**Уфимский Государственный Авиационный Технический Университет**

**Кафедра информатики**

Пояснительная записка к лабораторной работе № 1

“Визуализация 3D-точки”

по дисциплине “Математические и алгоритмические основы компьютерной графики”

Выполнил:

студент группы СТС-109

Шлёнкин Дмитрий

Проверил: Верхотуров

Михаил Александрович

Уфа 2020

Оглавление

[1 Краткая теория 4](#_Toc41770666)

[1.1 Ортогональная система трех плоскостей проекций 4](#_Toc41770667)

[2 Постановка задачи 7](#_Toc41770668)

[2.1 Содержательная постановка задачи 7](#_Toc41770669)

[2.2 Формальная постановка задачи 8](#_Toc41770670)

[3 Структура решения (Первый уровень детализации) 9](#_Toc41770671)

[4 Обзор и анализ методов решения (2-й уровень детализации) 10](#_Toc41770672)

[4.1 Ввод информации 10](#_Toc41770673)

[4.2 Преобразование 3D координат в набор 2D точек 11](#_Toc41770674)

[4.2.1 Преобразование для П.Ч. 11](#_Toc41770675)

[4.2.2 Преобразование для К.Ч. 13](#_Toc41770676)

[4.3 Визуализация 16](#_Toc41770677)

[4.3.1 Визуализация «пространственного» чертежа 17](#_Toc41770678)

[4.3.2 Визуализация комплексного чертежа 17](#_Toc41770679)

[5 Описание применяемых методов (3-й уровень детализации) 19](#_Toc41770680)

[5.1 Типы данных 19](#_Toc41770681)

[5.2 Структуры данных 19](#_Toc41770682)

[5.2.1 Переменные для П.Ч. 20](#_Toc41770683)

[5.2.2 Переменные для К.Ч. 20](#_Toc41770684)

[5.3 Укрупненный алгоритм 21](#_Toc41770685)

[5.3.1 Ввод информации о координатах точки и углах между горизонтальным вектором и осями координат 21](#_Toc41770686)

[5.3.2 Преобразование трехмерных координат точек в двумерные 22](#_Toc41770687)

[5.3.3 Визуализация «пространственного» чертежа 22](#_Toc41770688)

[5.3.4 Визуализация комплексного чертежа 23](#_Toc41770689)

[6 Детализированный алгоритм (4-й уровень детализации) 26](#_Toc41770690)

[6.1 Ввод координаты точек и углов 27](#_Toc41770691)

[6.2 Преобразование для П.Ч. 28](#_Toc41770692)

[6.3 Преобразования для К.Ч. 28](#_Toc41770693)

[6.4 Визуализация для П.Ч. 28](#_Toc41770694)

[6.5 Визуализация К.Ч. 29](#_Toc41770695)

[7 Руководство программиста 31](#_Toc41770696)

[7.1 Интерфейс 31](#_Toc41770697)

[7.2 Прикладная часть 33](#_Toc41770698)

[7.3 Как работает программа 38](#_Toc41770699)

[8 Руководство пользователя 41](#_Toc41770700)

# Краткая теория

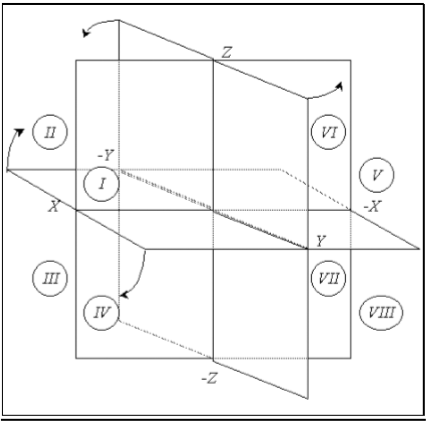
## Ортогональная система трех плоскостей проекций

В основу построения любого изображения положена операция проецирования.

Сущность метода ортогонального проецирования заключается в том, что предмет проецируется на две взаимно перпендикулярные плоскости лучами, ортогональными (перпендикулярными) к этим осям.

Одну из плоскостей проекций П1 располагают горизонтально, а вторую П2 - вертикально. Плоскость П1 называют горизонтальной плоскостью проекций, П2 - фронтальной. Плоскости П1 и П2 бесконечны и непрозрачны.

Три плоскости проекций делят пространство на восемь трехгранных углов - это так называемые октанты. Нумерация октантов дана на Рисунок 1.1



П3

1

П21

П11

Рисунок 1.1 – Ортогональная система плоскостей

Рассматривая ортогональные проекции, предполагают, что наблюдатель находится в первом октанте. Проекционный чертеж, на котором плоскости проекций со всем тем, что на них изображено, совмещены определенным образом одна с другой, называется эпюром. Для получения эпюра плоскости П1 и П3 вращают как показано на Рисунок 1.1.

Окончательный вид всех совмещенных плоскостей проекций дан на Рисунок 1.2.

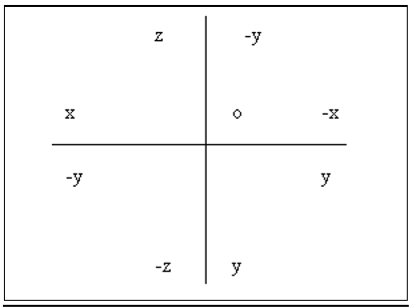
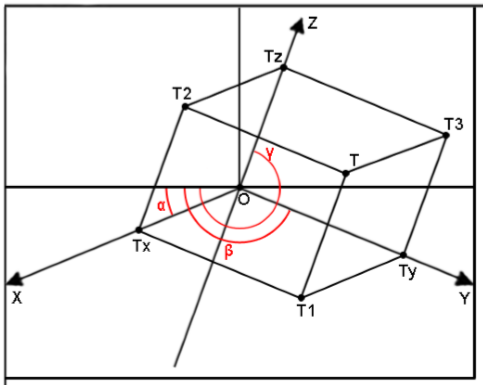


Рисунок 1.2 – Вид совмещенных плоскостей

Так как в данной лабораторной работе пользователь будет иметь возможность изменять положение координатных осей (Ox, Oy, Oz), то для предоставления данной возможности будем использовать углы α, β, γ между горизонтальным вектором, направленным влево, и положительными направлениями осей координат Ox, Oy, Oz соответственно.

На Рисунок 1.3 показано расположение точек Tx, Ty, Tz; проекций точки T: T1, T2, T3на координатные плоскости П1, П2 и П3 и углов α, β, γ на «пространственном» чертеже с используемыми обозначениями, которые предстоит отобразить на экране.



П31

П11

П21

Рисунок 1.3 – «Пространственный» чертеж

# Постановка задачи

## Содержательная постановка задачи

Визуализация 3D-точки на «пространственном» и комплексном чертежах.

