**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.**

|  |  |  |
| --- | --- | --- |
| Студент гр. 9301 |  | Иванов М. В. |
| Студент гр. 9301 |  | Пронин В. В. |

|  |  |  |
| --- | --- | --- |
| Студентка гр. 9302 |  | Синицкая В. А. |
| Преподаватель |  | Матвеева И. В. |

Санкт-Петербург

2022

# Отображение плоского объекта (ломаной линии, которая может составлять плоскую фигуру) относительно осей координат с изменением или без изменения масштаба изображения.

Минимальный плоский объект — точка. Плоский объект — любая фигура, линия, набор точек, их комбинация, умещающиеся в пределах двумерного пространства. В данной работе под плоским объектом будет пониматься набор точек, соединенных прямой линией в том порядке, в котором они следуют. Таким образом, соединив точки A, B, C, A, можно получить треугольник или прямую линию, или точку, если все три точки совпадают.

# Краткие теоретические сведения

Для отражения точки относительно оси Ox, используется матрица:

где (x’, y’) — координаты отраженной точки, (x, y) — координаты изначальной точки.

Матрицу будем называть матрицей преобразования. Если матрицу преобразования изменить на , точка будет отражаться уже относительно оси Oy.

Для масштабирования объекта, требуется найти его центр, переместить объект в начало координат, отмасштабировать его и вернуть его центр на изначальную позицию.

Для переноса точки используются однородное координаты, то есть пара координат (x, y) превращается в тройку (x, y, 1).

Матрица переноса имеет вид: . То есть, чтобы сдвинуть центр (x, y) в (x’, y’):

Нужно (x, y) умножить на указанную матрицу 3х3, а затем взять первые две координаты получившегося трехкоординатного вектора, таким образом возвращаясь от однородных координат к обычным.

Центр фигур считается по формуле центров тяжестей: считается, что все точки имеют вес, при том одинаковый, а линии ничего не весят.

Формула для нахождения центра имеет вид:

где () — координаты центра тяжести, — координаты i точки, где i изменяется от 1 до N, то есть точек всего N.

# Описание программы

Работа выполнена на языке C# при использовании Windows Forms в среде Visual Studio 2019.

Координаты записываются в виде строки.

По умолчанию задается относительный центр O(0, 0), где (0, 0) — вообще говоря, относительные, так как настоящий центр для элемента pictureBox, в котором происходит отрисовка, находится в левом верхнем углу.

Для хранения плоского объекта в программе был создан класс FlatOblject, листинг которого представлен ниже. Объект хранит в себе ручку, которой должен отрисовываться объект, радиус рисуемых точек и сами точки в том порядке, в котором они должны быть соединены.

1. public FlatObject(Pen p, List<Point> dots, uint dotR = defaultDorRadius)
2. {
3. orderedDots = dots; // coordinates are in real Oxy and in pixels
4. pen = p;
5. dotRadius = dotR;
6. }
7. public void addDot(Point point) { orderedDots.Add(point); }
8. public void setColor(Color cl) { pen.Color = cl; }
9. public void setDotRadius(uint R) { dotRadius = R; }
10. public List<Point> getDots() { return new List<Point>(orderedDots); }
11. public uint getDotRadius() { return dotRadius; }
12. public Color getColor() { return pen.Color; }
13. public void setPen(Pen p) { pen = p; }
14. public Pen getPen() { return pen; }
16. public void draw(Pen p, Graphics graphics)
17. {
18. List<Point> dots = getDots();
19. uint r = getDotRadius();
20. setPen(p);
21. int dotsSize = dots.Count();
22. for (int dotIndex = 0; dotIndex < dotsSize; dotIndex++)
23. {
24. graphics.DrawEllipse(pen, dots[dotIndex].X - r / 2, dots[dotIndex].Y - r / 2, r, r);
25. if (dotIndex + 1 < dotsSize)
26. graphics.DrawLine(pen, dots[dotIndex], dots[dotIndex + 1]);
27. }
28. }
29. }

Матрица представляется двумерным массивом целых чисел. Была написана функция для перемножения матриц:

1. static int[,] multiplyMatrix(int[,] a, int[,] b)
2. {
3. if (a.GetLength(1) != b.GetLength(0)) throw new Exception("Матрицы нельзя перемножить");
4. int[,] r = new int[a.GetLength(0), b.GetLength(1)];
5. for (int i = 0; i < a.GetLength(0); i++)
6. {
7. for (int j = 0; j < b.GetLength(1); j++)
8. {
9. for (int k = 0; k < b.GetLength(0); k++)
10. {
11. r[i, j] += a[i, k] \* b[k, j];
12. }
13. }
14. }
15. return r;
16. }

