# Глава 1
# Графика и визуализация

## § 1
## Технологии обработки графической информации

- *Растровая графика*
- *Цветовые модели*
- *Векторная графика*

Всем известно, что бóльшую часть сведений и представлений об окружающем мире человек получает с помощью зрения. Поэтому очень важно существование средств представления информации «для зрения» и автоматических средств обработки такой информации.

Общее направление, в рамках которого решаются такие задачи, получило название компьютерной графики. Другими словами, ***компьютерной графикой*** *называют область деятельности, в которой компьютеры и программное обеспечение используются в качестве инструмента создания и обработки изображений.*

Поскольку компьютер — устройство, предназначенное для обработки дискретной цифровой информации, то требуется способ формирования изображения на основе именно такой информации.

Способ, которым чаще всего представляется графическая информация, получил название *растрового.* ***Растровая графика*** *— способ представления и хранения изображения в виде обозначения цветов точек (пикселей), находящихся в узлах прямоугольной равномерной координатной сетки —* ***растра.***

Если сетка достаточно плотная, то точки получаются мелкие, поэтому человеческий глаз не воспринимает изображение как дискретное.

Основными параметрами изображения в растровой форме являются **разрешение линейное**, т. е. возможное количество точек на единицу площади, и **разрешение цветовое** — количество градаций цвета. Таким образом, различают разрешение линейное — количество столбцов по горизонтали и линий по вертикали, и

**Технико-историческая справка**

В декабре 1951 года инженер Массачусетского технологического института Джей У. Форрестер продемонстрировал новый компьютер «Вихрь», принципиальным отличием которого было устройство вывода, формировавшее изображение на экране электронно-лучевой трубки. Изображение формировалось из отдельных светящихся точек. Позднее для оперативного управления компьютером во время использования комплексов противовоздушной обороны было разработано первое интерактивное устройство ввода — световой пистолет.

Следующим шагом в развитии этого направления стала разработка в 1961–1962 годах Айвеном Сазерлендом первой интерактивной программы для выполнения чертежей (рис. 1.1) — Sketchpad (Блокнот).

**Рис. 1.1.** Первая интерактивная программа для выполнения чертежей

Программа впервые реализовала принцип интерактивного рисования отдельных графических примитивов (отрезков и дуг) из отдельных точек и последующих операций с ними. Интерактивность достигалась применением светового пера для указания необходимых координат. Примерно в то же время была разработана первая система автоматизированного проектирования (DAC-1), но она требовала ввода координат примитивов с клавиатуры.

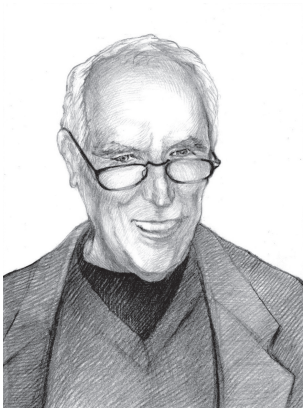
Наиболее существенной трудностью при работе с графикой была высокая загрузка центрального процессора и памяти, поскольку изображение полностью формировалось с помощью центрального процессора. Для преодоления этого затруднения были разработаны системы с памятью регенерации (позднее — видеопамятью), снимавшие с центрального процессора эту нагрузку. В таких системах каждая точка изображения описывалась некоторым числом. Практически все мониторы были монохромными, т. е. позволяли работать с одним цветом (не обязательно белым), большинство из них не допускало градаций цвета.

Поскольку наличие большой памяти (для каждой точки — не менее бита) делало такие системы крайне дорогими, то вместо них долгое время применялись системы с запоминающей электронно-лучевой трубкой, удерживавшей изображение около часа. Применение таких систем не давало возможности работать с изображением интерактивно, но сильно удешевляло производство.

Появление в начале 1980-х годов высокоскоростных и дешевых запоминающих устройств на основе микросхем позволило активно развивать направление создания устройств с памятью.

цветовое/оттеночное — количество оттенков или цветов у каждой точки. Линейное разрешение описывают возможным количеством точек, а цветовое — в виде количества битов, отводимых на описание каждой отдельной точки.

Чем больше количество точек на единицу площади и количество цветов каждой точки, тем выше возможное качество изображения.



**Сазерленд Айвен** (род. в 1938 г.) — американский ученый в области компьютерных наук, удостоен наиболее престижных академических наград и является членом высших научных обществ США. Ему принадлежит создание первого интерактивного графического программного пакета «Sketchpad», прообраза будущих систем автоматизированного проектирования (САПР). На рубеже 60-х и 70-х годов XX века разработки Айвена Сазерленда в области векторной графики с использованием графических примитивов, светового пера и возможностей компьютерного моделирования использовались в основном для военных целей. Благодаря этим разработкам современные пользователи имеют мощные графические пакеты для самых разных профессиональных целей. В то же время Сазерленд впервые применил объектно-ориентированный подход к программированию, создал первый шлем виртуальной реальности и алгоритм Коэна-Сазерленда, позволяющий эффективно находить отрезки прямых, находящихся внутри прямоугольника.

Один из самых важных вопросов при организации обработки графических данных — это представление и кодирование цвета.

В простейшем случае, когда всего два цвета, используется один бит, состояние которого и задает цвет. Если же цветов становится больше, то такой подход уже не позволяет решить поставленную задачу.

Существует несколько способов кодирования цвета, применяемых при обработке графики.

Для описания градации одного цвета применяется обычное кодирование, в котором номер обозначает **градацию**. Чем больше значение, тем сильнее проявляется цвет. Для мониторов (в которых точка самостоятельно излучает свет) обычно 0 соответствует отсутствию цвета, а максимальное значение (например, 255) — максимальной светимости точки. Таким образом, появляется возможность задавать **оттенок** на монохромном мониторе.

В случае, когда используется печатающее устройство и чернильная точка либо есть, либо нет, оттенок задается количеством цветовых точек некоторой матрицы (например, $4 \times 4$ точки).

В более сложных случаях, когда речь идет о кодировании сложного цвета с большим количеством оттенков, рассматривают разложение цвета на несколько отдельных компонентов, которые, смешиваясь в одной точке, образуют заданный цвет.

Для каждого конкретного изображения всё, что передается одним из компонентов цвета, также называется *каналом*.

Компоненты цвета и способ образования из них видимого оттенка и представляют собой **цветовую модель.**

Цветовые модели разрабатывались в психологии восприятия задолго до появления вычислительной техники. Существует большое количество цветовых моделей, которые создавались и вводились разными авторами для описания и исследования зрения человека. С появлением проекционной и печатающей аппаратуры, с учетом технических требований были разработаны новые модели, учитывающие в первую очередь физические и технические аспекты формирования конкретного цвета.

Наиболее популярны сейчас следующие цветовые модели.

**Модель восприятия HLS** (рис. 1.2 и рис. I на цветной вклейке) подразумевает образование цвета из трех основных компонентов:

- **H**ue — оттенка цвета;
- **L**ightness (**I**ntensity) — яркости (интенсивности);
- **S**aturation — насыщенности.