Необходимо написать программу, удовлетворяющую следующим требованиям:

1. «пространственный» чертёж с изображением точки Т, ее проекций Т1, Т2, Т3 и линий связи;
2. комплексный чертеж с изображением проекций Т1, Т2, Т3 и линий связи;
3. ползунковые переключатели для интерактивного изменения координат (X, Y, Z) точки Т.;
4. ползунковые переключатели для интерактивного изменения углов α, β, γ на «пространственном» чертеже, где α, β, γ – углы между горизонтальным вектором, направленным влево, и положительными направлениями осей Ox, Oy, Oz соответственно.

Так же, чтобы это было все в динамике, чтобы можно было задавать координату точки, и видеть изменение.

Обобщим:

**Входные данные**

Пустой экран

**Управляющие параметры**

Координаты изображаемой точки.

Углы, задающие оси координат на «пространственном» чертеже

**Выходные данные**

Изображение 3D-точки на «пространственном» чертеже. Изображение точки на комплексном чертеже.

## Формальная постановка задачи

Сделаем такую форму, которая описывает наше задание:



# Структура решения (Первый уровень детализации)

На этом уровне задача делится на классические этапы.

1. **Ввод** – Ввод информации о координатах точки T и углах между горизонтальным вектором и осями координат (для «пространственного» чертежа).

1. **Обработка Операция проецирования –** осуществляемая двумя способами, т.е. преобразование совокупности трехмерных точек в два набора двумерных точек:

1) Для «пространственного» чертежа;

2) Для комплексного чертежа.

1. **Вывод Визуализация изображения**:

1) «Пространственного» чертежа;

2) Комплексного чертежа.

Иными словами, эта классификация делит нашу задачу на 5 подзадач.

# Обзор и анализ методов решения (2-й уровень детализации)

## Ввод информации

**Дано:**

Пользователь вводит:

* координаты точки в пространстве (X, Y, Z);
* Углы поворота осей, относительно горизонтального вектора, направленного влево (α, β, γ);

**Задача** / **характеристика:**

Пользователю необходимо изменять, вводить координаты точки Т и углы поворота осей.

Специфика ввода: Координаты должны быть целочисленными значениями, причем ограничены на определенном промежутке.

Должен быть динамический ввод данных, чтобы можно было плавно менять значение.

**Возможное решение:**

Для ввода информации в данном случае подойдут подвижные элементы управления, чтобы было удобно менять значения данных параметров – ползунковые переключатели.

В среде программирования Visual Studio существует множество элементов, через которые можно вводить данные.

Под описанние ранее свойство подходит элемент TrackBar.

TrackBar содержит такое свойство, как Value, которое показывает целочисленное текущее значение ползунка, так же можно задать минимальное значение и мацсимальное значение ползунка, ем самым ограничив промежуток, который следует взять. Так же у TrackBar’а есть событие Scroll, которое вызывается при каждом изменении значения ползунка. Что как раз подходит под ввод данной задачи.

Обобщим:

Считать информацию с элементов управления TrackBar, спомощью Value (значения X, Y, Z, α, β, γ) и записать эти значения в соответствующие переменные.

## Преобразование 3D координат в набор 2D точек

Необходимо перевести 3D координаты точки в плоскость экрана (в 2D)

### Преобразование для П.Ч.

**Задание:**

1) получить экранные 2D-координаты 3D-точки в пространстве, ее проекций и точек, задающих оси координат и линии связи.

**Дано:**

1. 3D-координаты точки T в пространстве: x, y, z.

3D точки «пространственного» макета (T, T1, T2, T3, Tx, Ty, Tz, O, OX, OY, OZ, -OX, -OY, -OZ)

2) Углы между горизонтальным вектором, направленным влево, и положительными направлениями осей координат Ox, Oy, Oz: α, β, γ.

3) Ширина и высота области рисования: width, height.

**Найти:**

Экранные 2D-координаты (T’, T1’, T2’, T3’, Tx’, Ty’, Tz’, O’, OX’, OY’, OZ’, -OX’, -OY’, -OZ’):

Точек начала и конца осей Ox, Oy, Oz;

Точки пересечения осей координат (точка О, начало координат)

Проекции точки T

Исходя из данных, нужно вычислить формулы, для перевода точки из трехмерной системы координат в двухмерную. Как это сделать – хорошо видно на (рисунок 4.1).

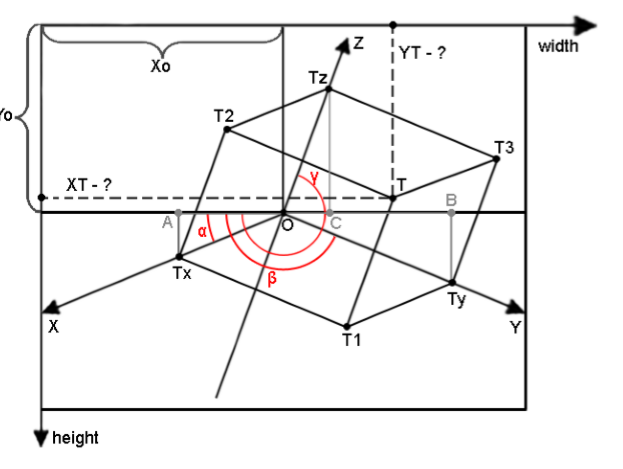
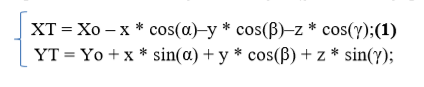


Рисунок 4.1 – Макет пространственного чертежа

*Точки 3D:*

T, T1, T2, T3, Tx, Ty, Tz, O, OX, OY, OZ, -OX, -OY, -OZ [[1]](#footnote-1)

Нужно найти координату (Xt,Yt) у точки, которая имеет координаты (X,Y,Z). Так же связять это с углами, которые введет пользователь, переработать остальные 13 точек чертежа в 2D формат. На (рисунок 4.1) указаны нужные переменные, обозначения. В итоге всех подстановок, выводов формул, получается система, а точнее 2 уравнения для точек Xt и Yt соответственно.(1)

 (1)

С помощью формул (1) можно преобразовать трехмерные координаты точек в двумерные для пространственного макета одинаковым образом. Таким образом, наиболее целесообразным будет хранение всех 14-ти точек в массиве 3D-точек, а преобразование проводить в цикле, используя формулы (1) для каждой точки одинаково, после чего записывать полученные результаты уже в массив из 14-ти 2D-точек

*Точки 2D для П.Ч.:*

(T’, T1’, T2’, T3’, Tx’, Ty’, Tz’, O’, OX’, OY’, OZ’, -OX’, -OY’, -OZ’)[[2]](#footnote-2)

**Обобщим:**

Каждую из 14-ти 3D-точек «пространственного» макета (T, T1, T2, T3, Tx, Ty, Tz, O, OX, OY, OZ, -OX, -OY, -OZ сноска(1)) преобразовать в двумерную точку «пространственного» чертежа(T’, T1’, T2’, T3’, Tx’, Ty’, Tz’, O’, OX’, OY’, OZ’, -OX’, -OY’, -OZ’) по формуле (1)

### Преобразование для К.Ч.

**Задание:**

1) получить экранные 2D-координаты проекций точки T на плоскость, точек, задающих оси, точки, соответствующие точке T на координатных осях Ox, Oy, Oz.

