МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Московский государственный технологический университет «СТАНКИН»

(ФГБОУ ВО «МГТУ «СТАНКИН»)

Институт
информационных технологий

Кафедра
Инженерной Графики

Основная образовательная программа 09.03.02
«Информационные системы и технологии»

Отчет по дисциплине «Компьютерная геометрия и графика»

по лабораторной работе № 5

Студент

группы ИДБ-21-06

Музафаров.К.Р

Преподаватель

к.т.н. , доц. Разумовский А.И

Москва, 2022 г.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

В заключительной лабораторной работе мы будем рассматривать: вращение объемных фигур, построения сплайнов, виды сплайнов.

Чтобы построить реалистичный трехмерный объект необходимо использовать перспективное преобразование координат. Для придания объекту большего реализма важно также настроить соотнесение его координат с источником света. Если объект также должен быть способен к вращению, то для этого необходимо соответствующим образом обрабатывать сообщения мыши WM_MOUSEMOVE и WM_MOUSEWHEEL.

ЗАДАНИЕ 1

Изобразить на экране вращающийся при помощи мыши куб с удалёнными гранями и закраской.

Код программы:

```
#include<Windows.h>
#include<tchar.h>
#include<math.h>
LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);
TCHAR WinName[] = _T("MainFrame");
int APIENTRY WinMain(HINSTANCE This, HINSTANCE Prev, LPSTR cmd, int mode) {
	HWND hWnd;
	MSG msg;
	WNDCLASS wc;
	wc.hInstance = This;
	wc.lpszClassName = WinName;
	wc.lpfnWndProc = WndProc;
	wc.style = CS_HREDRAW | CS_VREDRAW; wc.hIcon = LoadIcon(NULL, IDI_APPLICATION); wc.hCursor = LoadCursor(NULL, IDC_ARROW); wc.lpszMenuName = NULL;
	wc.cbClsExtra = 0;
	wc.cbWndExtra = 0;
	wc.hbrBackground = (HBRUSH)(COLOR_WINDOW + 1);
	if (!RegisterClass(&wc)) return 0;
	hWnd = CreateWindow(WinName, _T("Каркас Windows-приложения"),
		WS_OVERLAPPEDWINDOW, CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT, HWND_DESKTOP, NULL, This, NULL);
	ShowWindow(hWnd, mode);
	while (GetMessage(&msg, NULL, 0, 0)) {
		TranslateMessage(&msg);
		DispatchMessage(&msg);
	}
	return 0;
}
```

```
const int WIDTH = 400;
const int HEIGHT = 300;
float v11, v12, v13, v21, v22, v23, v31, v32, v33, v43; float rho = 300., thetta = 75., phi = 30., ScreenDist = 500.; float A, B, C, D, An, Bn, Cn;
float xt[3], yt[3], zt[3];
float Al, Bl, Cl;
float alpha;
float th, ph, costh, cosph, sinth, sinph;
float factor = atan(1.0) / 45.;
void VidMatCoeff(float rho, float thetta, float phi) {
	th = thetta * factor; ph = phi * factor;
	costh = cos(th); sinth = sin(th); cosph = cos(ph); sinph = sin(ph); v11 = -sinth; v12 = -cosph * costh; v13 = -sinph * costh;
	v21 = costh; v22 = -cosph * sinth; v23 = -sinph * sinth;
	v31 = 0.; v32 = sinph; v33 = -cosph; v43 = rho;
}
POINT Perspective(float x, float y, float z) {
	POINT point;
	float xe, ye, ze;
	VidMatCoeff(rho, thetta, phi);
	xe = v11 * x + v21 * y;
	ye = v12 * x + v22 * y + v32 * z;
	ze = v13 * x + v23 * y + v33 * z + v43; point.x = ScreenDist * xe / ze + 400.; point.y = ScreenDist * ye / ze + 300.; return point;
}
LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)
{
	PAINTSTRUCT ps; static HBRUSH hBrush; class TFPoint
	{
	public:
		float X; float Y; float Z;
	};
	TFPoint CubePoints[] = {
	{-50,-50, -50}, { 50,-50, -50}, { 50, 50, -50}, {-50, 50, -50}, {-50, 50, 50}, {-50,-50, 50}, { 50,-50, 50}, { 50, 50, 50}
	};
```

```
int Gran[6][4] = {
{0,3,4,5}, {0,5,6,1},

{2,7,4,3}, {7,6,5,4}, {0,1,2,3}, {2,1,6,7}
};
POINT point1[4];
HDC hdc;
int sx, sy, xPos, yPos, zDelta; switch (message)
{
case WM_CREATE:
break; case WM_SIZE:
	sx = LOWORD(lParam); sy = HIWORD(lParam); break;
case WM_MOUSEMOVE:
	sx = LOWORD(lParam);
	sy = HIWORD(lParam);
	thetta += ((sx % 180) - 90) / 10;
	phi += ((sy % 180) - 90) / 10; InvalidateRect(hWnd, NULL, TRUE);
break;
case WM_MOUSEWHEEL: zDelta = (int)wParam;
	ScreenDist -= zDelta / 1000000.; InvalidateRect(hWnd, NULL, TRUE);
break;
case WM_PAINT:
	hdc = BeginPaint(hWnd, &ps);
	th = thetta * factor; ph = phi * factor;
	costh = cos(th); sinth = sin(th); cosph = cos(ph); sinph = sin(ph); A = rho
* sinph * costh; B = rho * sinph * sinth; C = rho * cosph; Al = A / (sqrt(A * A + B
* B + C * C));
	Bl = B / (sqrt(A * A + B * B + C * C));
	Cl = C / (sqrt(A * A + B * B + C * C));
	for (int i = 0; i < 6; i++)
	{
		for (int j = 0; j < 3; j++) {
			xt[j] = CubePoints[Gran[i][j]].X; yt[j] = CubePoints[Gran[i]
[j]].Y; zt[j] = CubePoints[Gran[i][j]].Z;
		}
		A = yt[0] * (zt[1] - zt[2]) - yt[1] * (zt[0] - zt[2]) + yt[2] * (zt[0] -
zt[1]); B = -(xt[0] * (zt[1] - zt[2]) - xt[1] * (zt[0] - zt[2]) + xt[2] * (zt[0] - zt[1])); C
```

```
= xt[0] * (yt[1] - yt[2]) - xt[1] * (yt[0] - yt[2]) + xt[2] * (yt[0] - yt[1]); An = A /
(sqrt(A * A + B * B + C * C));
                Bn = B / (sqrt(A * A + B * B + C * C));
                Cn = C / (sqrt(A * A + B * B + C * C));
                alpha = (An * Al + Bn * Bl + Cn * Cl);

                    for (int j = 0; j < 4; j++) {
                        point1[j] = Perspective(CubePoints[Gran[i][j]].X,
CubePoints[Gran[i][j]].Y,
                                CubePoints[Gran[i][j]].Z);
                    }
                D = point1[0].x * (point1[1].y - point1[2].y) -
                    point1[1].x * (point1[0].y - point1[2].y) +
                    point1[2].x * (point1[0].y - point1[1].y); if (D < 0)
                {
                    hBrush = CreateSolidBrush(RGB((1 - alpha) * 255,
                        (1 - alpha) * 255, (1 - alpha) * 255)); SelectObject(hdc,
hBrush);
                    Polygon(hdc, point1, 4);
                }
            }
            EndPaint(hWnd, &ps); break;
        default:
            return DefWindowProc(hWnd, message, wParam, lParam);
        }
        return 0;
}
```

