МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«**Вятский государственный университет**»

**(ФГБОУ ВО «ВятГУ»)**

Факультет автоматики и вычислительной техники

Кафедра ЭВМ

Лабораторная работа № 8 по дисциплине

«Компьютерная графика»

**«Представление трёхмерных объектов»**

Выполнил студент группы ИВТ-21\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Птахова А.М

Проверил преподаватель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Коржавина А.С.

Киров 2021

**Цель работы:** Научиться программно выводить трёхмерные объекты.

**Задание:**

### Написать на языке Pascal программу, которая выводит каркас векторной полигональной модели.

**Краткие теоретические сведения**

Для описания пространственных объектов здесь используются следующие элементы: вершины: отрезки прямых (векторы); полилинии. полигоны, полигональные поверхности (рис. 1). Элемент «вершина» (vertex) — главный элемент описания, все другие являются производными. При использовании трехмерной декартовой системы координаты вершин определяются как (x,y,z). Каждый объект однозначно определяется координатами собственных вершин.

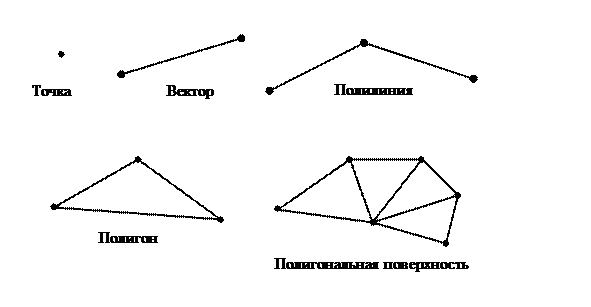


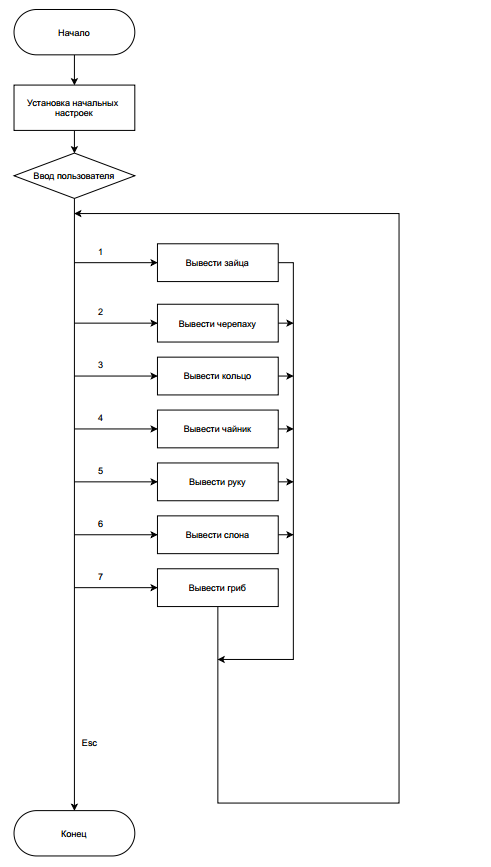
Рисунок 1 – Элементы описания пространственных объектов

Вершина может моделировать отдельный точечный объект, размер которого не имеет значения, а также может использоваться в качестве конечных точек для линейных объектов и полигонов. Двумя вершинами задается вектор. Несколько векторов составляют полилинию. Полилиния может моделировать отдельный линейный объект, толщина которого не учитывается, а также мо­жет представлять контур полигона. Полигон моделирует площадный объект Один полигон может описывать плоскую грань объемного объекта. Несколь­ко граней составляют объемный объект в виде полигональной поверхности – многогранник или незамкнутую поверхность (в литературе часто употребляется название «полигональная сетка»).

**Вывод:** В ходе данной лабораторной работы была реализована программа, с помощью которой можно выводить каркас векторной полигональной модели. Также познакомились с файловым расширением .off. В файле данного расширения содержится описание 3D-объектов.