**Дано:**

1. 3D-координаты точки T в пространстве: x, y, z.

3D точки «пространственного» макета:(T, T1, T2, T3, Tx, Ty, Tz, O, OX, OY, OZ, -OX, -OY, -OZ)

2) Ширина и высота области рисования: width, height.

**Найти:** Экранные 2D-координаты (MT1, MT2, MT3, MTx, MTy1, MTy3, MTz, MO, Mx, My1, My3, Mz,):

a. Точек начала и конца осей Ox, Oy, Oz;

b. Точки пересечения осей координат (точка О, начало координат);

c. точки, соответствующие точке T на координатных осях Ox, Oy, Oz (Tx, Ty, Tz в 3D);

d. Проекций точки T на координатные плоскости Ox:Oy, Oz:Oy, Oz:Ox;

Имея всевозможные координаты точки Т (сноска(1))

Для наглядности рассмотрим (рисунок 4.2).

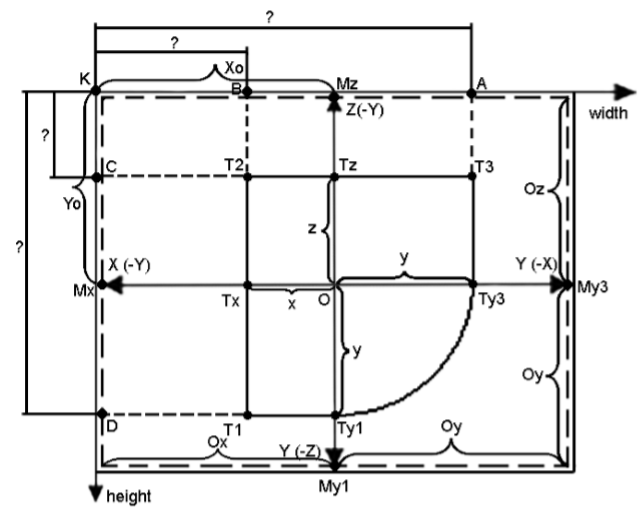


Рисунок 4.2 – Комплексный чертёж

Для начала нужно заметить, что если посчитать точек в 3D пространстве - 14. Для комплексного достаточно 12. Это происходит вследствие смещения осей таким образом, что две точки чертежа накрываются другими, то есть их координаты совпадают.

**Исходя из этого чертежа:**

1. Координаты точек начала и конца осей:

Mx = (Xo – Ox, Yo);

Mz = (Xo ,Yo – Oz)

My1 = (Xo, Yo + Oy)

My3 = (Xo + Oy, Yo)

Координаты точки O (начала координат):

MO = (Xo, Yo);

Координаты проекций точки T на координатные оси:

MTx = (BK, Yo) = (Xo – x, Yo);

MTz = (Xo, CK) = (Xo, Yo – z);

MTy1 = (Xo, DK) = (Xo, Yo + y);

MTy3 = (AK, Yo) = (Xo + y, Yo);

Координаты проекций точки T на координатные плоскости:

MT1 = (BK, DK) = (Xo - x, Yo + y);

MT2 = (BK, CK) = (Xo - x, Yo – z);

MT3 = (AK, CK) = (Xo + y, Yo – z);

*Точки для 2D К.Ч.:*

(MT1, MT2, MT3, MTx, MTy1, MTy3, MTz, MO, Mx, My1, My3, Mz,)[[3]](#footnote-3)

**Обобщим:**

Каждую 3D точку (сноска 1)

Для координат точек (OX, OY, Tx, Ty1, T1) 3D макета преобразовать в 2D точки (Mx, My1, MTx, MTy1, MT1) по формуле;

(Xo + y, Yo – z) (2)

Для координат точек (Oz, Tz, T2) 3D макета преобразовать в 2D точки (Mz, MTz, MT2) по формуле;

(Xo – x, Yo – z) (3)

Для координат точек(-Oy,Ty, T3) 3D макета преобразовать в 2D точки (My3, MTy3, MT3) по формуле;

(Xo – x, Yo + y) (4)

## Визуализация

Для визуализации изображения необходимо вывести на экран следующие компоненты:

1. Координатные оси

a. Три оси для «пространственного» чертежа;

b. Две оси для комплексного чертежа.

2. Точку T на «пространственном» чертеже

3. Проекции точки T на координатные оси

a. Для «пространственного» чертежа;

b. Для комплексного чертежа.

4. Проекции точки T на координатные плоскости

a. Для «пространственного» чертежа;

b. Для комплексного чертежа.

5. Линии связи.

Каждый из чертежей будет отрисовываться в соответствующем графическом окне, которое во многих средах разработки программного обеспечения для операционной системы Windows (например, VisualStudio) представлено классом PictureBox. Он имеет все необходимое для отрисовки изображения.

Для вывода данных компонентов воспользуемся отрисовкой:

* Линий (для осей координат и линий связи);
* Окружностей (для точки и ее проекций);
* Дуги (для соединения Ty1, Ty3 на комплексном чертеже).

Всё это будет отрисовываться спомощью такой подключенно библиотеки как System.Drawing. В ней присутствуют необходимые функции для отображения: Линий, точек, дуг и т.д.

* DrawLine(), отрисовка линии между двумя точками;
* DrawEllipse(), отрисовка эллипса (окружности);
* DrawArc(), отрисовка дуги(для комплексного чертежа).

Проблема с «мерцанием», связанная с отрисовкой нового изображения и затирания старого. Эту проблему в данной лабораторной работе можно решить спомощью Bitmap().

### Визуализация «пространственного» чертежа

1. Отрисовать координатные оси, соединив точки их начала и конца линиями, отрисовку которых осуществим при помощи метода DrawLine();

[Ox’] - [-Ox’] - ось Х

[Oy’] - [-Oy’] - Ось У

[Oz’]- [-Oz’] - Ось z

2. Отрисовать линии связи, соединив соответствующие точки проекций на координатные оси и плоскости линиями, отрисовку которых осуществим при помощи метода DrawLine();

[T’] - [Tx’], [T’] - [Ty’], [T’]- [Tz’], [T1’]- [Ty’], [T1’]- [Tz’],

[T2’]- [Tx’], [T2’]- [Tz’], [T3’]- [Tx’], [T3’]- [Ty’].