Так же были написаны функции преобразования координат из той системы, которая используется pictureBox и при рисовании в ту, которую наблюдает пользователь:

1. private Point realToO(Point point, Point O) // returns point's coordinates in coordinate system centered at point O
2. {
3. return new Point(point.X - O.X, O.Y - point.Y);
4. }
6. private Point oToReal(Point point, Point O) // translates coordinates in coordinate system centered at point O to real
7. {
8. return new Point(point.X + O.X, O.Y - point.Y);
9. }

Была написана функция для переноса точки из системы координат с центром в точке O в систему координат с центром в точке newO:

1. static Point moveOToNewO(Point point, Point O, Point newO) // returns point in O-coords in newO-coords
2. {
3. int[,] dot = new int[1, 3] { { point.X, point.Y, 1 } };
4. int[,] matrixMoveToO = new int[3, 3] { { 1, 0, 0 }, { 0, 1, 0 }, { O.X - newO.X, O.Y - newO.Y, 1 } };
5. int[,] newDot = multiplyMatrix(dot, matrixMoveToO);
6. return new Point(newDot[0, 0], newDot[0, 1]);
7. }

На основе этой функции была написана функция переноса плоского объекта:

1. private FlatObject moveObjectToNewO(FlatObject obj, Point oldO, Point newO)
2. {
3. List<Point> newDots = new List<Point>();
4. foreach(Point dot in obj.getDots())
5. {
6. newDots.Add(moveOToNewO(dot, oldO, newO));
7. }
8. return new FlatObject(obj.getPen(), newDots);
9. }

Была написана функция, выдающая преобразованный объект из данного посредством влияния на него матрицы (умножения на нее справа):

1. FlatObject matrixImpact(FlatObject init, int[,] matrix) // returns changed flat object
2. {
3. List<Point> newPoints = init.getDots();
4. for (int i = 0; i < newPoints.Count; i++)
5. {
6. newPoints[i] = realToO(newPoints[i], O);
7. int[,] dot = new int[1, 2] { { newPoints[i].X, newPoints[i].Y} };
8. dot = multiplyMatrix(dot, matrix);
9. newPoints[i] = new Point(dot[0, 0], dot[0, 1]);
10. newPoints[i] = oToReal(newPoints[i], O);
11. }
12. return new FlatObject(penChangedObject, newPoints);
13. }

Была также написана функция масштабирования плоского объекта с заданным коэффициентом:

1. FlatObject scaleObjectWith (FlatObject obj, double scaleParam = defaultScalingFactor)
2. {
3. // find gravity center to move it to O and scale and then return the object back
4. int[,] gravityCenter = new int[1, 2] { { 0, 0 } };
5. List<Point> dots = new List<Point>(obj.getDots());
6. foreach (Point point in dots)
7. {
8. gravityCenter[0, 0] += point.X;
9. gravityCenter[0, 1] += point.Y;
10. }
11. gravityCenter[0, 0] /= dots.Count; // loss
12. gravityCenter[0, 1] /= dots.Count; // loss
13. Point gravityCenterPoint = new Point(gravityCenter[0, 0], gravityCenter[0, 1]);
14. Point realCenter = new Point(0, 0);
16. for (int i = 0; i < dots.Count; i++)
17. {
18. dots[i] = moveOToNewO(dots[i], realCenter, gravityCenterPoint);
19. int x = (int)Math.Round(dots[i].X \* scaleParam);
20. int y = (int)Math.Round(dots[i].Y \* scaleParam);
21. Point currentPoint = new Point(x, y); // loss
22. dots[i] = moveOToNewO(currentPoint, gravityCenterPoint, realCenter);
23. }
24. return new FlatObject(penChangedObject, dots);
25. }

Для конечной отрисовки хранятся объекты O, Ox, Oy, initialObject, changedObject, которые представляют из себя начало координат, оси Ox и Oy, наблюдаемые пользователем, изначальный объект и объект, к которому были применены некоторые выбранные пользователем настройки соответственно.

В приложении А приведены ссылки на видео-демонстрацию работы программы и полный код.

# Выводы

1) Для хранения координат важно определиться, в каком виде хранятся координаты: в строчку или в столбец.

2) Для масштабирования объекта необходимо найти его центр и перенести объект в настоящее, абсолютное начало координат, чтобы все координаты масштабировались пропорционально, так как на самом деле масштабируются радиус-векторы точек объекта, и перенос его центра в начало координат помогает избавиться от ненужного сдвига.

3) С помощью матриц определенного вида можно переносить и отражать объекты (точки) выполнив одно действие — умножение на матрицу.

# Приложение А

Видео-демонстрация работы программы:

<https://drive.google.com/file/d/1cbIjnKLMv0T0ZJHSU07Ir3LCObj48E_5/view?usp=sharing>

Полный код программы: