MSIM 441/541 & ECE 406/506
Computer Graphics & Visualization

Programming Assignment One: Input Analysis Using Histograms

Thomas J Laverghetta
Tlave002@odu.edu
757-620-7607
Virtual
MSIM411
10/22/20

Introduction

The objective of this assignment is to create a visualization program that will visualize data for histogram analysis. Histogram analysis is the process of comparing a theoretical distribution to a histogram so that a theorical distribution can be used instead of data (gives larger range of values to use than purely data would). To do this automatically, I created a program that accepts data, produces histogram, allows users via a user-interface (UI) to choose a theorical distribution to compare, visualize histogram and theorical distribution, and lastly, the ability to modify the theorical distribution parameters so it better fits the histogram.

Program Design

The design documentation for the programming assignment was made using Doxygen. Doxygen is a tool for generating documentation from annotated C++ sources (classes, enums, variables, functions, etc.), and programming languages (C, Objective-C, Java, etc.). To make documentation using your source code with Doxygen you comment your source code with specified markers and description of the code. The markers give specific meaning to the comments (such as identifying the code is a class, enum, variable, function, etc.). The markers and other documentation on Doxygen can be found on their website (https://www.doxygen.nl/manual/docblocks.html). Using these markers, I commented my code and ran Doxygen wizard tool. This generated HTML design documentation (Assignment_1_Laverghetta_Thomas.chm). The HTML design documentation illustrates the functions and variables used. The functions and variables will all have short briefings explaining their use within the program and any corresponding dependencies. Doxygen will also illustrate how the functions are connected and where dependencies are.

Results

A user-interface (UI) menu allows users to choose what data files to input to produce histogram, what theorical distribution to augment over histogram to allow for visual comparison, what number of bins for histogram, and what parameter step size. With these inputs, the user can interact with the program to meet his/her data analysis requirements.

In this section, I will describe what tasks were done to achieve UI response requirements. In total there are five-tasks: reading input data, probability density histogram, probability density functions, input analysis, and visual perception.

Reading Input Data

The program reads data from input files to generate histogram. After a user has chosen what data file to run using user-interface menu (normal.dat, expo.dat, 5.dat, or 15.dat), the program will get the data file and start reading and parsing it. The data files are ASCII formatted. The first line of data files contains the number of data points in the file and each subsequent line contains one data point. Using this knowledge, the program will read number of points, create allocate array to hold data, and save the data to array. While saving the data, it will determine the maximum and minimum data points. The maximum (max) and minimum (min) will be used to find bin step sizes for the histogram.

Probability Density Histogram

After reading in data points, the program will start generating histogram values. The values it will be generating are bin step-size, location of each bin endpoint, probabilities for each bin, and density of probability maximum. The values are generated using data inputted and parsed and number of intervals (bins) chosen by user (30-, 40-, or 50-bins).

Bin step-size is calculated by taking the difference of max and min datapoints, from data file, divided by number of intervals. i.e., the distance of datapoints then segmenting by number of intervals.

Location of each bin endpoint is calculated by additively iterating step-by-step to each bin and getting its endpoint. In pseudocode:

Endpoints (step-size, number of intervals)
1. create endpoint array with number of intervals + 1 elements
2. endpoint [0] = minimum datapoint
3. for j in range(number of intervals):
*4. endpoints [j] = minimum + j * step-size*
5. end for
6. return endpoints

The probability of each bin is calculated by taking the counting points within a bin then dividing by total number of points and bin step size. This will produce the density of probability of that bin (area of bin is the probability of points occurring). The density of probability maximum can be found be finding the maximum value produced in this process.

Probability Density Function

The user can choose between two different probability density functions (distribution functions), normal and exponential distributions. The normal and exponential distribution functions is shown in equation (eqn) 1 and 2, respectively. These equations are used to compute the theorical curves given respective to their distribution parameters (μ, σ, β) and number of curve points (100 for this program). The distribution parameters can be altered during program execution using keyboard input. The parameter μ decrease and increase with Left and Right arrow keys, respectively, and parameters $\sigma$and β increase and decrease with Up and Down keys, respectively. All increasing and decreasing are the same amount, parameter step-size. Parameter step-size is chosen by user via UI (0.01-, 0.02-, or 0.05-parameter step-sizes). Once the theorical curves have been computed, they will be augmented over histogram (illustrated in Figure 1). The visualization will also display the parameter values for distribution.

$$Norm(x|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \qquad eqn.1$$

$$Expo(x|\beta) = f(x) = \begin{cases} \frac{1}{\beta} e^{-\frac{x}{\beta}}, & x \geq 0 \\ 0, & x < 0 \end{cases} \qquad eqn.2$$

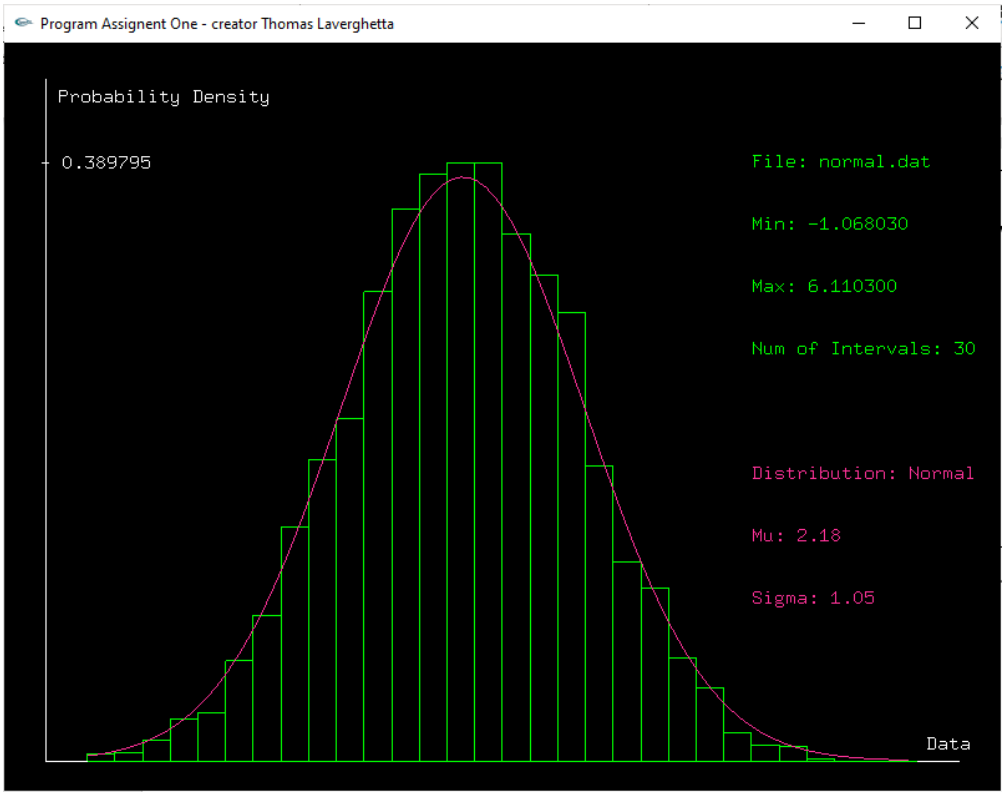

Figure 1. Theorical Curve Augmented on Histogram

Input Analysis

To test my program, I used two data files with unknown theorical distributions. The data files used were data files 5.dat and 15.dat (corresponding to my first and last initials – T, L). These data files were inputted into the system and I used visual comparison method to determine the best fitting theorical distribution. Figure 2 and Figure 3 show outputted histogram and the best theorical distribution ($Norm(x|\mu = 1.00, \sigma = 0.05)$ for Figure 2 and $Expo(x|\beta = 4.05)$ for Figure 3).