3. Отрисовать точки из соответствующего массива 14-ти 2D точек (Сноска(1)) при помощи метода DrawEllipse() (отрисовку проводим в цикле, так как все 14 точек рисуются одинаковым образом);

### Визуализация комплексного чертежа

1. **Отрисовать координатные оси**, соединив точки их начала и конца линиями, отрисовку которых, осуществим при помощи метода DrawLine();

[Mx] - [My3] - Ось(X (-Y) – Y(-X));

[Mz] - [My1] - Ось (Z(-Y) – Y(-Z));

2. **Отрисовать линии связи**, соединив соответствующие точки проекций на координатные оси и плоскости линиями, отрисовку которых мы осуществим при помощи метода DrawLine();

[MTz]- [MT1], [MTz]- [MTy], [MT3]- [MT2], [MT2]- [MT1]

1. **Отрисовать дугу**, которая будет соединять точки, соответствующие точке T на координатных осях Ox, Oy, Oz (Tx, Ty, Tz в 3D); при помощи метода DrawArc().

[MTy1]- [MTy3]

Причем отрисовывать дугу мы будем двумя способами (cразными начальными углами), в зависимости от того, какие знаки будут у точки (MTy1.y) проекций на ось Oy.

Если MTy1.y > СenterY:

Начальный угол, равный 180 градусов;

Полный угол, равный 90 градусов.

Иначе:

Начальный угол, равный 0 градусов;

Полный угол, равный 90 градусов.

4. **Отрисуем точки** из соответствующего массива 12-ти 2D точек (сноска(3) (MT1, MT2, MT3, MTx, MTy1, MTy3, MTz, MO, Mx, My1, My2, Mz,) при помощи метода DrawEllipse() (отрисовку проводим в цикле, так как все 12 точек рисуются одинаковым образом);

# Описание применяемых методов (3-й уровень детализации)

## Типы данных

* Struct Point3D
* Struct Point2D

Struct Point3D

{

X – координата точки по оси OX

Y– координата точки по оси OY

Z– координата точки по оси OZ

}

Struct Point2D

{

X – экранная координата точки по Ширине

Y– экранная координата точки по Высоте

}

## Структуры данных

Ввод данных осуществим спомощью ранее описанного метода – TrackBar.Value. То есть введем нужные типы данных:

x, y, z – установленные значения соответствующих ползунковых переключателей для осей Ox, Oy, Oz;

alfa, betta, gamma – установленные значения соответствующих ползунковых переключателей для углов между горизонтальным вектором, направленным влево и положительными направлениями координатных осей Ox, Oy, Oz;

Center.x, Center.y – координаты середины поля для отрисовки

lenAxis – длина полуоси;

offset – дополнительное смещение;

массив начальных 14-ти 3D-точек baseStorage3DPoints;, содержащий следующие точки (в таком порядке):

1. T– 3D-точка T.

2. Tx, Ty, Tz – точки, соответствующие точке T на координатных осях Ox, Oy, Oz.

3. T3, T2, T1 – проекции точки Tна координатные плоскости.

4. O – точка начала координат.

5. OX, -OX – начальная и конечная точки оси Оx.

6. OY, -OY – начальная и конечная точки оси Oy.

7. OZ, -OZ – начальная и конечная точки оси Oz.

### Переменные для П.Ч.

storageSpacial2D; - массив точек пространственного чертежа;

T’– 3D-точка T.

2. Tx’, Ty’, Tz’ – точки, соответствующие точке T на координатных осях Ox, Oy, Oz.

3. T3’, T2’, T1’ – проекции точки T на координатные плоскости.

4. O’ – точка начала координат.

5. OX’, -OX’ – начальная и конечная точки оси Оx.

6. OY’, -OY’ – начальная и конечная точки оси Oy.

7. OZ’, -OZ’ – начальная и конечная точки оси Oz.

### Переменные для К.Ч.

storageComplex2D; - массив точек для комплексного чертежа.

MO – точка начала координат;

Mx, My3, Mz, My1– конечные точки для осей Ox, Oy, Oz;

MT1, MT2, MT3– проекции точки T на координатные плоскости;

MTx, MTy1, MTy3, MTz – точки, соответствующие точке T на координатных осях Ox, Oy, Oz.

## Укрупненный алгоритм

### Ввод информации о координатах точки и углах между горизонтальным вектором и осями координат

Вводим координаты точки и углымежду горизонтальным вектором и положительными направлениями осей координат при помощи свойства класса ползунковых переключателей TrackBar.Value:

x = trackBarX.Value;

y = trackBarY.Value;

z = trackBarZ.Value;

alfa = trackBarAlfa.Value;

beta = trackBarBeta.Value;

gamma = trackBarGamma.Value.

Вычислим значение длины полуоси по следующей формуле:

lenAxis = min(Center.x, Center.y) – offset,

где offset = min(Center.x, Center.y)/3;

Задаем значения элементов массива baseStorage3DPoints:

(x, y, z),

(x, 0, 0),

(0, y, 0),

(0, 0, z),

(0, y, z),

(x, 0, z),

(x, y, 0),

(0, 0, 0),

(lenAxis, 0, 0),

(-lenAxis, 0, 0),

(0, lenAxis, 0),

(0, -lenAxis, 0),

(0, 0, lenAxis),

(0, 0, -lenAxis)

### Преобразование трехмерных координат точек в двумерные

* 1. *Для П.Ч.*

В цикле каждую из 14-ти трехмерных точек (T, T1, T2, T3, Tx, Ty, Tz, O, OX, OY, OZ, -OX, -OY, -OZ) из массива baseStorage3DPoints преобразуем в двумерную точку (T’, T1’, T2’, T3’, Tx’, Ty’, Tz’, O’, OX’, OY’, OZ’, -OX’, -OY’, -OZ’) из массива storageSpacial2D по формуле (1)

* 1. *Для К.Ч.*

Вычисляем массив двумерных точек

(MT1, MT2, MT3, MTx, MTy1, MTy3, MTz, MO, Mx, My1, My2, Mz,)

storageComplex2D комплексного чертежа для соответствующих трехмерных точек

(T, T1, T2, T3, Tx, Ty, Tz, O, OX, OY, OZ, -OX, -OY, -OZ) из массива baseStorage3DPoints «пространственного» чертежа по формулам:

*Для координат точек* (OX, OY, Tx, Ty1, T1) массива baseStorage3DPoints преобразовать в 2D точки (Mx, My1, MTx, MTy1, MT1) storageComplex2D по формуле **(2)**;

*Для координат точек* (Oz, Tz, T2) массива baseStorage3DPoints преобразовать в 2D точки (Mz, MTz, MT2) массива storageComplex2D по формуле **(3);**

*Для координат точек* (-Oy,Ty, T3) storageComplex2D преобразовать в 2D точки (My3, MTy3, MT3) массива storageComplex2D по формуле **(4)**;

### Визуализация «пространственного» чертежа

1. Отрисовка координатных осей

Необходимо соединить линиями, при помощи метода DrawLine() следующие пары точек:

Это и есть:

storageSpacial2D[8] - storageSpacial2D[9] - ось Х

[Ox’] - [-Ox’] - ось Х

storageSpacial2D[10] - storageSpacial2D[11] - Ось У

[Oy’] - [-Oy’] - Ось У

storageSpacial2D[12]- storageSpacial2D[13] –Ось z

[Oz’]- [-Oz’] - Ось z

1. Отрисовка линий связи между точкой T и ее проекциями:

storageSpacial2D[0] - storageSpacial2D[4]- [T’] - [Tx’],

storageSpacial2D[0] - storageSpacial2D[5] [T’] - [Ty’],

storageSpacial2D[0]- storageSpacial2D[6] [T’]- [Tz’],

storageSpacial2D[1]- storageSpacial2D[5] [T1’]- [Ty’],

storageSpacial2D[1]- storageSpacial2D[6] [T1’]- [Tz’],

storageSpacial2D[2]- storageSpacial2D[4] [T2’]- [Tx’],

storageSpacial2D[2]- storageSpacial2D[6] [T2’]- [Tz’],

storageSpacial2D[3]- storageSpacial2D[4] [T3’]- [Tx’],

storageSpacial2D[3]- storageSpacial2D[5] [T3’]- [Ty’].

1. Отрисовка точек

Необходимо отрисовать все точки (T’, T1’, T2’, T3’, Tx’, Ty’, Tz’, O’, OX’, OY’, OZ’, -OX’, -OY’, -OZ’) из массива storageSpacial2D при помощи DrawEllips().

### Визуализация комплексного чертежа

1. **Отрисовка координатных осей**

Необходимо соединить линиями, при помощи метода DrawLine() следующие пары точек:

[Mx] - [My3] - Ось(X (-Y) – Y(-X));

[Mz] - [My1] - Ось (Z(-Y) – Y(-Z));

:

storageComplex2D[8] - storageComplex2D[9] - Ось(X (-Y) – Y(-X));

storageComplex2D[10] - storageComplex2D[11] - Ось (Z(-Y) – Y(-Z));

1. **Отрисовка линий связи между проекциямиточки T**

Необходимо соединить линиями, при помощи метода DrawLine() следующие пары точек:

(MT1, MT2, MT3, MTx, MTy1, MTy3, MTz, MO, Mx, My1, My3, Mz,)

storageComplex2D[6]- storageComplex2D[0] [MTу1]- [MT1],

storageComplex2D[6]- storageComplex2D[2] [MTу3]- [MT3],

storageComplex2D[2]- storageComplex2D[1] [MT3]- [MT2],

storageComplex2D[1]- storageComplex2D[0] [MT2]- [MT1]

1. **Отрисовка дуги**

Необходимо соединить дугой, при помощи метода DrawArc() следующие пары точек:

(MTy1, MTy3)

:

storageComplex2D[5] - storageComplex2D[6]

**При** MTy1.y > СenterY

Эта отрисовка рассмотрена ниже:

Дуга должна отрисовываться в четвертом квадранте с параметрами:

1) Центр в точке O;

2) Радиус, равный расстоянию от точки O до ty1 (или до точки Ty3);

3) Начальный угол, равный 180 градусов;

4) Полный угол, равный 90 градусов.

**При** MTy1.y < СenterY

Дуга должна отрисовываться во втором квадранте с параметрами:

1) Центр в точке O;

2) Радиус, равный расстоянию от точки O до ty1 (или до точки Ty3);

3) Начальный угол, равный 0 градусов;

4) Полный угол, равный 90 градусов.

1. **Отрисовка точек**

Необходимо отрисовать все точки (MT1, MT2, MT3, MTx, MTy1, MTy3, MTz, MO, Mx, My1, My2, Mz,) из массива storageComplex2D при помощи DrawEllips().

# Детализированный алгоритм (4-й уровень детализации)

**Подготовка функций:**

*Для П.Ч.*

В связи с тем, что координаты каждой точки (T, T1, T2, T3, Tx, Ty, Tz, O, OX, OY, OZ, -OX, -OY, -OZ) вычисляются одинаковым образом (для «пространственного» чертежа), целесообразно реализовать данные вычисления как функцию, основанную на формуле (1):

*Функция\_переработки\_3D\_в\_2D\_П\_Ч(Point3D, alpha, beta, gamma,* CenterX, CenterY*)* **Вернуть** *Point2D*

{

ЕX = (CenterX - *Point3D*.х\* Cos(alpha) - *Point3D*.у \* Cos(beta) - *Point3D*.z \* Cos(gamma));

ЕY = (CenterY + *Point3D*.х \* Sin(alpha) + *Point3D*.y \* Cos(beta) + *Point3D*.z \* Sin(gamma));

*Point2D* = (ЕX, ЕY);

}

*Для К.Ч***.**

Поскольку точки высчитываются по трём разным формулам, то занесем все в функции.

Для точек

(Mx, My1, MTx, MTy1, MT1) **Вернуть** *Point2D*

*Функция\_переработки\_3D\_в\_2D\_К\_Ч\_1(Point3D, CenterX, CenterY)*

*{*

X = (CenterX - *Point3D*.x);

YY = (CenterY + *Point3D*.y);

*Point2D* = (X, YY) *}*

Для точек

(Mz, MTz, MT2) **Вернуть** *Point2D*

*Функция\_переработки\_3D\_в\_2D\_К\_Ч\_2(Point3D, CenterX, CenterY)*

*{*

X =(CenterX - *Point3D*.x);

YY = (CenterY - *Point3D*.z);

*Point2D* = (X, YY);

*}*

Для точек

(My3, MTy3, MT3)

*Функция\_переработки\_3D\_в\_2D\_К\_Ч\_3(Point3D, CenterX, CenterY)* **Вернуть** *Point2D*

*{*

X = (CenterX + *Point3D*.y);

YY =(CenterY - *Point3D*.z);

*Point2D* = (X, YY);

*}*

## Ввод координаты точек и углов

Считываем данные с ползунковых переключателей, производим необходимые действия и заносим данные в массив 14-ти 3D-точек baseStorage3DPoints.:

x = trackBarX.Value;

y = trackBarY.Value;

z = trackBarZ.Value;

alfa = trackBarAlfa.Value;

betta = trackBarBeta.Value;

gamma = trackBarGamma.Value;

baseStorage3DPoints =

{

(x, y, z), (x, 0, 0), (0, y, 0),(0, 0, z),(0, y, z),(x, 0, z),(x, y, 0),(0, 0, 0),

(lenAxis, 0, 0),(-lenAxis, 0, 0),(0, lenAxis, 0),(0, -lenAxis, 0),(0, 0, lenAxis),

(0, 0, -lenAxis)

};

## Преобразование для П.Ч.

Цикл от 0 до 13 выполнить:

storageSpacial2D*[i]=Функция\_переработки\_3D\_в\_2D\_П\_Ч (baseStorage3DPoints[i], alpha, beta, gamma)*

## Преобразования для К.Ч.

Вычислить 12-ть точек комплексного чертежа storageComplex2D используя соответствующие функции;

Цикл от 1 до 11 выполнить:

Если *baseStorage3DPoints[i].Z=0*

storageComplex2D = *Функция\_переработки\_3D\_в\_2D\_К\_Ч\_1 (baseStorage3DPoints[i], CenterX, CenterY)*

Если *baseStorage3DPoints[i].Y=0*

storageComplex2D = *Функция\_переработки\_3D\_в\_2D\_К\_Ч\_2 (baseStorage3DPoints[i], CenterX, CenterY)*

Если *baseStorage3DPoints[i].X=0*

storageComplex2D = *Функция\_переработки\_3D\_в\_2D\_К\_Ч\_3 (baseStorage3DPoints[i], CenterX, CenterY)*

## Визуализация для П.Ч.

1. ***Отрисовка координатных осей***

DrawLine(storageSpacial2D[8] - storageSpacial2D[9])

[Ox’] - [-Ox’] - ось Х

DrawLine(storageSpacial2D[10] - storageSpacial2D[11])

[Oy’] - [-Oy’] - Ось У

DrawLine(storageSpacial2D[12]- storageSpacial2D[13])

[Oz’]- [-Oz’] - Ось z

1. ***Отрисовка линий связи между точкой T и ее проекциями***

DrawLine(storageSpacial2D[0] - storageSpacial2D[4]) – T’-T1’

DrawLine (storageSpacial2D[0] - storageSpacial2D[5]) – T’-T2’

DrawLine (storageSpacial2D[0]- storageSpacial2D[6]) – T’-T3’

DrawLine (storageSpacial2D[1]- storageSpacial2D[5]) – T3’-Tу’

DrawLine (storageSpacial2D[1]- storageSpacial2D[6]) - T3’-Tz’

DrawLine (storageSpacial2D[2]- storageSpacial2D[4]) - T2’-Tx’

DrawLine (storageSpacial2D[2]- storageSpacial2D[6]) - T2’-Tz’

DrawLine (storageSpacial2D[3]- storageSpacial2D[4]) - T1’-Tx’

DrawLine( storageSpacial2D[3]- storageSpacial2D[5]) – T1’-Ty’

функция отрисовки линий связи:

1. Отрисовка точек

Цикл от 0 до 13 выполнить:

{

DrawEllipse(storageSpacial2D[i].x,storageSpacial2D[i].y);

}

## Визуализация К.Ч.

1. **Отрисовка координатных осей**

DrawLine(storageComplex2D[8] - storageComplex2D[9]) - Ось(X (-Y) – Y(-X));

DrawLine(storageComplex2D[10] - storageComplex2D[11]) - Ось (Z(-Y) – Y(-Z));

1. **Отрисовка линий связи между проекциямиточки T**

DrawLine(storageComplex2D[6]- storageComplex2D[0]): [MTу1]- [MT1],

DrawLine (storageComplex2D[6]- storageComplex2D[2]): [MTу3]- [MT3],

DrawLine (storageComplex2D[2]- storageComplex2D[1]): [MT3]- [MT2],

DrawLine (storageComplex2D[1]- storageComplex2D[0]): [MT2]- [MT1]

1. **Отрисовка дуги**

d = 2\*|Center.y – Ty3.y|;

Если storageComplex2D[5].у < CenterY

{

DrawArc(CenterX - d / 2) + 3, (CenterX - d / 2) + 3, d, d, 180, 90);

}

Или если storageComplex2D[5].у > CenterY

{

DrawArc(CenterX - d / 2) + 3, CenterY - d / 2) + 3, d, d, 0, 90);

}

1. Отрисовка точек

Цикл от 0 до длинны массива storageComplex2D выполнить:

{

DrawEllipse(storageComplex2D [i].x, storageComplex2D [i].y);

}

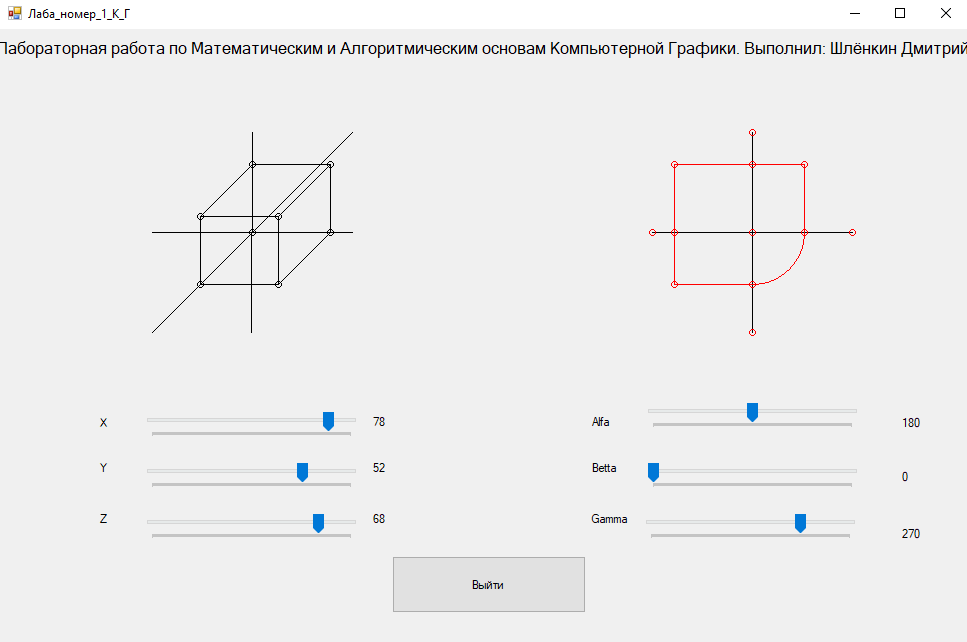
# Руководство программиста

Данная лабораторная работа разработана на платформе Microsoft .NETF rameworkверсии 4.9.0, в среде разработки VisualStudio 2019 версии 10.0.3,на языке программирования C#.

Программа состоит из двух частей – Интерфейса и Прикладной части.