Приложение А

Схемы алгоритмов



Приложение В

Листинг программы

|  |
| --- |
|  |

program lab8;

uses sysutils,ptcgraph,ptccrt;

type

Tcrd2=record

x,y:integer;

end;

Tcrd=array[0..2] of real;

Tconnection=array[0..2] of longint;

Tcrds=array of Tcrd;

Tconnections=array of Tconnection;

Tcrds2=array of Tcrd2;

var

f: text;

s: String;

info: Tconnection;

connections:Tconnections;

crds:Tcrds;

i,j:integer;

g,h:integer;

ch:char;

mn:integer=20;

crds2:Tcrds2;

a,b:integer;

mx:integer;

my:integer;

model:String;

function getInfo(s:String):Tconnection;

var i,j:integer;

tmpS:String;

tc:Tconnection;

begin

j:=1;

tmpS:='';

i:=0;

while i<3 do

begin

if(s[j]<>' ') and (j<=length(s)) then

begin

tmpS:=tmpS+s[j];

end

else

begin

//writeln(tmpS);

tc[i]:=StrToInt(tmpS);

//writeln(tc[i]);

tmpS:='';

Inc(i);

end;

Inc(j);

end;

Exit(tc);

end;

function getCon(s:String):Tconnection;

var i,j:integer;

tmpS:String;

tc:Tconnection;

begin

j:=3;

tmpS:='';

i:=0;

while i<3 do

begin

if(s[j]<>' ') and (j<=length(s)) then

begin

tmpS:=tmpS+s[j];

end

else

begin

tc[i]:=StrToInt(tmpS);

tmpS:='';

Inc(i);

end;

Inc(j);

end;

Exit(tc);

end;

function getCrd(s:String):Tcrd;

var i,j:integer;

tmpS:String;

tc:Tcrd;

begin

j:=1;

tmpS:='';

i:=0;

while i<3 do

begin

if (s[j]<>' ') and (j<=length(s)) then

begin

if (s[j]='.') then

begin

tmpS:=tmpS+',';

end

else

begin

tmpS:=tmpS+s[j];

end;

end

else

begin

tc[i]:=StrToFloat(tmpS);

tmpS:='';

Inc(i);

end;

Inc(j);

end;

Exit(tc);

end;

procedure GetCrds();

var i:longint;

begin

for i:=0 to info[0]-1 do

begin

readln(f,s);

crds[i]:=getCrd(s);

end;

end;

procedure GetConnetions();

var i:longint;

begin

for i:=0 to info[1]-1 do

begin

readln(f,s);

connections[i]:=getCon(s);

end;

end;

procedure Init;

begin

assign(f,model);

Reset (f);

readln(f,s);

readln(f,s);

info:=getInfo(s);

writeln(s);

setLength(crds,info[0]);

setLength(crds2, info[0]);

setLength(connections,info[1]);

GetCrds();

GetConnetions();

Close (f);

end;

procedure drawTriangle(c:Tconnection);

//var mx,my:integer;

begin

line(crds2[c[0]].x+mx,crds2[c[0]].y+my,crds2[c[1]].x+mx,crds2[c[1]].y+my);

line(crds2[c[1]].x+mx,crds2[c[1]].y+my,crds2[c[2]].x+mx,crds2[c[2]].y+my);

line(crds2[c[0]].x+mx,crds2[c[0]].y+my,crds2[c[2]].x+mx,crds2[c[2]].y+my);

end;

procedure drawFigure();

var i:longint;

begin

for i:=0 to info[1]-1 do

begin

drawTriangle(connections[i]);

end;

end;

procedure centralProjection();

var i:integer;

begin

ClearDevice;

for i:=0 to info[0]-1 do

begin

crds2[i].x:=round( crds[i][0]/(crds[i][2]/150+1)\*mn );

crds2[i].y:=round( crds[i][1]/(crds[i][2]/150+1)\*mn );

end;

end;

procedure Axonometric(p1,f1:real);

var p,f:real;

i:longint;

begin

p:=p1\*pi/180;

f:=f1\*pi/180;

for i:=0 to info[0]-1 do

begin

crds2[i].x:=round( mn\*( crds[i][0]\*cos(p)+crds[i][2]\*sin(p) ));

crds2[i].y:=round( mn\*(

crds[i][0]\*sin(p)\*sin(f)+

crds[i][1]\*cos(f)-

crds[i][2]\*sin(f)\*cos(p)

) );

end;

end;

procedure action(ch:char);

begin

case ch of

#49:begin

ClearDevice;

model:='models/bunny.off';

a:=0;

b:=180;

mn:=5000;

mx:=700;

my:=1000;

