ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ МОЛДОВЫ
ФАКУЛЬТЕТ ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ, ИНФОРМАТИКИ И МИКРОЭЛЕКТРОНИКИ
КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

ОТЧЕТ

По лабораторной работе №3
по Програмированию для Windows
Тема: Сплайны Безье (GDI)

Выполнила: ст. гр. TI-155 Зверкова К.

Проверил: Скроб С.

Кишинев 2017

1. Цель лабораторной работы

Изучение методов построения сплайнов Безье.

2. Теоретические понятия

Сплайны Безье — одни из самых популярных в программировании компьютерной графики. Это совсем недавнее усовершенствование в арсенале графических средств, доступных на уровне операционной системы, и оно пришло с неожиданной стороны. В шестидесятых годах автомобильная компания Renault переходила от ручного проектирования кузовов автомобилей (что требовало много глины) к компьютерному. Требовался математический аппарат, и Пьер Безье предложил набор формул, оказавшихся очень полезными в этой работе. С тех пор двумерная форма сплайна Безье показала себя как самая удобная кривая (после прямых линий и эллипсов) в компьютерной графике. Например, в языке PostScript сплайны Безье используются для всех кривых — эллиптические линии аппроксимируются из сплайнов Безье. Кривые Безье также используются для описания контуров символов различных шрифтов языка PostScript. (TrueType используют более простые и быстрые формы сплайнов.)

Простой двумерный сплайн Безье определяется четырьмя точками — двумя конечными и двумя контрольными. Концы кривой привязаны к двум конечным точкам. Контрольные точки выступают в роли магнитов для оттягивания кривой от прямой, соединяющей две крайние точки.

Сплайны Безье считаются полезными для компьютерного проектирования благодаря следующим характеристикам:

Во-первых, немного попрактиковавшись, вы можете легко манипулировать кривой для получения нужной формы.

Во-вторых, сплайны Безье очень легко управляются. В некоторых формах сплайнов кривая не может быть проведена через все определяющие точки. Сплайны Безье всегда "привязаны" к двум конечным точкам. (Это первое допущение, которое берет начало в формулах Безье.) Кроме того, существуют сплайны с бесконечными кривыми, которые имеют свои особенности. В компьютерном проектировании редко встречаются подобные типы сплайнов. Как правило, кривые Безье всегда ограничены четырехэлементной ломаной, называемой "выпуклым корпусом" (convex hull), которая получается соединением конечных и контрольных точек.

В-третьих, в сплайнах Безье существует связь между конечными и контрольными точками. Кривая всегда является касательной к прямой, соединяющей начальную точку и первую контрольную точку, и направленной в ту же сторону. Кривая также является касательной к прямой, соединяющей конечную точку и вторую контрольную точку, и направленной в ту же сторону. Это еще два допущения на основе формул Безье.

В-четвертых, сплайны Безье в основном хорошо смотрятся. Понятно, что это критерий субъективный, но так считают многие.

Раньше сплайны Безье создавались с помощью функции Polyline(). Вам следовало также знатьпараметрические уравнения, описывающие сплайны Безье. Начальная точка (x0,y0), конечная точка (x3, y3). Две контрольные точки (x1, y1) и (x2, y2). Кривая, отображаемая в интервале t от 0 до 1 описывалась так:

x(t) = (1—t)^3*x0 + 3t(1—t)^2*x1 + 3t^2(1—t)*x2 +t^3*x3

y(t) = (1—t)^3*x0 + 3t(1—t)^2*x1 + 3t^2(1—t)*x2 +t^3*x3

Теперь, чтобы нарисовать одну или более связанных сплайнов Безье, используются:

PolyBezier(hdc, pt, iCount) или PolyBezierTo(hdc, pt, iCount);

В обоих случаях pt — массив структур типа POINT. В функции PolyBezier первые четыре точки идут в таком порядке: начальная точка, первая контрольная точка, вторая контрольная точка, конечная точка кривой Безье. Каждая следующая кривая Безье требует три новых точки, поскольку начальная точка следующей кривой есть конечная точка предыдущей и т. д. Параметр iCount всегда равен единице плюс умноженному на три числу связанных кривых, которые вы хотите отобразить.

