**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра САПР**

отчет

**по лабораторной работе №1**

**по дисциплине «Компьютерная графика»**

Тема: **Исследование математических методов представления и преобразования графических объектов на плоскости и в пространстве**

|  |  |  |
| --- | --- | --- |
| Студенты гр. 8362 |  | Ларионова Е.Е.  Матвеев Н.Д. |
| Преподаватель |  | Матвеева И. В. |

Санкт-Петербург

2021

**ЗАДАНИЕ**

Поворот плоского объекта относительно произвольной точки плоскости на заданный угол. Необходимо предусмотреть возможность редактирования положения точки

**РЕАЛИЗАЦИЯ ПРОГРАММЫ**

Программа реализована на языке программирования C++ с помощью библиотек Qt. Программа работает в графическом режиме.

В начале работы программа открывает два окна «MainWindow» и «Form». Окно «MainWindow» запрашивает координаты точки и угол поворота квадрата относительно заданной точки (Рисунок 1). Окно «Form» служит для отрисовки квадрата, согласно заданным координатам в предыдущем окне (Рисунок 2).

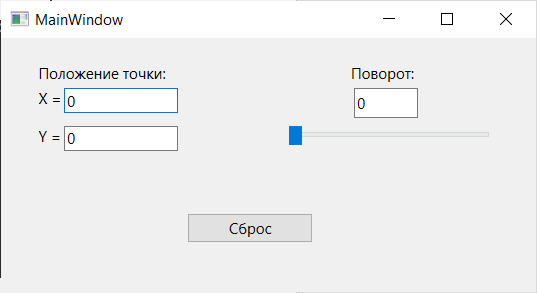


Рисунок 1 ­– Запрос координат точки и угла поворота

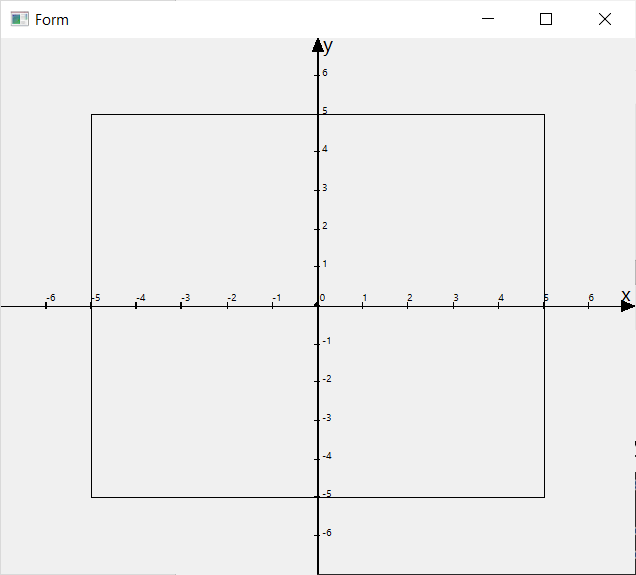


Рисунок 2 ­– Окно «Form»

После ввода координат и угла в окне «Form» появляется точка, с заданными параметрами. Также при изменении угла поворота, квадрат меняет свое положение (Рисунок 3 и Рисунок 4).

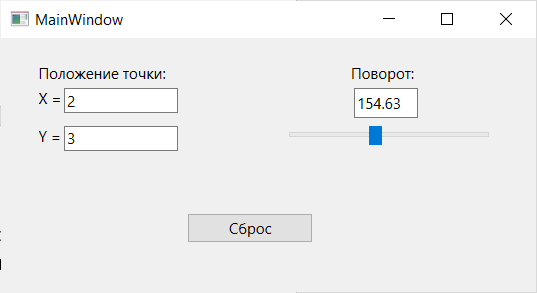


Рисунок 3 – Изменение координат точки и угла поворота

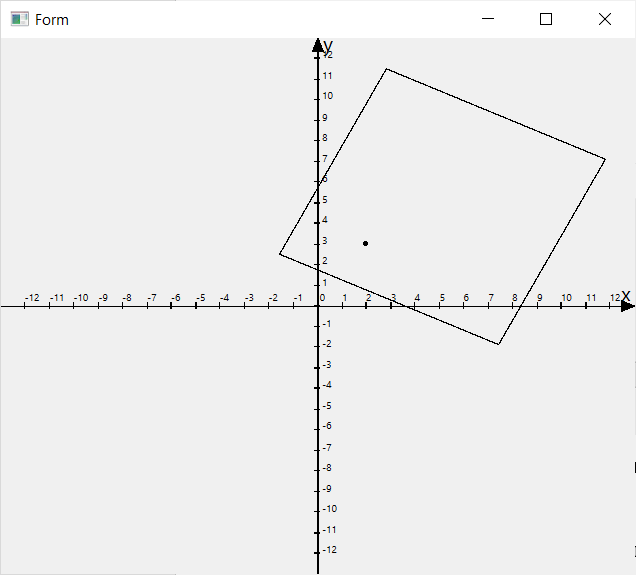


Рисунок 4 – Отображение повернутого квадрата

**ПРИЛОЖЕНИЕ 1. КОД ПРОГРАММЫ**

Файл Main.cpp

#include <application.h>

int main(int argc, char \*argv[])

