Министерство науки и высшего образования РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Курский государственный университет»

Кафедра программного
обеспечения и администрирования
информационных систем
Направление подготовки
математическое обеспечение
и администрирование
информационных систем
Форма обучения очная

Отчёт

по лабораторной работе №2

дисциплина «Компьютерная графика»

Выполнил:
студент группы 313.1 Андреенко А.Д.

Проверил:
доц. кафедры ПОиАИС Сухотерин Е. А.

Курск, 2021

Цель работы: Изучение математических методов аффинных преобразований на плоскости и практическое освоение приёмов программной реализации аффинных преобразований.

Задание

На самостоятельно выбранном языке программирования создать программу, реализующую следующие преобразования фигуры (многоугольника) на плоскости в зависимости от варианта:

I. Перенос:
   a. вдоль оси OX (ввод расстояния переноса);
   b. вдоль оси OY (ввод расстояния переноса);
   c. вдоль вектора OP (указание точки P);
   d. вдоль заданной стороны фигуры (указание стороны, расстояние переноса равно длине стороны).

II. Масштабирование:
   a. вдоль оси OX (ввод масштабного коэффициента);
   b. вдоль оси OY (ввод масштабного коэффициента);
   c. вдоль вектора OP (указание точки P, ввод масштабного коэффициента);
   d. вдоль заданной стороны фигуры (указание стороны, ввод масштабного коэффициента).

III. Отражение:
   a. относительно оси OX;
   b. относительно заданной вершины фигуры (указание вершины);
   c. относительно прямой с направляющим вектором OP, проходящей через начало координат (указание точки P);
   d. относительно заданной стороны фигуры (указание стороны).

IV. Поворот:
   a. вокруг начала координат (ввод величины угла поворота);

b. вокруг заданной точки плоскости (указание точки, ввод величины угла поворота);
c. вокруг заданной вершины фигуры (указание вершины, ввод величины угла поворота).

Требования к программе:

а) включать два окна (на одной форме или на разных), содержащих изображение фигуры до и после преобразования и управляющие элементы для ввода необходимой информации;

б) отображать координаты вершин исходной и преобразованной фигуры;

в) отображать итоговую матрицу преобразования и обеспечивать возможность внесения ручных изменений в эту матрицу и применения полученной матрицы к исходной фигуре;

г) иметь возможность отображать или скрывать координатные оси и масштабную сетку в окнах вывода фигур;

д) обеспечивать ввод фигуры для преобразования при помощи мыши посредством указания вершин фигуры;

е) для каждого преобразования обеспечивать возможность динамической визуализации многократного последовательного применения заданного преобразования к исходному многоугольнику (до остановки пользователем).

Работа выполняется по индивидуальному варианту №1, который представлен на рисунке 1.

Вариант	I	II	III	IV
1	(c)	(a)	(b)	(b)

Рисунок 1 – Вариант 1

Разработка алгоритма

Аффинное преобразование на плоскости — это линейное невырожденное преобразование плоскости в себя.

Афинное преобразование может быть представлено в матричном виде:

$$P` = P \cdot A + B$$

Здесь:

P = (x, y) – координаты исходной точки;

P` = (x`, y`) – координаты точки преобразования;

$A \in R^{(2\times2)}, B \in R^{(1\times2)}$ – матрицы преобразования, где $|A| \neq 0$.

Для упрощения реализации были введены однородные координаты.

Однородные координаты точки плоскости P с декартовыми координатами (x, y) — это тройка вида C = (a, b, c) = (x·h, y·h, h), где h — произвольное число, отличное от нуля. В данном случае, h был принят равным единице, то есть C = (x, y, 1).

В однородных координатах аффинные преобразования имеют одинаковую форму произведения вектора исходных координат на матрицу преобразования - C` = C · A.

Матрица C составного преобразования, заключающегося в последовательном выполнении n элементарных преобразований с матрицами $C_1, C_2, C_3, \dots, C_n$ определяется как произведение этих матриц слева направо в порядке выполнения:

$$C = C_1 \cdot C_2 \cdot C_3 \cdot \dots \cdot C_n$$

В ходе выполнения данной работы было необходимо реализовать следующие афинные преобразования:

1) Перенос вдоль вектора OP;
2) Масштабирование вдоль оси OX;
3) Отражение относительно заданной вершины фигуры;
4) Поворот вокруг заданной точки плоскости;

1. Перенос вдоль вектора.

