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 №3

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 и практическое освоение приёмов программной реализации проекционных преобразований.

Задание

Создать программу для построения проекций каркасной модели трёхмерного объекта-многогранника: наглядной и трёх основных ортографических видов, расположенных следующим образом:

Разновидность наглядной проекции определяется вариантом.

Требования к программе:

1) информация о координатах вершин объекта и порядке их соединения считывается из файла, имя которого указывается путём использования стандартного диалога открытия файла. Формат файла определяет разработчик программы. Следует подготовить несколько файлов с описанием трёхмерных объектов, наглядно демонстрирующих особенности указанного в задании вида проецирования;
2) окончательная матрица проецирования должна выводиться на экран. Программа должна обеспечивать возможность редактировать матрицу и выводить изображение на экран с использованием отредактированной матрицы;
3) программа должна обеспечивать реализацию ввода параметров проецирования и их плавное изменение с заданным пользователем шагом. При существовании ограничений на значения параметров

программа должна контролировать выполнение этих ограничений при вводе и информировать пользователя при их нарушении;

4) на экране должны быть показаны следующие параметры проекционного преобразования:

   – sx, sy, sz — координаты центра проецирования,

   – D — расстояние от начала координат до картинной плоскости,

   – ψx, ψy, ψz — углы между картинной плоскостью и осями координат OX, OY и OZ соответственно;

   Часть параметров (согласно индивидуальному заданию) вводится пользователем, остальные рассчитываются на основе введённой информации и выводятся на экран. В случае существования нескольких решений при расчёте программа должна выводить все варианты, обеспечивая пользователю возможность выбрать один из них;

5) исходные трёхмерные координаты вершин многогранника должны выводиться на экран;

6) в случае центрального проецирования на ортографических проекциях необходимо показать положение центра проецирования; для косоугольной проецирования — направляющий вектор проецирующего пучка;

7) на всех четырёх проекциях необходимо показать координатные оси с обозначениями, ортографические проекции должны содержать сетку с указанием масштаба.

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

Номер	Вид проецирования	Параметры, задаваемые пользователем
1	Центральное	$D, \psi_x, \psi_y, s_x, s_y, s_z$

Рисунок 1 – Вариант 1

Разработка алгоритма

Определение проецирования

Проецирование — это отображение трёхмерного объекта на двумерную картинную плоскость (в общем случае, n-мерного объекта на m-мерное пространство, n > m). Понижение размерности пространства приводит к потере части информации, вследствие чего однозначное восстановление прообраза по образу при проецировании невозможно. Таким образом, все проекционные преобразования являются вырожденными.

Для нахождения проекции объекта достаточно (но не всегда необходимо) найти проекции всех его точек, совокупность которых и будет представлять собой проекцию объекта. Таким образом, для построения проекции произвольного объекта достаточно уметь находить проекцию произвольной точки.

Классификация видов проецирования определяется типом и взаимным расположением элементов, определяющих конкретную разновидность проецирования: проецирующего пучка (пучка проецирующих лучей), картинной плоскости и трёхмерной системы координат OXYZ, в которой определяются координаты прообраза. В данной работе реализуется центральное проецирование – все проецирующие лучи исходят из одной точки, называемой центром проекции.

Для программной реализации проекционного образования следует определить его матрицу. Тогда любое проекционное преобразование точки можно представить в следующей общей алгебраической форме:

$$\mathbf{P}' = \mathbf{P} \cdot \mathbf{M},\ \mathbf{M} \in \mathbb{R}^{4\times4};\ \mathbf{P},\ \mathbf{P}' \in \mathbb{R}^{1\times4};\ |\mathbf{M}| = 0,$$

где P, P′ - однородные координаты точек прообраза и образа соответственно, M – матрица преобразования.

Реализация центрального проецирования

В общем виде, матрица центрального проецирования на плоскость OXY имеет следующий вид:

$$M^c = R_{YX}(\varphi_y, \varphi_x) \cdot T^{-1}(P_0) \cdot M_F^c,$$

где $R_{YX}(\varphi_y, \varphi_x)$ – матрица поворота, совмещающая картинную плоскость с плоскостью OXY. Углы φ_y, φ_x связаны с углами ψ_x, ψ_y, ψ_z связаны следующим соотношением:

$$\begin{cases} \varphi_x = \psi_y; \\ \varphi_y = -\arcsin \dfrac{\sin\psi_x}{\cos\psi_y} = \arccos \dfrac{\sin\psi_z}{\cos\psi_y}. \end{cases}$$

$T^{-1}(P_0)$ – матрица переноса на расстояние D от начала координат до картинной плоскости.

