Содержание

Посвящение

Эта книга посвящается памяти профессора Московского университета Евгения Викторовича Шикина. Под его руководством все мы постигали науку на факультете ВМК МГУ, занимались компьютерной графикой, дифференциальной геометрией, решали теоретические и прикладные задачи. Он одним из первых опубликовал в России книгу по компьютерной графике. В своих книгах он всегда уделял особое внимание математическим основам, которые другие авторы часто не считали необходимым приводить.

Евгений Викторович был выдающимся лектором, его лекции всегда были очень интересны и отличались образностью. Его стиль ведения лекций стал для всех нас тем эталоном, к которому мы все стремимся уже в своих лекциях.

Большой заслугой Евгения Викторовича было также и то, что именно благодаря ему дисциплина «Компьютерная графика» стала обязательной учебной дисциплиной сначала на факультете ВМК МГУ, а затем и в ряде других вузов. Евгений Викторович ввел новую форму экзамена для этого предмета – вместо теоретического экзамена предлагалось написать компьютерную программу, реализующую трехмерную динамическую сцену. Оценивалась не только сложность эффектов, которые реализовал автор, но и оригинальность замысла, а также качество программного кода.

Проверка экзаменационных работ превратилась для всех нас в увлекательное занятие, поскольку все работы были разные, а многие студенты создавали настоящие завораживающие виртуальные миры.

Сазонов В. В.
Березин С. Б.
Боресков А. В.

Благодарности

Хочется от всей души поблагодарить всех тех, кто своими замечаниями, советами и отзывами помогли сделать эту книгу лучше и полезнее. Особенную благодарность выражаю коллегам по Cadence Design Systems (как настоящим, так и бывшим) за многочисленные советы и замечания.

Введение

Для того чтобы уметь писать программы, строящие различные изображения (что и является целью компьютерной графики), нужно знать целый ряд понятий и алгоритмов.

Поскольку мы работаем с геометрическими объектами, то нам нужен способ их представления в компьютере, а также способ их преобразования.

Все геометрические объекты обычно представляются при помощи *координат*, т. е. упорядоченных наборов чисел (векторов) с введенными операциями над ними. В главе 1 мы рассмотрим координаты и преобразования в двухмерном случае (на плоскости), а в главе 3 – в трехмерном случае (в пространстве).

Также в главе 3 рассматриваются различные способы задания ориентации объектов в пространстве и задания поворотов. В качестве таких способов выступают *углы Эйлера* и *кватернионы*. Также в этой главе рассматриваются стандартные операции над кватернионами.

Существует большое количество часто используемых объектов и операций над ними как на плоскости, так и в пространстве. Соответственно, главы 2 и 4 рассматривают такие объекты и часто встречающиеся операции над ними на плоскости и в пространстве.

Когда мы имеем дело с большим количеством геометрических данных, то для ускорения работы с ними обычно используются специальные структуры данных, подобно тому как в системах управления базами данных используются различные типы индексов для ускорения операций над данными.

Использование таких структур данных (обычно называемых *пространственными индексами*) позволяет заметно сократить затраты на работу с большими массивами геометрических данных. Подобные структуры, также их построение и использование рассматриваются в главе 5.

Нашей конечной целью является построение изображений, т. е. наборов точек различных цветов. Соответственно, мы должны определиться с тем, что такое цвет и как его можно представить в компьютере. Вся глава 6 посвящена различным моделям для представления цвета, рассматриваются их преимущества для тех или иных задач.

Кроме того, в главе 6 также рассматривается такое понятие, как *гамма-коррекция*, моделирующее преобразование цвета в различных устройствах для отображения и получения изображений (мониторах, камерах и т. п.).

Практически все части современной компьютерной графики относятся к так называемой *растровой графике*, т. е. изображения представляются при помощи прямоугольных массивов точек – пикселов – различных цветов.