Результат работы программы представлен на рисунке 1:

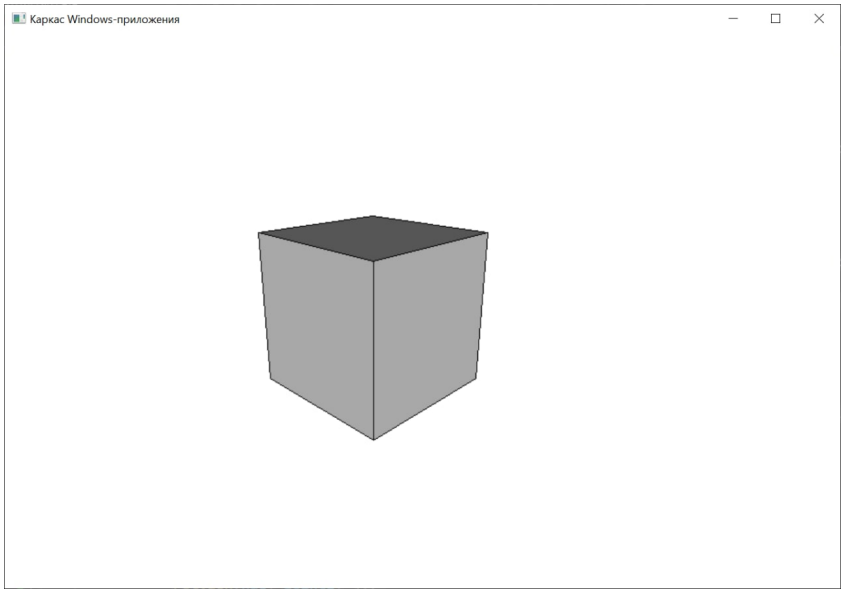

Рис.1. Вращающийся куб с удалёнными гранями и закраской.