$$M_F^c = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -c'_x & -c'_y & 0 & -c'_F \\ 0 & 0 & 0 & 1 \end{pmatrix}; \; c'_x = \frac{s'_x}{s'_z}, \; c'_y = \frac{s'_y}{s'_z}, \; c'_F = \frac{1}{s'_z};$$

$$(s'_x, s'_y, s'_z) = (s_x, s_y, s_z) \cdot R_{YX}(\varphi_y, \varphi_x) \cdot T^{-1}(P_0)$$

Здесь (s_x, s_y, s_z) и (s'_x, s'_y, s'_z) — координаты центра проекции в исходной системе координат, и, соответственно, системе, в которой картинная плоскость совпадает с одной из координатных.

При центральном проецировании не всякая точка имеет реальную проекцию. Критерии реальности центральной проекции имеют следующий вид:

$$\begin{cases} \text{реальна, если } h > 0; \\ \text{мнима, если } h < 0; \\ \text{вырождена, если } h = 0, \end{cases}$$

где h – четвертая однородная координата точки после преобразования.

Также, при проекции следует учитывать ситуацию, когда одна точка отрезка имеет реальную проекцию, а другая – мнимую. В таком случае, нужно заменить точку с мнимой проекцией на точку с проекцией реальной. Координаты новой точки имеют следующий вид:

$$\left(x_1 + \frac{(x_2 - x_1)(s_z - z_1)}{z_2 - z_1}, y_1 + \frac{(y_2 - y_1)(s_z - z_1)}{z_2 - z_1}, s_z\right).$$

В данной формуле, в силу конечности окна вывода, при программной реализации следует заменить s_x на $(1 - \varepsilon)s_x$. Где ε – достаточно маленькое число, чтобы создать иллюзию ухода отображаемого отрезка в бесконечность.

Средства программной реализации

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 центрального проецирования представлена ниже:

```
  projectEntity(edge, projMatrix, s, e, rotationMatrix, transMatrix) {
    if (isZeroMatrix(projMatrix)) return;
    const p1 = edge.p1;
    const p2 = edge.p2;
    let vec1 = [p1.x, p1.y, p1.z, 1];
    let vec2 = [p2.x, p2.y, p2.z, 1];
    let aligned = math.multiply(rotationMatrix, transMatrix);
    vec1 = math.multiply(vec1, aligned);
    vec2 = math.multiply(vec2, aligned);

    const h1 = 1 - vec1[2] / s.z;
    const h2 = 1 - vec2[2] / s.z;
    if (h1 <= 0 && h2 <= 0) return;
    if (h1 > 0 && h2 <= 0) {
      vec2 = [
        vec2[0] + ((vec2[0]-vec1[0])*((1 - e)*s.z - vec2[2]))/(vec2[2]-vec1[2]),
        vec2[1] + ((vec2[0]-vec2[0])*((1 - e)*s.z - vec2[2]))/(vec2[2]-vec1[2]),
        (1 - e)*s.z,
        1
      ];
    } else if (h2 > 0 && h1 <= 0) {
      vec1 = [
        vec1[0] + ((vec2[0]-vec1[0])*((1 - e)*s.z - vec1[2]))/(vec2[2]-vec1[2]),
        vec1[1] + ((vec2[1]-vec1[1])*((1 - e)*s.z - vec1[2]))/(vec2[2]-vec1[2]),
        (1 - e)*s.z,
        1
      ];
    }
    vec1 = math.multiply(vec1, projMatrix);
    vec2 = math.multiply(vec2, projMatrix);
    vec1 = normalize(vec1);
    vec2 = normalize(vec2);
    return new Edge(new Point(vec1), new Point(vec2));
  },
  toScreen(point) {
    const sX = (point.x * step + width / 2);
    const sY = (-point.y * step + height / 2) ;
    return new Point([sX, sY]);
  }
});
```

Тестирование программы

Тестирование программы представлено на рисунках 2 –

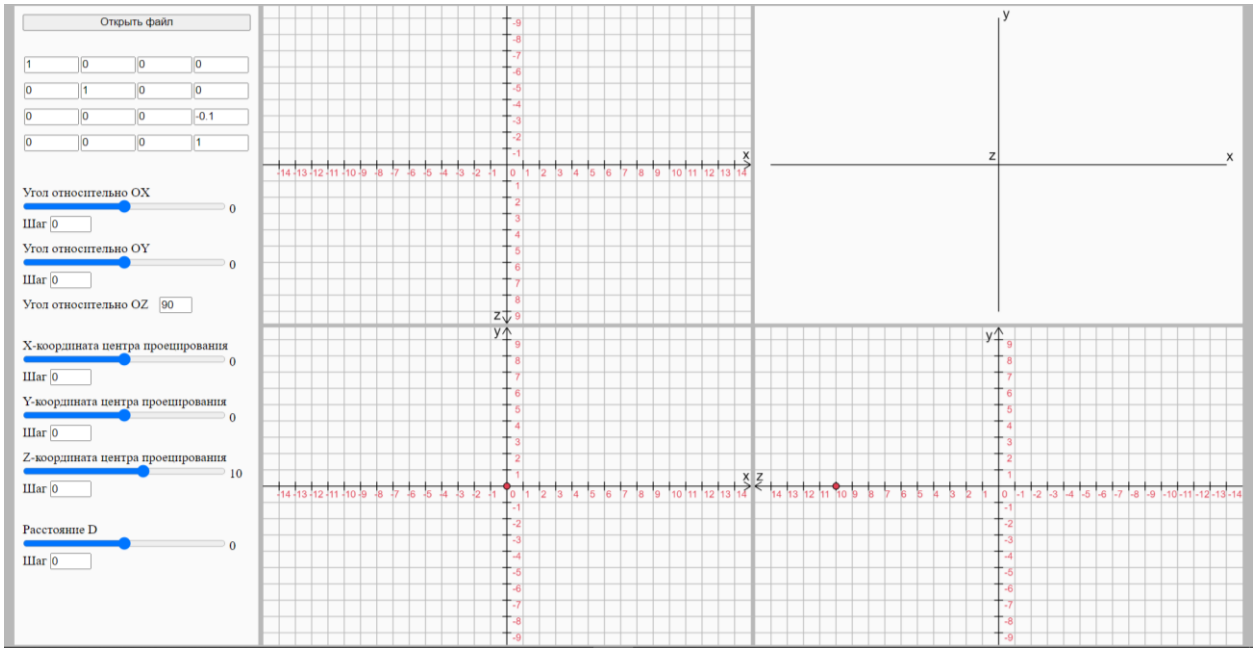

Рисунок 2 – Вид интерфейса приложения

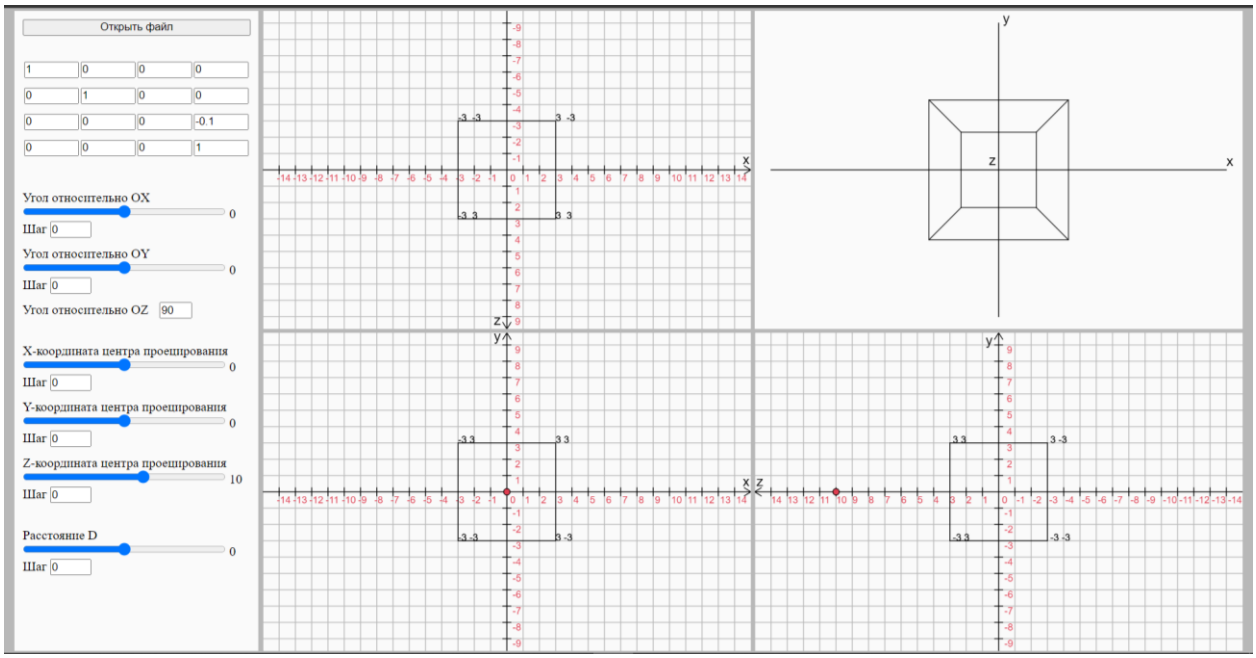