Соответственно, возникает задача перевода идеальных геометрических объектов, таких как отрезки, дуги, треугольники, в их растровое представление (процесс, называемый *растеризацией*). Вся глава 7 посвящена этой задаче.

Другой важной задачей, возникающей при построении трехмерных объектов (*рендеринге*), является задача *удаления невидимых поверхностей*. Она связана с тем, что при рендеринге трехмерных объектов мы проектируем их на двухмерную плоскость. При этом одни объекты могут закрывать (делать полностью или частично невидимыми) другие объекты. Задачей является определить, какие именно объекты из числа имеющихся будут видны и что именно будет видно в каждой части строящегося изображения.

Для решения этой задачи существует большое количество различных методов. В главе 8 дается общий обзор методов и указываются их основные плюсы и минусы.

Обычно мы хотим построить изображения объектов, освещаемых различными источниками света (а не просто висящих в темноте). Для этого нам нужно разобраться с тем, как именно различные объекты взаимодействуют с падающим на них светом.

В главе 9 рассматривается, как можно моделировать это взаимодействие. Для начала рассматриваются самые простые эмпирические модели (например, Ламберта и Фонга). Далее осуществляется переход к строгим физически корректным моделям освещения (например, Кука–Торранса). Также в этой главе рассматриваются играющие большую роль в современной графике *сферические гармоники* и изучается их применение для расчета освещения.

Глава 10 посвящена методу трассировки лучей – простому и красивому методу, позволяющему строить фотореалистичные изображения с точным расчетом таких эффектов, как преломление и отражение света.

Метод трассировки лучей, хотя и дает очень высокое качество получаемых изображений, является довольно медленным. Поэтому во многих случаях (например, симуляторы, обучающие программы, компьютерные игры) требуется *графика реального времени*.

Под этим термином подразумевается возможность построения изображений с достаточно высокой частотой, обычно это не менее 24 кадров в секунду. Для получения графики с такой частотой и достаточно высоким качеством обычно используются программируемые графические процессоры (GPU, Graphics Processing Unit).

Существует много различных GPU, поэтому работа с ними обычно идет не напрямую, а через специальные API (Application Program Interface). Двумя из наиболее распространенных из этих API являются кроссплатформенный OpenGL и ориентированный на платформу Microsoft Window Direct3D. В главе 12 будет рассмотрен OpenGL версии 3.3. В этой главе будет рассмотрен как вывод отдельных примитивов, так и работа с буферами и написание простейших шейдеров (программ, выполняемых на GPU).

Однако сама библиотека OpenGL не занимается созданием окон для рендеринга, обработкой ввода/вывода и т. п. Для этого обычно используют вспомогательные библиотеки. В главе 11 мы рассмотрим сразу две подобные библиотеки – freeglut и GLFW. Также мы рассмотрим библиотеку GLEW, позволяющую получить доступ к различным расширениям OpenGL. Она позволяет получить доступ к многочисленным функциям, которые обычно содержатся в самом драйвере, и к более поздним версиям OpenGL (так, для платформы Microsoft Window библиотека OpenGL32.dll обычно поддерживает OpenGL версии 1.5 или даже ниже).

Одной из неотъемлемых частей современной компьютерной графики являются различные спецэффекты. В главе 13 рассматриваются реализации на OpenGL большого набора различных спецэффектов, начиная с самых простых и заканчивая такими, как рендеринг меха, физически корректный рендеринг, вывод текста при помощи поля расстояний и т. п.

Почти для всех глав имеются многочисленные примеры исходного кода, которые доступны как в репозитории на github – **https://github.com/steps3d/graphics-book**, так и на сайте автора **http://steps3d.narod.ru**. Этот исходный код использует определенный набор библиотек и инструментов, которые не содержатся в репозитории. Для сборки всех примеров применяется утилита cmake. При ее помощи можно легко создать проекты для компиляции примеров под определенную среду разработки и операционную систему.

