Лабораторная работа № 2

по дисциплине комп. геометрия и графика

студента группы ИТ-32

Манукова Давида Альбертовича

Выполнение: \_\_\_\_\_\_\_\_\_\_ Защита: \_\_\_\_\_\_\_\_\_\_

**Основные задачи компьютерной геометрии**

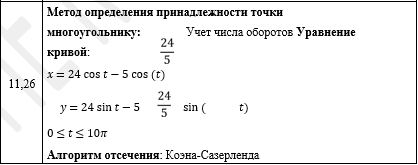
Цель работы: Получение практических применения различных математических инструментов для решения основных задач компьютерной геометрии.

Содержание работы

1. Разработать в Visual C++ приложение, которое решает задачу принадлежности точки многоугольнику с помощью метода, указанного в варианте задания.
2. Разработать в Visual C++ оконное приложение, которое рисует кривую согласно варианту задания.

*При рисовании кривой необходимо использовать алгоритм отсечения, указанный в варианте задания*

1. Изменить оконное приложение Win32, разработанное в лабораторной работе №1, так чтобы в нем использовалось отсечение с помощью регионов GDI+.
2. Протестировать работу приложений, разработанных в п. 1, 2 и 3. Результаты тестирования отразить в отчете.
3. Включить в отчет исходный программный код и выводы о проделанной работе.



Ход работы

1. Разработал в Visual C++ приложение, которое определяет принадлежность точки многоугольнику.

#include <iostream>

#include <vector>

using namespace std;

int pnpoly(int nvert, float\* vertx, float\* verty, float testx, float testy)

{

int i, j, c = 0;

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

if ((verty[i] == testy && verty[j] == testy) && (vertx[i] > testx) != (vertx[j] > testx))

{

c++;

break;

}

if ((vertx[i] == testx && vertx[j] == testx) && (verty[i] > testy) != (verty[j] > testy))

{

c++;

break;

}

float t2 = (verty[i] - testy) / (verty[i] - verty[j]);

float t1 = (vertx[i] - testx) + ((vertx[j] - vertx[i]) \* t2);

if (t2 == 0 && t1 == 0)

{

c = !c;

break;

}

if ((t2 >= 0 && t2 <= 1) && t1 >= 0)

{

if ((0 \* (vertx[i] - vertx[j]) + 1 \* (verty[i] - verty[j])) > 0)

{

c++;

}

else c--;

}

}

return c;

}

int main()

{

setlocale(LC\_ALL, "ru");

const int polygN = 8;

float polygX[polygN] = { 2, 2, 5, 10, 13, 13, 10, 5 };

float polygY[polygN] = { 5, 8, 11, 11, 8, 5, 2, 2 };

float testpointX = 7;

float testpointY = 10;

int c = pnpoly(polygN, polygX, polygY, testpointX, testpointY);

if (c != 0)

{

cout << "Точка внутри многоугольника";

}

else cout << "Точка вне многоугольника";

return 0;

}

1. Разработал оконное приложение, которое рисует кривую, при этом используется алгоритм отсечения.

#include "framework.h"

#include "CGG-Lab\_2.2.h"

#define \_USE\_MATH\_DEFINES

#include <math.h>

#include <vector>

#include <windows.h>

#include <objidl.h>

#include <gdiplus.h>

#include "Resource.h"

using namespace Gdiplus;

#pragma comment (lib,"Gdiplus.lib")

#define MAX\_LOADSTRING 100

#define LEFT 1

#define RIGHT 2

#define BOT 4

#define TOP 8

class WorldWindow

{

public:

float Left;

float Top;

float Right;

float Bottom;

inline WorldWindow(float left, float top, float right, float bottom) : Left(left), Top(top), Right(right), Bottom(bottom) {}

};

class ViewPort : public Gdiplus::Rect

{

public:

inline ViewPort(int left, int top, int right, int bottom) : Gdiplus::Rect(left, top, right-left, bottom-top) {}

};

inline void WorldToViewport(const WorldWindow& w, const ViewPort& vp, Gdiplus::PointF\* points, unsigned int count)

{

float A = vp.Width / (w.Right - w.Left);

float B = vp.Height / (w.Bottom - w.Top);

float C = vp.X - A \* w.Left;

float D = vp.Y - B \* w.Top;

for (unsigned int i = 0; i < count; i++)

{

points[i].X = A \* points[i].X + C;

points[i].Y = B \* points[i].Y + D;

}

}

int vcode(RectF\* r, PointF\* p)

{

int code = 0;

if (p->X < r->GetLeft())

{

code += LEFT;

}

if (p->X > r->GetRight())

{

code += RIGHT;

}

if (p->Y > r->GetBottom())

{

code += BOT;

}

if (p->Y < r->GetTop())

{

code += TOP;

}

return code;

}

void Display(HDC hdc, PointF\* a, PointF\* b)

{

Graphics g(hdc);

Pen pen(Color::Black);

g.DrawLine(&pen, \*a, \*b);

}

int cohen\_sutherland(RectF\* r, PointF\* points, unsigned int count, HDC hdc)

{

int code\_a, code\_b, code; /\* код концов отрезка \*/

PointF\* c; /\* одна из точек \*/

int i, j = 0;

for (i = 0, j = count - 1; i < count; j = i++)

{

code\_a = vcode(r, &points[i]);

code\_b = vcode(r, &points[j]);

while (code\_a | code\_b) {

if (code\_a) {

code = code\_a;

c = &points[i];

}

else {

code = code\_b;

c = &points[j];

}

if (code & LEFT) {

c->Y += (&points[i].Y - &points[j].Y) \* (float)(r->GetLeft() - c->X) / (&points[i].X - &points[j].X);

c->X = r->GetLeft();

}

else if (code & RIGHT) {

c->Y += (&points[i].Y - &points[j].Y) \* (float)(r->GetRight() - c->X) / (&points[i].X - &points[j].X);

c->X = r->GetRight();

}

else if (code & BOT) {

c->X += (&points[i].X - &points[j].X) \* (float)(r->GetBottom() - c->Y) / (&points[i].Y - &points[j].Y);

c->Y = r->GetBottom();

}

else if (code & TOP) {

c->X += (&points[i].X - &points[j].X) \* (float)(r->GetTop() - c->Y) / (&points[i].Y - &points[j].Y);

c->Y = r->GetTop();

}

if (code == code\_a) {

points[i].X = c->X;

points[i].Y = c->Y;

code\_a = vcode(r, &points[i]);

}

else {

points[j].X = c->X;

points[j].Y = c->Y;

code\_b = vcode(r, &points[j]);

}

}

Display(hdc, &points[i], &points[j]);

}

return 0;

}

// Глобальные переменные:

HINSTANCE hInst; // текущий экземпляр

WCHAR szTitle[MAX\_LOADSTRING]; // Текст строки заголовка

WCHAR szWindowClass[MAX\_LOADSTRING]; // имя класса главного окна

GdiplusStartupInput gdiplusStartupInput;

ULONG\_PTR gdiplusToken;

ATOM MyRegisterClass(HINSTANCE hInstance);

BOOL InitInstance(HINSTANCE, int);

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

INT\_PTR CALLBACK About(HWND, UINT, WPARAM, LPARAM);

int APIENTRY wWinMain(\_In\_ HINSTANCE hInstance,

\_In\_opt\_ HINSTANCE hPrevInstance,

\_In\_ LPWSTR lpCmdLine,

\_In\_ int nCmdShow)

{

GdiplusStartup(&gdiplusToken, &gdiplusStartupInput, NULL);

UNREFERENCED\_PARAMETER(hPrevInstance);

UNREFERENCED\_PARAMETER(lpCmdLine);

// TODO: Разместите код здесь.