## Интерфейс

Проект сделан на Form.(Рисунок 7.1)



Button1

TrackBar6

TrackBar5

TrackBar4

TrackBar1

TrackBar3

TrackBar2

Label13

Label12

Label11

Label10

Label7

Label8

Label9

Label4

Label5

Label6

Label3

Label1

Label2

PictureBox2

PictureBox1

Рисунок 7.1 - Вид формы

* Label

Label1 – подпись для координаты X

Label2 - подпись для координаты Y

Label3 - подпись для координаты Z

Label4- подпись Вывода для координаты X

Label5 - подпись Вывода для координаты Y

Label6 - подпись Вывода для координаты Z

Label7 - подпись угла между горизонтальным вектором, направленным влево и положительным направлением Ox

Label8 - подпись угла между горизонтальным вектором, направленным влево и положительным направлением Oy

Label9 - подпись угла между горизонтальным вектором, направленным влево и положительным направлением Oz

Label10 - подпись вывода угла между горизонтальным вектором, направленным влево и положительным направлением Ox

Label11 - подпись вывода угла между горизонтальным вектором, направленным влево и положительным направлением Oy

Label12 - подпись вывода угла ввод угла между горизонтальным вектором, направленным влево и положительным направлением Oz

Label13 – Подпись автора

* PictureBox

PictureBox1 - вывод «пространственного» чертежа

PictureBox2 вывод комплексного чертежа

* TrackBar

TrackBar1 - Ввод для координаты X

TrackBar2 - Ввод для координаты Y

TrackBar3 - Ввод для координаты Z

TrackBar4 - Ввод угла между горизонтальным вектором, направленным влево и положительным направлением Ox

TrackBar5 - Ввод угла между горизонтальным вектором, направленным влево и положительным направлением Oy

TrackBar6 - Ввод угла между горизонтальным вектором, направленным влево и положительным направлением Oz

* Button

Button1 – Закрытие программы

## Прикладная часть

Программа состоит из трёх внутренних классов:

1. **FromForm()**
2. **CPoint3D**
3. **CPoint2D**

И одного внешнего класса

1. **Form1**

Рассмотрим более подробно классы:

1. **FromForm()**

Внутренний класс, содержащий такие процедуры, как

* Input
* createstorageSpacial3D
* createstorageSpacial2D
* Paint\_Ortogonal
* Paint\_Complex

**Paint\_Ortogonal** – процедура, для построения аксонометрического чертежа. В ней содержится отрисовка точек для пространственного чертежа.

**Paint\_Complex** – аналогичная процедура для построение комплексного чертежа. В ней так же отрисовка всего чертежа.

**Input** – процедура, высчитывающая:

CPoint3D[] storageSpacial3D[14] - поле(массив), содержащее 14 начальных 3D-точек

int xCenter; - поле, содержащее координатуx точки О;

int yCenter; - поле, содержащее координатуy точки O;

double alfa; поле, содержащее угол между горизонтальным вектором экрана, направленным влево и положительным направлением Ox;

double betta; поле, содержащее угол между горизонтальным вектором экрана, направленным влево и положительным направлением Oy;

double gamma; поле, содержащее угол между горизонтальным вектором экрана, направленным влево и положительным направлением Oz

**createstorageSpacial3D** – процедура для обработки точек для пространственного чертежа, в которой вызывается функция переработки набора 3D точек в 2D:

storageSpacial2D *= Calculate\_3D\_in\_2D\_Ox (baseStorage3DPoints, alpha, beta, gamma)*

*{*

*}*

**createstorageSpacial2D** – процедура для обработки точек для комплексного чертежа:

Цикл от 0 до 11:

в котором вызывается функция переработки:

Для обработки точек (Mx, My1, MTx, MTy1, MT1)

Если *baseStorage3DPoints[i].Z=0*

storageComplex2D=*Calculate\_3D\_in\_2D\_Co\_1(baseStorage3DPoints, CenterX, CenterY)*

*{*

*}*

Для обработки точек (Mz, MTz, MT2)

Если *baseStorage3DPoints[i].Y=0*

storageComplex2D=*Calculate\_3D\_in\_2D\_Co\_2(baseStorage3DPoints, CenterX, CenterY)*

*{*

*}*

Для обработки точек (My3, MTy3, MT3)

Если *baseStorage3DPoints[i].X=0*

storageComplex2D=*Calculate\_3D\_in\_2D\_Co\_3(baseStorage3DPoints, CenterX, CenterY)*

*{*

*}*

1. **CPoint3D** - Внутренний класс программы, описывающий 3Dточку и содержащий следующие поля и методы:

int x; поле, содержащее координату трехмерной точки по оси Ox;

int y; поле, содержащее координату трехмерной точки по оси Oу;

int z; поле, содержащее координату трехмерной точки по оси Oz;

CPoint3D(int x, int y, int z)

Конструктор класса с 3-мя параметрами: координаты трехмерной точки по каждой из трех осей координат.

1. **CPoint2D**

Внутренний класс программы, описывающий 2D точку и содержащий следующие поля и методы:

int x; поле, содержащее координату двумерной точки по оси Ox;

int y; поле, содержащее координату двумерной точки по оси Oу;

CPoint2D(intx, inty)

конструктор класса с 2-мя параметрами: координаты трехмерной точки по каждой из двух осей координат.

1. **Form1()**

Внешний класс приложения, служит для вызова процедур с внутреннего класса.

Содержит:

1. private void Form1\_Paint(object sender, PaintEventArgs e)
2. public void trackBar3\_Scroll(object sender, EventArgs e)
3. public void trackBar2\_Scroll(object sender, EventArgs e)
4. public void trackBar1\_Scroll(object sender, EventArgs e)
5. public void trackBar4\_Scroll(object sender, EventArgs e)
6. public void trackBar5\_Scroll(object sender, EventArgs e)
7. public void trackBar6\_Scroll(object sender, EventArgs e)
8. private void button1\_Click(object sender, EventArgs e)

Покажем на функциональной схеме, как взаимодействуют между собой функции и методы, а также как происходит их взаимодействие с элементами интерфейса (Рисунок 7.2):

Trackbar6

Trackbar5

Trackbar4

Trackbar3

Trackbar2

Trackbar1

Alpha = trackbar1.value

….

…..

**Input()**

*Calculate\_3D\_in\_2D\_Co\_1 ()*

*Calculate\_3D\_in\_2D\_Co\_2 ()*

*Calculate\_3D\_in\_2D\_Co\_2*

*Calculate\_3D\_in\_2D\_Ox*

**createstorageSpacial2D()**

**createstorageSpacial3D()**

DrawPoints**()**

DrawPoints()

**Paint\_Complex()**

**Paint\_Ortogonal()**

DrawLines**()**

DrawLines()

DrawArc()

Label10

PictureBox2

PictureBox1

Label5

Label11

Label4

Label6

Label12

Рисунок 7.2 - Функциональная схема программы

## Как работает программа

При запуске программы происходит инициализация главной формы и с главной формы идет вызов процедуры Input(), в которую отправляется информация о текущем положении трэкбара, спомощью Value (Input (Trackbar1.Value, Trackbar2.Value, Trackbar3.Value,Trackbar4.Value, Trackbar5.Value, Trackbar6.Value)). В этой процедуре происходит присвоение нужным переменным сооветствующие значения ((X, Y, Z, *alpha, beta, gamma) – информация с трэкбаров,* CenterX, CenterY, lenAxis,). Происходит создание массива 3D точек baseStorage3DPoints спомощью класса CPoint3D.