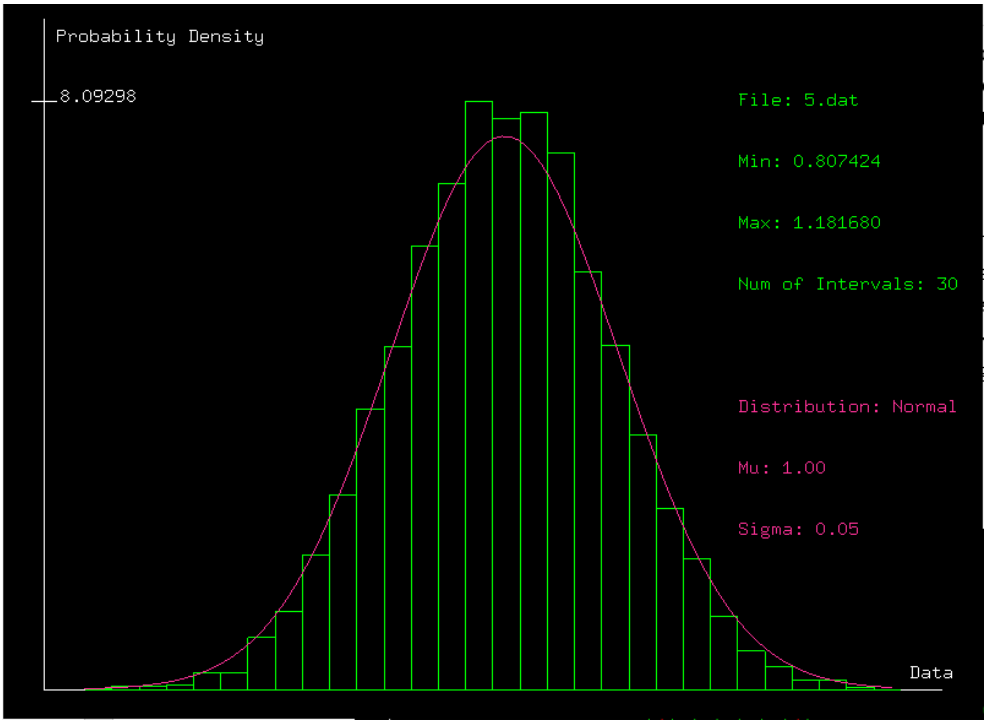

Figure 2. Data file 5 (5.dat) with best fitness ($Norm(x|\mu = 1.00, \sigma = 0.05)$)

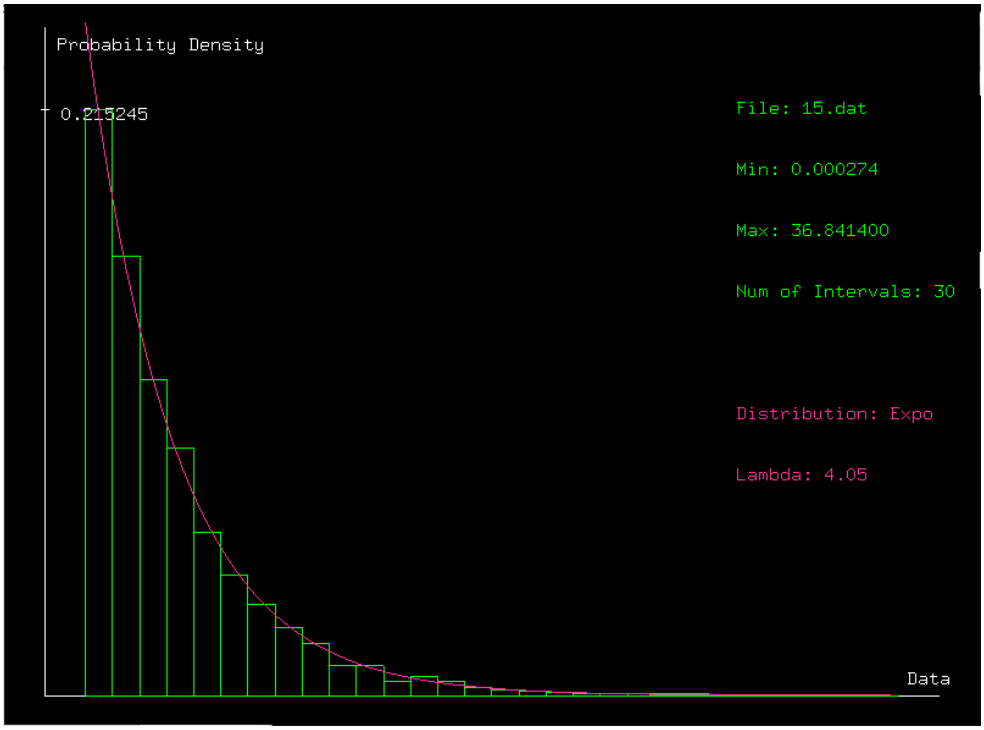

Figure 3. Data file 15 (15.dat) with best fitness ($Expo(x|\beta = 4.05)$)