// Инициализация глобальных строк

LoadStringW(hInstance, IDS\_APP\_TITLE, szTitle, MAX\_LOADSTRING);

LoadStringW(hInstance, IDC\_CGGLAB22, szWindowClass, MAX\_LOADSTRING);

MyRegisterClass(hInstance);

// Выполнить инициализацию приложения:

if (!InitInstance (hInstance, nCmdShow))

{

return FALSE;

}

HACCEL hAccelTable = LoadAccelerators(hInstance, MAKEINTRESOURCE(IDC\_CGGLAB22));

MSG msg;

// Цикл основного сообщения:

while (GetMessage(&msg, nullptr, 0, 0))

{

if (!TranslateAccelerator(msg.hwnd, hAccelTable, &msg))

{

TranslateMessage(&msg);

DispatchMessage(&msg);

}

}

GdiplusShutdown(gdiplusToken);

return (int) msg.wParam;

}

ATOM MyRegisterClass(HINSTANCE hInstance)

{

WNDCLASSEXW wcex;

wcex.cbSize = sizeof(WNDCLASSEX);

wcex.style = CS\_HREDRAW | CS\_VREDRAW;

wcex.lpfnWndProc = WndProc;

wcex.cbClsExtra = 0;

wcex.cbWndExtra = 0;

wcex.hInstance = hInstance;

wcex.hIcon = LoadIcon(hInstance, MAKEINTRESOURCE(IDI\_CGGLAB22));

wcex.hCursor = LoadCursor(nullptr, IDC\_ARROW);

wcex.hbrBackground = (HBRUSH)(COLOR\_WINDOW+1);

wcex.lpszMenuName = MAKEINTRESOURCEW(IDC\_CGGLAB22);

wcex.lpszClassName = szWindowClass;

wcex.hIconSm = LoadIcon(wcex.hInstance, MAKEINTRESOURCE(IDI\_SMALL));

return RegisterClassExW(&wcex);

}

BOOL InitInstance(HINSTANCE hInstance, int nCmdShow)

{

hInst = hInstance; // Сохранить маркер экземпляра в глобальной переменной

HWND hWnd = CreateWindowW(szWindowClass, szTitle, WS\_OVERLAPPEDWINDOW,

CW\_USEDEFAULT, 0, CW\_USEDEFAULT, 0, nullptr, nullptr, hInstance, nullptr);

if (!hWnd)

{

return FALSE;

}

ShowWindow(hWnd, nCmdShow);

UpdateWindow(hWnd);

return TRUE;

}

LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

{

switch (message)

{

case WM\_COMMAND:

{

int wmId = LOWORD(wParam);

// Разобрать выбор в меню:

switch (wmId)

{

case IDM\_ABOUT:

DialogBox(hInst, MAKEINTRESOURCE(IDD\_ABOUTBOX), hWnd, About);

break;

case IDM\_EXIT:

DestroyWindow(hWnd);

break;

default:

return DefWindowProc(hWnd, message, wParam, lParam);

}

}

break;

case WM\_PAINT:

{

PAINTSTRUCT ps;

HDC hdc = BeginPaint(hWnd, &ps);

Graphics g(hdc);

g.Clear(Color::White);

Pen pen(Color::Black);

RectF r(100.f, 100.f, 500.f, 300.f);

g.DrawRectangle(&pen, r);

//PointF a(50.f, 50.f);

//PointF b(550.f, 550.f);

//int result = cohen\_sutherland(&r, &a, &b, hdc);

WorldWindow wWnd(-50, -50, 50, 50);

ViewPort vPort(150, 50, 300, 150);

vPort.Width = 600;

vPort.Height = 600;

const int n = 64;

PointF points[n];

float tMax = 10 \* M\_PI;

float tMin = 0;

float dt = (tMax - tMin) / (float)(n - 1);

for (int i = 0; i < n; i++)

{

points[i].X = 24 \* cos(dt \* i) - 5 \* (float)cos(24 / 5 \* dt \* i);

points[i].Y = 24 \* sin(dt \* i) - 5 \* (float)sin(24 / 5 \* dt \* i);

}

WorldToViewport(wWnd, vPort, points, n);

int result = cohen\_sutherland(&r, points, n, hdc);

//g.DrawCurve(&pen, points, n);

EndPaint(hWnd, &ps);

}

break;

case WM\_DESTROY:

PostQuitMessage(0);

break;

default:

return DefWindowProc(hWnd, message, wParam, lParam);

}

return 0;

}

// Обработчик сообщений для окна "О программе".

