GRAPHICS PROGRAMMING

PROJECT REPORT

ON
VISUALIZING MULTIVARIATE DATA

TEAM MEMBERS

Ridham Saini(1910991376)

Davesh Mehta(1910991376)

UNDER THE GUIDANCE OF

Mr. Narinder Pal Singh

CONTENTS

INTRODUCTION

The Team has built a Data Visualization Tool, which is capable of reading large amounts of data from a CSV file and representing the given data in a multidimensional space dynamically.

The first main part is reading the CSV data file, which is being done by using the various C++ Data Management Tools.

The next step is storing each value separately so it can be manipulated accordingly if needed.

Then comes Data Normalization, large data values obtained from the step above which are hard to represent in a limited space, are normalized accordingly to make its representation easier.

After normalizing the values, the data can be used to render its Multidimensional representation.

READING DATA

Data given in the form of CSV file is being read by the code written in C++ using basic Data Management tools like fstream, ios::in etc. After reading the data, it is stored separately for better readability and easier manipulation.

DATA READING CODE

zecooEngine - SceneData.cpp

SceneData.cpp

zecooEngine (Global Scope)

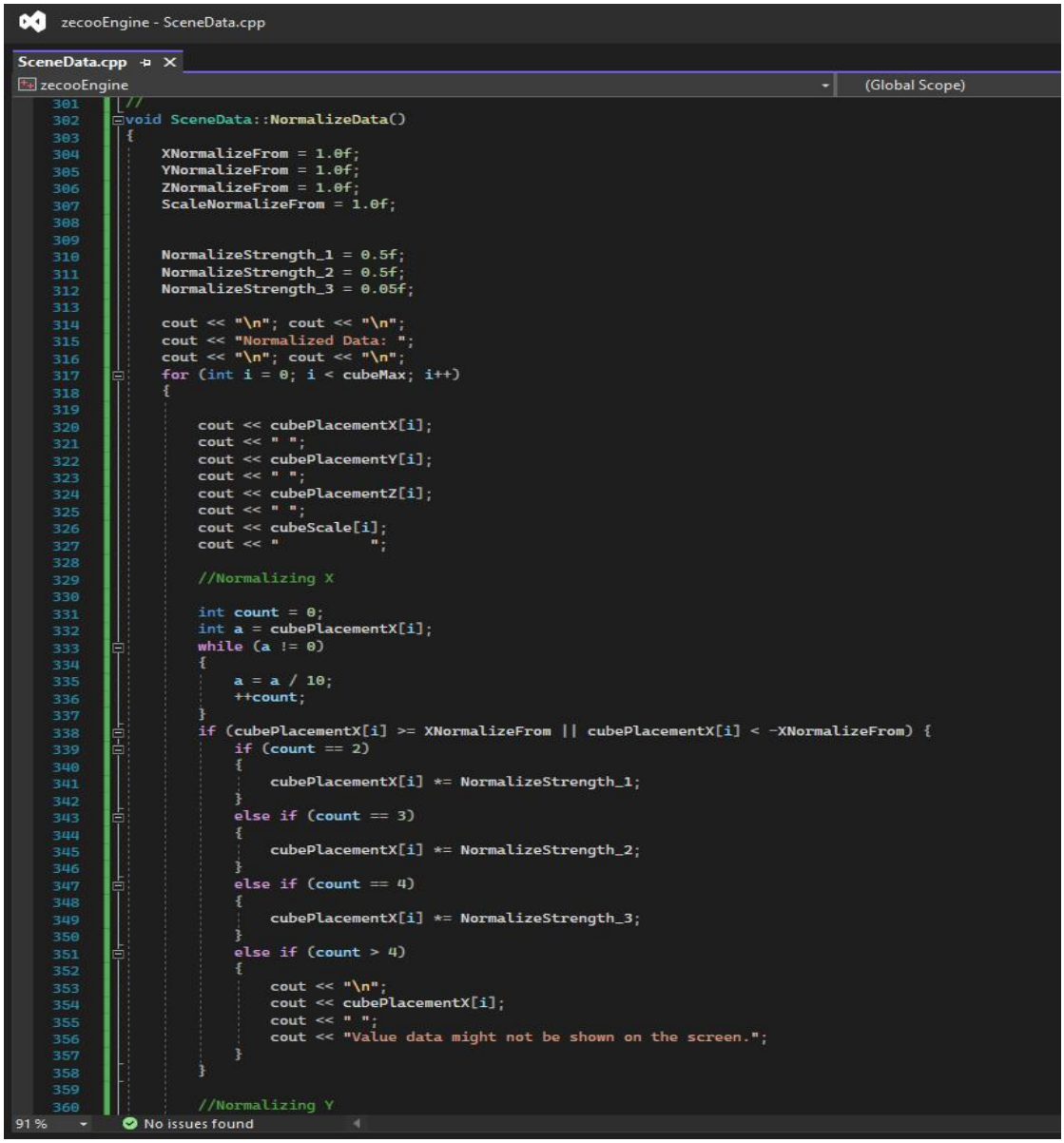
```
94
95      }
96
97      int SceneData::ReadFile()
98      {
99          fstream file(fname, ios::in);
100         if (file.is_open())
101         {
102             while (getline(file, line))
103             {
104                 row.clear();
105
106                 stringstream str(line);
107
108                 while (getline(str, word, ','))
109                     row.push_back(word);
110                 content.push_back(row);
111             }
112         }
113         else
114             cout << "Could not open the file\n";
115
116         cout << "\n"; cout << "\n";
117         cout << "Data Read: \n";
118         cout << "\n";
119
120         for (int i = 0; i < content.size(); i++)
121         {
122
123
124             for (int j = 0; j < content[i].size(); j++)
125             {
126                 if (j == 0) {
127                     cubePlacementX[cubeMax] = std::stof(content[i][j]);
128                     cout << content[i][j] << " ";
129                 }
130                 else if (j == 1) {
131                     cubePlacementY[cubeMax] = std::stof(content[i][j]);
132                     cout << content[i][j] << " ";
133                 }
134                 else if (j == 2) {
135                     cubePlacementZ[cubeMax] = std::stof(content[i][j]);
136                     cout << content[i][j] << " ";
137                 }
138                 else if (j == 3) {
139                     cubeScale[cubeMax] = std::stof(content[i][j]);
140                     cout << content[i][j] << " ";
141                 }
142                 else
143                     cout << ("Only Float Input Acepted");
144
145             }
146             cubeMax++;
147             cout << ("\n");
148         }
149         return 0;
150     }
151
152
153
```