Init;

Axonometric(0,180);

drawFigure();

end;

#50:begin

ClearDevice;

model:='models/elephant.off';

a:=0;

b:=180;

mn:=15;

mx:=500;

my:=500;

Init;

Axonometric(0,180);

drawFigure();

end;

#51:begin

ClearDevice;

model:='models/hand.off';

a:=0;

b:=180;

mn:=70;

mx:=1000;

my:=-700;

Init;

Axonometric(0,180);

drawFigure();

end;

#52:begin

ClearDevice;

model:='models/ico.off';

a:=0;

b:=180;

mn:=100;

mx:=500;

my:=500;

Init;

Axonometric(0,180);

drawFigure();

end;

#53:begin

ClearDevice;

model:='models/mug.off';

a:=0;

b:=180;

mn:=100;

mx:=500;

my:=500;

Init;

Axonometric(0,180);

drawFigure();

end;

#54:begin

ClearDevice;

model:='models/mushroom.off';

a:=0;

b:=180;

mn:=200;

mx:=500;

my:=500;

Init;

Axonometric(0,180);

drawFigure();

end;

#55:begin

ClearDevice;

model:='models/teapot.off';

a:=0;

b:=180;

mn:=100;

mx:=500;

my:=500;

Init;

Axonometric(0,180);

drawFigure();

end;

#56:begin

ClearDevice;

model:='models/torus.off';

a:=0;

b:=180;

mn:=200;

mx:=500;

my:=500;

Init;

Axonometric(0,180);

drawFigure();

end;

#57:begin

ClearDevice;

model:='models/turtle.off';

a:=0;

b:=180;

mn:=8;

mx:=800;

my:=1000;

Init;

Axonometric(0,180);

drawFigure();

end;

#1:begin

a:=a+5;

ClearDevice;

Axonometric(a,b);

drawFigure();

end;

end;

end;

begin

//writeln(connections[19][2]);

g := detect;

initgraph(g,h,'');

setcolor($FFFFFF);

//centralProjection();

a:=0;

//b:=90;

repeat

ch:=ReadKey();

action(ch);

until ch=#27;

CloseGraph;

end.