Параметром преобразования является вектор $X_0 = (x_0, y_0)$, на который совершается перенос. Матрица преобразования T имеет вид:

$$T(X_0) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ x_0 & y_0 & 1 \end{pmatrix}.$$

2. Масштабирование вдоль координатных осей с заданными коэффициентами.

Преобразование имеет два параметра: коэффициент масштабирования s_x вдоль оси OX и коэффициент масштабирования s_y вдоль оси OY. Матрица преобразования $S(s_x, s_y)$ имеет вид:

$$S(s_x, s_y) = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

3. Отражение относительно заданной вершины фигуры.

Параметром преобразования является вершина фигуры $P_0(x_0, y_0)$. Данное преобразование можно представить в виде последовательности трех элементарных преобразований:

1) Перенос каждой точки фигуры на вектор -OP_0. Таким образом вершина P_0 окажется в начале координат;
2) Масштабирование с матрицей $S(-1,-1)$, то есть отражение относительно начала координат;
3) Перенос каждой точки фигуры на вектор OP_0.

4. Поворот вокруг заданной точки плоскости.

Параметром преобразования являются точка плоскости $P_0(x_0, y_0)$ и угол поворота φ. Данное преобразование можно представить в виде последовательности трех элементарных преобразований:

1) Перенос каждой точки фигуры на вектор -OP_0. Таким образом фигура окажется в том же положении относительно начала координат, в котором она была относительно точки P_0 до переноса;
2) Поворот каждой точки фигуры на угол φ относительно начала координат с матрицей преобразования R(φ):

$$R(\varphi) = \begin{pmatrix} \cos(\varphi) & \sin(\varphi) & 0 \\ -\sin(\varphi) & \cos(\varphi) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

3) Перенос каждой точки фигуры на вектор OP_0.

Поскольку в однородных координатах все афинные преобразования точки выполняются единообразно, алгоритм выполнения произвольного афинного преобразования заданного многоугольника для различных преобразований отличается только в части формирования матрицы преобразования. Блок-схема данного алгоритма представлена на рисунке 2.