91 % No issues found

NORMALIZING DATA

Larger Values of data read from the file are hard to represent on a limited viewport space so they need to be normalized accordingly and then passed to the next step of rendering it in a multi dimensional space.

DATA NORMALIZATION CODE:



DYNAMIC RENDERING

The Normalized data is used as input for mapping the graphical visualization of the data whose plotting density changes according to the amount data provided in the file. This was done using C++ and OpenGL.

RENDERING DATA

zecooEngine - SceneData.cpp

SceneData.cpp

zecooEngine (Global Scope)

```
61      cubeMaterial_2 = new Material(cubeShader3 , color3);
62      cubeMaterial_2->linkLight(dlight);
63      cubeMaterial_2->linkCamera(camera);
64
65
66
67
68
69      //DATA.................
70
71      ReadFile();
72      NormalizeData();
73      grid = new Grid(camera);
74
75
76
77
78      for (int i = 0; i < cubeMax; i++)
79      {
80          if(cubePlacementX[i] <= 30)
81              cube[i] = new Cube(cubeMaterial_0, NULL);
82
83          else if (cubePlacementX[i] >= 30 && cubePlacementX[i] <= 60 )
84              cube[i] = new Cube(cubeMaterial_1, NULL);
85
86          else if (cubePlacementX[i] >= 60)
87              cube[i] = new Cube(cubeMaterial_2, NULL);
88
89          cube[i]->transform->translate(glm::vec3(cubePlacementX[cubeNum], cubePlacementY[cubeNum], cubePlacementZ[cubeNum]));
90          cube[i]->transform->scale(glm::vec3(1 * cubeScale[cubeNum], 1 * cubeScale[cubeNum], 1 * cubeScale[cubeNum]));
91          cubeNum++;
92      }
93
94
95  }
96
97  int SceneData::ReadFile()
98  {
99      fstream file(fname, ios::in);
100     if (file.is_open())
101     {
```

133 % No issues found

CONCLUSION

This project has helped us better understand the applications of Data Visualization and its importance in day to day life along with giving us a chance to work on some of the important skills like team work and coordination.

Many hurdles were faced during the development phase but to overcome those was a lesson we learned on how the team could have worked more efficiently. It has provided us with the invaluable knowledge about OpenGL, C++ and many other resources used to complete the project.

Moreover, we thank the faculty provided to us for guiding us through every step and making us capable enough to do this project on our own.

The Project could be hosted on some online websites and clients where the user just need to provide the data in CSV format and the data will be shown on the user's screen in no time.