Далее вызывается процедура **createstorageSpacial3D**, которая, в совю очередь в цикле вызывает функцию для вычисления точек «пространственного» чертежа в цикле от 0 до 13:

storageSpacial2D *=Calculate\_3D\_in\_2D\_Ox (*baseStorage3DPoints[i], *alpha, beta, gamma).*

После этого вызывается процедура **createstorageСomplex2D**, которая, в совою очередь вызывает функции для вычисления точек для комплексного чертежа Для разных точек:

Цикл от 0 до 11:

в котором вызывается функция переработки:

Для обработки точек (Mx, My1, MTx, MTy1, MT1)

Если *baseStorage3DPoints[i].Z=0*

storageComplex2D=*Calculate\_3D\_in\_2D\_Co\_1(baseStorage3DPoints[i], CenterX, CenterY)*

*{*

*}*

Для обработки точек (Mz, MTz, MT2)

Если *baseStorage3DPoints[i].Y=0*

storageComplex2D=*Calculate\_3D\_in\_2D\_Co\_2(baseStorage3DPoints[i], CenterX, CenterY)*

*{*

*}*

Для обработки точек (My3, MTy3, MT3)

Если *baseStorage3DPoints[i].X=0*

storageComplex2D=*Calculate\_3D\_in\_2D\_Co\_3(baseStorage3DPoints[i], CenterX, CenterY)*

*{*

*}*

Далее, из главной формы идет вызов процедур для отрисовки чертежей: Paint\_Ortogonal(pictureBox1) и Paint\_Complex(pictureBox2). Они содержат методы для отрисовки точек линий связи и отрисовки координатных осей. (DrawPoints(),DrawLines()) – в Paint\_Ortogonal, и (DrawPoints(),DrawLines(), DrawArc()) – в Paint\_Complex.

В итоге пространственный и комплексный чертежи выводятся соответственно в “пикчербоксы” (pictureBox1 и pictureBox2), а текущие значения координат точки T: x, y, z, и углов между горизонтальным вектором, направленным влево и положительными направлениями осей: Ox, Oy, Oz выводятся соответственно в “лэйблы” (Label4, Label5, Label6, Label10, Label11, Label12)

При нажатии на кнопку (Button) button1 (выход из программы), программа завершает свою работу.

При изменении положения ползунковых переключателей (Trackbar1, Trackbar2, Trackbar3,Trackbar4, Trackbar5, Trackbar6) вызывается событие Scroll, в обработке которого фиксируются значения, и передаются в функцию Input() и производятся все вышеописанные действия.

Кроме того, событие Scroll вызывает функцию обновления (перерисовки) изображения Refresh() (для изображений пространственного и комплексного чертежей в случае изменения значений координат точки T и только для пространственного чертежа, в случае изменения значений углов между горизонтальным вектором, направленным влево, и положительными направлениями осей координат).

# Руководство пользователя

Данная программа предназначена для визуализации «пространственного» и комплексного чертежа 3D-точки T.

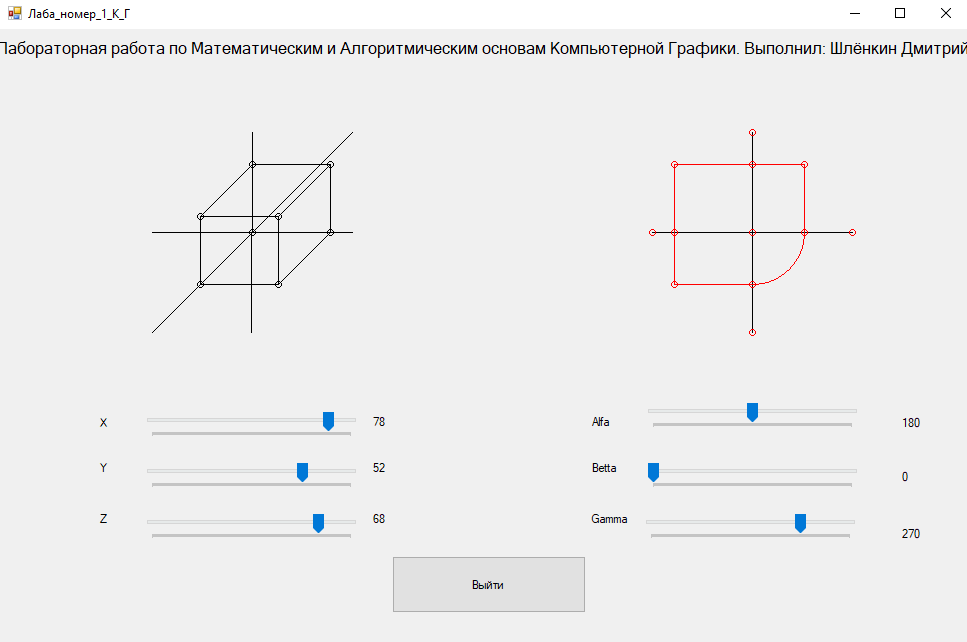


Рисунок 8.1 – Приложение

Рассмотрим рисунок 8.1

Левое изображение является изображением «пространственного» чертежа. Правое изображение является изображением комплексного чертежа.

Для того, чтобы задать нужные значения координат точки T: x, y, z необходимо использовать соответствующие ползунковые переключатели в левой нижней части экрана.

Для того, чтобы задать нужные значения углов между горизонтальным вектором, направленным влево и положительными направлениями осей Ox,

Oy, Oz: α, β, γнеобходимо использовать соответствующие ползунковые переключатели в правой нижней части экрана

Минимальные и максимальные возможные значения величин находятся рядом (соответственно с левой и правой сторон) с соответствующими им ползунковым переключателями. Для ползунковых переключателей, задающих значения координат точки Tтакже обозначен переход через 0. Справа от каждого ползункового переключателя отображено текущее значение, зафиксированное пользователем.

Для выхода из программы необходимо нажать кнопку “Выйти из программы”, находящуюся снизу, посередине экрана.

1. Координаты проекций, осей и точки Т (3D). [↑](#footnote-ref-1)
2. Точки 2D для Пространственного чертежа. [↑](#footnote-ref-2)
3. Переработанные 2D точки для К.Ч. [↑](#footnote-ref-3)