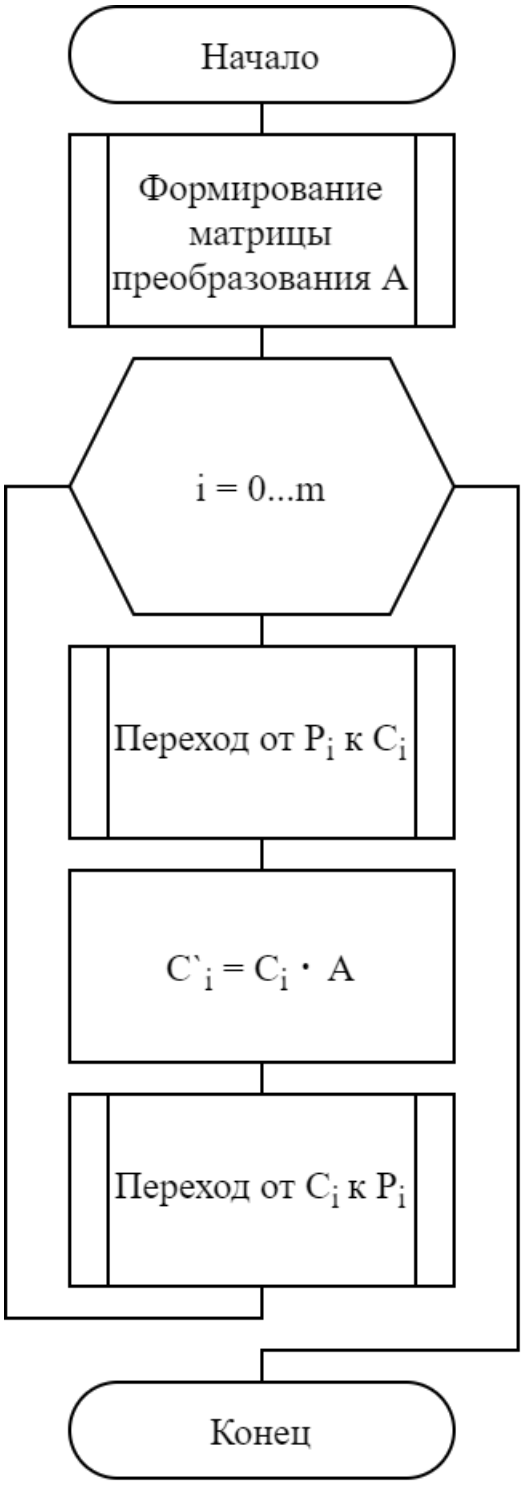

Рисунок 2 – Блок-схема алгоритма реализации произвольного афинного преобразования m-угольника. Здесь: $P_i = (x_i, y_i)$, $C_i = (x_i, y_i, 1)$ – декартовы и однородные координаты i-й вершины исходного многоугольника; $C`_i = (x`_i, y`_i, 1)$ $P`_i = (x`_i, y`_i)$ – однородные и декартовы координаты образа исходного многоугольника; m – количество вершин многоугольника.

В качестве средств программной реализации для данной лабораторной работы был выбран язык программирования JavaScript и графическая библиотека p5.js. Выбор данных средств реализации обоснован наличием опыта работы с ними и простотой их использования.

Текст программы

Программная реализация данных афинных преобразований приведена ниже:

```
translate() {
    let x = this.translation.x;
    let y = this.translation.y;
    const translationMatrix = [
      [1, 0, 0],
      [0, 1, 0],
      [x, y, 1]
    ];
    this.resultMatrix = math.multiply(this.resultMatrix, translationMatrix);
    this.updateTransformed();
    this.updateTable();
  }

  scale() {
    let x = this.scaling.x;
    let y = this.scaling.y;
    const scalingMatrix = [
      [x, 0, 0],
      [0, y, 0],
      [0, 0, 1]
    ];

    this.resultMatrix = math.multiply(this.resultMatrix, scalingMatrix);
```

```
          this.updateTransformed();
          this.updateTable();
        }

        reflect() {
          // saving previous values to later restore them
          const [scaleX, scaleY] = [this.scaling.x, this.scaling.y];
          const [translateX, translateY] = [this.translation.x,
this.translation.y];

          this.translation.x = -this.reflection.x;
          this.translation.y = -this.reflection.y;
          this.translate();

          this.scaling.x = -1;
          this.scaling.y = -1;
          this.scale();

          this.translation.x = this.reflection.x;
          this.translation.y = this.reflection.y;
          this.translate();

          [this.scaling.x, this.scaling.y] = [scaleX, scaleY];
          [this.translation.x, this.translation.y] = [translateX, translateY];
        }

        rotate() {
          const [translateX, translateY] = [this.translation.x,
this.translation.y];
          const angle = -transformed.radians(this.rotation.angle);
          const rotationMatrix = [
            [Math.cos(angle), -Math.sin(angle), 0],
            [Math.sin(angle), Math.cos(angle),  0],
            [0, 0, 1]
          ];
```

```
            this.translation.x = -this.rotation.x;
            this.translation.y = -this.rotation.y;
            this.translate();

            this.resultMatrix = math.multiply(this.resultMatrix,
rotationMatrix);

            this.translation.x = this.rotation.x;
            this.translation.y = this.rotation.y;
            this.translate();

            [this.translation.x, this.translation.y] = [translateX, translateY];
        }
```

