**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра ТВ**

**отчЁт**

**по лабораторной работе № 1**

**по дисциплине «Компьютерный синтез трехмерных изображений»**

**Тема: ИССЛЕДОВАНИЕ ГЕОМЕТРИЧЕСКИХ ПРЕОБРАЗОВАНИЙ НА ПЛОСКОСТИ (2D ПРЕОБРАЗОВАНИЙ)**

**Вариант 4**

|  |  |  |
| --- | --- | --- |
| Студент гр. 9105 |  | Шаривзянов Д. Р. |
| Преподаватель |  | Сирый Р. С. |

Санкт-Петербург

2024

ИССЛЕДОВАНИЕ ГЕОМЕТРИЧЕСКИХ ПРЕОБРАЗОВАНИЙ НА ПЛОСКОСТИ (2D ПРЕОБРАЗОВАНИЙ)

**Цель работы:**

Целью лабораторной работы является знакомство с аффинными преобразованиями в 2D пространстве и средствами отображения фигур с помощью C++ Window Forms.

**Код программы:**

#include<math.h>

const int vert\_num = 4; // количество вершин фигуры

const int dimension = 3;// число столбцов матрицы однородных координат

//3-для двумерной графики, 4 для трехмерной

double hmg\_p[vert\_num][dimension] = { 0 }; // матрица однородных координат // фигуры

int dek\_p[vert\_num][dimension - 1] = { 0 }; // матрица экранных координат //фигуры

double matrix\_T[dimension][dimension] = { 0 };// матрица преобразования

private: System::Void matrix\_mult(int number\_of\_vertex\_A, double A[][dimension], double B[][dimension], double C[][dimension]) {

int i, j, k;

for (i = 0; i < number\_of\_vertex\_A; i++)

for (j = 0; j < dimension; j++) {

C[i][j] = 0;

for (k = 0; k < dimension; k++)

C[i][j] += (A[i][k] \* B[k][j]);

}

}

private: System::Void hmg2dek(int number\_of\_vertex\_A, double HMG[][dimension], int DEK[][dimension - 1]) {

for (int i = 0; i < number\_of\_vertex\_A; i++) {

double a = HMG[i][dimension - 1];

if (a != 0) {

DEK[i][0] = HMG[i][0] / a;

DEK[i][1] = HMG[i][1] / a;

}

else {

System::Windows::Forms::MessageBox::Show("division by zero is impossible! Check the coordinates of the shape and the transformation matrix");

break;

}

}

}

private: System::Void make\_trans\_mat(double x, double y, double result\_mat[dimension][dimension]) {

result\_mat[0][0] = 1;

result\_mat[0][1] = 0;

result\_mat[0][2] = 0;

result\_mat[1][0] = 0;

result\_mat[1][1] = 1;

result\_mat[1][2] = 0;

result\_mat[2][0] = x;

result\_mat[2][1] = y;

result\_mat[2][2] = 1;

}

private: System::Void btn\_draw\_Click(System::Object^ sender, System::EventArgs^ e) {

// считывание введенных пользователем координат в матрицу. Использует //-ся функция Convert::ToInt32 для преобразования типов данных

hmg\_p[0][0] = Convert::ToInt32(textBox\_x1->Text);

textBox\_y1->Text = textBox\_x1->Text;

hmg\_p[0][1] = Convert::ToInt32(textBox\_y1->Text);

hmg\_p[0][2] = Convert::ToInt32(textBox\_z1->Text);

textBox\_x2->Text = textBox\_x4->Text;

hmg\_p[1][0] = Convert::ToInt32(textBox\_x2->Text);

textBox\_y2->Text = textBox\_y1->Text;

hmg\_p[1][1] = Convert::ToInt32(textBox\_y2->Text);

hmg\_p[1][2] = Convert::ToInt32(textBox\_z2->Text);

textBox\_y4->Text = textBox\_x4->Text;

hmg\_p[3][0] = Convert::ToInt32(textBox\_x4->Text);

hmg\_p[3][1] = Convert::ToInt32(textBox\_y4->Text);

hmg\_p[3][2] = Convert::ToInt32(textBox\_z4->Text);

textBox\_x3->Text = textBox\_x1->Text;

hmg\_p[2][0] = Convert::ToInt32(textBox\_x3->Text);

textBox\_y3->Text = textBox\_y4->Text;

hmg\_p[2][1] = Convert::ToInt32(textBox\_y3->Text);

hmg\_p[2][2] = Convert::ToInt32(textBox\_z3->Text);

hmg2dek(vert\_num, hmg\_p, dek\_p); // вызов функции перевода координат из однородных в экранные

pictureBox1->Refresh(); // вызов перерисовки элемента pictureBox.

