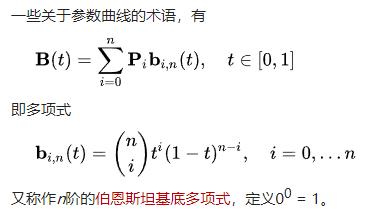
计算机图形学 Homework 9 - Bezier Curve

16340299 赵博然

使用这个公式描绘贝塞尔曲线. 节选自中文维基百科.



部分算法.

全局变量如下.

// 数组最大长度

#define MAX\_SIZE 1024

// 最大点数, 点数过多会溢出, 即使用了unsigned long long数据类型, 懒得搞高精度

#define MAX\_POINT 20

const unsigned int window\_width = 800;

const unsigned int window\_height = 600;

const float point\_size = 8.0f;

// 步长

const float step = 0.01f;

float currentX;

float currentY;

float vertices[MAX\_SIZE];

float bonus\_vertices[MAX\_SIZE];

float q\_vertices[4];

// 当前点数

int number = 0;

// 时间, 放在while (!glfwWindowShouldClose(window))循环中累加, 用来动态生成曲线, 其实就是曲线函数的自变量

float TIME = 0;

#define MAX\_TIME 1.000

迭代求组合数.

unsigned long long binomial\_coefficient(const int &n, const int &k) {

if (n > k && k >= 0) {

unsigned long long numerator = 1;

unsigned long long denominator = 1;

for (int i = 0; i < n - k; i++) {

numerator \*= n - i;

denominator \*= i + 1;

}

return unsigned long long(numerator / denominator);

}

else {

return 1;

}

}

鼠标监听器.

void mouse\_callback(GLFWwindow\* window, int button, int action, int) {

if (action == GLFW\_PRESS) {

switch (button) {

case GLFW\_MOUSE\_BUTTON\_LEFT:

TIME = 0;

if (number < MAX\_POINT) {

vertices[number \* 2] = currentX / (float)(window\_width / 2);

vertices[number \* 2 + 1] = currentY / (float)(window\_height / 2);

number++;

}

break;

case GLFW\_MOUSE\_BUTTON\_RIGHT:

TIME = 0;

if (number > 0) {

number--;

}

break;

default:

break;

}

}

}

生成贝塞尔曲线.

q\_vertices[0] = vertices[0];

q\_vertices[1] = vertices[1];

// 生成曲线

for (float t = 0.0f; t < 1.0f; t += step) {

for (int i = 0; i < number; i++) {

q\_vertices[2] += vertices[i \* 2] \* binomial\_coefficient(number - 1, i) \* pow(t, i) \* pow((1 - t), number - 1 - i);

q\_vertices[3] += vertices[i \* 2 + 1] \* binomial\_coefficient(number - 1, i) \* pow(t, i) \* pow((1 - t), number - 1 - i);

}

unsigned int qVAO, qVBO;

glGenBuffers(1, &qVBO);

glBindBuffer(GL\_ARRAY\_BUFFER, qVBO);

glBufferData(GL\_ARRAY\_BUFFER, sizeof(q\_vertices), q\_vertices, GL\_STATIC\_DRAW);

glGenVertexArrays(1, &qVAO);

glBindVertexArray(qVAO);

glVertexAttribPointer(0, 2, GL\_FLOAT, GL\_FALSE, 2 \* sizeof(float), (void\*)0);

glEnableVertexAttribArray(0);

glBindBuffer(GL\_ARRAY\_BUFFER, 0);

glBindVertexArray(0);

if (number > 0) {

glBindVertexArray(qVAO);

glDrawArrays(GL\_LINE\_STRIP, 0, 2);

}

glDeleteVertexArrays(1, &qVAO);

glDeleteBuffers(1, &qVBO);

q\_vertices[0] = q\_vertices[2];

q\_vertices[1] = q\_vertices[3];

q\_vertices[2] = 0.0f;

q\_vertices[3] = 0.0f;

}

bonus部分.

// bonus部分, 显示曲线的生成过程

for (int i = 0; i < MAX\_SIZE; i++) {

bonus\_vertices[i] = vertices[i];

}

for (int i = number; i > 2; i--) {

for (int j = 0; j < i - 1; j++) {

bonus\_vertices[j \* 2] = bonus\_vertices[j \* 2] \* (1 - TIME) + bonus\_vertices[(j + 1) \* 2] \* TIME;

bonus\_vertices[j \* 2 + 1] = bonus\_vertices[j \* 2 + 1] \* (1 - TIME) + bonus\_vertices[(j + 1) \* 2 + 1] \* TIME;

}

unsigned int bonus\_VAO, bonus\_VBO;

glGenBuffers(1, &bonus\_VBO);

glBindBuffer(GL\_ARRAY\_BUFFER, bonus\_VBO);

glBufferData(GL\_ARRAY\_BUFFER, sizeof(bonus\_vertices), bonus\_vertices, GL\_STATIC\_DRAW);

glGenVertexArrays(1, &bonus\_VAO);

glBindVertexArray(bonus\_VAO);

glVertexAttribPointer(0, 2, GL\_FLOAT, GL\_FALSE, 2 \* sizeof(float), (void\*)0);

glEnableVertexAttribArray(0);

glBindBuffer(GL\_ARRAY\_BUFFER, 0);

glBindVertexArray(0);

glBindVertexArray(bonus\_VAO);

glDrawArrays(GL\_LINE\_STRIP, 0, i);

glDeleteVertexArrays(1, &bonus\_VAO);

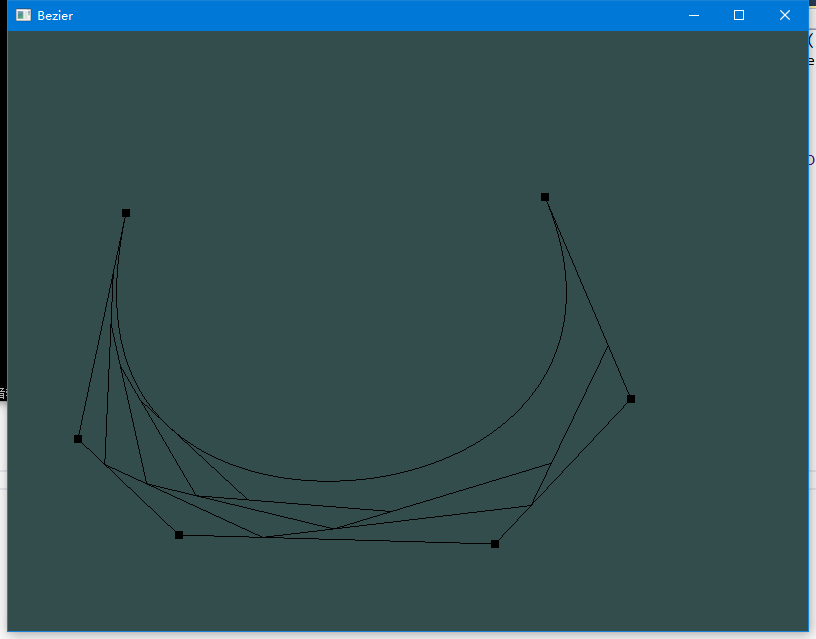
glDeleteBuffers(1, &bonus\_VBO);

}

TIME = TIME <= MAX\_TIME ? TIME + step / 4.0 : 0;

代码详见main.cpp.

程序截图.



演示视频见video.mp4.

源程序见program.exe.