Тестирование программы

Тестирование программы представлено на рисунках 3 –

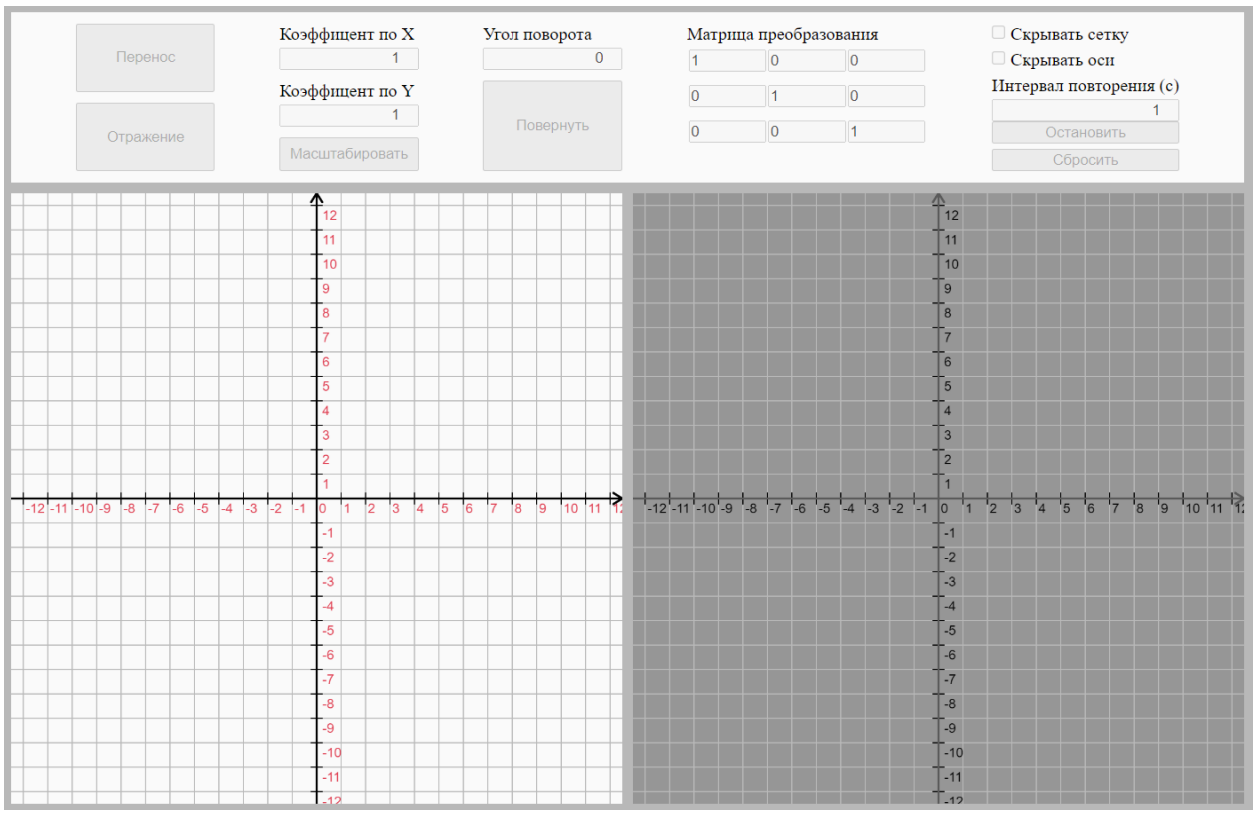

Рисунок 3 – Начальный вид формы