ЗАДАНИЕ 2

С помощью фрагментов кода на основе приложения «Каркас» построить приложение «Кривая Безье».

Код программы:

```
#include <math.h>
#include <windows.h>
#include <tchar.h>
#include <fstream>
LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);
TCHAR WinName[] = _T("MainFrame");
int APIENTRY WinMain(HINSTANCE This, HINSTANCE Prev, LPSTR cmd, int
mode) {
	HWND hWnd;
	MSG msg;
	WNDCLASS wc;
	wc.hInstance = This;
	wc.lpszClassName = WinName;
	wc.lpfnWndProc = WndProc;
	wc.style = CS_HREDRAW | CS_VREDRAW; wc.hIcon = LoadIcon(NULL,
IDI_APPLICATION); wc.hCursor = LoadCursor(NULL, IDC_ARROW);
wc.lpszMenuName = NULL;
	wc.cbClsExtra = 0;
	wc.cbWndExtra = 0;
	wc.hbrBackground = (HBRUSH)(COLOR_WINDOW + 1); if (!
RegisterClass(&wc)) return 0;
	hWnd = CreateWindow(WinName, _T("Каркас Windows-приложения"),
		WS_OVERLAPPEDWINDOW, CW_USEDEFAULT,
CW_USEDEFAULT,
		CW_USEDEFAULT, CW_USEDEFAULT, HWND_DESKTOP, NULL,
This, NULL);
	ShowWindow(hWnd, mode);
	while (GetMessage(&msg, NULL, 0, 0)) {
		TranslateMessage(&msg);
		DispatchMessage(&msg); return 0;
```

```
	}
	static int sx, sy;
	const int SCALE = 1000;
	const int MARK = 4;
	const int WIDTH = 400;
	const int HEIGHT = 300;
	float v11, v12, v13, v21, v22, v23, v31, v32, v33, v43; float rho = 300., thetta
= 75., phi = 30., ScreenDist = 500.; float A, B, C, D, An, Bn, Cn;
	float Al, Bl, Cl;
	float xt[3], yt[3], zt[3];
	float alpha;
	float th, ph, costh, cosph, sinth, sinph;
	float factor = atan(1.0) / 45.;
	void DclnLp(POINT & point) {
		point.x = point.x * SCALE / sx;
		point.y = SCALE - point.y * SCALE / sy;
	}
	void transform(HDC& hdc) {
		SetMapMode(hdc, MM_ANISOTROPIC);
		SetWindowExtEx(hdc, SCALE, -SCALE, NULL);
SetViewportExtEx(hdc, sx, sy, NULL); SetViewportOrgEx(hdc, 0, sy, NULL);
	}
}
static HPEN hDash, hBezier;
static HBRUSH hRect, hSel;
static POINT pt[20];
static POINT point;
RECT rt;
static int count, index;
static bool capture;
int i;
std::ifstream in;
std::ofstream out;
LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM
wParam, LPARAM lParam) {
	POINT point1[4]; PAINTSTRUCT ps; HDC hdc;
	switch (message)
```

```
{
case WM_CREATE:
in.open("dat.txt"); if (in.fail())
{
MessageBox(hWnd, _T("Файл dat.txt не найден"), _T("Открытие
файла"),
MB_OK | MB_ICONEXCLAMATION); PostQuitMessage(0);
return 1;
}
for (count = 0; in >> pt[count].x; count++) in >> pt[count].y; in.close();
hDash = CreatePen(PS_DASH, 1, 0);
hBezier = CreatePen(PS_SOLID, 4, RGB(0, 0, 255));
hRect = CreateSolidBrush(RGB(128, 0, 128)); hSel =
CreateSolidBrush(RGB(255, 0, 0)); break;
case WM_SIZE:
sx = LOWORD(lParam);
sy = HIWORD(lParam);
break;
case WM_LBUTTONDOWN:
point.x = LOWORD(lParam); point.y = HIWORD(lParam);
DclnLp(point);
for (i = 0; i <= count; i++)
{
SetRect(&rt, pt[i].x - MARK, pt[i].y - MARK,
pt[i].x + MARK, pt[i].y + MARK); if (PtInRect(&rt, point))
}
}
{
index = i;
capture = true;
hdc = GetDC(hWnd); transform(hdc); FillRect(hdc, &rt, hSel);
ReleaseDC(hWnd, hdc); SetCapture(hWnd); return 0;
break;
case WM_LBUTTONUP:
if (capture) {
ReleaseCapture();
capture = false;
```

```
            }
            break;
        case WM_MOUSEMOVE:
            if (capture) {
                point.x = LOWORD(lParam);
                point.y = HIWORD(lParam); DclnLp(point);
                pt[index] = point; InvalidateRect(hWnd, NULL, TRUE);
            }
            break;
        case WM_PAINT:
        {
            hdc = BeginPaint(hWnd, &ps); transform(hdc); SelectObject(hdc,
hDash); Polyline(hdc, pt, count); SelectObject(hdc, hBezier); PolyBezier(hdc, pt,
count);
            for (int i = 0; i < count; i++) {
                SetRect(&rt, pt[i].x - MARK, pt[i].y - MARK, pt[i].x + MARK,
pt[i].y + MARK);
                FillRect(hdc, &rt, hRect);
            }
            break; }
        case WM_DESTROY: PostQuitMessage(0);
            DeleteObject(hDash); DeleteObject(hBezier); DeleteObject(hRect);
DeleteObject(hSel); out.open("dat.txt");
            for (i = 0; i < count; i++) out << pt[i].x << '\t' << pt[i].y << '\n';
                out.close();
        break; default:
            return DefWindowProc(hWnd, message, wParam, lParam); }
        return 0;
}
```

