

6 Векторная графика. Координатный метод

Векторная графика базируется на координатном методе, который был введен в XVII веке французскими математиками Р. Декартом и П. Ферма. На этом методе базируется аналитическая геометрия, которую можно считать фундаментом компьютерной графики. В современной компьютерной графике широко используется координатный метод, т.к.:

- Каждая точка на экране (или на бумаге при печати на принтере) задается координатами — например, пикселями.
- Координаты используются для описания объектов, которые будут отображаться как пространственные. Например, объекты микромира, объекты на поверхности Земли, объекты космического пространства и тому подобное. Даже тогда, когда отображается нечто, не имеющее привязки к положению в пространстве (например, случайные цветовые пятна в каком-то видеоэффекте), то здесь используются координаты для учета взаиморасположения отдельных элементов.
- При выполнении многих промежуточных действий для отображения объекта используют разные системы координат и преобразования из одной системы в другую.

В основе координатного метода лежат координатные преобразования. Координатный метод лежит в основе векторной графики. Поэтому остановимся более подробно на изучении координатных преобразований.

6.1 Преобразование координат

Сначала рассмотрим общие вопросы преобразования координат. Пусть задана n -мерная система координат в базисе (k_1, k_2, \dots, k_n) , описывающая положения точки в пространстве с помощью числовых значений k_i . В компьютерной графике чаще все используется двумерная ($n=2$) и трехмерная ($n=3$) системы координат.

Если задать другую N -мерную систему координат в базисе (m_1, m_2, \dots, m_N) и поставить задачу определения координат объекта в новой системе, зная

координаты в старой, то решение (если оно существует) можно записать в таком виде:

$$\begin{cases} m_1 = f_1(k_1, k_2, \dots, k_n), \\ m_2 = f_2(k_1, k_2, \dots, k_n), \\ \dots \\ m_N = f_N(k_1, k_2, \dots, k_n), \end{cases}$$

где f_i — функция пересчета i -й координаты,

аргументы — координаты в системе (k_1, k_2, \dots, k_n) .

Можно поставить и обратную задачу — по известным координатам (m_1, m_2, \dots, m_N) определить координаты (k_1, k_2, \dots, k_n) . Решение обратной задачи запишется так:

$$\begin{cases} k_1 = F_1(m_1, m_2, \dots, m_N), \\ k_2 = F_2(m_1, m_2, \dots, m_N), \\ \dots \\ k_n = F_n(m_1, m_2, \dots, m_N), \end{cases}$$

где F_i — функция *обратного* преобразования.

В случае, когда размерности систем координат не совпадают ($n \neq N$), осуществить однозначное преобразование координат зачастую не удастся. Например, по двумерным экраным координатам нельзя без дополнительных условий однозначно определить трехмерные координаты отображаемых объектов. Если размерности систем совпадают ($n = N$), то также возможны случаи, когда нельзя однозначно решить прямую или обратную задачи.

Преобразование координат классифицируется:

- по системам координат — например, преобразование из полярной системы координат в прямоугольную (декартову);
- по виду функций преобразования f_i .

По виду функций преобразования различают *линейные* и *нелинейные* преобразования. Если при всех $i = 1, 2, \dots, n$ функции f_i — линейные относительно аргументов (k_1, k_2, \dots, k_n) , то есть

$$f_i = a_{i1}k_1 + a_{i2}k_2 + \dots + a_{in}k_n + a_{i,n+1},$$

где a_{ij} — константы, то такие преобразования называются линейными, а при $n = N$ — *аффинными*.

Если хотя бы для одного i функция f_i — нелинейная относительно (k_1, k_2, \dots, k_n) , тогда преобразование координат в целом не линейно.

Например, преобразование

$$\begin{aligned} X &= 3x + 5y \\ Y &= 4xy + 10y \end{aligned}$$

нелинейное, так как в выражении для Y присутствует xy .

Линейные преобразования наглядно записываются в матричной форме:

$$\begin{bmatrix} m_1 \\ m_2 \\ \dots \\ m_N \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} & a_{1,n+1} \\ a_{21} & a_{22} & \dots & a_{2n} & a_{2,n+1} \\ \dots & \dots & \dots & \dots & \dots \\ a_{N1} & a_{N2} & \dots & a_{Nn} & a_{N,n+1} \end{bmatrix} \cdot \begin{bmatrix} k_1 \\ k_2 \\ \dots \\ k_n \end{bmatrix}.$$

Здесь матрица коэффициентов (a_{ij}) умножается на матрицу-столбец (k_i) и в результате получается матрица столбец (m_i) .

6.2 Аффинные преобразования на плоскости

На плоскости любую точку представляют с помощью 2-х ее координат. Их

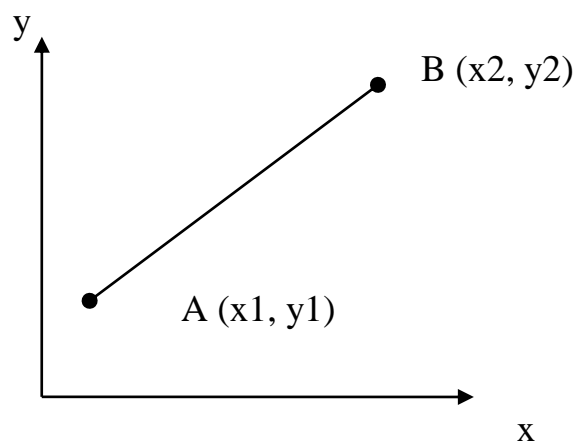


Рисунок 6.1

значения можно рассматривать как элементы вектора-строки $[x, y]$ или вектора-столбца $\begin{bmatrix} x \\ y \end{bmatrix}$.