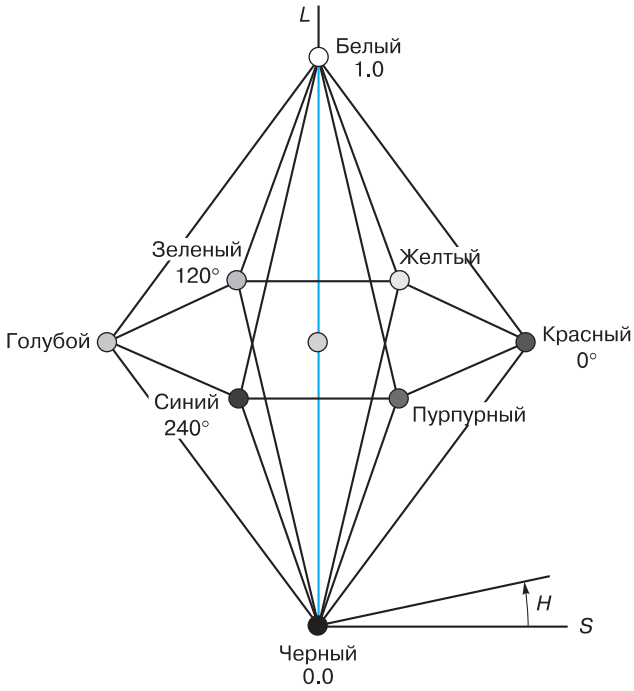



**Рис. 1.2.** Модель восприятия HLS

При использовании этой модели считается, что все оттенки заданы на едином цветовом круге. Поэтому первый параметр задает градус поворота от эталонного оттенка (0 — белый). Остальные параметры задают в процентах положение между максимальными и минимальными доступными значениями.

Модель также известна под названиями HSL, HSI.

Эта модель наиболее приближена к человеческому восприятию и описанию цвета. Она применяется в основном для описания цвета при анализе восприятия цвета человеком.

В **аддитивной цветовой модели RGB** (рис. 1.3 и рис. III на цветной вклейке) цвет образуется смешиванием трех компонентов:

- **R**ed — красного;
- **G**reen — зеленого;
- **B**lue — голубого.

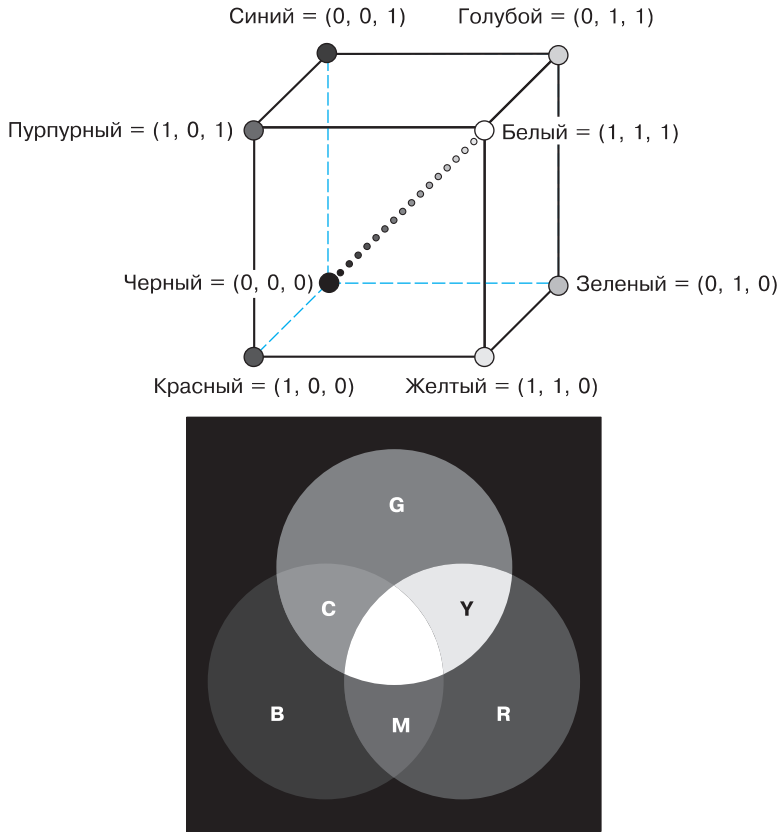



**Рис. 1.3.** Аддитивная цветовая модель RGB

Эта модель описывает цвет, который образуется из «суммы» света, излучаемого несколькими источниками. Эта модель является **аддитивной** (от лат. *additio* — прибавляю).

Самым популярным примером использования этой модели являются мониторы (в которых цвет каждого пикселя растра складывается из трех компонентов), проекторы и сканеры (которые чаще всего регистрируют отраженный свет).

Именно такая цветовая модель используется и в описании возможностей различных графических устройств. Цветовое пространство в этом случае описывают количеством битов, отводимых на сохранение цвета. Чаще всего используются режимы HighColor (16 битов, в соотношении 5:6:5 или 5:5:5) и TrueColor (24 бита, в соотношении 8:8:8).

Профессиональные программы обработки графической информации позволяют работать с расширенным представлением, когда на одну компоненту отводится не 8, а 16 битов.