Результат выполнения программы представлен на рисунке 2:

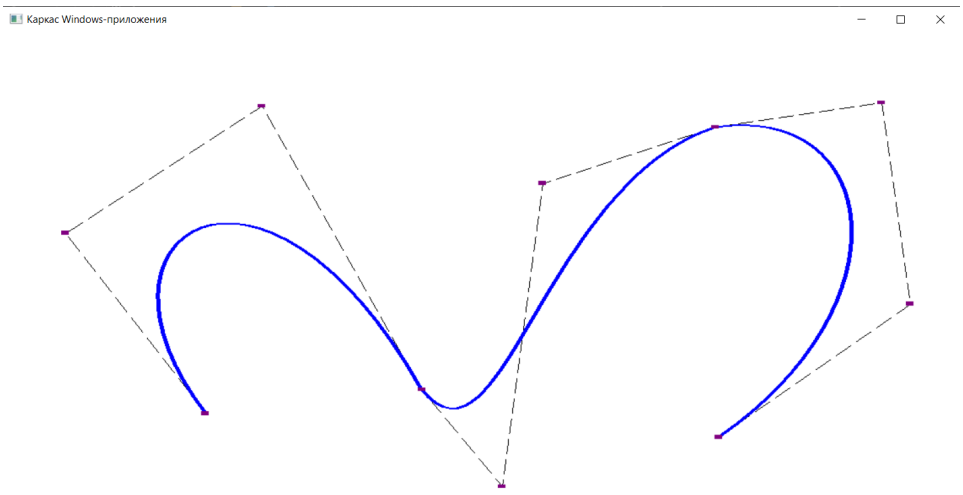

Рис.2. Результат работы приложения Безье.

ЗАДАНИЕ 3

По примеру кода «Безье» адаптировать код программы, написанной в MS-DOS для построения кривой B-сплайна