Последовательность точек, образующих объект, может быть представлена в виде матрицы чисел. Например, отрезок на рисунке 6.1 можно представить следующим образом:

$$\begin{bmatrix} x1 & y1 \\ x2 & y2 \end{bmatrix} \text{ или } \begin{bmatrix} x1 & x2 \\ y1 & y2 \end{bmatrix}.$$

Другой пример, фигуру на рисунке 6.2 можно записать так:

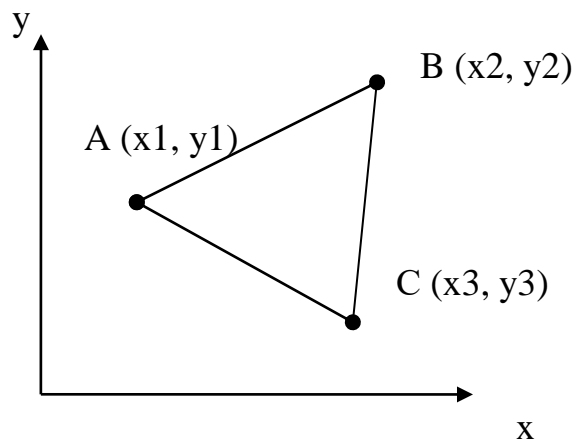


Рисунок 6.2

$$\begin{bmatrix} x1 & y1 \\ x2 & y2 \\ x3 & y3 \end{bmatrix} \text{ или } \begin{bmatrix} x1 & x2 & x3 \\ y1 & y2 & y3 \end{bmatrix}$$

В результате многие задачи можно привести к следующей формулировке.

Пусть даны матрицы A и B и задана их взаимосвязь: $AT = B$; необходимо найти матрицу

преобразования T .

С другой стороны, матрицу T можно рассматривать как оператор, а умножение A на T - как геометрическое преобразование над системой точек, содержащихся в матрице A . При этом матрицы A и T должны быть известны. Такая интерпретация является основой математических преобразований, используемых в КГ.

6.2.1 Преобразование точек

Рассмотрим результаты матричного умножения матрицы $\begin{bmatrix} x & y \end{bmatrix}$, определяющей точку P на плоскости, и матрицы преобразования 2×2 общего вида:

$$\begin{bmatrix} x & y \end{bmatrix} \cdot \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} (ax + cy) & (bx + dy) \end{bmatrix} = \begin{bmatrix} x^* & y^* \end{bmatrix} \quad (6.1)$$

Эта запись означает, что координаты точки $P(x, y)$ преобразованы в $P^*(x^*, y^*)$, где $x^* = a \cdot x + c \cdot y$; $y^* = b \cdot x + d \cdot y$. Проведем анализ этого преобразования, рассматривая отдельные частные случаи.

1) $a = d = 1, c = b = 0$

$$\begin{bmatrix} x & y \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} x & y \end{bmatrix} = \begin{bmatrix} x^* & y^* \end{bmatrix} \quad (6.2)$$

Положение точки P не изменилось.

$$2) \quad d = 1, b = c = 0$$

$$\begin{bmatrix} x & y \end{bmatrix} \cdot \begin{bmatrix} a & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} ax & y \end{bmatrix} = \begin{bmatrix} x^* & y^* \end{bmatrix} \quad (6.3)$$

Получили изменение масштаба по оси x : $x^* = a \cdot x$ или перемещение точки P по оси x . Если $a > 1$, то происходит увеличение масштаба (перемещение вправо); если $0 < a < 1$, то уменьшение масштаба (перемещение влево). Если $a < 0$, то происходят аналогичные изменения, но с отображением относительно оси y .

$$3) \quad a = 1, b = c = 0$$

$$\begin{bmatrix} x & y \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \\ 0 & d \end{bmatrix} = \begin{bmatrix} x & dy \end{bmatrix} = \begin{bmatrix} x^* & y^* \end{bmatrix} \quad (6.4)$$

Здесь происходят аналогичные изменения масштаба, но по оси y , и отображение (если $d < 0$) относительно оси x .

$$4) \quad b = c = 0$$

$$\begin{bmatrix} x & y \end{bmatrix} \cdot \begin{bmatrix} a & 0 \\ 0 & d \end{bmatrix} = \begin{bmatrix} ax & dy \end{bmatrix} = \begin{bmatrix} x^* & y^* \end{bmatrix} \quad (6.5)$$

Изменение масштаба (перемещение) происходит по обеим осям (если $a = d$, то координаты изменяются пропорционально), отображение, если $a < 0$ или $d < 0$, то относительно осей y или x соответственно, а если $a = d < 0$ вместе, то относительно начала координат.

$$5) \quad a = d = 1, c = 0$$

$$\begin{bmatrix} x & y \end{bmatrix} \cdot \begin{bmatrix} 1 & b \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} x + 0y & bx + 1y \end{bmatrix} = \begin{bmatrix} x^* & y^* \end{bmatrix} \quad (6.6)$$

Координата x точки P^* не изменяется, а координата y линейно зависит от начала координат. Этот эффект называется смещением.

$$6) \quad a = d = 1, b = 0$$

$$\begin{bmatrix} x & y \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \\ c & 1 \end{bmatrix} = \begin{bmatrix} x + cy & y \end{bmatrix} = \begin{bmatrix} x^* & y^* \end{bmatrix} \quad (6.7)$$

Здесь все аналогично, но смещение происходит вдоль оси x на величину cy .

В заключение отметим, что преобразование общего вида (6.1), примененное к началу координат, не изменяет положения начала координат, т.е.:

$$\begin{bmatrix} 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} 0 & 0 \end{bmatrix} \quad (6.8)$$

Иными словами, начало координат инвариантно к преобразованию 2×2 . Это является недостатком такого преобразования, который будет преодолен введением однородных координат.

6.2.2 Преобразование прямых линий

Прямая линия может быть задана координатами двух ее точек. Отрезок прямой задается координатами его концов. Покажем, что преобразование 2×2 преобразует прямую линию в другую прямую линию.

Пусть координаты концов отрезка равны $A = [x_1 \ y_1]$, $B = [x_2 \ y_2]$

В результате преобразования 2×2 получим:

$$A^* = [x_1 \ y_1] \cdot \begin{bmatrix} a & b \\ c & d \end{bmatrix} = [ax_1 + cy_1 \ bx_1 + dy_1] = [x_1^* \ y_1^*] \quad (6.9)$$

$$B^* = [x_2 \ y_2] \cdot \begin{bmatrix} a & b \\ c & d \end{bmatrix} = [ax_2 + cy_2 \ bx_2 + dy_2] = [x_2^* \ y_2^*]$$

Преобразование отрезка можно записать в более компактной форме, представив его в виде матрицы координат концов

$$L = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix}$$

Тогда преобразованный отрезок получится как

$$L^* = LT = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix} * \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} ax_1 + cy_1 & bx_1 + dy_1 \\ ax_2 + cy_2 & bx_2 + dy_2 \end{bmatrix} = \begin{bmatrix} x_1^* & y_1^* \\ x_2^* & y_2^* \end{bmatrix} \quad (6.10)$$

Получили одинаковые результаты (формулы (6.9) и (6.10)), т.е. матрица преобразования 2×2 преобразует прямую линию в другую прямую линию. Говоря об отрезке, легко показать, что между точками исходного и преобразованного отрезка существует взаимно однозначное соответствие. Концы отрезка преобразуются в концы нового отрезка. Покажем, что средняя точка преобразуется в среднюю точку.

Координаты средней точки исходного отрезка равны:

$$M = \left[\frac{x_1 + x_2}{2} \quad \frac{y_1 + y_2}{2} \right]; \quad (6.11)$$

Координаты средней точки преобразованного отрезка

$$M^* = \left[\frac{x_1^* + x_2^*}{2} \quad \frac{y_1^* + y_2^*}{2} \right] = \left[\frac{a(x_1 + x_2) + c(y_1 + y_2)}{2} \quad \frac{b(x_1 + x_2) + d(y_1 + y_2)}{2} \right] \quad (6.12)$$

Применение преобразования 2x2 к точке М дает:

$$MT = \left[\frac{x_1 + x_2}{2} \quad \frac{y_1 + y_2}{2} \right] * \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \left[\frac{a(x_1 + x_2)}{2} + \frac{c(y_1 + y_2)}{2} \quad \frac{b(x_1 + x_2)}{2} + \frac{d(y_1 + y_2)}{2} \right], \quad (6.13)$$

то есть точку M^* .

Очевидно, что это соответствие выполнится для любой точки отрезка, делящей его в соотношении 1/к (во всех выражениях вместо «2» подставить «к»), т.е. вообще для любой точки отрезка.

Итак: при применении компьютерной графики для того, чтобы получить преобразованный отрезок, необходимо преобразовать его концы и соединить полученные точки.

6.2.3 Параллельные линии

Покажем, что преобразование 2x2 преобразует две параллельные линии в две другие параллельные линии.

Пусть даны две линии

$$L_1 = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix} \quad \text{и} \quad L_2 = \begin{bmatrix} x_3 & y_3 \\ x_4 & y_4 \end{bmatrix} \quad \text{параллельные друг другу. Значит, тангенс}$$

угла наклона этих линий одинаков и равен

$$m_1 = \frac{y_2 - y_1}{x_2 - x_1} = \frac{y_4 - y_3}{x_4 - x_3} \quad (6.14)$$

Преобразование L_1 по формуле (6.1) даст линию L_1^*

$$L_1^* = L_1 T = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \end{bmatrix} * \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} ax_1 + cy_1 & bx_1 + dy_1 \\ ax_2 + cy_2 & bx_2 + dy_2 \end{bmatrix} = \begin{bmatrix} x_1^* & y_1^* \\ x_2^* & y_2^* \end{bmatrix}$$

Ее наклон:

$$m_2 = \frac{y_2^* - y_1^*}{x_2^* - x_1^*} = \frac{b(x_2 - x_1) + d(y_2 - y_1)}{a(x_2 - x_1) + c(y_2 - y_1)} = \frac{b + d \frac{y_2 - y_1}{x_2 - x_1}}{a + c \frac{y_2 - y_1}{x_2 - x_1}} = \frac{b + dm_1}{a + cm_1} \quad (6.15)$$

Для L_2 - аналогично получаем: наклон прямой L_2^* равен:

$$m_2 = \frac{y_4^* - y_3^*}{x_4^* - x_3^*} = \frac{b(x_4 - x_3) + d(y_4 - y_3)}{a(x_4 - x_3) + c(y_4 - y_3)} = \frac{b + d \frac{y_4 - y_3}{x_4 - x_3}}{a + c \frac{y_4 - y_3}{x_4 - x_3}} = \frac{b + dm_1}{a + cm_1} \quad (6.16)$$

Получили одинаковые результаты (формулы (6.15) и (6.16)). Значит, линии L_1^* и L_2^* имеют одинаковый наклон, т.е. они параллельны. Следовательно, преобразование 2×2 преобразует параллельные линии в параллельные.

Это означает, в частности, что преобразованием 2×2 параллелограмм преобразуется в другой параллелограмм.

6.2.4 Пересекающиеся линии

Преобразование пересекающихся линий рассмотрим на примере (рис.6.3).

Дано 4 точки и следующее преобразование над ними:

$$\begin{matrix} A: \\ B: \\ C: \\ D: \end{matrix} \begin{bmatrix} 0 & 4 \\ 4 & 0 \\ -1 & -1 \\ 3 & 3 \end{bmatrix} * \begin{bmatrix} 1 & 2 \\ 1 & -2 \end{bmatrix} = \begin{bmatrix} 4 & -8 \\ 4 & 8 \\ -2 & 0 \\ 6 & 0 \end{bmatrix} \begin{matrix} A^* \\ B^* \\ C^* \\ D^* \end{matrix}$$

Точка пересечения
прямых АВ и CD и ее
преобразование:

$$\begin{bmatrix} 2 & 2 \end{bmatrix} * \begin{bmatrix} 1 & 2 \\ 1 & -2 \end{bmatrix} = \begin{bmatrix} 4 & 0 \end{bmatrix}$$

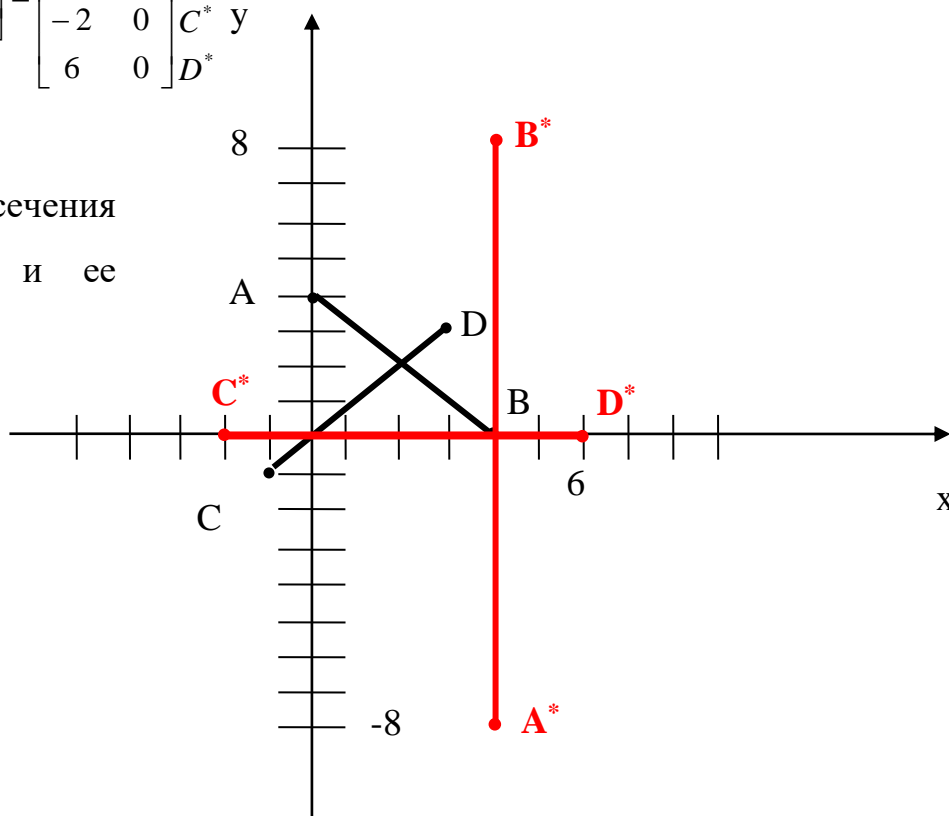


Рисунок 6.3

Видно, что точка пересечения прямых АВ и CD преобразуется в точку пересечения прямых A^*B^* и C^*D^* . Этот вывод является общим для любых пересекающихся прямых.

6.2.5 Преобразование плоских фигур

Рассмотрим преобразование матрицей 2×2 плоских фигур на примере треугольника.

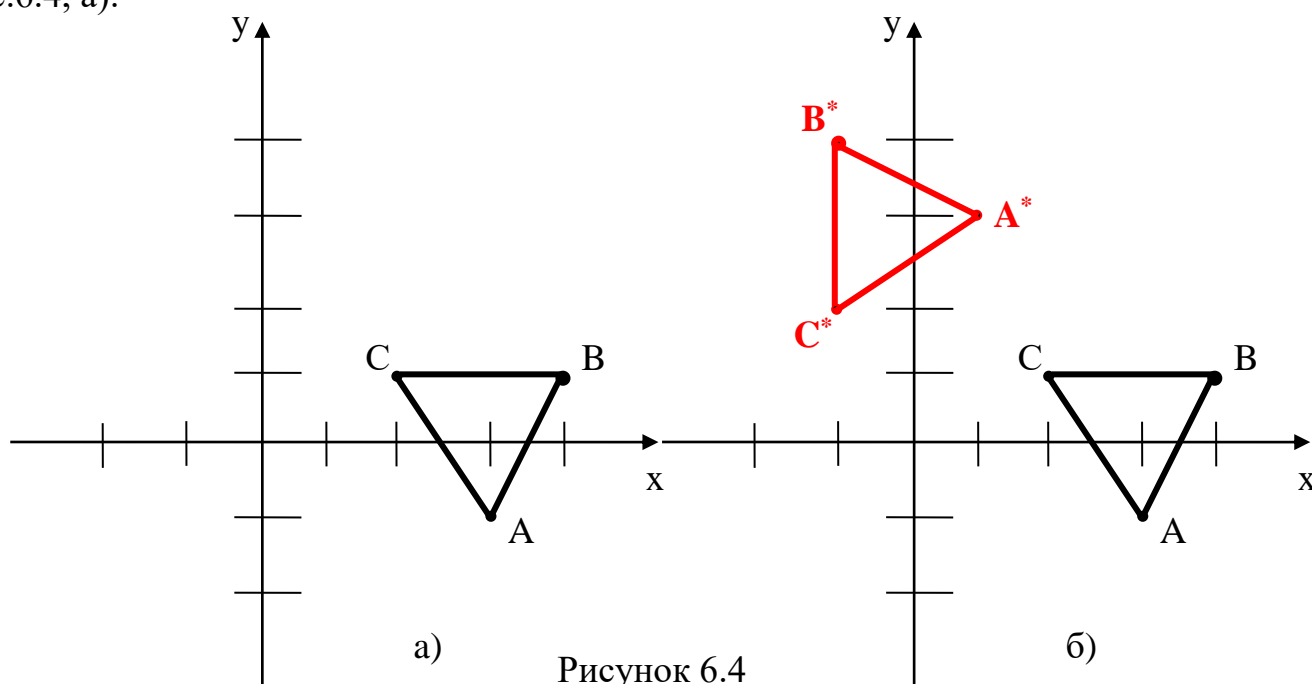
1) Вращение

Вращение вокруг начала координат против часовой стрелки на угол 90° реализуется матрицей $T = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$.

На угол 180° - матрицей $T = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$.

На угол 270° - матрицей $T = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$

Например, дан треугольник с координатами вершин $A(3, -1)$, $B(4, 1)$, $C(2, 1)$ (рис.6.4, а):



Осуществим поворот на 90° (рисунок 6.4, б):

$$\begin{matrix} A: \\ B: \\ C: \end{matrix} \begin{bmatrix} 3 & -1 \\ 4 & 1 \\ 2 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 3 \\ -1 & 4 \\ -1 & 2 \end{bmatrix} \begin{matrix} A^* \\ B^* \\ C^* \end{matrix}$$

Отметим, что вращение осуществляется вокруг оси, перпендикулярной к плоскости XOY.

2) Отображение

Отображение относительно прямой $Y = X$ осуществляется матрицей

$$T = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

Отображение относительно прямой $Y=0$ (оси OX) - матрицей $T = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$.

Отображение относительно прямой $X=0$ (оси OY) - матрицей $T = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$.

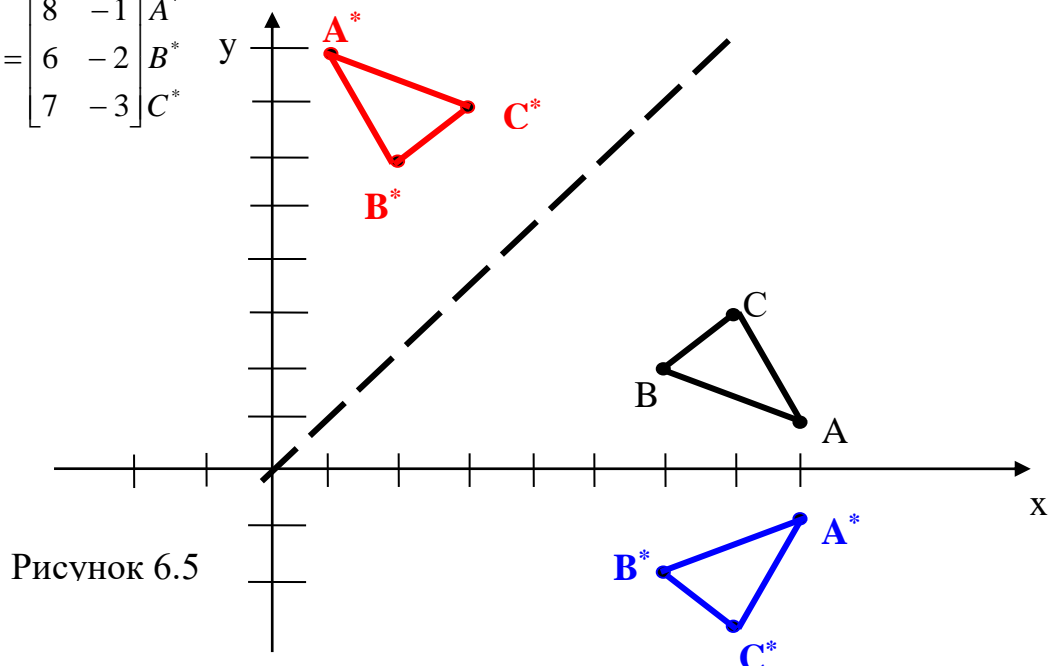
Пример.

Осуществим отображение относительно прямой $Y=X$: (рисунок 6.5, красный треугольник):

$$\begin{matrix} A: \\ B: \\ C: \end{matrix} \begin{bmatrix} 8 & 1 \\ 6 & 2 \\ 7 & 3 \end{bmatrix} * \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 8 \\ 2 & 6 \\ 3 & 7 \end{bmatrix} \begin{matrix} A^* \\ B^* \\ C^* \end{matrix}$$

Осуществим отображение относительно оси OX (прямая $Y=0$): (рисунок 6.5, синий треугольник):

$$\begin{matrix} A: \\ B: \\ C: \end{matrix} \begin{bmatrix} 8 & 1 \\ 6 & 2 \\ 7 & 3 \end{bmatrix} * \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} = \begin{bmatrix} 8 & -1 \\ 6 & -2 \\ 7 & -3 \end{bmatrix} \begin{matrix} A^* \\ B^* \\ C^* \end{matrix}$$



3) Изменение масштаба

Изменение масштаба, как уже отмечалось, определяется значением элементов главной диагонали матрицы 2×2 .

Пример.

$$\begin{matrix} A: \begin{bmatrix} 3 & 1 \\ 3 & 2 \end{bmatrix} \\ B: \begin{bmatrix} 3 & 2 \\ 1 & 2 \end{bmatrix} \\ C: \end{matrix} * \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} = \begin{matrix} \begin{bmatrix} 6 & 2 \\ 6 & 4 \end{bmatrix} A^* \\ \begin{bmatrix} 6 & 4 \\ 2 & 4 \end{bmatrix} B^* \\ \begin{bmatrix} 2 & 4 \end{bmatrix} C^* \end{matrix}$$

Наглядно данный пример представлен на рисунке 6.6.

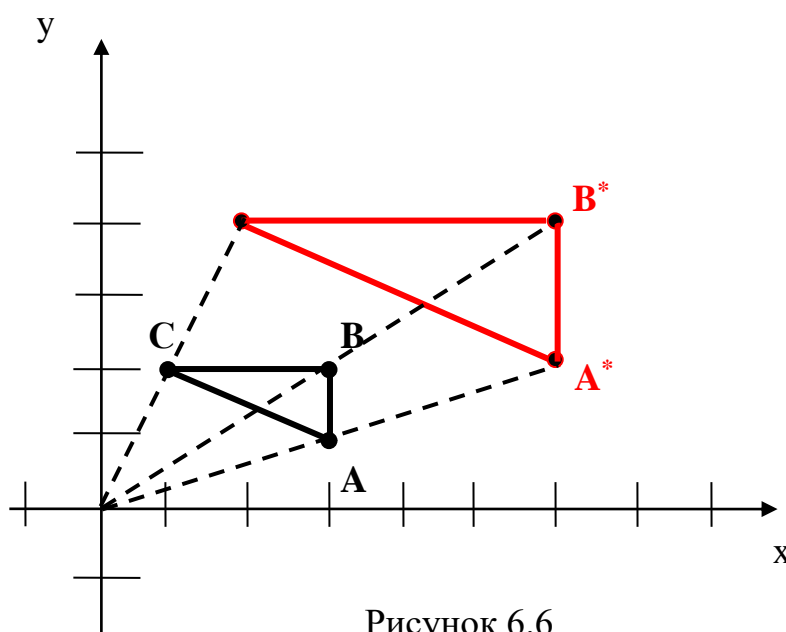


Рисунок 6.6

Если элементы главной диагонали матрицы 2×2 не равны, то происходит искажение фигуры.

6.2.6 Композиция (последовательность) преобразований

Часто для достижения желаемого результата требуется выполнить последовательность преобразований, например, ряд преобразований, рассмотренных ранее. При этом важно помнить, что операция умножения матриц некоммукативна, т.е. играет роль порядок выполнения преобразований.

Рассмотрим пример.

Имеются матрицы: вращение на 90°

$$T_1 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix};$$

отображение относительно прямой $X=0$ (оси OY) - матрица $T_2 = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$.

Выполним последовательность действий над некоторой точкой $A = [x \ y]$:

$$AT_1 = [x \ y] \cdot \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} = [-y \ x]$$

$$(AT_1)T_2 = [-y \ x] \cdot \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} = [y \ x]$$

В другом порядке:

$$AT_2 = [x \ y] \cdot \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} = [-x \ y]$$

$$(AT_2)T_1 = [-x \ y] \cdot \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} = [-y \ -x]$$

Как видим, результаты получились разными.

То же самое можно показать несколько иначе. В начале получим общую матрицу преобразования для комбинации $T = T_1 * T_2$:

$$T = T_1 T_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \cdot \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},$$

а затем для комбинации $T^* = T_2 * T_1$:

$$T^* = T_2 T_1 = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}$$

Видно, что $T \neq T^*$, значит, и результаты преобразования будут различны.

В заключении отметим, что хотя композиция (умножение) матриц в общем случае не коммутативна, но в некоторых частных случаях коммутативность имеет место быть, т.е. $T_1 * T_2 = T_2 * T_1$, если T_1 и T_2 :

- Перенос, перенос;
- Масштабирование, масштабирование
- Поворот, поворот;
- Общее масштабирование, поворот.

6.2.7 Преобразование с помощью однородных координат

Очень часто используемым преобразованием в компьютерной графике является преобразование переноса начала координат. Выше было показано, что такое преобразование нельзя реализовать умножением на матрицу 2×2 (начало координат инвариантно к умножению на матрицу 2×2). Эту трудность можно устранить, введя третью компоненту в вектор координат точки и третью строку в матрицу преобразования. Получим:

$$\begin{bmatrix} x & y & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ m & n \end{bmatrix} = \begin{bmatrix} x+m & y+n \end{bmatrix} = \begin{bmatrix} x^* & y^* \end{bmatrix} \quad (6.17)$$

Т.е. введенная третья строка как раз и реализует требуемый перенос.

Однако матрица преобразования 3×2 не является квадратной и, значит, не имеет обратной матрицы. Это часто оказывается неудобным. Поэтому дополним ее до квадратной матрицы, получим:

$$\begin{bmatrix} x & y & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ m & n & 1 \end{bmatrix} = \begin{bmatrix} x+m & y+n & 1 \end{bmatrix} = \begin{bmatrix} x^* & y^* & 1 \end{bmatrix} \quad (6.18)$$

Итак, теперь имеем трехмерный вектор координат точки на плоскости и матрицу преобразования размера 3×3 .

В случае произвольных значений элементов матрицы 3×3 получим вектор преобразованных координат вида $\begin{bmatrix} \tilde{x} & \tilde{y} & h \end{bmatrix}$.

Представление двумерного вектора трехмерным или, в общем случае, представление n -мерного вектора $(n+1)$ -мерным называется *однородным координатным воспроизведением*; координаты $\begin{bmatrix} \tilde{x} & \tilde{y} & h \end{bmatrix}$ - *однородными координатами*.

Однородные координаты интерпретируются как результат масштабного преобразования точки $\begin{bmatrix} x & y \end{bmatrix}$, лежащей в плоскости $z = 1$, в точку $\begin{bmatrix} x & y \end{bmatrix}$ в плоскости $z = h$.

При однородном координатном преобразовании n -мерного вектора оно выполняется в $(n+1)$ -мерном пространстве, а конечные результаты в n -мерном пространстве (физические или мировые координаты) получают как:

$$x^* = \frac{\tilde{x}}{h}, \quad y^* = \frac{\tilde{y}}{h}. \quad (6.19)$$

Очевидно, что для любой точки на плоскости существует бесконечное множество однородных координатных представлений $[x \ y] \rightarrow [hx \ hy \ h]$. Например, однородные координаты (16, 12, 4), (8, 6, 2), (4, 3, 1) представляют одну и ту же точку (4, 3).

Отметим, что все преобразования матрицей 2×2 (вращение, отображение, покоординатное масштабирование, смещение) реализуются в однородных координатах с помощью матрицы:

$$\begin{bmatrix} a & b & 0 \\ c & d & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$[x \ y \ 1] \cdot \begin{bmatrix} a & b & 0 \\ c & d & 0 \\ 0 & 0 & 1 \end{bmatrix} = [ax + cy \ bx + dy \ 1] = [\tilde{x} \ \tilde{y} \ h] = [x^* \ y^* \ 1] \quad (6.20)$$

$$x^* = \frac{\tilde{x}}{h} = ax + cy, \quad y^* = \frac{\tilde{y}}{h} = bx + dy, \quad (6.12)$$

Т.е. результат тот же, что и для матрицы 2×2 :

$$[x \ y] \cdot \begin{bmatrix} a & b \\ c & d \end{bmatrix} = [ax + cy \ bx + dy] = [x^* \ y^*] \quad (6.22)$$

6.2.8 Преобразования на плоскости с помощью матрицы однородных координат (3x3)

Используя однородные координаты, матрицу преобразования на плоскости в общем виде можно представить так:

$$\begin{bmatrix} a & b & p \\ c & d & q \\ m & n & s \end{bmatrix}$$

Введение однородных координат и применение матрицы преобразования 3×3 общего вида дает довольно много дополнительных преимуществ.

Влияние элементов a, b, c, d остается аналогично рассмотренным ранее. Они в разных комбинациях выполняют вращение, смещение, отображение, масштабирование (см. п.6.2.5).

Влияние элементов m, n третьей строки матрицы 3×3 тоже рассмотрены выше - они реализуют перенос:

$$\begin{bmatrix} x & y & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ m & n & 1 \end{bmatrix} = \begin{bmatrix} x+m & y+n & 1 \end{bmatrix} \quad (6.23)$$

Рассмотрим влияние элементов третьего столбца: p, q, s .

1) матрица $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & s \end{bmatrix}$, тогда

$$\begin{bmatrix} x & y & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & s \end{bmatrix} = \begin{bmatrix} x & y & s \end{bmatrix} \quad (6.24)$$

$$\begin{bmatrix} x^* & y^* \end{bmatrix} = \begin{bmatrix} x/s & y/s \end{bmatrix}$$

Пример.

