**Лабораторна робота №5**

**Застосування афінних перетворень до зсуву, повороту, перенесення, віддзеркалення та проектування дво- і тривимірних зображень**

**Мета:** навчитись застосування афінних перетворень до зсуву, повороту, перенесення, віддзеркалення та проектування дво- і тривимірних зображень.

**Варіант №7**

**Теоретичні відомості**

**Афінним** називається таке відображення площини (простору) на іншу площину (простір), при якому кожна пряма відображається в пряму.

**Афінне перетворення** – це афінне відображення площини на себе.

В бібліотеці Open GL перетворення фігур досягається шляхгм відповідного перетворення системи координат перед виводом фігури. Для перенесення системи координат застосовуються наступні процедури:

**glTranslated(dx, dy, dz: double);**

де dx, dy, dz – величини зсуву системи координат по відповідним осям.

**glRotated(Angle, x, y, z: double);**

де x, y, z – координати радіус-вектора, навоколо якого здійснюється поворот системи координат (довжина вектора має дорівнювати 1); Angle – кут повороту в градусах (додатній напрямок проти годинникової стрілки, від’ємний – за годинниковою стрілкою).

**glScaled(kx, ky, kz: double);**

де kx, ky,kz – коефіцієнти стиснення / розтягу масштабу відповідних координатних осей.

Для збереження поточного стану координатної системи застосовується процедура **pushmatrix**, а для його відновлення – **popmatrix**.

**Завдання**

1. З застосуванням Open GL написати процедуру для виводу на екран зображення кола з центром в початку координат та радіусом 1.

2. Шляхом перетворення координат та виклику написаної процедури вивести на екран кілька кіл в різних позиціях та різного радіусу.

3. Переробити програму малювання графіка з попередньої лабораторної роботи, так, щоб по заданому інтервалу зміни координати х автоматично визначалися необхідні межі виводу по координаті y та на форму виводився весь графік функції цілком.

**Код для завдань 1, 2 (рис. 1):**

**unit UnitMain;**

**interface**

**uses**

Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, ExtCtrls, OpenGL;

**type**

TFormMain = class(TForm)

PaintBox: TPaintBox;

procedure FormCreate(Sender: TObject);

procedure The\_FormPaint(Sender: TObject);

procedure FormDestroy(Sender: TObject);

private

{ Private declarations }

public

hrc: HGLRC;

end;

**var**

FormMain: TFormMain;

**implementation**

**{$R \*.dfm}**

**procedure SetDCPixelFormat(hdc: HDC);**

**var**

pfd: TPixelFormatDescriptor;

nPixelFormat: integer;

**begin**

FillChar(pfd, SizeOf(pfd), 0);

nPixelFormat:= ChoosePixelFormat(hdc, @pfd);

SetPixelFormat(hdc, nPixelFormat, @pfd);

**end;**

**procedure TFormMain.FormCreate(Sender: TObject);**

**begin**

SetDCPixelFormat(Canvas.Handle);

hrc:= wglCreateContext(Canvas.Handle);

**end;**

**procedure Draw\_Circle;**

**var**

j:word;

x:real;

**begin**

glLineWidth(2);

glEnable(GL\_LINE\_SMOOTH);

glBegin(GL\_LINE\_STRIP);

for j:= 0 to 500 do

begin

x:= j/180\*Pi;

glVertex2f(cos(x),sin(x));

end;

glEnd;

glDisable(GL\_LINE\_SMOOTH);

glLineWidth(1);

**end;**

**procedure TFormMain.The\_FormPaint (Sender: TObject);**

**begin**

wglMakeCurrent(Canvas.Handle, hrc);

glClearColor(0.95,0.95,0.95,1);

glClear(GL\_COLOR\_BUFFER\_BIT);

glColor3d(1,1,1);

glColor3d(0,0.65,0);

glPushMatrix;

**Draw\_Circle; //осн.круг**

glRotated(-15,0,0,1);

glTranslated(0.5,0.5,0);

glScaled(0.1,0.1,1);

**Draw\_Circle; //правый глаз**

glTranslated(-10, 0, 0);

**Draw\_Circle; //левій глаз**

**glTranslated(5, -8, 0);**

glScaled(8, 0.3, 1);

Draw\_Circle; //рот

glPopMatrix;

wglMakeCurrent(0,0);

**end;**

**procedure TFormMain.FormDestroy(Sender: TObject);**

**begin**

wglDeleteContext(hrc);