Рисунок 3 – Результат работы приложения после открытия тестового файла

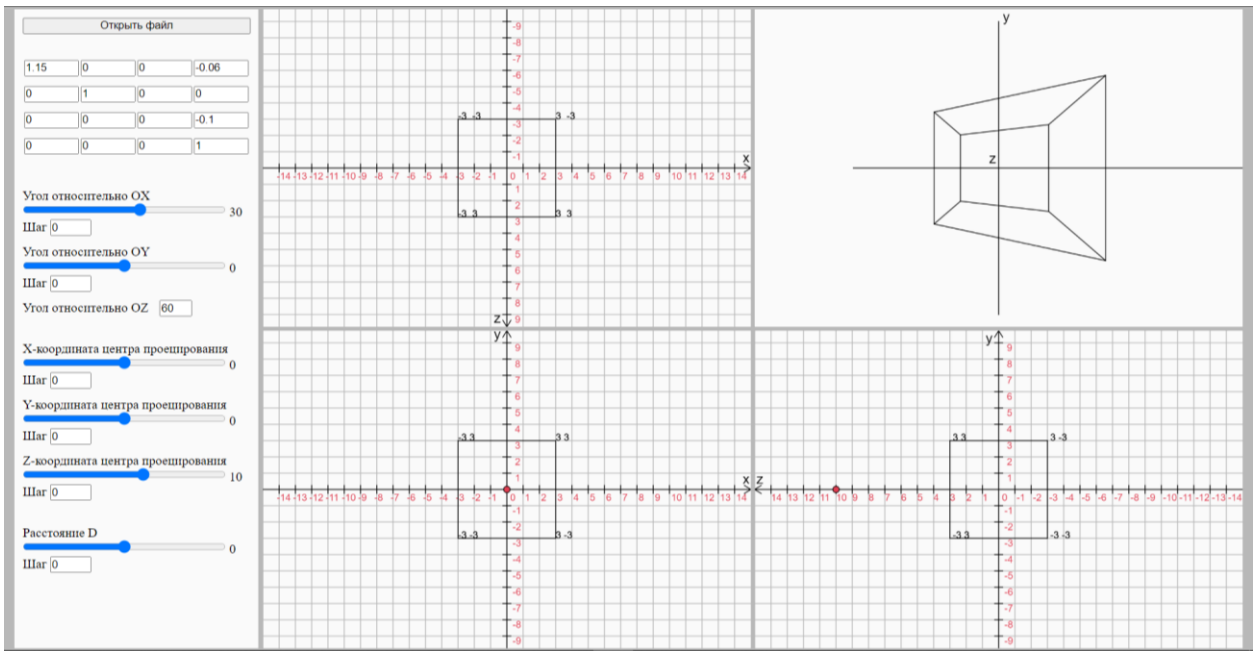

Рисунок 4 – Результат работы приложения после установки параметра “Угол относительно OX” в 30 единиц