Visual Perception

For the last task, I tried different color combinations to choose the best combination for displaying information. For this task, I tried four different color combinations as shown in Figure 4, Figure 5, Figure 6, and Figure 7. In the first figure, I used a black background with green to display data information (histogram, file info, min and max, and number of interval used), red to display distribution information (theorical curve, distribution name, and distribution parameters used), and white for the axis information (xy-axis, density of probability max, and title "Probability Density"). In the second figure, I used black background, green to display data information, tan to display distribution information, and yellow to display axis information. For the third figure, I used grey background with green data information, red distribution information, and pink axis information. Lastly, the fourth figure, I used black background, green data information, pink distribution information, and white axis information. Based on these four figures, the best one is Figure 7.

The reasoning behind Figure 7. Figure 5 and Figure 6 are the worst combination with Figure 5 looking aesthetically unpleasing to me (i.e., yellow and tan combination) and Figure 6 being barely readable due to the bright grey background and other bright colors used. Therefore, leaving the other two figures, Figure 4 and Figure 7. Figure 4 is a good choice, however, I found the dark red to blend with the black background unlike the other two colors (green and white) which are brighter (making them clearer). Hence, I changed the shade of red to pink producing Figure 7. The pink pops out from the dark background allowing for a user to easily see the difference.

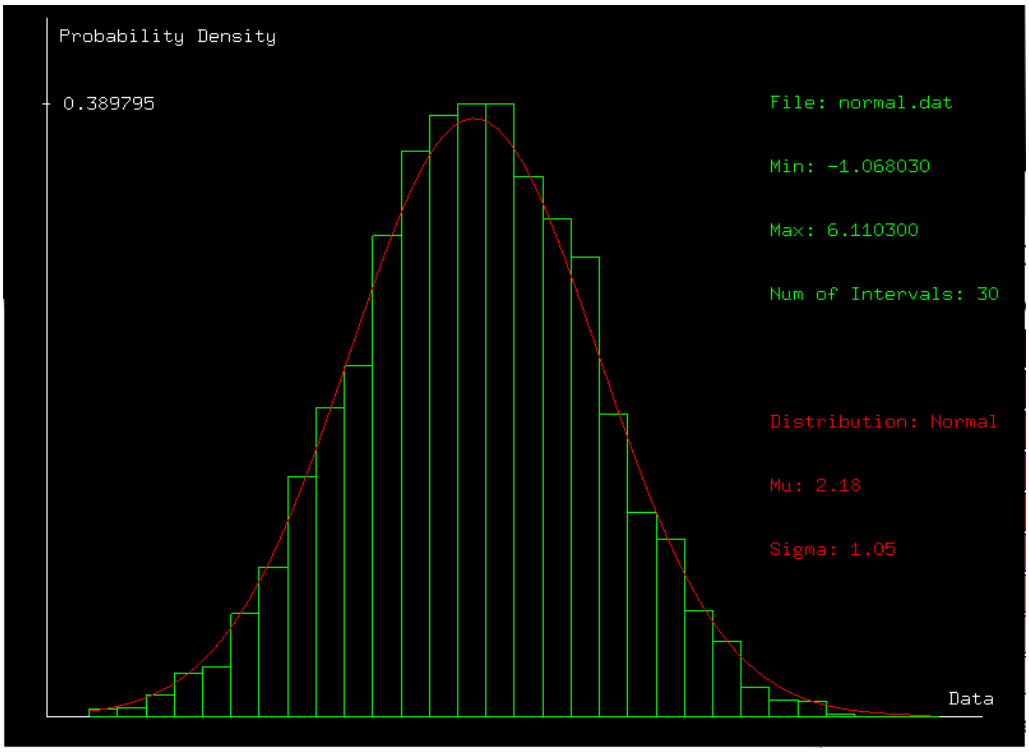

Figure 4. Black Background, Green Data Info, White Axis Info, and Red Distribution Info