}

private: System::Void pictureBox1\_Paint(System::Object^ sender, System::Windows::Forms::PaintEventArgs^ e) {

//int side = x2 - x1;

//

//e->Graphics->DrawRectangle(System::Drawing::Pens::Red, x1, y1, side+10, side);

int x1 = dek\_p[0][0];

int y1 = dek\_p[0][1];

//int z1 = dek\_p[0][2];

int x2 = dek\_p[1][0];

int y2 = dek\_p[1][1];

//int z2 = dek\_p[1][2];

int x3 = dek\_p[2][0];

int y3 = dek\_p[2][1];

//int z3 = dek\_p[2][2];

int x4 = dek\_p[3][0];

int y4 = dek\_p[3][1];

//int z4 = dek\_p[3][2];

e->Graphics->DrawLine(System::Drawing::Pens::Red, x1, y1, x2, y2);

e->Graphics->DrawLine(System::Drawing::Pens::Red, x1, y1, x3, y3);

e->Graphics->DrawLine(System::Drawing::Pens::Red, x2, y2, x4, y4);

e->Graphics->DrawLine(System::Drawing::Pens::Red, x3, y3, x4, y4);

}

private: System::Void button\_transfer\_Click(System::Object^ sender, System::EventArgs^ e) {

double matrix\_T\_loc1[dimension][dimension] = { 0 };// матрица преобразования

textBox\_transx3->Text = textBox\_transx->Text;

textBox\_transy3->Text = textBox\_transy->Text;

int cx = Convert::ToDouble(textBox\_transx3->Text);

int cy = Convert::ToDouble(textBox\_transy3->Text);

make\_trans\_mat(cx, cy, matrix\_T\_loc1);

double result\_mat1[vert\_num][dimension] = { 0 };

matrix\_mult(vert\_num, hmg\_p, matrix\_T\_loc1, result\_mat1);

hmg2dek(vert\_num, result\_mat1, dek\_p); // вызов функции перевода координат из однородных в экранные

for (int i = 0; i < vert\_num; i++)

for (int j = 0; j < dimension; j++)

hmg\_p[i][j] = result\_mat1[i][j];

pictureBox1->Refresh(); // вызов перерисовки элемента pictureBox.

}

private: System::Void button\_scale\_Click(System::Object^ sender, System::EventArgs^ e) {

double matrix\_T\_loc1[dimension][dimension] = { 0 };// матрица преобразования

double matrix\_T\_loc2[dimension][dimension] = { 0 };// матрица преобразования

double matrix\_T\_loc3[dimension][dimension] = { 0 };// матрица преобразования

int cx = (dek\_p[0][0] + dek\_p[1][0] + dek\_p[2][0] + dek\_p[3][0]) / 4;

int cy = (dek\_p[0][1] + dek\_p[1][1] + dek\_p[2][1] + dek\_p[3][1]) / 4;

make\_trans\_mat(-cx, -cy, matrix\_T\_loc1);

make\_trans\_mat(cx, cy, matrix\_T\_loc3);

textBox\_scalex1->Text = textBox\_scalex->Text;

matrix\_T\_loc2[0][0] = Convert::ToDouble(textBox\_scalex1->Text);

matrix\_T\_loc2[0][1] = Convert::ToDouble(textBox\_scaley1->Text);

matrix\_T\_loc2[0][2] = Convert::ToDouble(textBox\_scalez1->Text);

matrix\_T\_loc2[1][0] = Convert::ToDouble(textBox\_scalex2->Text);

textBox\_scaley2->Text = textBox\_scaley->Text;

matrix\_T\_loc2[1][1] = Convert::ToDouble(textBox\_scaley2->Text);

matrix\_T\_loc2[1][2] = Convert::ToDouble(textBox\_scalez2->Text);

matrix\_T\_loc2[2][0] = Convert::ToDouble(textBox\_scalex3->Text);

matrix\_T\_loc2[2][1] = Convert::ToDouble(textBox\_scaley3->Text);

matrix\_T\_loc2[2][2] = Convert::ToDouble(textBox\_scalez3->Text);

double result\_mat1[vert\_num][dimension] = { 0 };

double result\_mat2[vert\_num][dimension] = { 0 };

matrix\_mult(vert\_num, hmg\_p, matrix\_T\_loc1, result\_mat1);

matrix\_mult(vert\_num, result\_mat1, matrix\_T\_loc2, result\_mat2);

matrix\_mult(vert\_num, result\_mat2, matrix\_T\_loc3, result\_mat1);

hmg2dek(vert\_num, result\_mat1, dek\_p); // вызов функции перевода координат из однородных в экранные

for (int i = 0; i < vert\_num; i++)

for (int j = 0; j < dimension; j++)

hmg\_p[i][j] = result\_mat1[i][j];

pictureBox1->Refresh(); // вызов перерисовки элемента pictureBox.