3. Задание

Построить кривую Безье, используя стандартные функции GDI и рядом построить кривую Безье по формулам.

4. Листинг программы

```
#include <windows.h>

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);
```

```
int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, PSTR szCmdLine, int
iCmdShow) {

   static char szAppName[] = "Bezier";

   HWND hwnd;

   MSG msg;

   WNDCLASSEX wndclass;

   wndclass.cbSize = sizeof(wndclass);

   wndclass.style = CS_HREDRAW | CS_VREDRAW;

   wndclass.lpfnWndProc = WndProc;

   wndclass.cbClsExtra = 0;

   wndclass.cbWndExtra = 0;

   wndclass.hInstance = hInstance;

   wndclass.hIcon = LoadIcon(NULL, IDI_APPLICATION);

   wndclass.hCursor = LoadCursor(NULL, IDC_ARROW);

   wndclass.hbrBackground = (HBRUSH)GetStockObject(WHITE_BRUSH);

   wndclass.lpszMenuName = NULL;

   wndclass.lpszClassName = szAppName;

   wndclass.hIconSm = LoadIcon(NULL, IDI_APPLICATION);

   RegisterClassEx(&wndclass);

   hwnd = CreateWindow( szAppName,

      "Lab #3 Program",

      WS_OVERLAPPEDWINDOW,

      CW_USEDEFAULT,

      CW_USEDEFAULT,

      CW_USEDEFAULT,

      CW_USEDEFAULT,

      NULL,

      NULL,

      hInstance,

      NULL );

   ShowWindow(hwnd, iCmdShow);

   UpdateWindow(hwnd);

   while (GetMessage(&msg, NULL, 0, 0)) {
```

```
      TranslateMessage(&msg);

      DispatchMessage(&msg);
   }

   return msg.wParam;
}

LRESULT CALLBACK WndProc(HWND hwnd, UINT iMsg, WPARAM wParam, LPARAM lParam) {

   static POINT pt[4];

   HDC hdc;

   int cxClient, cyClient;

   PAINTSTRUCT ps;

   switch (iMsg) {

   case WM_PAINT:

      hdc = BeginPaint(hwnd, &ps);

      HPEN hpen1;

      hpen1 = CreatePen(PS_SOLID, 2, RGB(255, 0, 102));

      SelectObject(hdc, hpen1);

      pt[0].x = 100;
      pt[0].y = 200;
      pt[1].x = 200;
      pt[1].y = 100;
      pt[2].x = 300;
      pt[2].y = 300;
      pt[3].x = 400;
      pt[3].y = 200;

      PolyBezier(hdc, pt, 4);


      hpen1 = CreatePen(PS_SOLID, 2, RGB(0, 0, 102));

      SelectObject(hdc, hpen1);

      MoveToEx(hdc, 200, 300, NULL);

      int x, y;

      pt[0].x = 200;
      pt[0].y = 300;
      pt[1].x = 300;
      pt[1].y = 200;
      pt[2].x = 400;
      pt[2].y = 400;
      pt[3].x = 500;
      pt[3].y = 300;

      for (double t = 0; t <= 1; t += 0.05) {

         x = (1 - t) * (1 - t) *(1 - t) * pt[0].x + 3 * (1 - t)* (1 - t) * t * pt[1].x + 3 * t
* t * (1 - t)*pt[2].x + t*t*t*pt[3].x;
```

```
        y = (1 - t) * (1 - t) *(1 - t) * pt[0].y + 3 * (1 - t)* (1 - t) * t * pt[1].y + 3 * t
* t * (1 - t)*pt[2].y + t*t*t*pt[3].y;

        LineTo(hdc, x, y);
      }

      EndPaint(hwnd, &ps);

      return 0;

   case WM_DESTROY:

      PostQuitMessage(0);

      return 0;
   }

   return DefWindowProc(hwnd, iMsg, wParam, lParam);
    }
```

5. Результат работы прогрммы

Вывод: В данной лабораторной работе были изучены методы построения сплайнов Безье. Была реализована программа, результатом работы которой являются два построенных сплайна Безье – первый, используя встроенную функцию PolyBozier() (малинового цвета) и второй, по параметрическим уравнениям описывающим сплайны Безье (синего цвета).