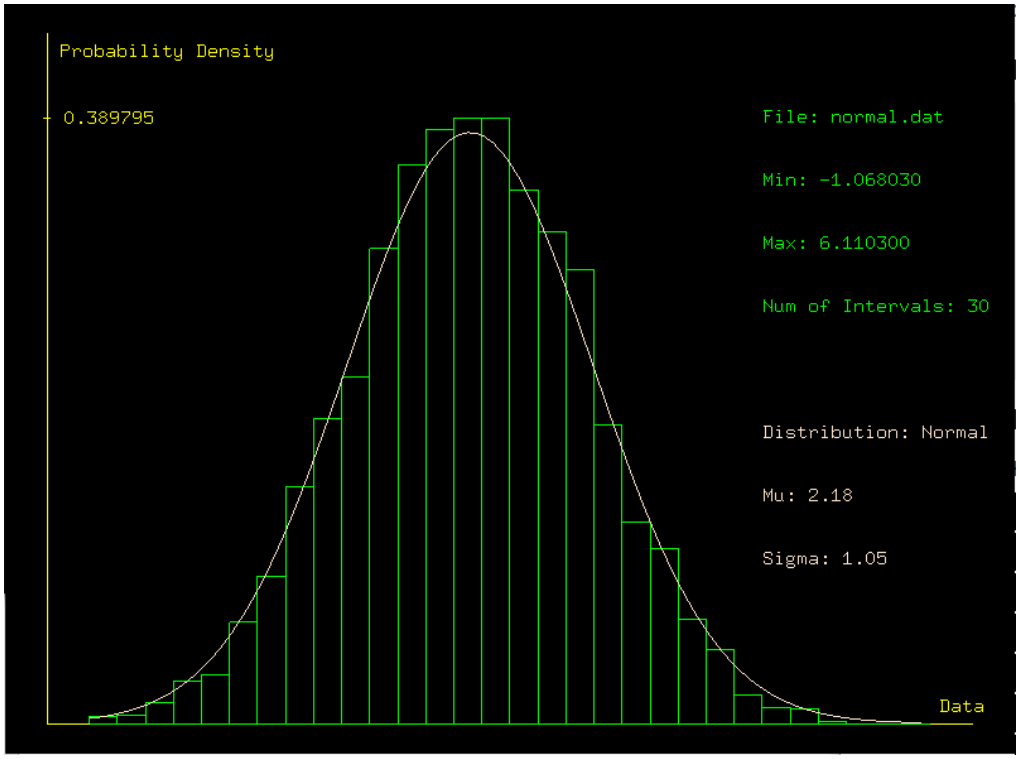

Figure 5. Black Background, Green Data Info, Yellow Axis Info, and Tan Distribution Info