}

private: System::Void button\_rotate\_Click(System::Object^ sender, System::EventArgs^ e) {

double matrix\_T\_loc1[dimension][dimension] = { 0 };// матрица преобразования

double matrix\_T\_loc2[dimension][dimension] = { 0 };// матрица преобразования

double matrix\_T\_loc3[dimension][dimension] = { 0 };// матрица преобразования

int cx = (dek\_p[0][0] + dek\_p[1][0] + dek\_p[2][0] + dek\_p[3][0]) / 4;

int cy = (dek\_p[0][1] + dek\_p[1][1] + dek\_p[2][1] + dek\_p[3][1]) / 4;

make\_trans\_mat(-cx, -cy, matrix\_T\_loc1);

make\_trans\_mat(cx, cy, matrix\_T\_loc3);

double alpha = (Convert::ToDouble(textBox\_rotatealpha->Text) \* (acos(-1))) / 180;

textBox\_rotatex1->Text = (cos(alpha)).ToString();

matrix\_T\_loc2[0][0] = Convert::ToDouble(textBox\_rotatex1->Text);

textBox\_rotatey1->Text = (-sin(alpha)).ToString();

matrix\_T\_loc2[0][1] = Convert::ToDouble(textBox\_rotatey1->Text);

matrix\_T\_loc2[0][2] = Convert::ToDouble(textBox\_rotatez1->Text);

textBox\_rotatex2->Text = (sin(alpha)).ToString();

matrix\_T\_loc2[1][0] = Convert::ToDouble(textBox\_rotatex2->Text);

textBox\_rotatey2->Text = (cos(alpha)).ToString();

matrix\_T\_loc2[1][1] = Convert::ToDouble(textBox\_rotatey2->Text);

matrix\_T\_loc2[1][2] = Convert::ToDouble(textBox\_rotatez2->Text);

matrix\_T\_loc2[2][0] = Convert::ToDouble(textBox\_rotatex3->Text);

matrix\_T\_loc2[2][1] = Convert::ToDouble(textBox\_rotatey3->Text);

matrix\_T\_loc2[2][2] = Convert::ToDouble(textBox\_rotatez3->Text);

double result\_mat1[vert\_num][dimension] = { 0 };

double result\_mat2[vert\_num][dimension] = { 0 };

matrix\_mult(vert\_num, hmg\_p, matrix\_T\_loc1, result\_mat1);

matrix\_mult(vert\_num, result\_mat1, matrix\_T\_loc2, result\_mat2);

matrix\_mult(vert\_num, result\_mat2, matrix\_T\_loc3, result\_mat1);

hmg2dek(vert\_num, result\_mat1, dek\_p); // вызов функции перевода координат из однородных в экранные

for (int i = 0; i < vert\_num; i++)

for (int j = 0; j < dimension; j++)

hmg\_p[i][j] = result\_mat1[i][j];

pictureBox1->Refresh(); // вызов перерисовки элемента pictureBox.

}

private: System::Void button\_mirror\_Click(System::Object^ sender, System::EventArgs^ e) {

double matrix\_T\_loc1[dimension][dimension] = { 0 };// матрица преобразования

double matrix\_T\_loc2[dimension][dimension] = { 0 };// матрица преобразования

double matrix\_T\_loc3[dimension][dimension] = { 0 };// матрица преобразования

int vertex = Convert::ToInt32(numericUpDown1->Value);

int cx = dek\_p[vertex - 1][0];

int cy = dek\_p[vertex - 1][1];

make\_trans\_mat(-cx, -cy, matrix\_T\_loc1);

make\_trans\_mat(cx, cy, matrix\_T\_loc3);

double alpha = acos(-1);

// Set up the rotation matrix

matrix\_T\_loc2[0][0] = cos(alpha);

matrix\_T\_loc2[0][1] = -sin(alpha);

matrix\_T\_loc2[0][2] = 0;

matrix\_T\_loc2[1][0] = sin(alpha);

matrix\_T\_loc2[1][1] = cos(alpha);

matrix\_T\_loc2[1][2] = 0;

matrix\_T\_loc2[2][0] = 0;

matrix\_T\_loc2[2][1] = 0;

matrix\_T\_loc2[2][2] = 1;

double result\_mat1[vert\_num][dimension] = { 0 };

double result\_mat2[vert\_num][dimension] = { 0 };

matrix\_mult(vert\_num, hmg\_p, matrix\_T\_loc1, result\_mat1);

matrix\_mult(vert\_num, result\_mat1, matrix\_T\_loc2, result\_mat2);

matrix\_mult(vert\_num, result\_mat2, matrix\_T\_loc3, result\_mat1);

hmg2dek(vert\_num, result\_mat1, dek\_p); // вызов функции перевода координат из однородных в экранные

for (int i = 0; i < vert\_num; i++)

for (int j = 0; j < dimension; j++)

hmg\_p[i][j] = result\_mat1[i][j];

pictureBox1->Refresh(); // вызов перерисовки элемента pictureBox.