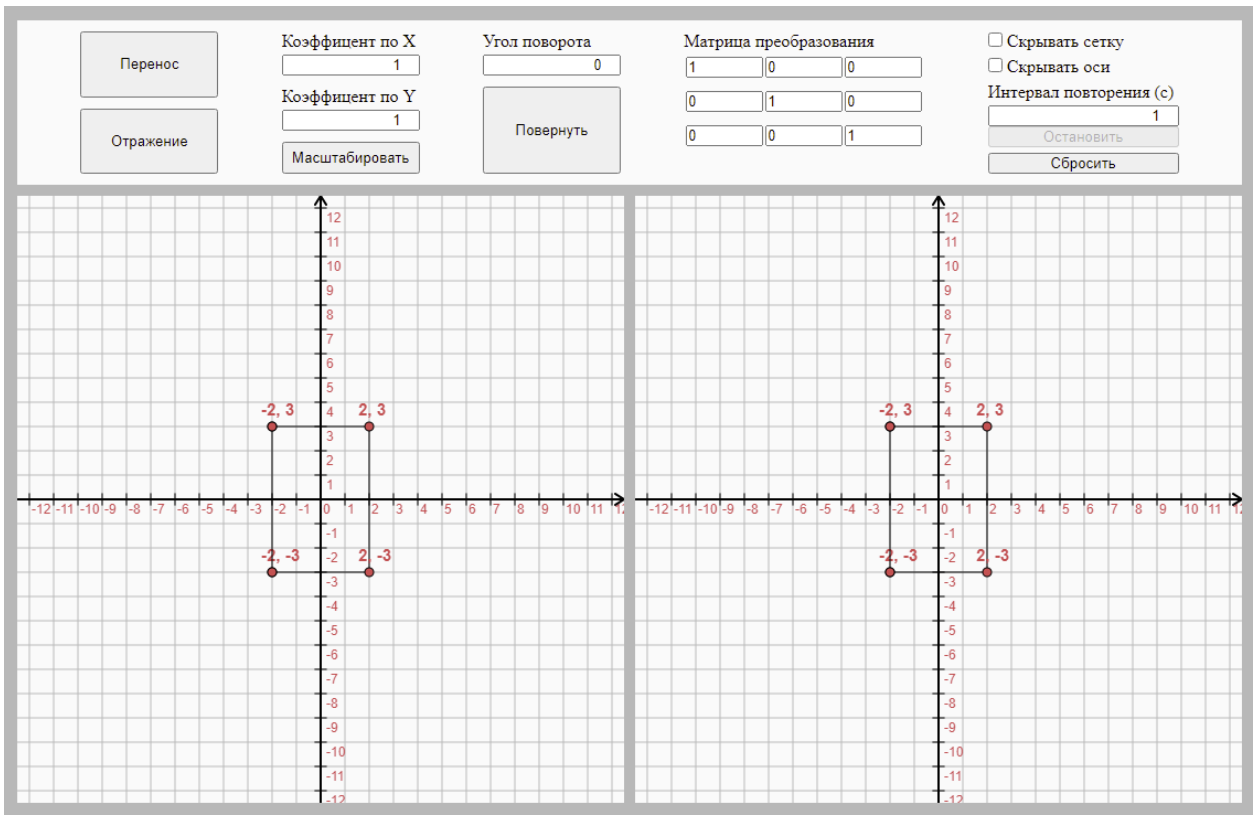

Рисунок 4 – Результат работы приложения после ввода фигуры

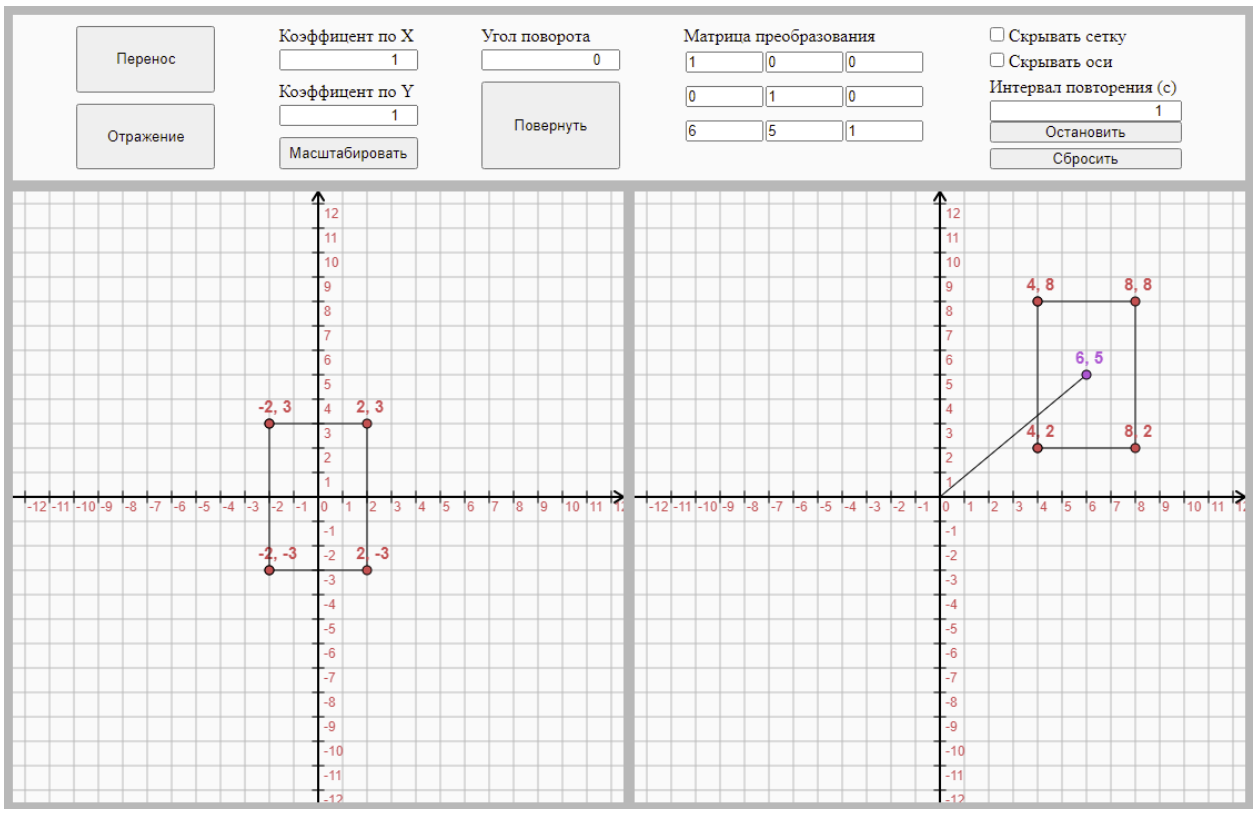

Рисунок 5 - Результат работы приложения после применения переноса

