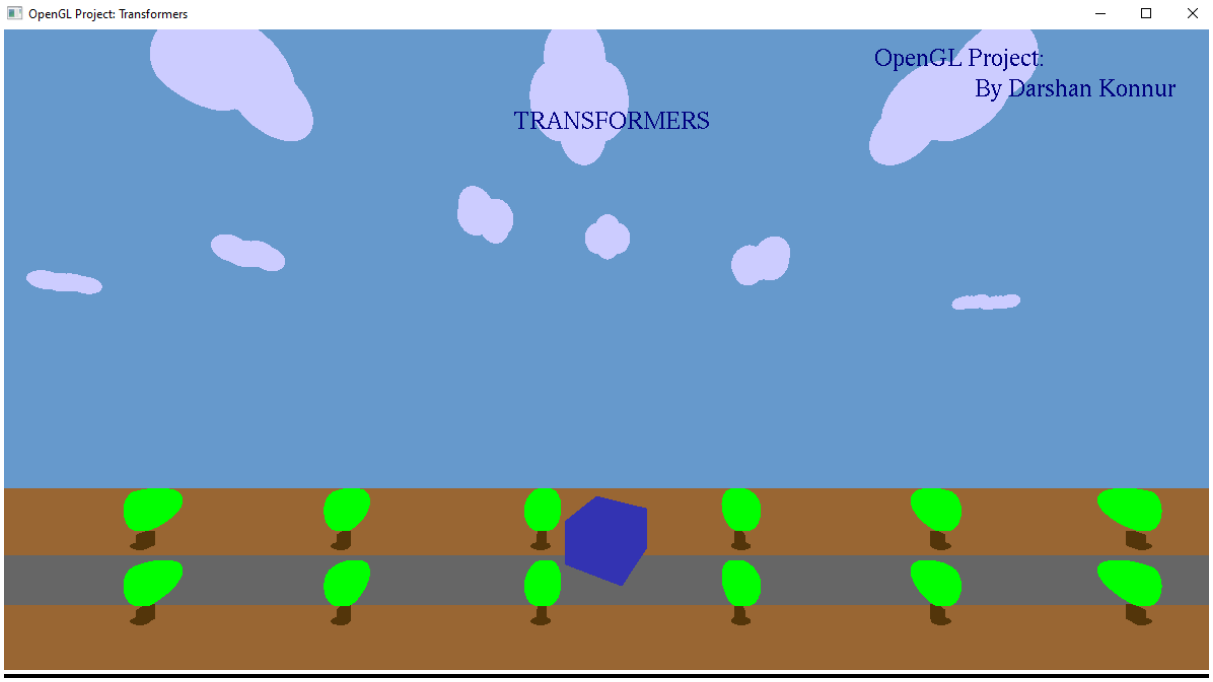

Fig 3. The robot rotates to look like a vehicle

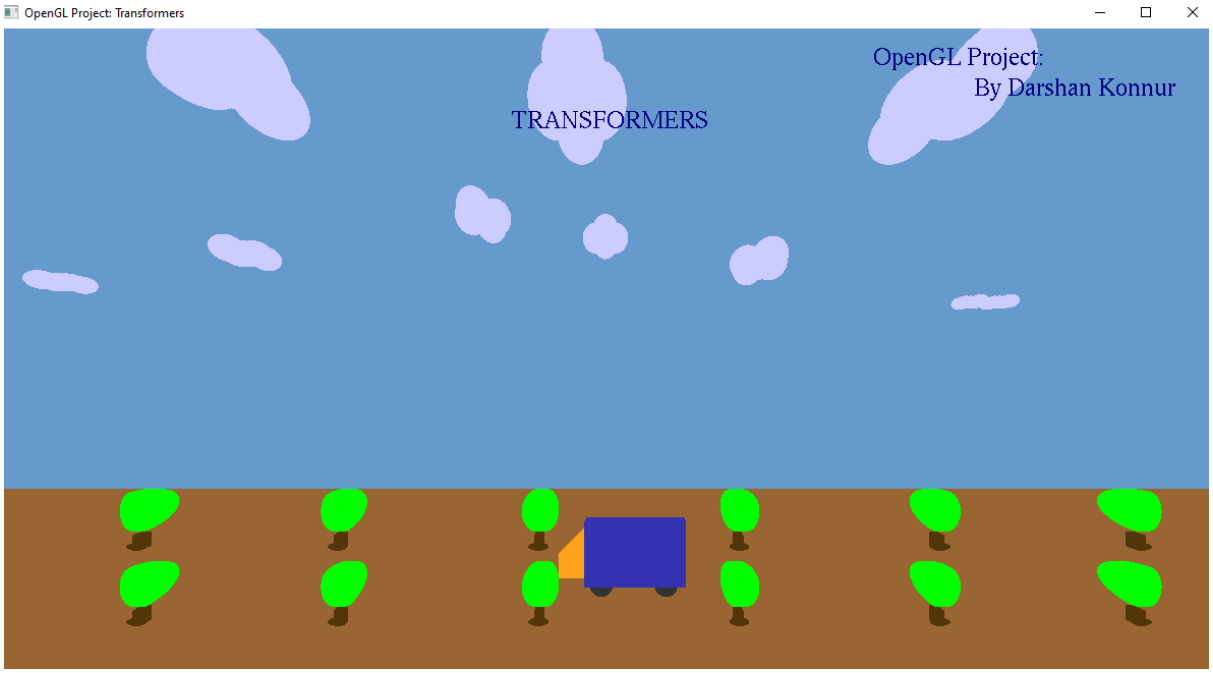

Fig 4. The vehicle moves along the way

5. CONCLUSION

OpenGL is the premier environment for developing portable, interactive 2D and 3D graphics applications. Since its introduction in 1992, OpenGL has become the industry's most widely used and supported 2D and 3D graphics application programming interface (API), bringing thousands of applications to a wide variety of computer platforms [4].

● **Industry standard:** An independent consortium, the OpenGL Architecture Review Board, guides the OpenGL specification. With broad industry support, OpenGL is the only truly open, vendor-neutral, multiplatform graphics standard.

● **Stable:** OpenGL implementations have been available for more than seven years on a wide variety of platforms. Additions to the specification are well controlled, and proposed updates are announced in time for developers to adopt changes. Backward compatibility requirements ensure that existing applications do not become obsolete.

● **Reliable and portable:** All OpenGL applications produce consistent visual display results on any OpenGL API-compliant hardware, regardless of operating system or windowing system.