{

Application a(argc, *argv*);

return a.exec();

}

Файл MainWindow.cpp

#include "mainwindow.h"

#include "ui\_mainwindow.h"

MainWindow::**MainWindow**(QWidget \*parent)

: QMainWindow(*parent*)

, ui(new Ui::MainWindow)

{

ui->setupUi(this);

this->setAttribute(Qt::WA\_DeleteOnClose);

}

MainWindow::~***MainWindow***()

{

delete ui;

}

void MainWindow::**res**()

{

on\_pushButton\_reset\_clicked();

}

Control\_State MainWindow::**get\_state**()

{

Control\_State tmp;

tmp.is\_reset = false;

tmp.point\_changed = false;

tmp.p\_x = ui->lineEdit\_x->text().toDouble();

tmp.p\_y = ui->lineEdit\_y->text().toDouble();

tmp.angle = ui->horizontalSlider->value()/100;

return tmp;

}

void MainWindow::**on\_pushButton\_reset\_clicked**()

{

ui->lineEdit\_x->setText(QString::number(0));

ui->lineEdit\_y->setText(QString::number(0));

ui->lineEdit\_angle->setText(QString::number(0));

Control\_State tmp = get\_state();

tmp.is\_reset = true;

emit update(tmp);

}

void MainWindow::**on\_lineEdit\_x\_textChanged**(const QString &arg1)

{

ui->lineEdit\_angle->setText(QString::number(0));

Control\_State tmp = get\_state();

tmp.point\_changed = true;

emit update(tmp);

}

void MainWindow::**on\_lineEdit\_y\_textChanged**(const QString &arg1)

{

ui->lineEdit\_angle->setText(QString::number(0));

Control\_State tmp = get\_state();

tmp.point\_changed = true;

emit update(tmp);}

void MainWindow::**on\_horizontalSlider\_valueChanged**(int value)

{

ui->lineEdit\_angle->setText(QString::number(value\*1.0/100));

emit update(get\_state());

}

void MainWindow::**on\_lineEdit\_angle\_textEdited**(const QString &arg1)

{

ui->horizontalSlider->setValue(arg1.toDouble()\*100);

}

Файл Drawwindow.cpp

#include "drawwindow.h"

#include "ui\_drawwindow.h"

DrawWindow::**DrawWindow**(QWidget \*parent) :

QWidget(*parent*),

ui(new Ui::DrawWindow)

{

ui->setupUi(this);

this->setAttribute(Qt::WA\_DeleteOnClose);

}

DrawWindow::~***DrawWindow***()

{

delete ui;

}

void DrawWindow::**recive\_state**(State new\_state)

{

draw\_state = new\_state;

repaint();

}

void DrawWindow::***paintEvent*** (QPaintEvent \*event)

{

Q\_UNUSED(event);

QPainter painter(this);

QFont font;

qreal cw = 0.5\*rect().width();

qreal ch = 0.5\*rect().height();

qreal cr = 0.015\*(cw>ch?ch:cw);

qreal ca = 0.05\*(cw>ch?ch:cw);

qreal caa = ca \* 0.4;

qreal cf = 0.06 \*(cw>ch?ch:cw);

int num = draw\_state.max + 2;

qreal cnw = cw / num;

qreal cnh = ch / num;

qreal can = ca \*0.2;

qreal cnf = cf \* 0.5;

font.setPointSize(cf);

painter.setFont(font);

painter.setPen(QPen(Qt::black));

painter.drawLine(QLineF(0,ch,2\*cw,ch));

painter.drawLine(QLineF(cw,0,cw,2\*ch));

QPointF arr1[3],arr2[3];

arr1[0] = QPointF(cw,0);

arr1[1] = QPointF(cw-caa,ca);

arr1[2] = QPointF(cw+caa,ca);

arr2[0] = QPointF(2\*cw,ch);

arr2[1] = QPointF(2\*cw-ca,ch+caa);

arr2[2] = QPointF(2\*cw-ca,ch-caa);

painter.setBrush(QBrush(Qt::black));

painter.drawPolygon(arr1,3);

painter.drawPolygon(arr2,3);

painter.drawText(arr1[2],"y");

painter.drawText(arr2[2],"x");

font.setPointSize(cnf);

painter.setFont(font);