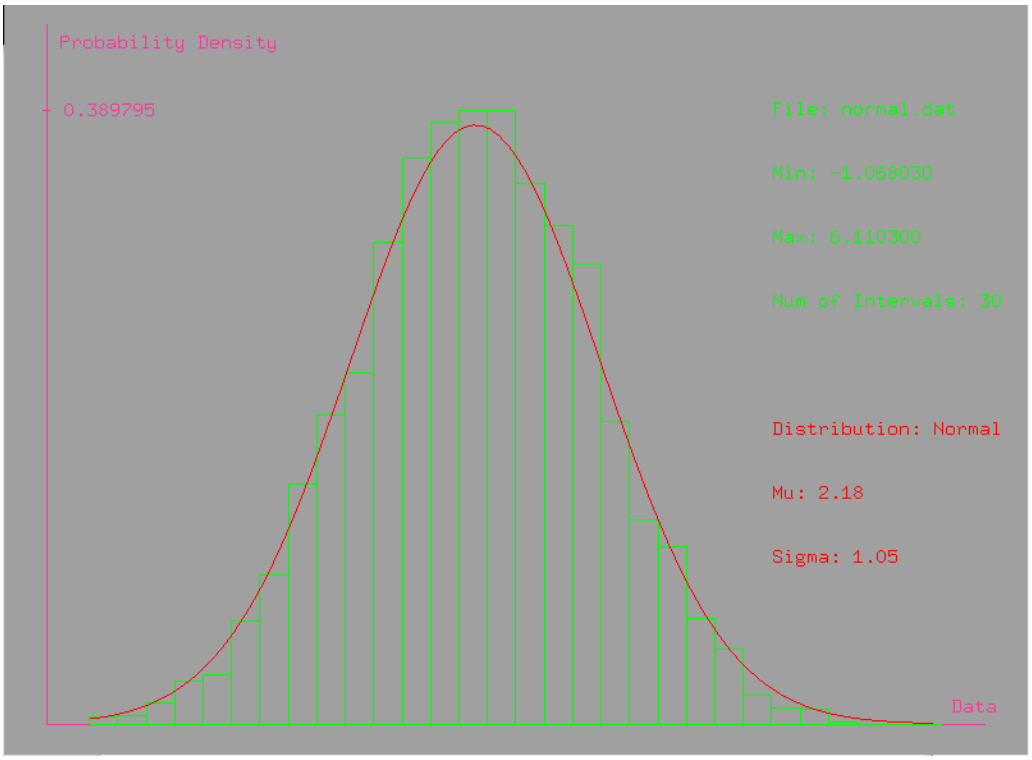

Figure 6. Grey background, Green Data Info Font, Pink Axis Info, and Red Distribution Info

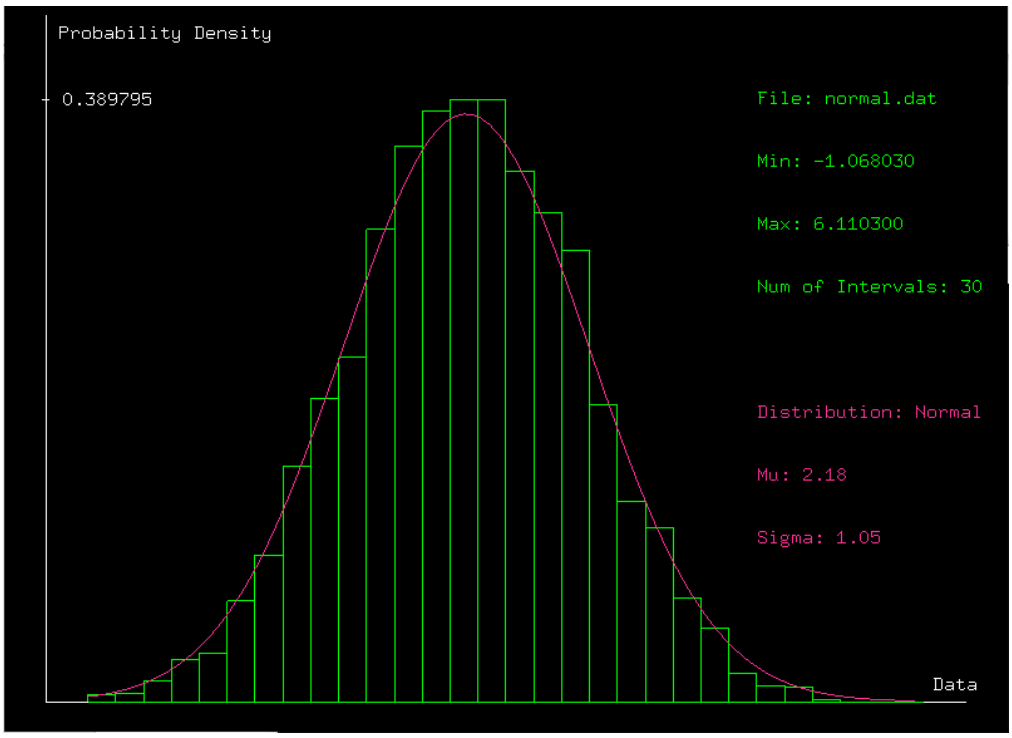

Figure 7. Black Background, Green Data info, Pink Distribution Info, and White Axis Info

Conclusion and Discussion

In this section, I will discuss my accomplishments, learning outcomes, and difficulties and how I can do better for this assignment.

Accomplishments

I felt really pleased with the outcome of this assignment. The assignment passed all the requirements set before me by Dr. Shen and the course. The program can read data files, output histogram with different number of bins, and able to produce modifiable normal and exponential distributions via user keyboard input which allows users to conduct histogram analysis.

Learning Outcomes

I feel I have better understanding how OpenGL works than before I did this assignment. Before I did this assignment, I had a really hard time understanding how everything worked in OpenGL, especially because there were no variables associated with objects created unlike all previous programming classes I have had. However, after spending hours doing this assignment, I started to understand that in OpenGL the objective is to create scene frames and associate variables that allow to create new scenes every frame.

Difficulties

I had no difficulties other than the learning curve of OpenGL frames.