Дан треугольник координатами своих вершин: А (6, 2), В (6, 4), С (2, 4)

В однородных координатах матрица его координат представляется так:

$$\begin{matrix} A: \\ B: \\ C: \end{matrix} \begin{bmatrix} 6 & 2 & 1 \\ 6 & 4 & 1 \\ 2 & 4 & 1 \end{bmatrix}$$

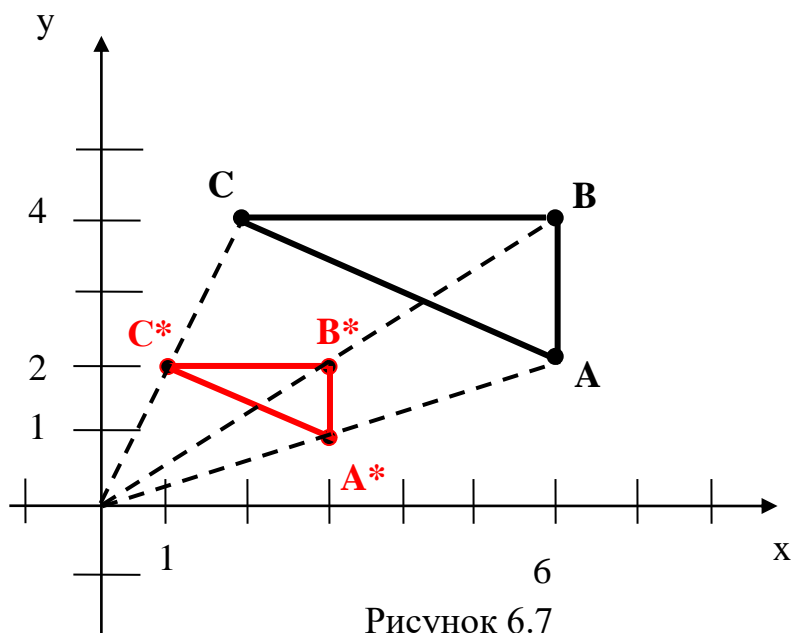
Найдем результат рассматриваемого преобразования.

$$\begin{matrix} A: \\ B: \\ C: \end{matrix} \begin{bmatrix} 6 & 2 & 1 \\ 6 & 4 & 1 \\ 2 & 4 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \end{bmatrix} = \begin{bmatrix} 6 & 2 & 2 \\ 6 & 4 & 2 \\ 2 & 4 & 2 \end{bmatrix}$$

Или переходя к *нормированным однородным координатам*, получаем новые физические координаты треугольника в 2-хмерном пространстве:

$$\begin{bmatrix} 6 & 2 & 2 \\ 6 & 4 & 2 \\ 2 & 4 & 2 \end{bmatrix} \Rightarrow \begin{bmatrix} 3 & 1 & 1 \\ 3 & 2 & 1 \\ 1 & 2 & 1 \end{bmatrix} \begin{matrix} A^* \\ B^* \\ C^* \end{matrix}$$

Наглядно данный пример представлен на рисунке 6.7.



Видно, что элемент s реализует *полное изменение масштаба* (при $s > 1$ – уменьшение; при $s < 1$ – увеличение), т.е. то же самое, что реализуется в случае использования обычных координат и матрицы 2×2 вида

$$\begin{bmatrix} 1/s & 0 \\ 0 & 1/s \end{bmatrix}$$

2) матрица $\begin{bmatrix} 1 & 0 & p \\ 0 & 1 & q \\ 0 & 0 & 1 \end{bmatrix}$

В общем виде

$$\begin{bmatrix} x & y & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & p \\ 0 & 1 & q \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} x & y & px + qy + 1 \end{bmatrix} \quad (6.25)$$

$$\begin{bmatrix} x^* & y^* \end{bmatrix} = \begin{bmatrix} \frac{x}{px + qy + 1} & \frac{y}{px + qy + 1} \end{bmatrix}$$

Получаем центральную проекцию. Центр проецирования лежит в начале координат.

Итак, элементы матрицы 3x3 общего вида выполняют следующие функции:

$$\begin{bmatrix} a & b & p \\ c & d & q \\ m & n & s \end{bmatrix}$$

a, b, c, d – по координатное изменение масштаба, сдвиг; вращение;

m, n – смещение;

s – полное изменение масштаба;

p, q – получение центральных проекций; центр проецирования в начале координат.

6.2.9 Матрица вращения (поворота) вокруг начала координат на плоскости общего вида

Выведем матрицу поворота точки $P(x, y)$ вокруг начала координат O против часовой стрелки на угол φ в точку $P^*(x^*, y^*)$ (рис.6.8).

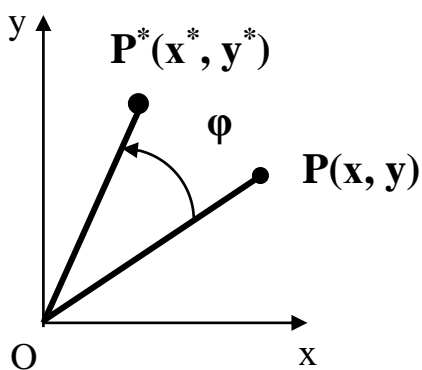


Рисунок 6.8

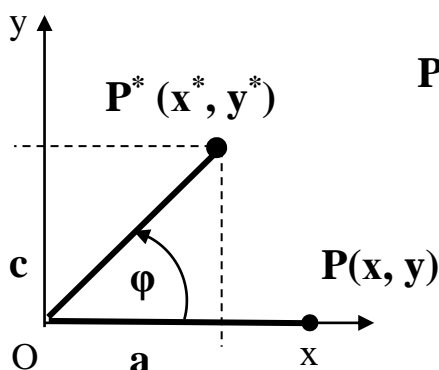


Рисунок 6.9

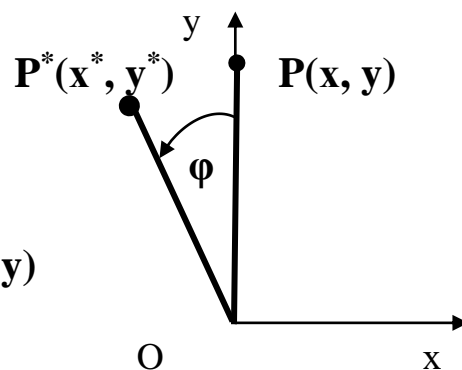


Рисунок 6.10

Новые координаты точки рассчитываются с помощью системы уравнений:

$$\begin{cases} x^* = ax + by \\ y^* = cx + dy \end{cases}$$

Если $P = (1, 0)$, то $x^* = a$, $y^* = c$. Из рисунка 6.9 следует, что

$$a = |OP^*| \cdot \cos \varphi = 1 \cdot \cos \varphi = \cos \varphi, \quad c = |OP^*| \cdot \sin \varphi = 1 \cdot \sin \varphi = \sin \varphi.$$

Аналогично, если $P = (0, 1)$: $b = -\sin \varphi$, $d = \cos \varphi$ (см. рис. 6.10).