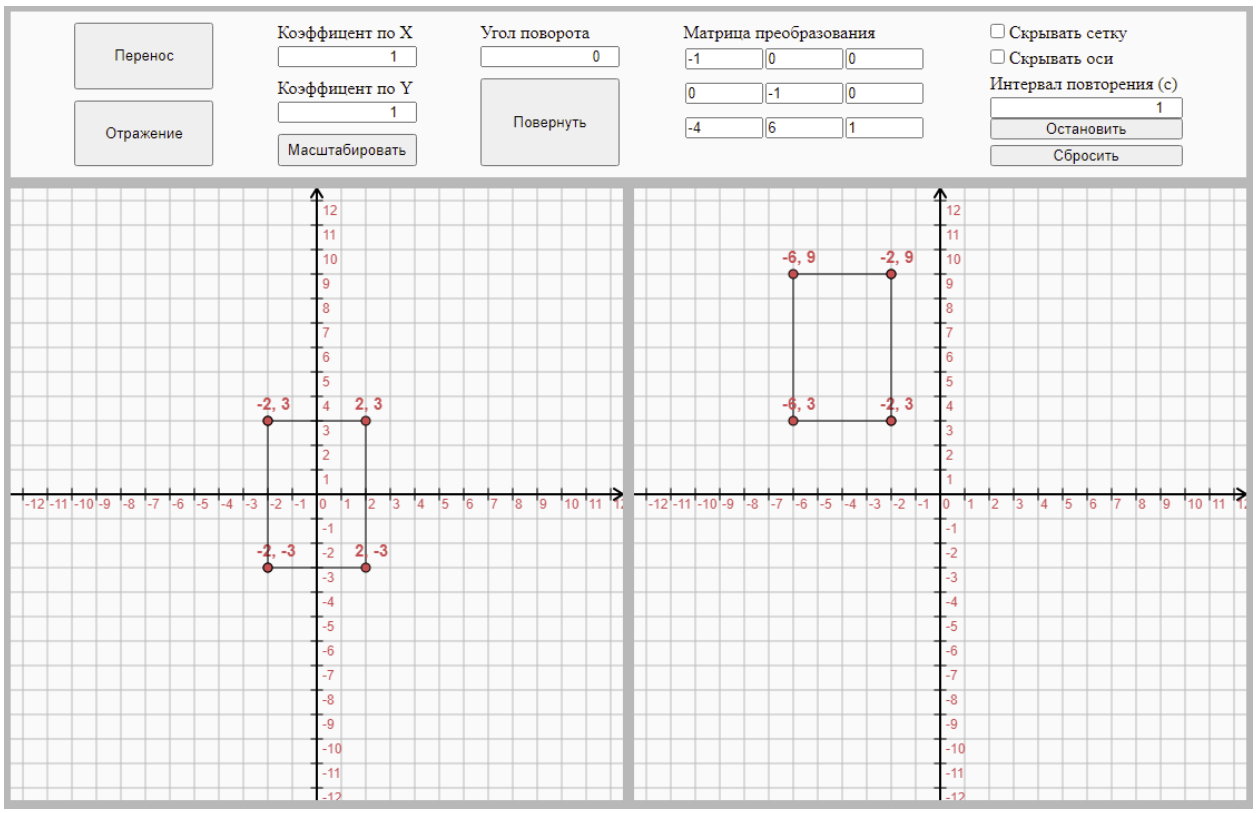

Рисунок 6 - Результат работы приложения после применения отражения относительно верхней левой вершины