Каждый компонент задается силой светимости, 0 соответствует отсутствию света. Таким образом, цвет 0-0-0 — это черный, цвет из равных долей каждого компонента — один из оттенков серого, а цвет с максимальными значениями компонентов — белый.

**Субтрактивная цветовая модель CMYK** (рис. 1.4 и рис. II на цветной вклейке). Если необходимо сформировать цвет точки из не светящихся самостоятельно компонентов, то аддитивная модель применяться не может, поскольку точка формируется из отраженного излучения. Поэтому для формирования цвета при печати была разработана субтрактивная, т. е. вычитающая модель цвета (удобнее рассматривать отраженную, а не поглощенную компоненту). В ней цвет формируется из трех основных компонентов:

- **C**yan — голубого;
- **M**agenta — ярко-красного;
- **Y**ellow — желтого.

Эти цвета получаются вычитанием из чистого белого света цветов аддитивной модели.

Формально при смешивании в равных максимальных долях эти цвета должны давать черный цвет.

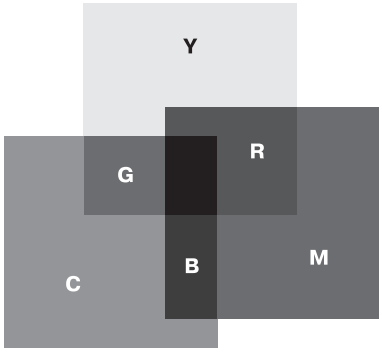



**Рис. 1.4.** Субтрактивная цветовая модель CMYK

Поскольку на практике точного черного цвета при смешивании не получается, то в модель добавляется компенсирующий четвертый компонент blac**K** — черный. Почему именно последняя буква взята в сокращение CMYK, точно не известно.

Эта модель формирования цвета используется при печати как в типографиях, так и в современных печатающих устройствах. В некоторые модели для достижения точности оттенков добавляют еще четыре цвета — осветленных.

Следует отметить, что преобразование из трехкомпонентной модели в четырехкомпонентную не может быть математически точным и всегда проходит с некоторыми искажениями. По этой причине оборудование при профессиональном использовании требует калибровки, а печать — учета большого количества параметров.

Именно из-за использования такой модели часто при печати сложных материалов указывают «печать в три краски» или «печать в четыре краски».

**Векторная графика.** Растровая графика является чрезвычайно мощным способом представления и хранения изображений, особенно фотографических, но в ряде случаев прямое использование такого подхода неудобно. Это случаи, когда изображение создается из типовых элементов — **графических примитивов** (точек, прямых и кривых линий и т. п.). Представление таких элементов в виде точек лишает нас возможности менять параметры примитива без перерисовки изображения, ограничивает возможности геометрических преобразований, требует много места при хранении.

Для преодоления этих ограничений применяется подход, подразумевающий хранение и обработку изображения не в виде растра, а в виде некоторых описаний отдельных элементов. Элементами обычно являются математические объекты с заданными конкретными параметрами. Параметры позволяют выполнить визуализацию элементов на устройстве вывода **(растеризацию)**, исходя из его характеристик и заданного «окна» просмотра.

Поскольку пространственное положение примитивов и способ отображения задаются с помощью координат, этот способ хранения и обработки изображений получил название **векторной графики**.

Одним из наиболее существенных достоинств векторной формы представления изображения является ее компактность и малая зависимость объема от размеров изображения.

К минусам этой формы представления относится отсутствие общих стандартов (практически у каждого редактора есть свои собственные форматы и особенности) и высокие требования к системным ресурсам, особенно вычислительным.

В программах подготовки векторных изображений (рис. 1.5) работа строится вокруг **объектов** (примитивов), обладающих некоторыми свойствами.

Наиболее распространенными примитивами являются: **отрезки**, **прямоугольники** и их производные (со сглаженными углами), **эллипсы** и их части, **кривые Безье** (математические кривые третьего порядка, задаваемые четырьмя точками), а также составленные из них **сложные контуры**. Одним из типовых объектов является текст, написанный, как правило, контурным шрифтом (векторным по сути).