Полный список используемых библиотек и инструментов содержится в приложении.

Предисловие от издательства

Отзывы и пожелания

Мы всегда рады отзывам наших читателей. Расскажите нам, что вы думаете об этой книге – что понравилось или, может быть, не понравилось. Отзывы важны для нас, чтобы выпускать книги, которые будут для вас максимально полезны.

Вы можете написать отзыв прямо на нашем сайте **www.dmkpress.com**, зайдя на страницу книги, и оставить комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу **dmkpress@gmail.com**, при этом напишите название книги в теме письма.

Если есть тема, в которой вы квалифицированы, и вы заинтересованы в написании новой книги, заполните форму на нашем сайте по адресу **http://dmkpress.com/authors/publish_book/** или напишите в издательство по адресу **dmkpress@gmail.com**.

Скачивание исходного кода примеров

Скачать файлы с дополнительной информацией для книг издательства «ДМК Пресс» можно на сайте **www.dmkpress.com** или **www.дмк.рф** на странице с описанием соответствующей книги.

Список опечаток

Хотя мы приняли все возможные меры для того, чтобы удостовериться в качестве наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг — возможно, ошибку в тексте или в коде, — мы будем очень благодарны, если вы сообщите нам о ней. Сделав это, вы избавите других читателей от расстройств и поможете нам улучшить последующие версии этой книги.

Если вы найдете какие-либо ошибки в коде, пожалуйста, сообщите о них главному редактору по адресу **dmkpress@gmail.com**, и мы исправим это в следующих тиражах.

Нарушение авторских прав

Пиратство в интернете по-прежнему остается насущной проблемой. Издательства «ДМК Пресс» и Packt очень серьезно относятся к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в интернете с незаконно выполненной копией любой нашей книги, пожалуйста, сообщите нам адрес копии или веб-сайта, чтобы мы могли применить санкции.

Пожалуйста, свяжитесь с нами по адресу электронной почты **dmkpress@gmail.com** со ссылкой на подозрительные материалы.

Мы высоко ценим любую помощь по защите наших авторов, помогающую нам предоставлять вам качественные материалы.

Глава 1

Координаты и преобразования на плоскости

Координаты являются ключевым понятием, с которым мы будем сталкиваться на протяжении всей книги. Рассмотрение координат и преобразований мы начнем с двухмерного случая (2D). С одной стороны, одномерный случай слишком прост, а с другой – мы будем использовать изложенное для двухмерного случая как основу в более сложном, трехмерном (3D) случае.

Фактически координаты – это механизм, позволяющий вместо работы с геометрическими объектами на самом деле работать с алгебраическими объектами методами алгебры.

Основным нашим понятием будет понятие *координатной плоскости* – это такая плоскость, на которой каждой ее точке P сопоставлена некоторая уникальная пара чисел (x, y), называемая *координатами* этой точки.

Самым простым способом введения координат на плоскости является задание двух координатных осей Ox и Oy. *Координатная ось* – это прямая, на которой обозначены начало отсчета – точка O – и дополнительная точка E. Точка E служит для обозначения направления оси и задания на ней единицы длины. Начало координат O соответствует нулю, а точка E соответствует единице (рис. 1.1) (длина отрезка OE равна единице).

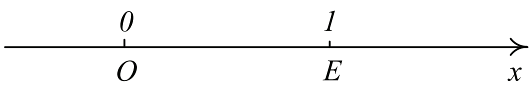

Рис. 1.1 ❖ Координатная ось *Ox*

В двухмерной системе координат обычно используются сразу две координатные оси с общим началом отсчета – точкой O. При этом чаще всего рассматриваются так называемые *декартовы* (Cartesian[1]) системы координат, в которых координатные оси перпендикулярны друг другу (см. рис. 1.2), но могут быть и недекартовы системы координат.

[1] Для многих важных понятий в скобках мы будем давать их английский эквивалент. Это связано с тем, что очень много литературы на компьютерной графике доступно именно на английском языке.

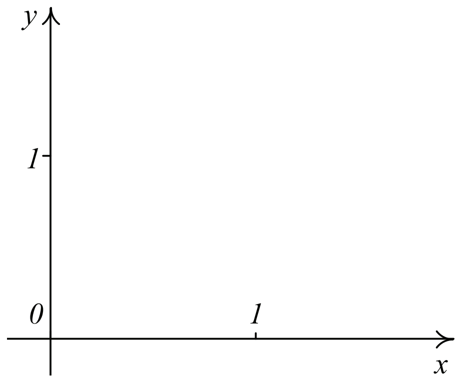

Рис. 1.2 ❖ Декартова система координат на плоскости

В качестве примера *недекартовой* давайте рассмотрим *полярную систему координат*. В этой системе координат используется всего одна ось (точнее, луч, выходящий из начала координат и идущий в положительном направлении). В качестве координат произвольной точки A выступают расстояние r от нее до начала координат O и угол ϕ между положительным направлением оси и лучом OA (рис. 1.3).

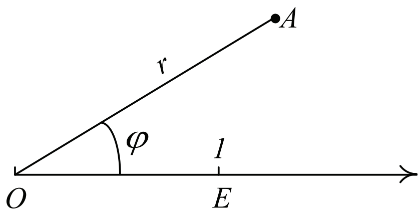

Рис. 1.3 ❖ Полярная система координат

Обратите внимание, что при $r = 0$ угол не определен.

Пусть у нас есть декартова система координат и некоторая точка A. Тогда мы можем получить координаты этой точки, просто спроектировав ее на каждую из координатных осей (вдоль направления другой оси) и взяв соответствующие значения проекций как координаты точки (рис. 1.4).

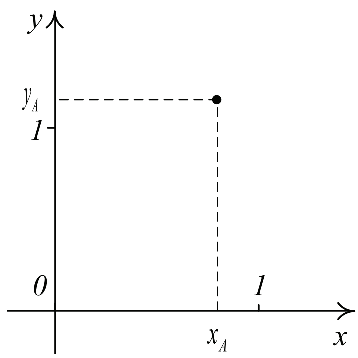

Рис. 1.4 ❖ Декартовы координаты точки A

На одной и той же плоскости мы можем ввести сразу несколько различных систем координат. В результате произвольной точке этой плоскости можно сопоставить сразу несколько различных пар координат (см. рис. 1.5).

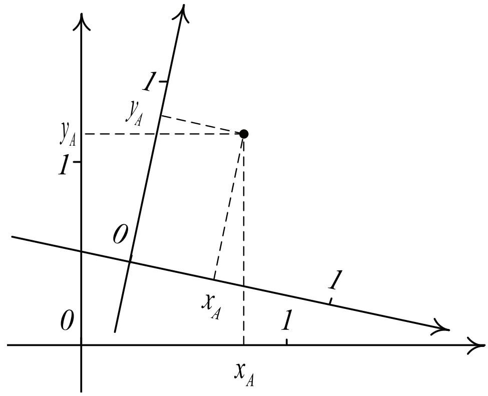

Рис. 1.5 ❖ Различные системы координат на плоскости

При помощи системы координат на плоскости устанавливается *взаимно однозначное* соответствие между множеством всех точек плоскости и множеством пар всевозможных вещественных чисел. Такое множество (пар) мы будем далее обозначать $\mathbb{R}^2$.