Код программы:

```
#include <math.h>
#include <windows.h>
#include <tchar.h>
#include <fstream>
#define MAX 100
#define N 30
LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);
TCHAR WinName[] = _T("MainFrame");
int APIENTRY WinMain(HINSTANCE This, HINSTANCE Prev, LPSTR cmd, int
mode)
{
	HWND hWnd;
	MSG msg;
	WNDCLASS wc;
	wc.hInstance = This;
	wc.lpszClassName = WinName;
	wc.lpfnWndProc = WndProc;
	wc.style = CS_HREDRAW | CS_VREDRAW; wc.hIcon = LoadIcon(NULL,
IDI_APPLICATION); wc.hCursor = LoadCursor(NULL, IDC_ARROW);
		wc.lpszMenuName = NULL;
	wc.cbClsExtra = 0;
	wc.cbWndExtra = 0;
	wc.hbrBackground = (HBRUSH)(COLOR_WINDOW + 1); if (!
RegisterClass(&wc)) return 0;
```

```
	hWnd = CreateWindow(WinName, _T("Каркас Windows-приложения"),
WS_OVERLAPPEDWINDOW, CW_USEDEFAULT,
		CW_USEDEFAULT,
		CW_USEDEFAULT, CW_USEDEFAULT, HWND_DESKTOP, NULL,
This, NULL);
	ShowWindow(hWnd, mode);
	while (GetMessage(&msg, NULL, 0, 0)) {
		TranslateMessage(&msg);
		DispatchMessage(&msg);
	}
	return 0;
}
float x[MAX], y[MAX], eps = 4, X, Y, t, xA, xB, xC, xD, yA, yB, yC, yD, a0, a1,
a2, a3, b0, b1, b2, b3;
int n, i, j, first;
PAINTSTRUCT ps;
const int WIDTH = 400; const int HEIGHT = 300; static int sx = 0, sy = 0;
const int SCALE = 1000; const int MARK = 4;
void DclnLp(POINT& point) {
	point.x = point.x * SCALE / sx;
	point.y = SCALE - point.y * SCALE / sy;
}
void transform(HDC& hdc) {
	SetMapMode(hdc, MM_ANISOTROPIC); SetWindowExtEx(hdc, SCALE,
-SCALE, NULL); SetViewportExtEx(hdc, sx, sy, NULL); SetViewportOrgEx(hdc,
0, sy, NULL);
}
static HPEN hLine, hMarker, hPunct; static HBRUSH hRect, hSel;
static POINT pt[20];
static POINT point;
RECT rt;
static int count, index;
static bool capture;
std::ifstream in;
std::ofstream out;
```

```
LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM
wParam, LPARAM lParam)
{
	POINT point1[4]; HDC hdc;
	switch (message) {
	case WM_CREATE: in.open("dat.txt");
		if (in.fail()) {
			MessageBox(hWnd, _T("Файл dat.txt не найден"), _T("Открытие
файла"),
				MB_OK | MB_ICONEXCLAMATION); PostQuitMessage(0);
			return 1;
		}
		for (count = 0; in >> pt[count].x; count++) in >> pt[count].y; in.close();
		hLine = CreatePen(PS_SOLID, 2, RGB(0, 0, 255));
		hMarker = CreatePen(PS_SOLID, 2, RGB(255, 0, 0)); hPunct =
CreatePen(PS_DASH, 1, 0);
		hRect = CreateSolidBrush(RGB(128, 0, 128)); hSel =
CreateSolidBrush(RGB(255, 0, 0)); break;
	case WM_SIZE:
		sx = LOWORD(lParam);
		sy = HIWORD(lParam);
		break;
	case WM_LBUTTONDOWN:
		MARK);
		point.x = LOWORD(lParam); point.y = HIWORD(lParam);
DclnLp(point);
		for (i = 0; i <= count; i++) {
			SetRect(&rt, pt[i].x - MARK, pt[i].y - MARK, pt[i].x + MARK,
pt[i].y +
				if (PtInRect(&rt, point)) {
				}
		}
		break;
	case WM_LBUTTONUP:
		if (capture)
			index = i;
		capture = true;
```

```
            hdc = GetDC(hWnd); transform(hdc); FillRect(hdc, &rt, hSel);
ReleaseDC(hWnd, hdc); SetCapture(hWnd); return 0;
            {
                ReleaseCapture();
                capture = false; }
            break;
        case WM_MOUSEMOVE:
            if (capture) {
                point.x = LOWORD(lParam);
                point.y = HIWORD(lParam); DclnLp(point);
                pt[index] = point; InvalidateRect(hWnd, NULL, TRUE);
            }
            break;
        case WM_PAINT:
        {
            hdc = BeginPaint(hWnd, &ps); transform(hdc); SelectObject(hdc,
hPunct); Polyline(hdc, pt, count); SelectObject(hdc, hMarker); for (i = 0; i <=
count; i++)
            {
                X = pt[i].x; Y = pt[i].y;
                MoveToEx(hdc, pt[i].x - eps, pt[i].y - eps, 0); LineTo(hdc, pt[i].x +
eps, pt[i].y + eps); MoveToEx(hdc, pt[i].x + eps, pt[i].y - eps, 0); LineTo(hdc,
pt[i].x - eps, pt[i].y + eps);
            }
            SelectObject(hdc, hLine); first = 1;
            for (i = 1; i < count - 2; i++) {
                xA = pt[i - 1].x; xB = pt[i].x; xC = pt[i + 1].x; xD = pt[i + 2].x; yA
= pt[i - 1].y; yB = pt[i].y; yC = pt[i + 1].y; yD = pt[i + 2].y; a3 = (-xA + 3 * (xB -
xC) + xD) / 6.0;
                b3 = (-yA + 3 * (yB - yC) + yD) / 6.0;
                a2 = (xA - 2 * xB + xC) / 2.0; b2 = (yA - 2 * yB + yC) / 2.0; a1 =
(xC - xA) / 2.0;
                b1 = (yC - yA) / 2.0;
                a0 = (xA + 4 * xB + xC) / 6.0; b0 = (yA + 4 * yB + yC) / 6.0; for (j
= 0; j <= N; j++)
                {
                    t = (float)j / (float)N;
```

```
                    X = ((a3 * t + a2) * t + a1) * t + a0;
                    Y = ((b3 * t + b2) * t + b1) * t + b0;
                    if (first) { first = 0; MoveToEx(hdc, X, Y, 0); }
                    else LineTo(hdc, X, Y);
                }
            }
            EndPaint(hWnd, &ps); }
        break;
        case WM_DESTROY:
            DeleteObject(hLine); DeleteObject(hMarker); DeleteObject(hRect);
DeleteObject(hSel); out.open("dat.txt");
            for (i = 0; i < count; i++)
                out << pt[i].x << '\t' << pt[i].y << '\n';
            out.close();
        break; default:
            return DefWindowProc(hWnd, message, wParam, lParam);
            break;
        }
        return 0;
}
```