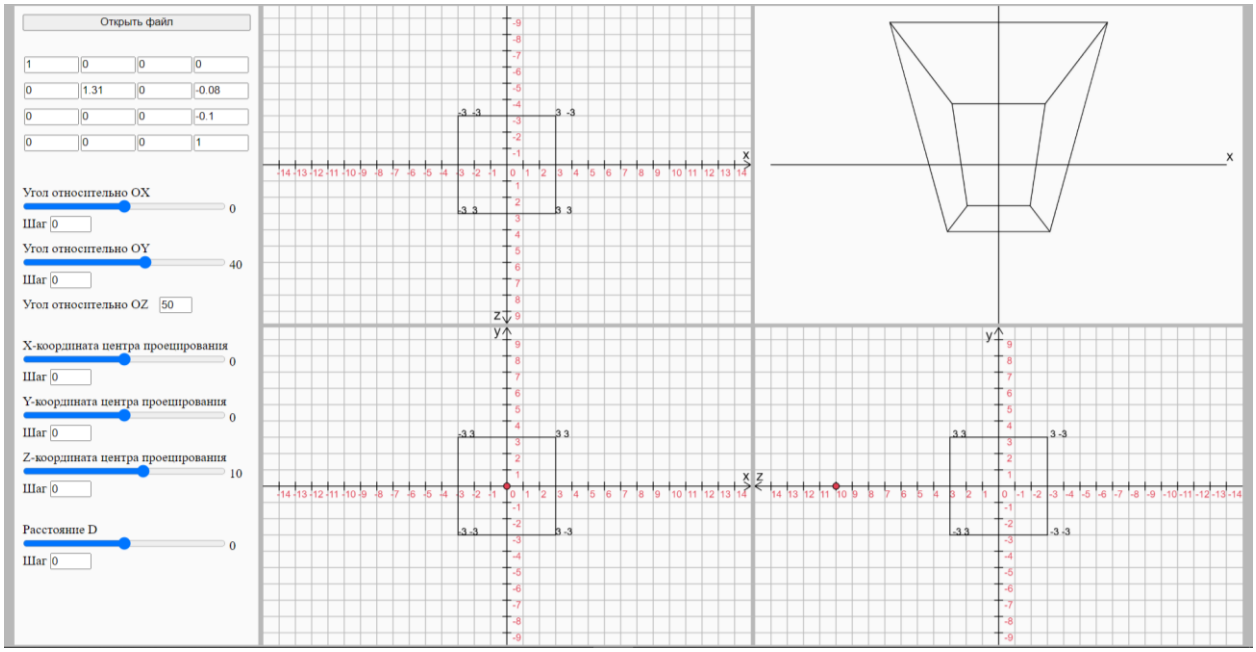

Рисунок 5 – Результат работы приложения после установки параметра “Угол относительно OY” в 40 единиц

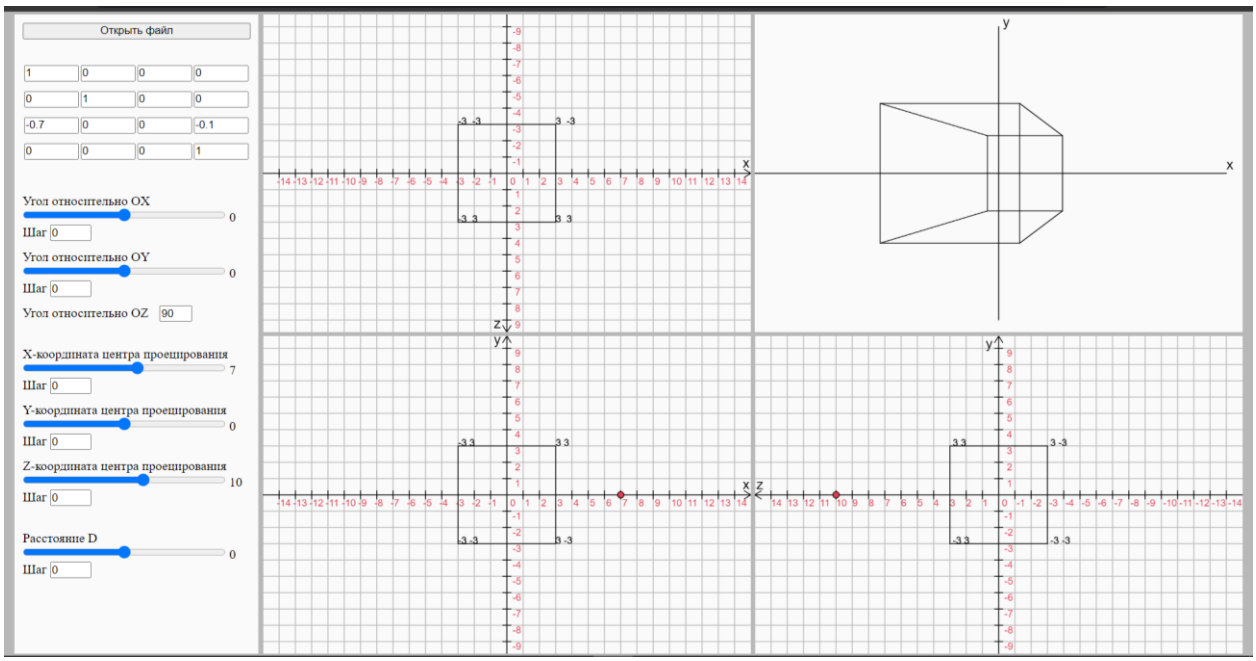

Рисунок 6 – Результат работы приложения после установки параметра “X-координата центра проецирования” в 7 единиц

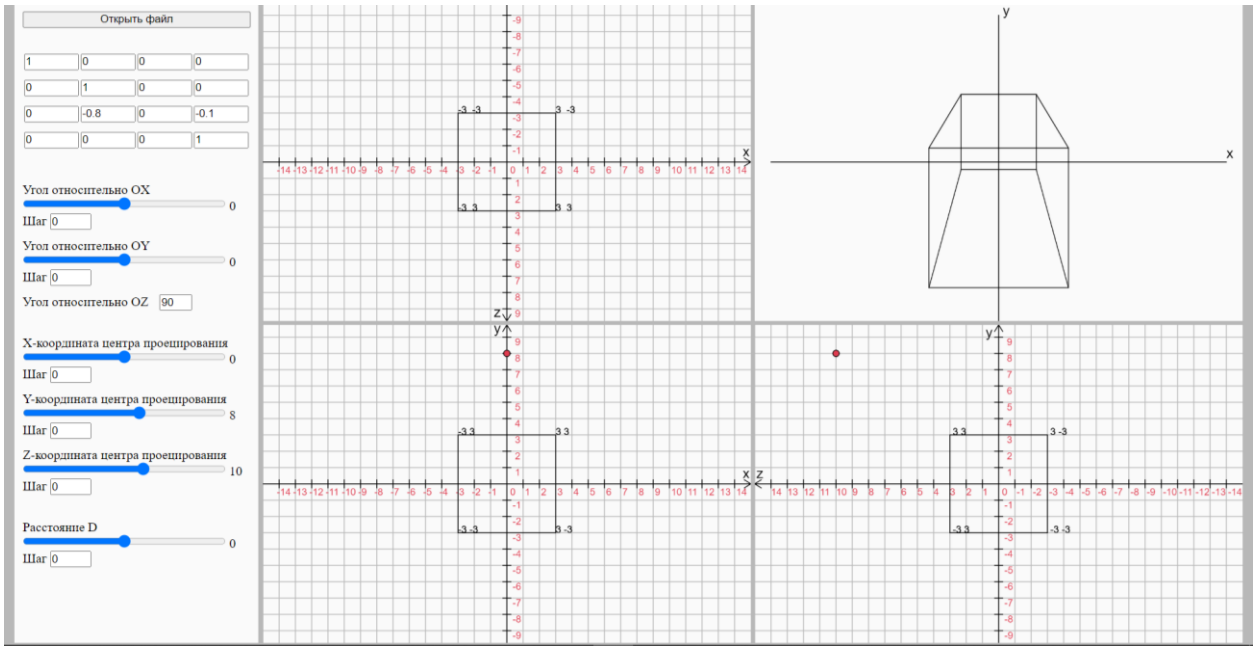

Рисунок 7 – Результат работы приложения после установки параметра “Y-координата центра проецирования” в 8 единиц

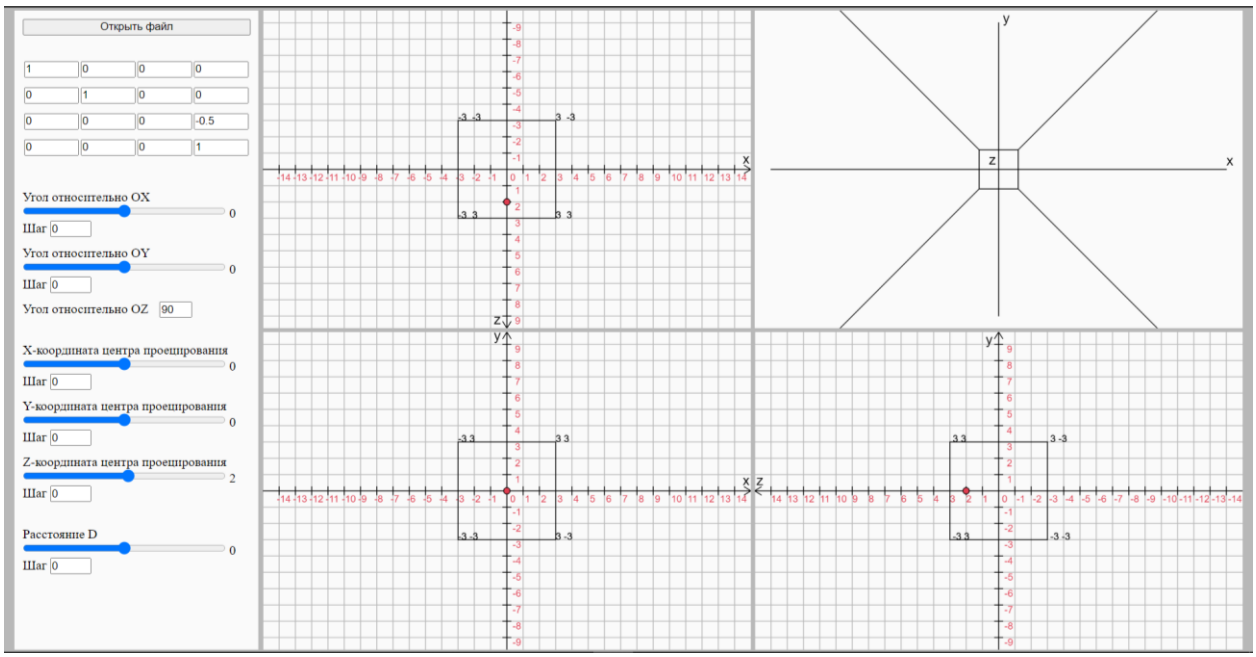

Рисунок 8 – Результат работы приложения после установки параметра “Z - координата центра проецирования” в 2 единицы