Пару чисел x_A и y_A, являющуюся координатами точки A, можно записать либо как строку $(x_A \quad y_A)$, либо как столбец $\begin{pmatrix} x_A \\ y_A \end{pmatrix}$. Всюду далее мы будем записывать координаты в виде столбцов. На самом деле, какая именно запись выбрана, не играет особой роли, обе они полностью эквивалентны, но нам нужно выбрать один какой-то определенный способ. Можно ввести операцию *транспонирования*, которая переводит строку в столбец и наоборот:

$$\begin{pmatrix} x & y \end{pmatrix}^T = \begin{pmatrix} x \\ y \end{pmatrix},$$

$$\begin{pmatrix} x \\ y \end{pmatrix}^T = \begin{pmatrix} x & y \end{pmatrix}.$$

Соответственно, вместо рассмотрения плоскости мы можем на самом деле рассматривать множество $\mathbb{R}^2$ и различные операции на нем. Вместо точек на плоскости мы будем далее рассматривать просто пары чисел, являющиеся координатами точек.

Если у нас есть точка A, то мы можем ввести вектор OA. Координатами этого вектора мы будем называть координаты точки A. Из школьного курса геометрии мы знаем, что векторы можно складывать между собой и умножать на числа. Давайте рассмотрим, что происходит с координатами векторов при их сложении и умножении на различные числа.

Далее тот факт, что некоторый вектор u имеет координаты $\begin{pmatrix} x_u \\ y_u \end{pmatrix}$, мы будем далее записывать следующим образом:

$$u = \begin{pmatrix} x_u \\ y_u \end{pmatrix}.$$

Пусть у нас есть два вектора $u = \begin{pmatrix} x_u \\ y_u \end{pmatrix}$, $v = \begin{pmatrix} x_v \\ y_v \end{pmatrix}$ и некоторое вещественное число α. Тогда легко можно убедиться в том, что сумме векторов u и v соответствует пара чисел, являющаяся покомпонентной суммой координат исходных векторов:

$$u + v = \begin{pmatrix} x_u + x_v \\ y_u + y_v \end{pmatrix}.$$

Аналогично можно убедиться в том, что при умножении вектора u на число α происходит покомпонентное умножение координат вектора u на число α:

$$\alpha \cdot u = \begin{pmatrix} \alpha x_u \\ \alpha y_u \end{pmatrix}.$$

Через O обозначим так называемый *нулевой вектор* $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$. Тогда мы можем для операций сложения векторов и умножения вектора на число записать следующие свойства:

$$u + v = v + u;$$

$$(u + v) + w = u + (v + w);$$

$$u + O = u;$$

$$1 \cdot u = u;$$

$$\alpha \cdot (u + v) = \alpha \cdot u + \alpha \cdot v;$$

$$(\alpha + \beta) \cdot u = \alpha \cdot u + \beta \cdot u.$$

Также для произвольного вектора u можно ввести *обратный вектор* $-u$, такой что $u + (-u) = O$. Легко убедиться, что обратный вектор $-u$ всегда существует на самом деле и имеет следующие координаты:

$$-u = \begin{pmatrix} -x_u \\ -y_u \end{pmatrix}.$$

Тем самым для обратного вектора справедливо равенство

$$-u = (-1) \cdot u.$$

Помимо введенных операций сложения и умножения на числа, часто используется еще одна операция – *скалярное произведение* векторов (*dot product*). Скалярное произведение векторов – это операция, сопоставляющая произвольной паре векторов u и v некоторое вещественное число, обозначаемое (u, v) или $u \cdot v$. Скалярное произведение удовлетворяет следующим свойствам:

$(u,v)=(v,u)$;

$(u+v,w)=(u,w)+(v,w)$;

$(\alpha u,v)=\alpha(u,v)$;

$(u,u)\geq 0$.

При этом скалярный квадрат вектора (u, u) равен нулю тогда и только тогда, когда сам вектор u равен нулевому вектору. Скалярное произведение легко может быть записано через координаты следующим образом:

$$(u, v) = u_x v_x + u_y v_y.$$

Поскольку для любого вектора u всегда справедливо $(u, u) \geqslant 0$, то можно ввести величину, называемую *нормой*, или *длиной*, вектора и обозначаемую как $||u||$ или $|u|$, следующим образом:

$$u=\sqrt{(u,u)}.$$

Введенная таким образом норма вектора удовлетворяет следующим основным свойствам:

$||\alpha \cdot u|| = |\alpha| \cdot ||u||$;

$||u + v|| \leqslant ||u|| + ||v||$;

$||u|| = 0$ тогда и только тогда, когда $u = 0$;

$|(u, v)| \leqslant ||u|| \cdot ||v||$.