INT\_PTR CALLBACK About(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam)

{

UNREFERENCED\_PARAMETER(lParam);

switch (message)

{

case WM\_INITDIALOG:

return (INT\_PTR)TRUE;

case WM\_COMMAND:

if (LOWORD(wParam) == IDOK || LOWORD(wParam) == IDCANCEL)

{

EndDialog(hDlg, LOWORD(wParam));

return (INT\_PTR)TRUE;

}

break;

}

return (INT\_PTR)FALSE;

}

1. Изменил оконное приложение, разработанное в ЛР№1, так чтобы в нем использовалось отсечение с помощью регионов GDI+.

#include "framework.h"

#include "CGG-Lab\_2.3.h"

#include <windows.h>

#include <objidl.h>

#include <gdiplus.h>

#include "Resource.h"

using namespace Gdiplus;

#pragma comment (lib,"Gdiplus.lib")

#define MAX\_LOADSTRING 100

// Глобальные переменные:

GdiplusStartupInput gdiplusStartupInput;

ULONG\_PTR gdiplusToken;

HINSTANCE hInst; // текущий экземпляр

WCHAR szTitle[MAX\_LOADSTRING]; // Текст строки заголовка

WCHAR szWindowClass[MAX\_LOADSTRING]; // имя класса главного окна

RECT clientRect;

HWND mhWnd;

// Отправить объявления функций, включенных в этот модуль кода:

ATOM MyRegisterClass(HINSTANCE hInstance);

BOOL InitInstance(HINSTANCE, int);

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

INT\_PTR CALLBACK About(HWND, UINT, WPARAM, LPARAM);

VOID Display(HDC);

int APIENTRY wWinMain(\_In\_ HINSTANCE hInstance,

\_In\_opt\_ HINSTANCE hPrevInstance,

\_In\_ LPWSTR lpCmdLine,

\_In\_ int nCmdShow)

{

GdiplusStartup(&gdiplusToken, &gdiplusStartupInput, NULL);

UNREFERENCED\_PARAMETER(hPrevInstance);

UNREFERENCED\_PARAMETER(lpCmdLine);

// TODO: Разместите код здесь.

// Инициализация глобальных строк

LoadStringW(hInstance, IDS\_APP\_TITLE, szTitle, MAX\_LOADSTRING);

LoadStringW(hInstance, IDC\_CGGLAB23, szWindowClass, MAX\_LOADSTRING);

MyRegisterClass(hInstance);

// Выполнить инициализацию приложения:

if (!InitInstance(hInstance, nCmdShow))

{

return FALSE;

}

HACCEL hAccelTable = LoadAccelerators(hInstance, MAKEINTRESOURCE(IDC\_CGGLAB23));

MSG msg;

// Цикл основного сообщения:

while (GetMessage(&msg, nullptr, 0, 0))

{

if (!TranslateAccelerator(msg.hwnd, hAccelTable, &msg))

{

TranslateMessage(&msg);

DispatchMessage(&msg);

}

}

GdiplusShutdown(gdiplusToken);

return (int)msg.wParam;

}

//

// ФУНКЦИЯ: MyRegisterClass()

//

// ЦЕЛЬ: Регистрирует класс окна.