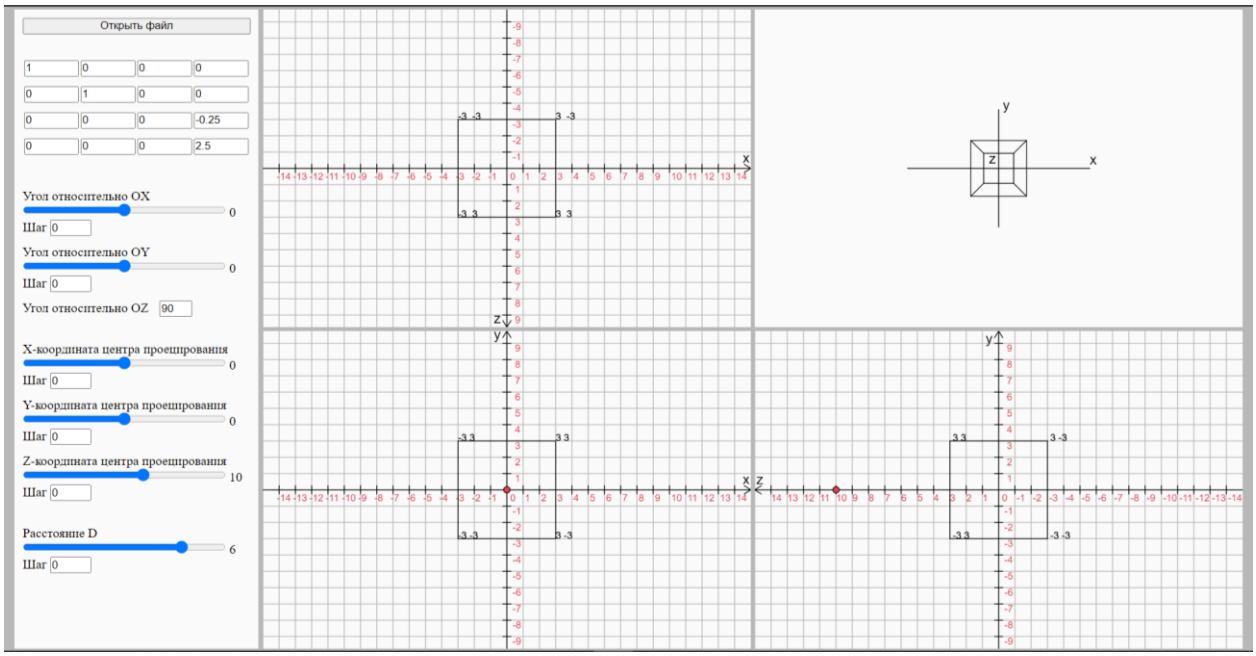

Рисунок 9 – Результат работы приложения после установки параметра “Расстояние D” в 6 единиц

