UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE INFORMÁTICA
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

CARLOS MAGNO DA SILVA – Matrícula: 20160143331

INTRODUÇÃO A COMPUTAÇÃO GRÁFICA

Trabalho 1 - Algoritmos de Rasterização Utilizados em Computação Gráfica

JOÃO PESSOA
2018

UFPB - Universidade Federal da Paraíba
Centro de Informática – CI

Disciplina: Introdução a Computação Grafica – ICG

Professor: Christian Azambuja Pagot

Aluno(s): Carlos Magno da Silva – **Matrícula:** 20160143331

Cursos: Ciênca da Computação

Semestre: 2017.2

Algoritmos de Rasterização Utilizados em Computação Gráfica

1. INTRODUÇÃO

O presente projeto foi solicitado pelo professor da disciplina e consiste na primeira atividade da disciplina de ICG, no presente periodo, tendo como finalidade familiarizar os alunos com os algoritmos de rasterização utilizados em computação gráfica.

2. ATIVIDADE

A atividade solicitada consiste em implementar algoritmos para a rasterização de pontos e linhas. O desenho de Triângulos, que também foi solicitado, foram desenhados através da rasterização das linhas que compõem suas arestas.

Como a rasterização destas primitivas são feitas através da escrita direta na memória de vídeo e considerando que os sistemas operacionais atuais protegem a memória quanto ao acesso direto, utilizou-se de um ***framework*** fornecido pelo professor, para simular o acesso à memória de vídeo.

3. DESENVOLVIMENTO

Neste trabalho foram desenvolvidas 03(três) funções cujas funcionalidades estão discriminadas abaixo:

1-) PutPixel: Função que rasteriza um ponto na memória de vídeo, recebendo como parâmetros a posição do pixel na tela (x,y) e sua cor (RGBA);

2-) DrawLine: Função que rasteriza uma linha na tela, recebendo como parâmetros os seus vértices (inicial e final, representados respectivamente pelas tuplas ***(x0,y0)*** e ***(x1,y1)***), e as cores (no **formato RGBA**) de cada vértice. As cores dos ***pixels*** ao longo da linha rasterizada devem ser obtidas através da ***interpolação linear*** das cores dos vértices. O algoritmo de rasterização a ser implementado deve ser o **Algoritmo de Bresenham.**

3-) DrawTriangle: Função que desenha as arestas de um triângulo na tela, recebendo como parâmetros as posições dos 03(três) vértices (xa,ya), (xb,yb) e (xc,yc) bem como as **cores (RGBA)** de cada um dos vértices. As cores dos ***pixels*** das arestas do triângulo devem ser obtidas através da ***interpolação linear*** das cores de seus vértices. **Não é necessário o preenchimento do triângulo.**

As funções, conforme solicitado, foram escritas na **IDE CodeBlocks**, utilizando a **Linguagem C/C++** com as **bibliotecas GLUT** e o **OpenGL**.

4. TELAS DE RESULTADOS DO PROCESSAMENTO

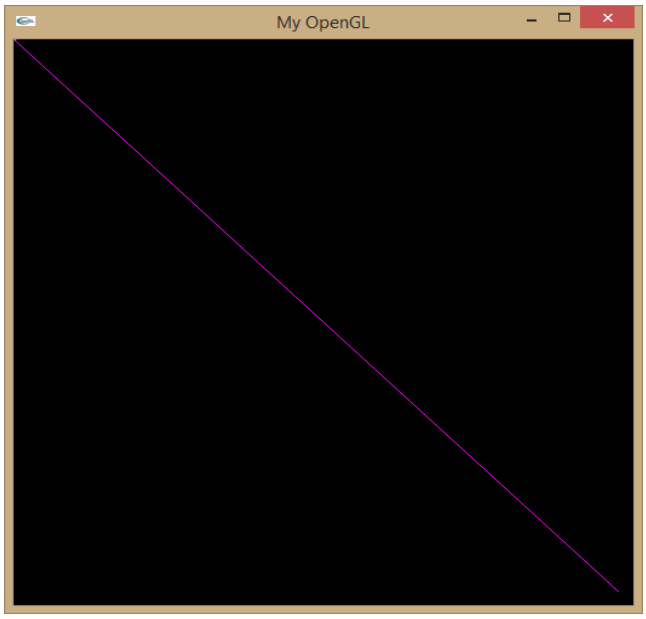

Figura 01 – Rasterização de Linha

```
void DesenhaLinha (void) {

    for (unsigned int i=0; i<250; i++) {

        FBptr[4*i + 4*i*IMAGE_WIDTH + 0]  = 255;

        FBptr[4*i + 4*i*IMAGE_WIDTH + 1]  = 0;

        FBptr[4*i + 4*i*IMAGE_WIDTH + 2]  = 255;

        FBptr[4*i + 4*i*IMAGE_WIDTH + 3]  = 255;

    }

}
```

Figura 02 – Trecho de Codificação da Figura 01

5. CONCLUSÃO

Principais Dificuldades

1 - Entendimento, do ponto de vista matemático, do funcionamento do Algorítmo de Bresenham, sendo necessário relembrar conceitos matemáticos básicos que foram estudados em períodos anteriores da graduação, como equação da reta, coeficiente angular, conceitos de geometria, etc.

2 - Algumas dificuldades surgiram durante a instalação e configuração das bibliotecas do opengl e da glut na **IDE do CodeBlocks**, que foi utilizada neste projeto;

3 - Entendimento do ***framework*** encaminhado, onde o projeto deveria funcionar, simulando a **memória de vídeo**, do computador, um vez que, os sistemas operacionais atuais protegem a memória quanto ao acesso direto.

Possíveis Melhoras

Utilizando-se de conceitos de estruturas de dados, principalmente os que se referem ao uso de ponteiros, de alocação dinâmica de mémoria; e tendo-se um bom domínio matemático, principalmente na área de matrizes, e conhecendo-se, além do algorítmo de Bresenham, outros algorítmos que possam ser usados, na rasterização de de primtivas de imagens, pode-se melhorar o desempenho de aplicações em Computação Gráfica..

Referências

1 – Tutorial de Utilização de OpenGl – Marcionílio Barbosa Sobrinho – Belo Horizonte – MG – 2003

2 – Introdução à OpenGL - Professora Isabel Harb Manssour – https://www.inf.pucrs.br/~manssour/OpenGL/Desenhando.html. Acesso em: 18/03/2018