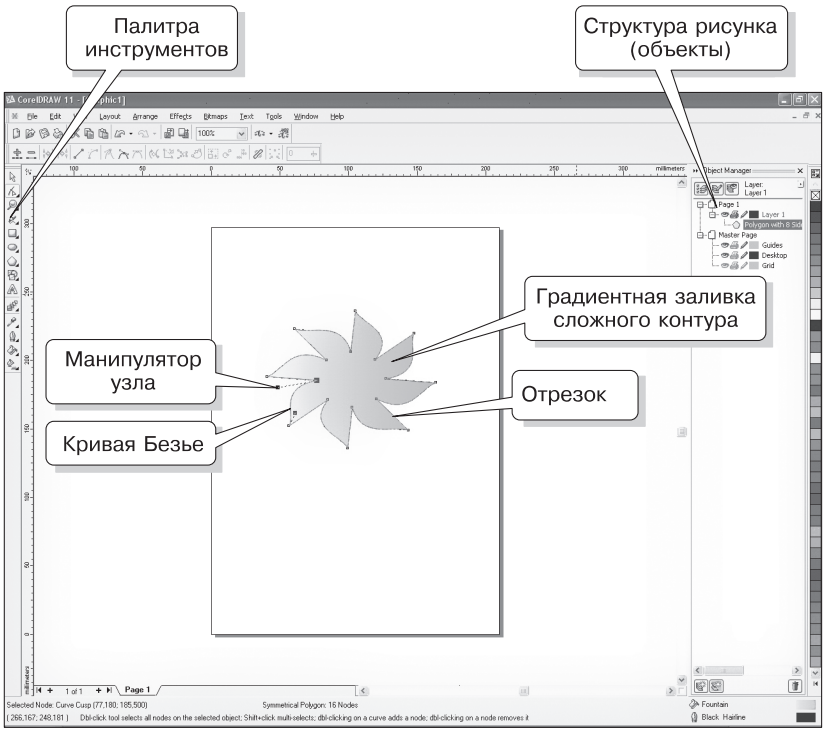



**Рис. 1.5.** Окно программы для обработки векторных изображений

Каждый объект может обладать целым рядом **свойств**. К ним, в частности, относят: **толщину линий** и способ их стыковки, **цвет**, **заливку** — способ заполнения замкнутого контура, накладываемые на объект **эффекты**. Параметрами является и положение объекта.

С объектами редактор векторной графики может выполнять большое количество разнообразных операций. К таким операциям относятся: **повороты**, **масштабирование**, **геометрические искажения** всевозможных видов, тиражирование готовых объектов. Специфика формы представления такова, что операции выполняются без искажений.

Современные редакторы векторной графики могут импортировать и использовать как готовые объекты, так и изображения растровой графики.

Редакторы векторной графики позволяют **группировать** объекты и создавать таким образом сложные объекты для выполнения операций над ними как над единым целым.

Объекты могут быть распределены на плоскости как «на поверхности», так и «по вертикали» (в разных слоях). Объекты отрисовываются по порядку и упорядочиваются друг относительно друга. Как и программы растровой графики, программы векторной графики поддерживают работу со слоями.

Векторная графика применяется в программах автоматизированного проектирования, для подготовки графических печатных материалов (например, плакатов), подготовки анимационных роликов к публикации в сети Интернет, презентаций.

## Вопросы и задания

1. Рассчитайте объем видеопамяти, необходимый для работы с монитором в режиме 1280 × 1024 пикселей в режиме TrueColor.
2. Какие дефекты возникнут, если увеличить растровое изображение формальным пересчетом пикселей в квадраты размером 8 × 8 пикселей с сохранением цвета?
3. На первых мониторах и растровых печатающих устройствах пиксели не имели оттенков. Предложите или найдите способ формирования изображений, содержащих полутона. Сформулируйте требования к аппаратуре, позволяющие применять такие методы.

4*. Изменение изображения на экране путем прямых вычислений с содержимым видеопамяти — достаточно длительный процесс. Предложите или найдите метод, который позволяет организовать анимацию при наличии достаточного количества видеопамяти.

# § 2
# Некоторые алгоритмы и методы машинной графики

- *Преобразования координат и фигур*
- *Построение фигур (алгоритм Брезенхема)*
- *Обработка растрового изображения*
- *Хранение изображений*
- *Фотореалистичные изображения. Моделирование физического мира*

При создании алгоритмов обработки изображений необходимо учитывать, что количество действий может оказаться большим из-за количества точек в растре. Это означает, что при обработке изображений нежелательно выполнять ресурсоемкие вычисления, иначе работа программы станет очень долгой.

**Преобразования координат и фигур.** Исторически первые задачи, которые решались в компьютерной (машинной) графике — это задачи создания изображения (графика, чертежа и т. д.). При подготовке таких рисунков очень часто возникает задача преобразования координат для отображения, преобразования объектов — масштабирования, поворота и т. д.

Такие задачи решаются методами аналитической геометрии с помощью преобразования координат.

Простейший случай — расчет координат точки на экране по ее математическим координатам.

Предположим, что у нас есть область для отображения, заданная координатами углов $(X_0, Y_0)$, $(X_1, Y_1)$. Нам нужно отобразить в ней объект (например, график функции), координаты которого укладываются в пределы $(x_0, y_0)$, $(x_1, y_1)$.

Тогда предварительно рассчитаем коэффициенты:

$$kx = (X_1 - X_0)/(x_1 - x_0) \text{ и } ky = (Y_1 - Y_0)/(y_1 - y_0).$$

Координаты точки на экране можно будет рассчитать так:

$$\begin{cases} X = \left[(x - x_0) \cdot kx + X_0\right]; \\ Y = \left[(y - y_0) \cdot ky + Y_0\right]. \end{cases}$$

Квадратными скобками мы обозначим операцию получения целой части числа. Если важно соблюдать масштаб, то коэффициент выбирается один — наименьший.

Фигуру, заданную координатами точек (всех или только ключевых), можно легко преобразовывать.

Сдвиг фигуры — это просто добавление числа к ключевым координатам.

Увеличить или уменьшить фигуру (и не обязательно оставлять ее на месте) можно просто, умножив координаты ее точек на коэффициент. Если фигура должна сохранить логическое положение на «листе», то координаты предварительно придется пересчитать в систему координат с центром в середине фигуры, определяемым как среднее арифметическое координат.

Важное свойство таких преобразований — сохранение формы фигуры. То есть линия преобразуется в линию, а поэтому можно пересчитать только координаты начала и конца отрезка, а промежуточные точки рассчитывать заново не надо.

**Построение фигур.** Как уже говорилось в § 1, одна из задач, которую необходимо решать при синтезе изображений — **растеризация**, т. е. задание цвета точек растра по математическому описанию фигуры. Приведем пример одного из основных алгоритмов, используемых в машинной графике в этом случае, — алгоритма построения отрезка. На его примере можно увидеть некоторые основные приемы решения таких задач.

Построение отрезка для нас означает заполнение тех точек растра, которые находятся на отрезке, соединяющем точки с заданными координатами. Очевидно, что просто вычислить эти координаты точно нельзя — поскольку не может быть «дробных» точек.

Самый распространенный алгоритм рисования точек получил название *алгоритма Брезенхема*.

Основная идея этого алгоритма — не использовать вычисления с плавающей точкой, а просто «пройти» по одной из осей и определить для каждой координаты в пределах линии, нужно ли для нее «сдвинуть» значение второй координаты.