//

ATOM MyRegisterClass(HINSTANCE hInstance)

{

WNDCLASSEXW wcex;

wcex.cbSize = sizeof(WNDCLASSEX);

wcex.style = CS\_HREDRAW | CS\_VREDRAW;

wcex.lpfnWndProc = WndProc;

wcex.cbClsExtra = 0;

wcex.cbWndExtra = 0;

wcex.hInstance = hInstance;

wcex.hIcon = LoadIcon(hInstance, MAKEINTRESOURCE(IDC\_CGGLAB23));

wcex.hCursor = LoadCursor(nullptr, IDC\_ARROW);

wcex.hbrBackground = (HBRUSH)(COLOR\_WINDOW + 1);

wcex.lpszMenuName = MAKEINTRESOURCEW(IDC\_CGGLAB23);

wcex.lpszClassName = szWindowClass;

wcex.hIconSm = LoadIcon(wcex.hInstance, MAKEINTRESOURCE(IDI\_SMALL));

return RegisterClassExW(&wcex);

}

BOOL InitInstance(HINSTANCE hInstance, int nCmdShow)

{

hInst = hInstance;

HWND hWnd = CreateWindowW(szWindowClass, szTitle, WS\_OVERLAPPEDWINDOW,

100, 100, 1280, 720, nullptr, nullptr, hInstance, nullptr);

if (!hWnd)

{

return FALSE;

}

ShowWindow(hWnd, nCmdShow);

UpdateWindow(hWnd);

mhWnd = hWnd;

return TRUE;

}

LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

{

switch (message)

{

case WM\_COMMAND:

{

int wmId = LOWORD(wParam);

// Разобрать выбор в меню:

switch (wmId)

{

case IDM\_ABOUT:

DialogBox(hInst, MAKEINTRESOURCE(IDD\_ABOUTBOX), hWnd, About);

break;

case IDM\_EXIT:

DestroyWindow(hWnd);

break;

default:

return DefWindowProc(hWnd, message, wParam, lParam);

}

}

break;

case WM\_PAINT:

{

PAINTSTRUCT ps;

HDC hdc = BeginPaint(hWnd, &ps);

Display(hdc);

EndPaint(hWnd, &ps);

}

break;

case WM\_DESTROY:

PostQuitMessage(0);

break;

default:

return DefWindowProc(hWnd, message, wParam, lParam);

}

return 0;

}

// Обработчик сообщений для окна "О программе".

INT\_PTR CALLBACK About(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam)

{

UNREFERENCED\_PARAMETER(lParam);

switch (message)

{

case WM\_INITDIALOG:

return (INT\_PTR)TRUE;

case WM\_COMMAND:

if (LOWORD(wParam) == IDOK || LOWORD(wParam) == IDCANCEL)

{

EndDialog(hDlg, LOWORD(wParam));

return (INT\_PTR)TRUE;

}

break;

}

return (INT\_PTR)FALSE;

}

VOID Display(HDC hdc)

{

Graphics g(hdc);

g.Clear(Color::White);

g.SetSmoothingMode(SmoothingModeHighQuality);

// Колеса

Region rgn;

Region rgn1(Rect(350, 350, 110, 180));

Pen compPen1(Color::Black, 10.f);

float comp[6] = {

0.0f, 0.2f,

0.3f, 0.7f,

0.8f, 1.0f };

compPen1.SetCompoundArray(comp, 6);

g.SetClip(&rgn1);

g.DrawEllipse(&compPen1, 400, 400, 120, 120); //заднее колесо

g.SetClip(&rgn);

Region rgn2(Rect(460, 350, 110, 180));

Pen compPen2(Color::DarkCyan, 10.f);

compPen2.SetCompoundArray(comp, 6);

g.SetClip(&rgn2);

g.DrawEllipse(&compPen2, 400, 400, 120, 120);

g.SetClip(&rgn);

Region rgn3(Rect(600, 350, 80, 180));

g.SetClip(&rgn3);

g.DrawEllipse(&compPen2, 620, 400, 120, 120);

g.SetClip(&rgn);

Region rgn4(Rect(680, 350, 80, 180));

g.SetClip(&rgn4);

g.DrawEllipse(&compPen1, 620, 400, 120, 120); //переднее колесо

g.SetClip(&rgn);