Два вектора, u и v, называются *ортогональными* (или *перпендикулярными*), если их скалярное произведение равно нулю, т. е. $(u, v) = 0$.

Используя скалярное произведение, можно не только ввести понятие перпендикулярности векторов, но и угол между ними. Угол φ между ненулевыми векторами u и v определяется из следующего равенства:

$$\cos\varphi=\frac{(u,v)}{||u||\cdot||v||}.$$

Кроме векторов, важную роль играют и *матрицы*. Матрица 2×2 – это упорядоченная запись четырех чисел a_{11}, a_{12}, a_{21} и a_{22} в виде таблицы из двух строк и двух столбцов:

$$A=\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}.$$

Матрицу, состоящую из одних нулей, называют *нулевой* и обозначают O, а через I обозначают *единичную* матрицу следующего вида:

$$I=\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

Элементы матрицы a_{11} и a_{22} называются *главной диагональю* матрицы A. Матрица, у которой все элементы, не лежащие на главной диагонали, равны нулю, называется *диагональной*.

Для матриц 2×2 можно, как и для векторов, ввести операции сложения и умножения на число, определив их покомпонентно:

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} + \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11}+b_{11} & a_{12}+b_{12} \\ a_{21}+b_{21} & a_{22}+b_{22} \end{pmatrix};$$

$$\alpha \cdot \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} = \begin{pmatrix} \alpha a_{11} & \alpha a_{12} \\ \alpha a_{21} & \alpha a_{22} \end{pmatrix}.$$

Эти операции удовлетворяют аналогичным свойствам соответствующих операций для векторов. Множество всех матриц 2×2 обозначают через $\mathbb{R}^{2\times 2}$.

Кроме уже введенных операций над матрицами, можно также ввести еще две – транспонирование матриц и перемножение матриц. Обратите внимание, что они не являются поэлементными, в отличие от сложения матриц и умножения матрицы на число. Транспонирование матриц определяется следующим образом:

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}^T = \begin{pmatrix} a_{11} & a_{21} \\ a_{12} & a_{22} \end{pmatrix}.$$

Матрица C называется *произведением* матриц A и B (и обозначается $C = AB$), если элемент матрицы C с индексами i и j (т. е. элемент c_{ij}) задается следующей формулой:

$$c_{ij} = \sum_{k=1}^{2} a_{ik} b_{kj}.$$

Для операций транспонирования и умножения справедливы следующие свойства:

$(AB)C = A(BC)$;
$A(B + C) = AB + AC$;
$(\alpha A)B = \alpha(AB)$;
$AI = IA = A$;
$AO = OA = O$;
$(AB)^T = B^T A^T$;
$(A + B)^T = A^T + B^T$;
$(\alpha A)^T = \alpha A^T$;
$(A^T)^T = A$.

Обратите внимание, что в общем случае $AB \neq BA$ подобные операции называют *некоммутативными*. В этом легко можно убедиться на следующем примере:

$$\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \neq \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.$$

Для матрицы A размером 2×2 также можно ввести понятие *детерминанта* матрицы, обозначаемого $\det A$, используя следующую формулу:

$$\det A = a_{11}a_{22} - a_{12}a_{21}.$$

Матрицы, детерминант которых равен нулю, называются *вырожденными*, все остальные матрицы называются *невырожденными*. Можно показать, что у вырожденной матрицы строки (или столбцы) пропорциональны друг другу.