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

{

painter.drawLine(cw+i\*cnw,ch+can,cw+i\*cnw,ch-can);

painter.drawLine(cw-i\*cnw,ch+can,cw-i\*cnw,ch-can);

painter.drawLine(cw-can,ch+i\*cnh,cw+can,ch+i\*cnh);

painter.drawLine(cw-can,ch-i\*cnh,cw+can,ch-i\*cnh);

if (i == 0)

{

painter.drawText(cw+i\*cnw+can,ch-2\*can,QString::number(i));

}

else

{

painter.drawText(cw+i\*cnw,ch-2\*can,QString::number(i));

painter.drawText(cw-i\*cnw,ch-2\*can,QString::number(-i));

painter.drawText(cw+2\*can,ch+i\*cnh,QString::number(-i));

painter.drawText(cw+2\*can,ch-i\*cnh,QString::number(i));

}

}

painter.drawEllipse(cw + draw\_state.p.x()\*cnw - 0.5\*cr,ch - draw\_state.p.y()\*cnh - 0.5\*cr,cr,cr);

painter.setBrush(QBrush(Qt::transparent));

QPointF tmp[4];

for (int i = 0; i<4; i++)

{

tmp[i].setX(cw + draw\_state.f[i].x()\*cnw);

tmp[i].setY(ch - draw\_state.f[i].y()\*cnh);

}

painter.drawPolygon(tmp,4);

}

Файл Application.cpp

#include "drawwindow.h"

#include "ui\_drawwindow.h"

DrawWindow::**DrawWindow**(QWidget \*parent) :

QWidget(*parent*),

ui(new Ui::DrawWindow)

{

ui->setupUi(this);

this->setAttribute(Qt::WA\_DeleteOnClose);

}

DrawWindow::~***DrawWindow***()

{

delete ui;

}

void DrawWindow::**recive\_state**(State new\_state)

{

draw\_state = new\_state;

repaint();

}

void DrawWindow::***paintEvent*** (QPaintEvent \*event)

{

Q\_UNUSED(event);

QPainter painter(this);

QFont font;

qreal cw = 0.5\*rect().width();

qreal ch = 0.5\*rect().height();

qreal cr = 0.015\*(cw>ch?ch:cw);

qreal ca = 0.05\*(cw>ch?ch:cw);

qreal caa = ca \* 0.4;

qreal cf = 0.06 \*(cw>ch?ch:cw);

int num = draw\_state.max + 2;

qreal cnw = cw / num;

qreal cnh = ch / num;

qreal can = ca \*0.2;

qreal cnf = cf \* 0.5;

font.setPointSize(cf);

painter.setFont(font);

painter.setPen(QPen(Qt::black));

painter.drawLine(QLineF(0,ch,2\*cw,ch));

painter.drawLine(QLineF(cw,0,cw,2\*ch));

QPointF arr1[3],arr2[3];

arr1[0] = QPointF(cw,0);

arr1[1] = QPointF(cw-caa,ca);

arr1[2] = QPointF(cw+caa,ca);

arr2[0] = QPointF(2\*cw,ch);

arr2[1] = QPointF(2\*cw-ca,ch+caa);

arr2[2] = QPointF(2\*cw-ca,ch-caa);

painter.setBrush(QBrush(Qt::black));

painter.drawPolygon(arr1,3);

painter.drawPolygon(arr2,3);

painter.drawText(arr1[2],"y");

painter.drawText(arr2[2],"x");

font.setPointSize(cnf);

painter.setFont(font);

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

{

painter.drawLine(cw+i\*cnw,ch+can,cw+i\*cnw,ch-can);

painter.drawLine(cw-i\*cnw,ch+can,cw-i\*cnw,ch-can);

painter.drawLine(cw-can,ch+i\*cnh,cw+can,ch+i\*cnh);

painter.drawLine(cw-can,ch-i\*cnh,cw+can,ch-i\*cnh);

if (i == 0)

{

painter.drawText(cw+i\*cnw+can,ch-2\*can,QString::number(i));

}

else

{

painter.drawText(cw+i\*cnw,ch-2\*can,QString::number(i));

painter.drawText(cw-i\*cnw,ch-2\*can,QString::number(-i));

painter.drawText(cw+2\*can,ch+i\*cnh,QString::number(-i));

painter.drawText(cw+2\*can,ch-i\*cnh,QString::number(i));

}

}

painter.drawEllipse(cw + draw\_state.p.x()\*cnw - 0.5\*cr,ch - draw\_state.p.y()\*cnh - 0.5\*cr,cr,cr);

painter.setBrush(QBrush(Qt::transparent));

QPointF tmp[4];

for (int i = 0; i<4; i++)

{

tmp[i].setX(cw + draw\_state.f[i].x()\*cnw);

tmp[i].setY(ch - draw\_state.f[i].y()\*cnh);

}

painter.drawPolygon(tmp,4);

}