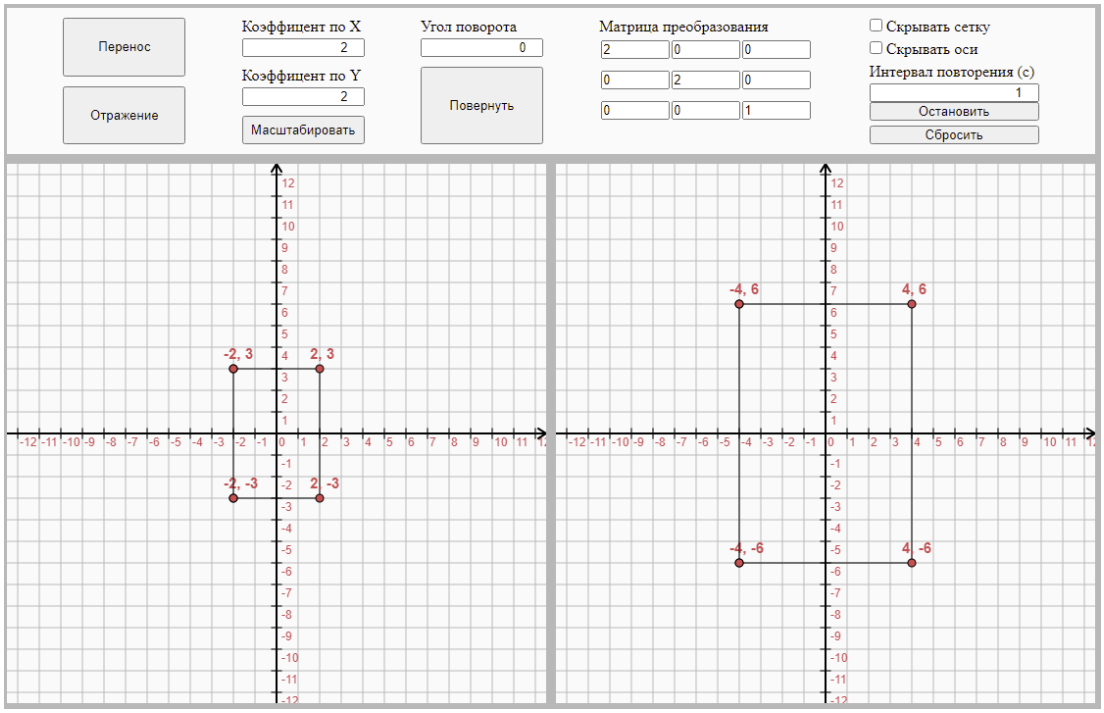

Рисунок 7 – Результат работы приложения после применения масштабирования

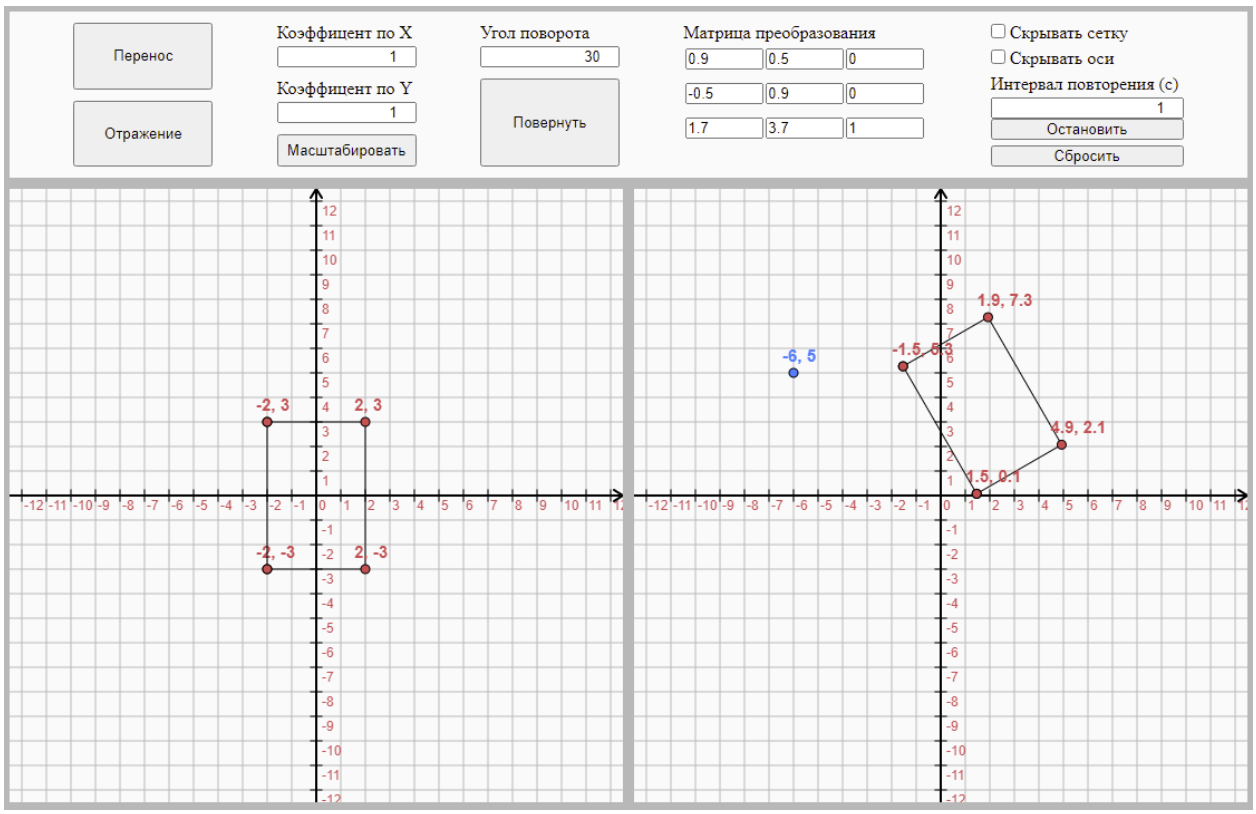

Рисунок 8 – Результат работы приложения после применения поворота

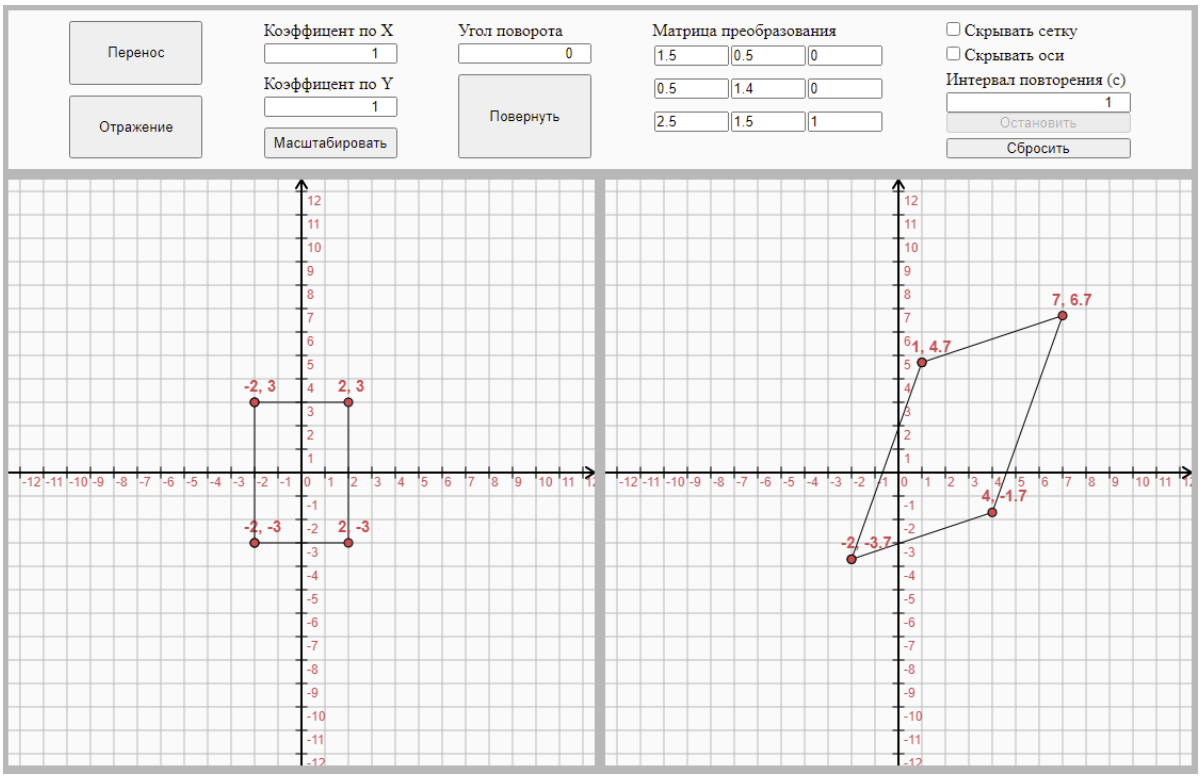

Рисунок 9 – Результат работы приложения после внесения произвольных изменений в матрицу преобразования

Выводы

В результате выполнения работы были изучены математические методов аффинных преобразований на плоскости и практически освоены приёмы программной реализации аффинных преобразований.