● **Evolving:** Because of its thorough and forward-looking design, OpenGL allows new hardware innovations to be accessible through the API via the OpenGL extension mechanism. In this way, innovations appear in the API in a timely fashion, letting application developers and hardware vendors incorporate new features into their normal product release cycles.

● **Scalable:** OpenGL API-based applications can run on systems ranging from consumer electronics to PCs, workstations, and supercomputers. As a result, applications can scale to any class of machine that the developer chooses to target.

● **Easy to use:** OpenGL is well structured with an intuitive design and logical commands. Efficient OpenGL routines typically result in applications with fewer lines of code than those that make up programs generated using other graphics libraries or packages. In addition, OpenGL drivers encapsulate information about the underlying hardware, freeing the application developer from having to design for specific hardware features.

6. REFERENCES

[1] **OpenGL Projection Matrix[Online]**. Available: http://www.songho.ca/opengl/gl_projectionmatrix.html

[2] **INTERACTIVE COMPUTER GRAPHICS A TOP-DOWN APPROACH WITH SHADER-BASED OPENGL® 6th Edition[Online]**. Available: https://inspirit.net.in/books/academic/Interactive%20Computer%20Graphics.pdf

[3] **OpenGL About Page [Online]**. Available: https://www.opengl.org/about/

[4] Woo, Mason, Jackie Neider, Tom Davis, and Dave Shreiner. OpenGL programming guide: the official guide to learning OpenGL, version 1.2. Addison-Wesley Longman Publishing Co., Inc., 1999.