Если матрица A невырожденная ($\det A \neq 0$), то всегда существует такая матрица B, называемая *обратной* к A, что выполняется равенство

$$AB = BA = I.$$

Матрица, обратная к матрице A, обозначается как A^{-1}. Для обратных матриц справедливы следующие свойства:

$$(A^{-1})^{-1} = A;$$
$$\det A^{-1} = \frac{1}{\det A};$$
$$(AB)^{-1} = B^{-1}A^{-1};$$
$$(A^T)^{-1} = (A^{-1})^T.$$

Для невырожденной матрицы A можно явно выписать ее обратную матрицу:

$$A^{-1} = \frac{1}{\det A}\begin{pmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{pmatrix}.$$

Кроме операции перемножения матриц 2×2, можно также ввести операцию умножения матрицы 2×2 на вектор следующим образом:

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} a_{11}x + a_{12}y \\ a_{21}x + a_{22}y \end{pmatrix}.$$

Для этой операции справедливы следующие свойства:

$$(A + B)u = Au + Bu;$$
$$A(Bu) = (AB)u;$$
$$A(u + v) = Au + Av;$$
$$A(\alpha u) = \alpha Au;$$
$$(Au, v) = (u, A^T v).$$

Преобразования на плоскости

Под *преобразованием* на плоскости понимается любое отображение плоскости на себя $f : \mathbb{R}^2 \to \mathbb{R}^2$. Простейшим примером преобразования является *тождественное* преобразование $f(u) \equiv u$.

Композицией h преобразований f и g называется результат последовательного применения этих преобразований:

$$h(u) = (fg)(u) = f(g(u)).$$

Введем еще одно понятие – преобразование g называется *обратным* к преобразованию f (и обозначается как $g = f^{-1}$), если для всех векторов u выполнено равенство

$$f(g(u)) = g(f(u)) = u.$$

Среди всех преобразований на плоскости выделяется очень важный класс *линейных* преобразований. Преобразование f называется линейным, если для любых векторов u и v и любого вещественного числа α выполняются следующие равенства:

$f(u + v) = u + v;$
$f(\alpha u) = \alpha f(u).$

Примером линейного преобразования может служить любое преобразование вида $f(u) = Au$, где A – произвольная матрица. Можно показать, что для любого линейного преобразования f всегда существует такая матрица A, что $f(u) = Au$ для всех векторов u. В то же время преобразование $f(u) = u + a$ не является линейным при ненулевом векторе a.

Тем самым вместо рассмотрения линейных преобразований на плоскости мы можем рассматривать просто различные матрицы, поскольку каждому линейному преобразованию соответствует преобразование, задаваемое какой-либо матрицей.

Для линейных преобразований обратному преобразованию соответствует обратная матрица, и композиции преобразований соответствует произведение матриц соответствующих преобразований.

Далее мы рассмотрим некоторые стандартные преобразования на плоскости и выпишем соответствующие им матрицы.

Масштабирование

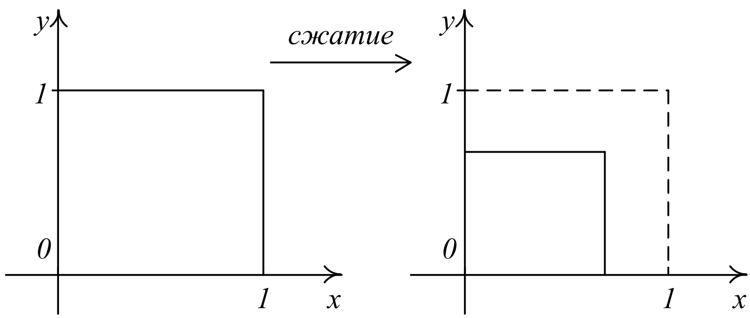

Рис. 1.6 ❖ Преобразование масштабирования

Простейшим линейным преобразованием на плоскости является *однородное масштабирование*. Оно задается при помощи следующей матрицы:

$$S_\lambda = \begin{pmatrix} \lambda & 0 \\ 0 & \lambda \end{pmatrix}, \lambda > 0.$$