Таким образом,

$$\begin{cases} x^* = x \cos \varphi - y \sin \varphi \\ y^* = x \sin \varphi + y \cos \varphi \end{cases} \quad (6.26)$$

Или в матричном виде:
$$\begin{bmatrix} x & y \end{bmatrix} \cdot \begin{bmatrix} \cos \varphi & \sin \varphi \\ -\sin \varphi & \cos \varphi \end{bmatrix} = \begin{bmatrix} x^* & y^* \end{bmatrix}$$

В однородных координатах матрица поворота против часовой стрелки на угол φ вокруг начала координат O :

$$T = \begin{bmatrix} \cos \varphi & \sin \varphi & 0 \\ -\sin \varphi & \cos \varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (6.27)$$

6.2.10 Поворот (вращение) вокруг произвольной точки на плоскости

Вращение фигуры вокруг произвольной точки на плоскости не может быть реализовано одной простой матрицей. Для реализации поворота фигуры на угол φ вокруг произвольной точки с координатами (m, n) необходимо выполнить несколько этапов:

1) перенос точки (m, n) в начало координат;

$$\begin{bmatrix} x & y & 1 \end{bmatrix} \cdot T_1 = \begin{bmatrix} x_1 & y_1 & 1 \end{bmatrix}, \text{ где } T_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -m & -n & 1 \end{bmatrix}$$

2) поворот на угол φ относительно начала координат;

$$\begin{bmatrix} x_1 & y_1 & 1 \end{bmatrix} \cdot R = \begin{bmatrix} x_2 & y_2 & 1 \end{bmatrix}, \text{ где } R = \begin{bmatrix} \cos \varphi & \sin \varphi & 0 \\ -\sin \varphi & \cos \varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3) перенос из начала координат в точку (m, n) .

$$[x_2 \ y_2 \ 1] \cdot T_2 = [x_3 \ y_3 \ 1], \text{ где } T_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ m & n & 1 \end{bmatrix}$$

Найдем **итоговую матрицу** вращения фигуры вокруг произвольной точки с координатами (m, n) на угол φ , путем перемножения матриц преобразований:

$$T_1 \cdot R \cdot T_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -m & -n & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos \varphi & \sin \varphi & 0 \\ -\sin \varphi & \cos \varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ m & n & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} \cos \varphi & \sin \varphi & 0 \\ -\sin \varphi & \cos \varphi & 0 \\ -m(\cos \varphi - 1) + n \sin \varphi & -m \sin \varphi - n(\cos \varphi - 1) & 1 \end{bmatrix}$$

Пример программной реализации преобразований на плоскости

На первом этапе спроектируйте оконный интерфейс. Разместите на форме кнопки для двумерных преобразований (переноса, масштабирования, отражения и поворота). При необходимости разместите элементы для ввода значений: например для ввода значений углов поворота.

Опишите и задайте начальные значения для матрицы, где будут храниться координаты двумерной фигуры. Назовем эту матрицу матрицей тела и опишем как двумерный массив Sq типа *double*. Опишите и задайте как единичную матрицу преобразования 3×3 (двумерный массив T типа *double*). Поскольку массивы Sq и T будут использоваться повсеместно, то целесообразно задать их в качестве полей класса доступных во всех методах и не тратить время на передачу в качестве параметров.

При нажатии на любую кнопку двумерного преобразования должен запускаться процесс, приведённый на рисунке.



Рисунок - Основной процесс визуализации

Этап задания коэффициентов матрицы преобразования сводится к заданию элементов двумерного массива.

Как видим из рисунка, на втором этапе выбрана стратегия сохранения изменений при преобразованиях в матрице тела, а не накапливание изменений в матрице преобразований. Исходя из этого тело метода, реализующего данный этап, может выглядеть следующим образом:

```

b = new double[3]; //массив для хранения промежуточных данных
for (int j=0; j<4; j++) //Цикл по вершинам фигуры (4 вершины для квадрата)
{
    for (int i = 0; i < 3; i++)
    {
        b[i] = 0;
        for (int k = 0; k < 3; k++)
            b[i] = b[i] + Sq[j, k]*T[k, i];
    }
    for (int k=0; k<3; k++)
        Sq[j, k]= b[k];
    }
  
```

Третий этап нормализации приводит к единице последнюю координату путем деления:

```

for (int j = 0; j < 4; j++)
{
    Sq[j, 0] = Sq[j, 0] / Sq[j, 2];
    Sq[j, 1] = Sq[j, 1] / Sq[j, 2];
    Sq[j, 2] = 1;
}
  
```

Перерисовка - это соединение вершин фигуры линиями. При этом необходимо произвести сдвиг изображения к центру экрана:

```
Graphics gfx = e.Graphics;
Pen skyBluePen = new Pen(Brushes.DeepSkyBlue);
int cx = ClientSize.Width/2;
int cy = ClientSize.Height/2;
gfx.DrawLine(skyBluePen, cx, 0, cx, ClientSize.Height);
gfx.DrawLine(skyBluePen, 0, cy, ClientSize.Width, cy);
Pen SqPen = new Pen(Brushes.BlueViolet);
gfx.DrawLine(SqPen, (int) (cx + Sq[0, 0]), (int) (cy - Sq[0, 1]),
               (int) (cx + Sq[1, 0]), (int) (cy - Sq[1, 1]));
gfx.DrawLine(SqPen, (int) (cx + Sq[1, 0]), (int) (cy - Sq[1, 1]),
               (int) (cx + Sq[2, 0]), (int) (cy - Sq[2, 1]));
gfx.DrawLine(SqPen, (int) (cx + Sq[2, 0]), (int) (cy - Sq[2, 1]),
               (int) (cx + Sq[3, 0]), (int) (cy - Sq[3, 1]));
gfx.DrawLine(SqPen, (int) (cx + Sq[3, 0]), (int) (cy - Sq[3, 1]),
               (int) (cx + Sq[0, 0]), (int) (cy - Sq[0, 1]));
```

Переменные *cx* и *cy* это координаты центра окна, используются для прорисовки осей и самой фигуры. Приведение к целому типу, а следовательно, округление, производится непосредственно перед прорисовкой и не сохраняется в матрице *Sq*. Поэтому не будет происходить накопление ошибок округления.