**end;**

**end.**

**Код для завдань 3 (рис. 2):**

procedure TFormMain.PaintBox1Paint(Sender: TObject);

const

E= 600;

var

yMax,yMin,x,xMax,xMin,k,xPMax,xPMin,yPMax,yPMin,yP,xP: real;

j: integer;

y: array[0..E] of real;

begin

xMax:= 4.7;

xMin:= -10;

if xMax<xMin then

begin

k:= xMin;

xMin:= xMax;

xMax:= k;

end;

k:= xMax-xMin;

yMax:= 0;

yMin:=0;

for j:= 0 to E do

begin

x:=xMin+k\*j/E;

y(.j.):= cos(10\*x)/sin(3\*x);

if y(.j.)>yMax then yMax:= y(.j.);

if y(.j.)<yMin then yMin:= y(.j.);

end;

if xMax<-xMin\*0.05 then xPMax:= -xMin\*0.05

else xPMax:= xMax;

if xMin>-xMax\*0.05 then xPMin:= -xMax\*0.05

else xPMin:= XMin;

if yMax<-yMin\*0.05 then yPMax:= -yMin\*0.05

else yPMax:= yMax;

if yMin>-yMax\*0.05 then yPMin:= -yMax\*0.05

else yPMin:= yMin;

xP:= 0.2/(xPMin/(xPMax+xPMin)-xPMax/(xPMax+xPMin));

yP:= 0.2/(yPMin/(yPMax+yPMin)-yPMax/(yPMax+yPMin));

wglMakeCurrent(Canvas.Handle, hrc);

glPushMatrix;

glClearColor(0.2, 0.5, 0.75, 1.0);

glClear(GL\_COLOR\_BUFFER\_BIT);

glColor3d(1,1,1);

glTranslated(xP, yP, 0);

glEnable(GL\_LINE\_STIPPLE);

glLineStipple(1,$F00F);

glBegin(GL\_LINES);

for j:=-10 to 10 do

begin

glVertex2f(j\*0.2, 2);

glVertex2f(j\*0.2, -2);

glVertex2f(2, j\*0.2);

glVertex2f(-2, j\*0.2);

end;

glEnd;

glDisable(GL\_LINE\_STIPPLE);

glBegin(GL\_LINES);

glColor3d(0,0,0);

glVertex2f(2, 0);

glVertex2f(-2, 0);

glVertex2f(0, 2);

glVertex2f(0, -2);

glEnd;

glBegin(GL\_LINE\_STRIP);

glVertex2f(0.96-xP, 0.015);

glVertex2f(1-xP, 0);

glVertex2f(0.96-xP, -0.015);

glEnd;

glBegin(GL\_LINE\_STRIP);

glVertex2f(0.015, 0.96-yP);

glVertex2f(0, 1-yP);

glVertex2f(-0.015, 0.96-yP);

glEnd;

glScaled(3/(xPMax-xPMin), 3/(yPMax-yPMin), 1);

glColor3d(1,0,0);

glLineWidth(2);

glEnable(GL\_LINE\_SMOOTH);

glBegin(GL\_LINE\_STRIP);

for j:=0 to E do

begin

glVertex2f(xMin+k\*j/E, y(.j.));

end;

glEnd;

glDisable(GL\_LINE\_SMOOTH);

glLineWidth(1);

glPopMatrix;

wglMakeCurrent(0,0);

end;

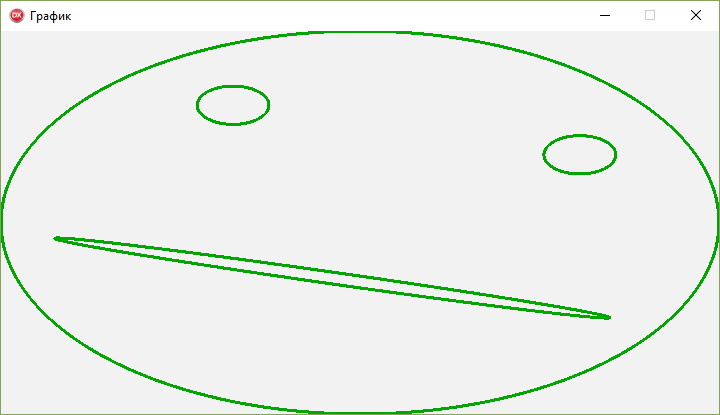


Рис. 1. Кілька кіл в різних позиціях та різного радіусу

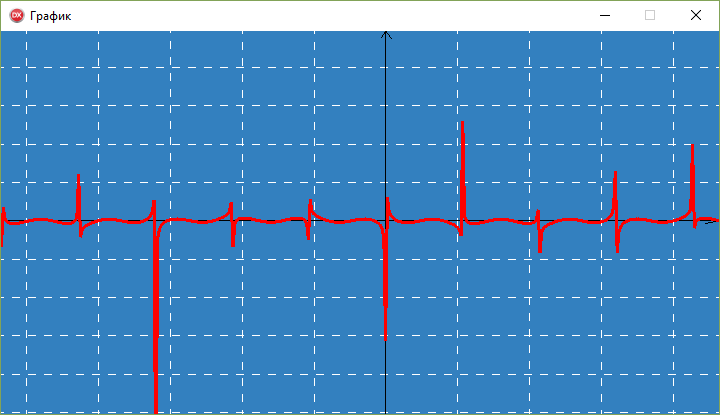


Рис. 2. Перероблена програма малювання графіка

**Висновок.** Під час виконання лабораторної роботи я навчилась застосуванню афінних перетворень до зсуву, повороту, перенесення, віддзеркалення та проектування дво- і тривимірних зображень.