Например, рассмотрим алгоритм для рисования линии на рис. 1.6, приняв координаты концов отрезка $x_0$, $y_0$ и $x_1$, $y_1$. Чтобы упростить понимание алгоритма, сначала запишем его с использованием дробных чисел, считая, что исходное изображе-

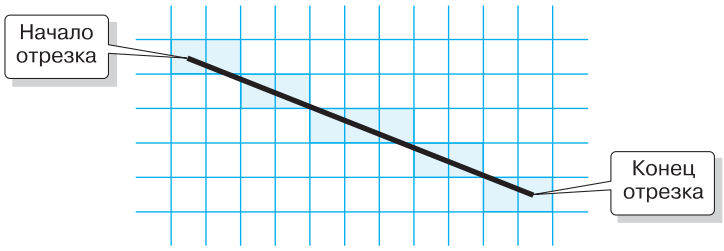



**Рис. 1.6.** Пример построения отрезка

ние — объект Image, все пиксели в нем — внутренняя матрица Pixel.

```
Deltax = x1 - x0
Deltay = y1 - y0
error = 0
deltaErr = Deltay/Deltax
y = y0
for x = x0 to x1
   Image.Pixel[x,y] = 1
   error = error + deltaErr
   if error > 0.5
      y = y + 1
      error = error - 1
```

Таким образом, мы вычислили коэффициент наклона, и увеличивали *y* тогда, когда прошли больше половины пикселя (т. е. накопили ошибку в переменной *error* больше, чем 0,5). Чтобы перейти к целым числам, просто умножим дробные числа на *Deltax* и на 2 (там, где использовался коэффициент 0,5):

```
Deltax = x1 - x0
Deltay = y1 - y0
error = 0
deltaErr = Deltay
y = y0
for x = x0 to x1
   Image.Pixel[x,y] = 1
   error = error + deltaErr
   if 2*error > Deltax
      y = y + 1
      error = error - Deltax
```

Чтобы получить полный алгоритм, нужно учесть и другие случаи наклона отрезка. Обычно их учитывают, меняя местами координаты и направления сдвига точек. Аналогично можно реализовать алгоритмы построения окружностей и эллипсов с осями, параллельными осям координат.

Алгоритм можно использовать и для большего разнообразия линий, например, задав шаблон в виде битовой карты (рисовать точку или не рисовать), получать пунктирные линии, использовать не просто точку, а фигуру сложной формы (кисть) и т. д.

**Обработка растрового изображения.** Рассмотренные нами алгоритмы предназначены для формирования растрового изображения по его векторному представлению, т. е. описанию отдельных элементов. Как вы знаете, очень часто такого описания у нас нет, а есть изображение, полученное в виде готовой иллюстрации (т. е. растра — матрицы точек).

Растровое изображение — основное средство представления и обработки фотографических изображений, стилизованных художественных рисунков, всевозможных диаграмм, текста. Тем не менее следует помнить, что смысл изображению придает человек (например, выделяет объекты) и при обработке растрового изображения мы не можем автоматически его учесть. Поэтому для манипулирования отдельными объектами и частями изображения программы работы с растровой графикой (рис. 1.7) предусматривают средства создания составных изображений с помощью механизма **слоев** (layers) — накладывающихся друг на друга плоскостей.

При обработке таких изображений часто применяют **фильтры** — преобразования цвета точек растра, которые применяются к какой-то зоне растра, преобразовывая (фильтруя) имеющиеся данные. Стоит заметить, что растр сам по себе никуда не движется, и все преобразования сдвига, поворота и т. п. — это фактически тоже перекрашивание точек.

Общий принцип работы фильтра таков: к каждой точке растра, находящейся в зоне действия фильтра, применяется некоторое действие для расчета ее нового цвета. Очень часто это действие основано на *весовой матрице*, которая называется *ядром преобразования*, т. е. матрице с нечетной длиной и шириной, в которой указан вес (т. е. доля участия) пикселя в итоговом значении. Матрица накладывается на изображение так, чтобы ее центр (именно поэтому длина и ширина выбраны нечетными) совпал с изменяемым пикселем. Параметры пикселей изменяемой зоны,