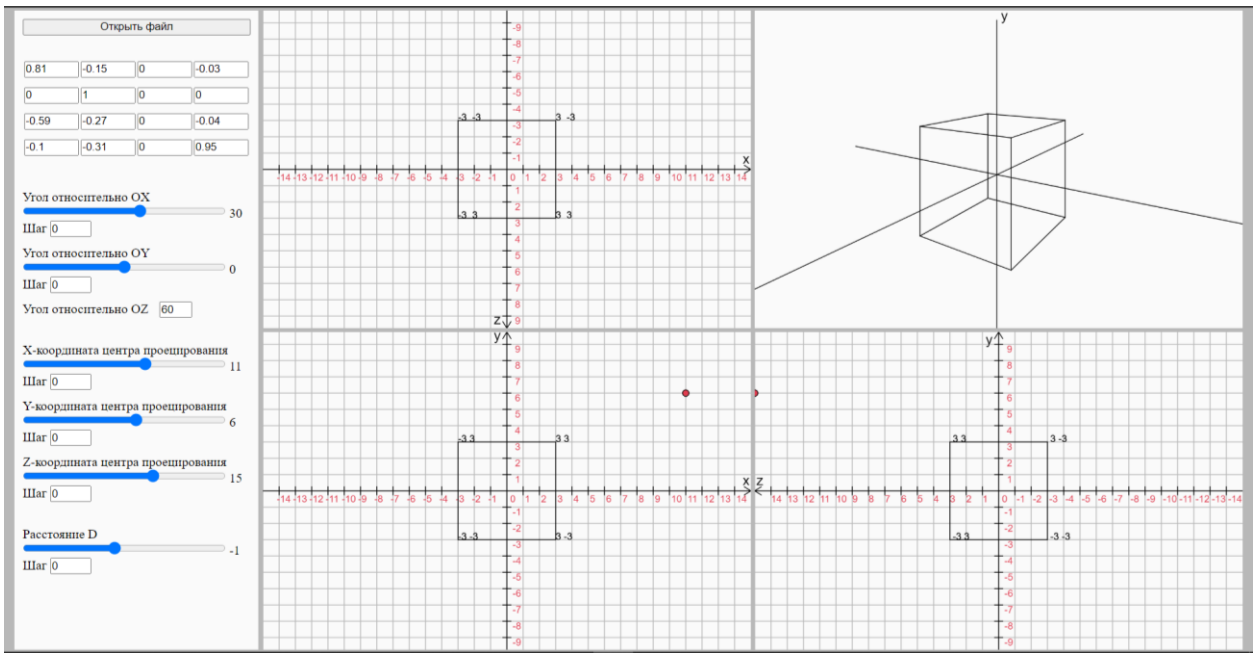

Рисунок 10 – Результат работы приложения после произвольной установки параметров (конкретные значение приведены на рисунке)

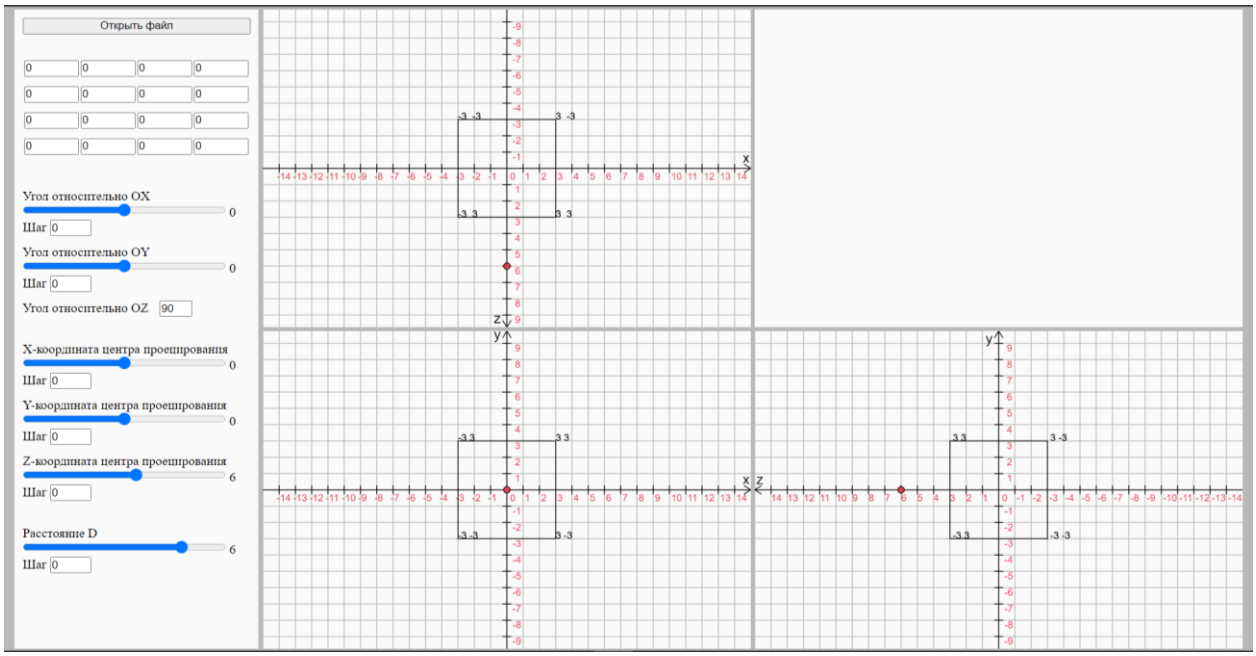

Рисунок 11 – Результат работы приложения после присваивания параметрам “Z-координаты центра проецирования” и “Расстояния D” одинакового значения

Выводы

В результате выполнения работы были изучены математические методы и практически освоены приёмы программной реализации проекционных преобразований.