В случае $\lambda > 1$ мы получаем однородное растяжение, а при $0 < \lambda < 1$ – однородное сжатие. Поскольку $\det S_\lambda = \lambda^2 > 0$, то это преобразование всегда невырожденное, обратным к нему также является преобразование однородного масштабирования:

$$S_\lambda^{-1} = \begin{pmatrix} \frac{1}{\lambda} & 0 \\ 0 & \frac{1}{\lambda} \end{pmatrix}.$$

Помимо однородного масштабирования, есть еще и *неоднородное масштабирование*, когда степень растяжения/сжатия отличается для разных направлений (рис. 1.7). Неоднородное масштабирование задается матрицей

$$S_{\lambda\mu} = \begin{pmatrix} \lambda & 0 \\ 0 & \mu \end{pmatrix}, \ \lambda > 0, \ \mu > 0.$$

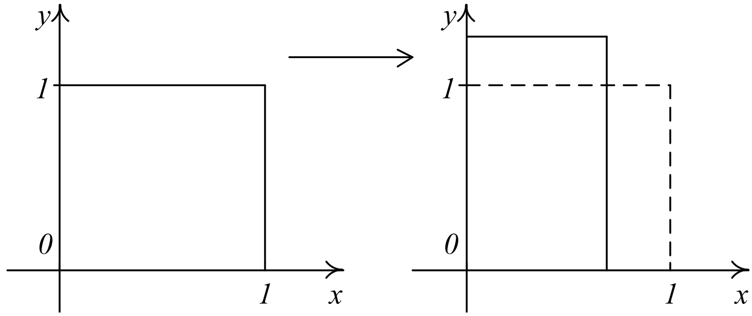

Рис. 1.7 ❖ Неоднородное масштабирование

Неоднородное масштабирование также является невырожденным и поэтому тоже всегда обратимо. Обратным к нему также является преобразование масштабирования со следующей матрицей:

$$S_{\lambda\mu}^{-1} = \begin{pmatrix} \frac{1}{\lambda} & 0 \\ 0 & \frac{1}{\mu} \end{pmatrix}.$$

Отражение

Еще одним простым линейным преобразованием на плоскости является отражение относительно координатной оси (рис. 1.8 и 1.9).

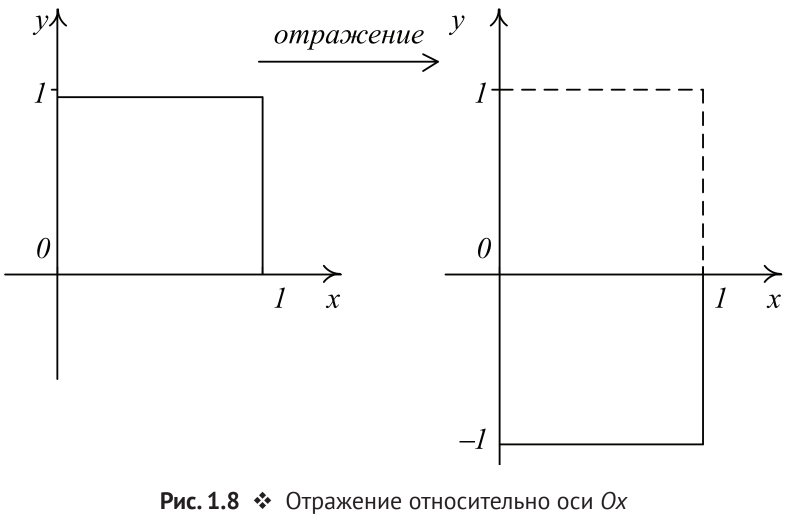

Рис. 1.8 ❖ Отражение относительно оси *Ox*