Приложение С

Экранные формы

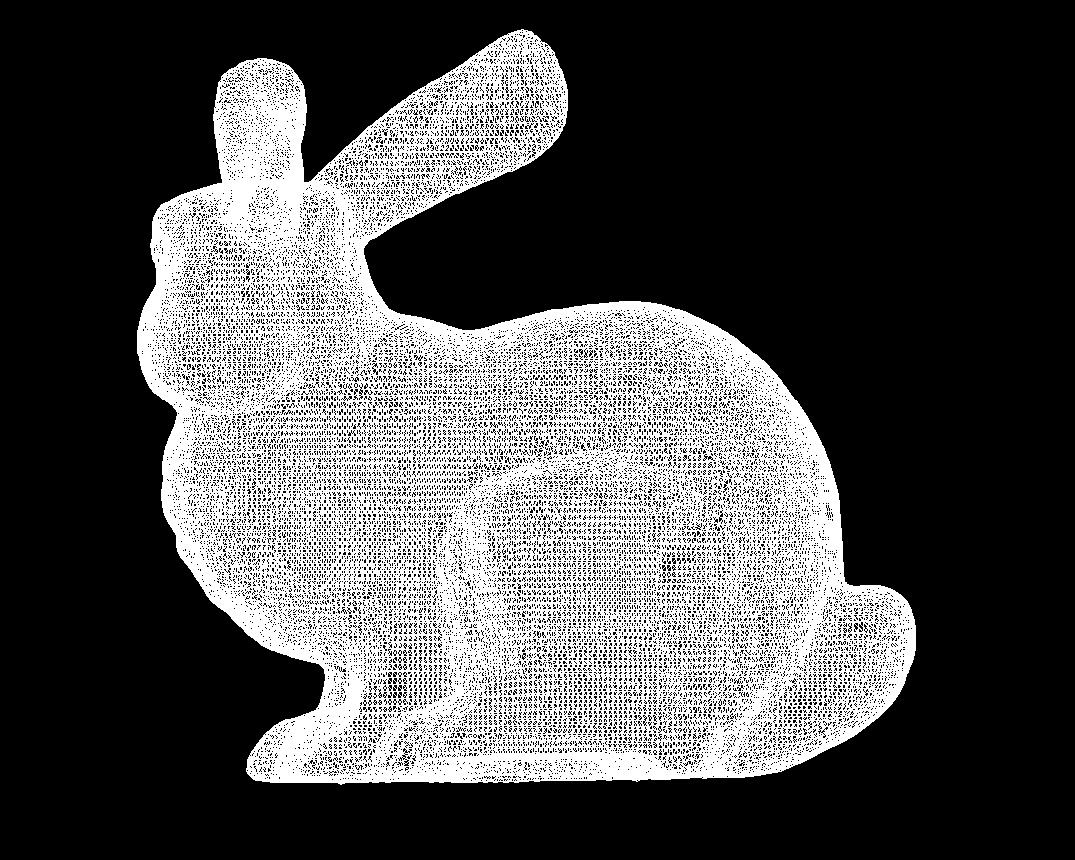


Рисунок 2 – Заяц

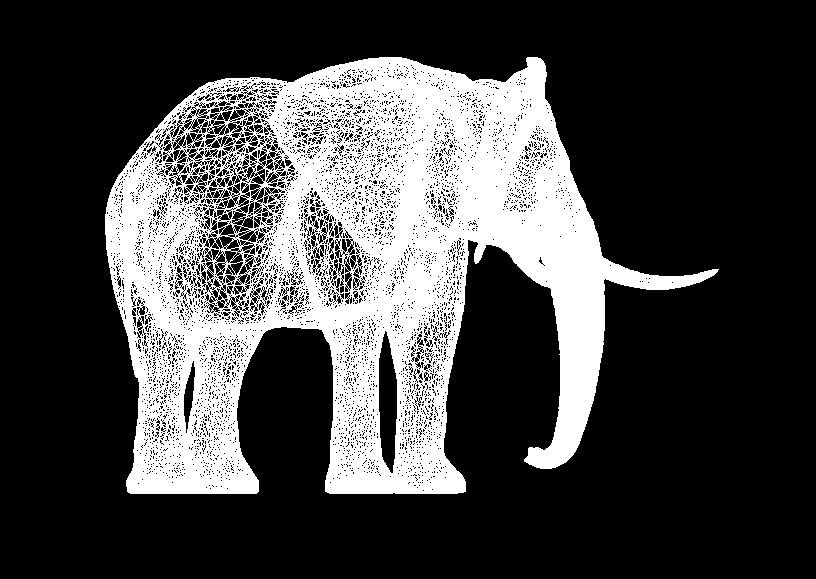


Рисунок 3 – Слон

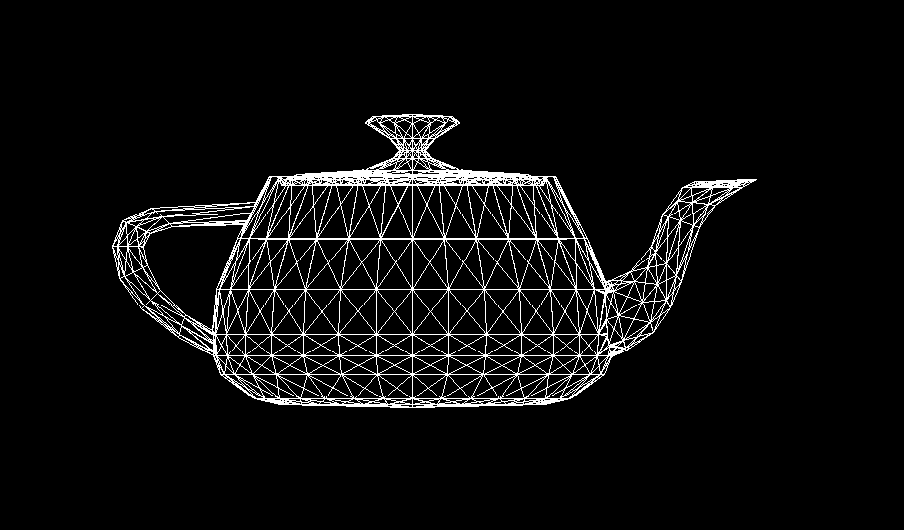


Рисунок 4 – Чайник