}

1. Основные преобразования

|  |  |  |
| --- | --- | --- |
| Вариант | Вид многоугольника | Вид преобразования |
| 4 | Квадрат | Зеркальное отображение многоугольника относительно одной из его вершин |

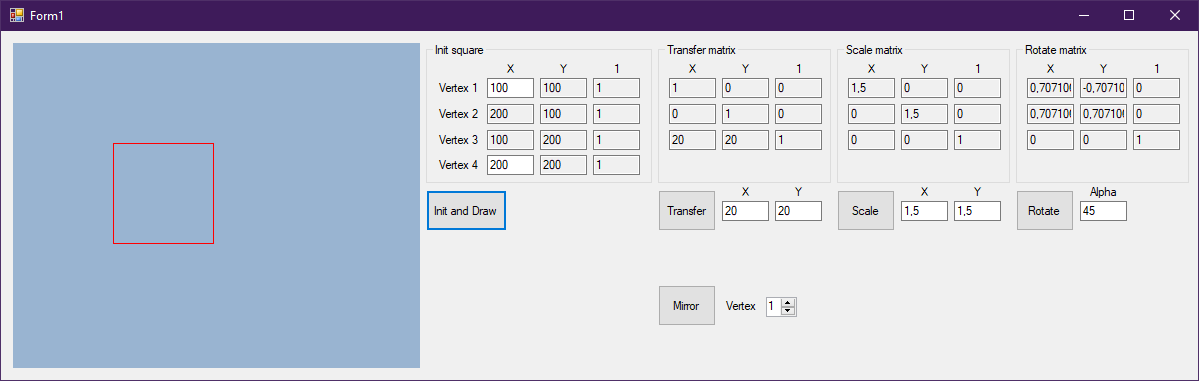
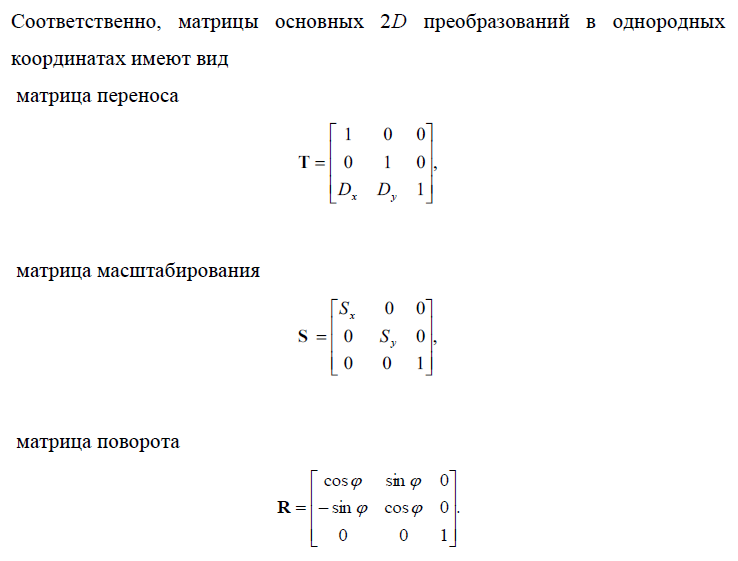


Рис 1. Основная форма программы.



1. Специфичное преобразование

Зеркальное отображение квадрата относительно одной из его вершин – это его поворот на 180 градусов относительно этой вершины. Соответственно, для данного преобразования необходимо 3 основных:

1. Перемещение квадрата в положение, где одна из его вершин в центре координат: перемещение 4-х вершин на отрицательные координаты выбранной вершины
2. Поворот на 180 градусов
3. Обратное преобразование перемещения

**Выводы:**

Были изучены аффинные преобразования перемещения, поворота, масштабирования, а также преобразование отражения по одной из вершин. В ходе работы было замечено, что при повороте фигуры на 360 градусов (в исходное положение) происходит незначительное смещение её исходного положения из-за многократного округления чисел с плавающей точкой при переводе координат из однородных в экранные.