Результат выполнения программы представлен на рисунке 3:

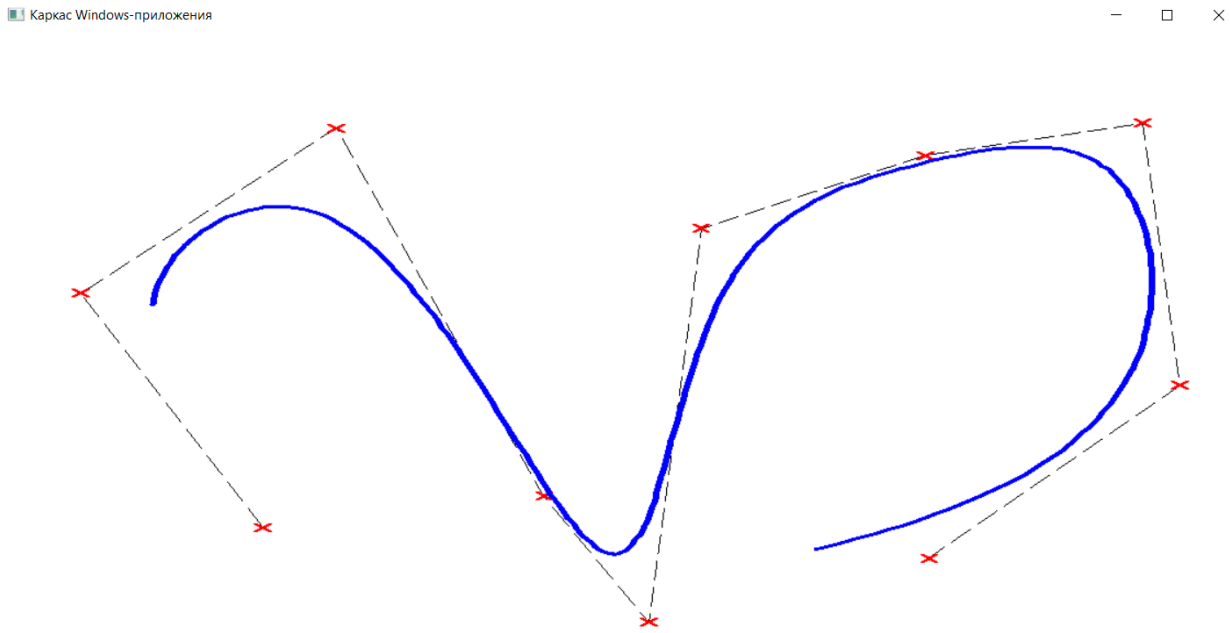

Рис.3. Результат работы приложения «Кривая B-сплайн»

ВЫВОДЫ

В ходе лабораторной работы было написаны 3 программы, одна из которых отрисовывает объемную, 3-х мерную, вращающуюся фигуру и 2 программы которые обрисовываю интерактивные кривые. Были освоены навыки вращения объемных фигур и построения сплайнов, вращающуюся фигуру.