HatchBrush hatchBrush(HatchStyleOutlinedDiamond, Color::Gray, Color::White);

g.FillEllipse(&hatchBrush, 400, 400, 120, 120); //заднее колесо

g.FillEllipse(&hatchBrush, 620, 400, 120, 120); //переднее колесо

Pen redPen(Color::Red, 8.f);

redPen.SetStartCap(LineCapRound);

redPen.SetLineJoin(LineJoinRound);

redPen.SetEndCap(LineCapRound);

// Рама

Point framePoints[4] = {

Point(460, 460),

Point(550, 460),

Point(650, 360),

Point(490, 360)

};

g.DrawPolygon(&redPen, framePoints, 4);

// Руль

Point wheelPoints[3] = {

Point(680, 460),

Point(637, 320),

Point(610, 320)

};

g.DrawLines(&redPen, wheelPoints, 3);

// Сиденье

Point Wpt1(550, 460);

Point Wpt2(475, 340);

g.DrawLine(&redPen, Wpt1, Wpt2);

Pen blackPen(Color::Black, 8.f);

blackPen.SetStartCap(LineCapRound);

blackPen.SetEndCap(LineCapRound);

Point Qpt1(460, 340);

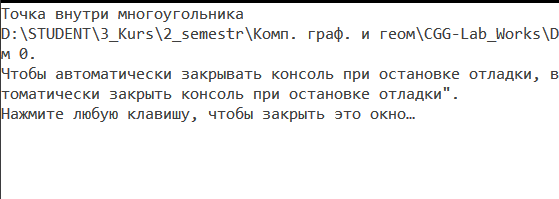
Point Qpt2(490, 340);

g.DrawLine(&blackPen, Qpt1, Qpt2);

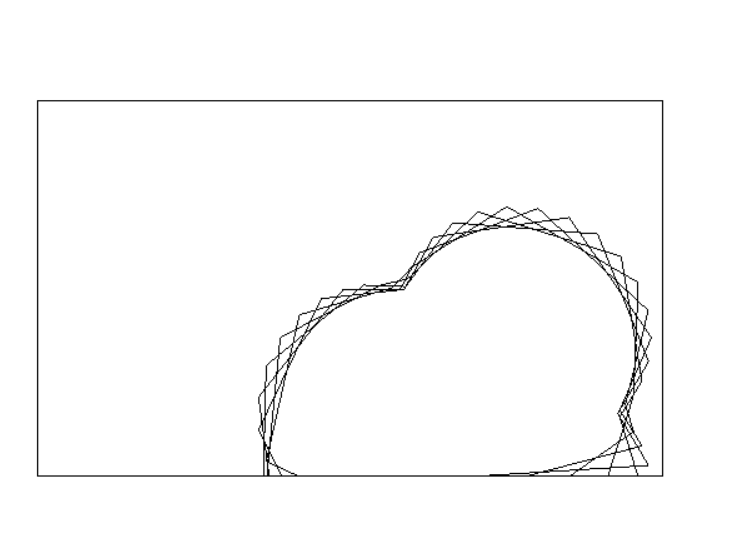
}

1. Результат работы

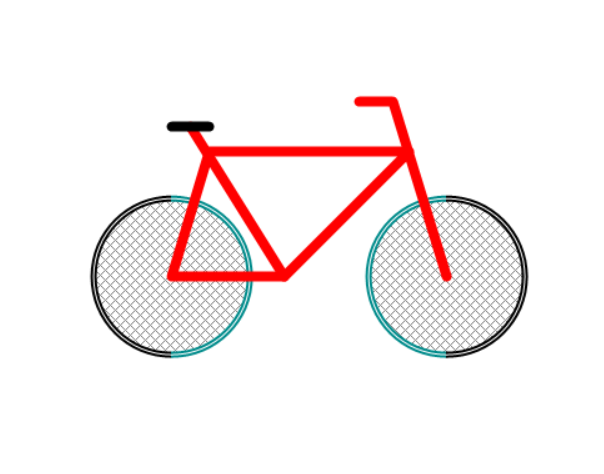
1 Задание



2 Задание



3 Задание



Вывод: получил практические применения различных математических инструментов для решения основных задач компьютерной геометрии.