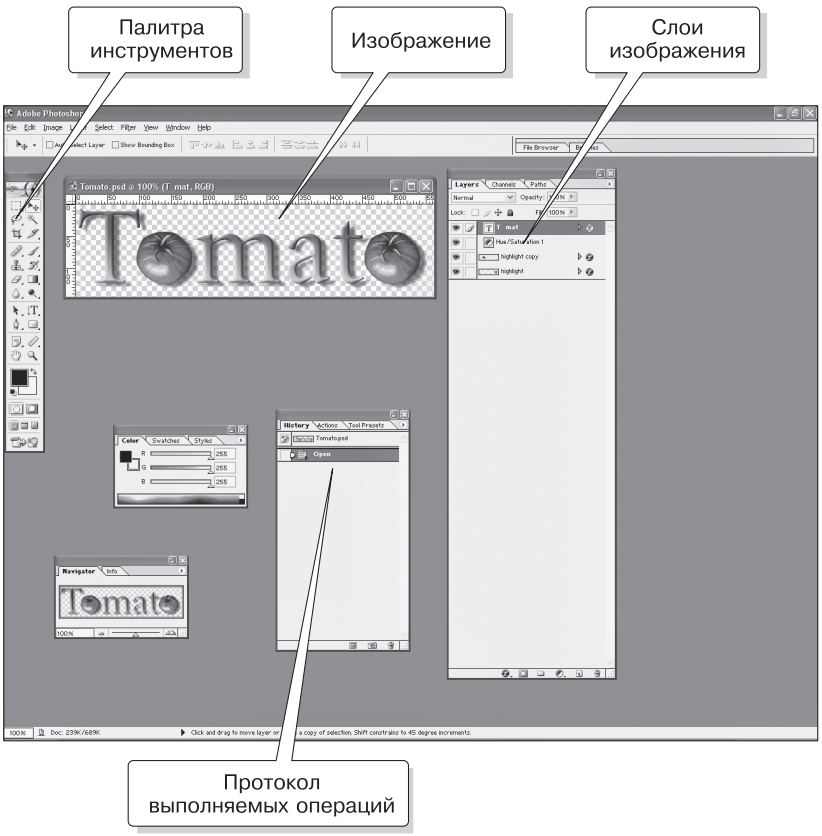



**Рис. 1.7.** Окно программы для обработки растровых изображений

попавшие под матрицу, суммируются с весом, указанным в матрице, результат нормируется (т. е. пересчитывается в доли от общего веса) и записывается в центральный пиксель. После этого матрица сдвигается на 1 пиксель, и все повторяется.

Например, для фильтра «Гауссово размытие» ядро может быть таким:

$$\begin{pmatrix} 1 & 2 & 3 & 2 & 1 \\ 2 & 4 & 5 & 4 & 2 \\ 3 & 5 & 6 & 5 & 3 \\ 2 & 4 & 5 & 4 & 2 \\ 1 & 2 & 3 & 2 & 1 \end{pmatrix}.$$

Такое преобразование будет «смешивать цвета» в зависимости от расстояния — чем дальше, тем меньше влияет.

Для прямоугольной области изображения алгоритм применения фильтра несложно записать на псевдокоде, считая, что исходное изображение — объект *Image*, все пиксели в нем — внутренняя матрица *Pixel*, и у каждого пикселя есть свойства *red*, *green* и *blue* — цветовые компоненты. Аналогично будет устроен объект-результат: *resultImage*. Ядро преобразования мы будем хранить в аналогичной структуре *Kernel*, в которой укажем ширину и высоту ядра, и весовую матрицу.

```
For x = 1 to Image.Width
   For y = 1 to Image.Height
      rSigma = 0
      gSigma = 0
      bSigma = 0
      kSum = 0
      for i = 1 to Kernel.Width
         for j = 1 to Kernel.Height
            curPixX = x + (i — Kernel.Width/2)
            curPixY = y + (j — Kernel.Height/2)
            if ((curPixX < 0) or
                        (curPixY < 0) or
                        (curPixX > Image.Width) or
                        (curPixY > Image.Heigth)
               continue
            rSum = rSum + Image.Pixel[curPixX,
                      curPixY].red*kernel.Weght[i,j]
            gSum = gSum + Image.Pixel[curPixX,
                     curPixY].green*kernel.Weght[i,j]
            bSum = bSum + Image.Pixel[curPixX,
                      curPixY].blue*kernel.Weght[i,j]
            kSum = kSum + kernel.Weght[i,j]
      resultImage.Pixel[x,y].red = rSum/kSum
      resultImage.Pixel[x,y].green = gSum/kSum
      resultImage.Pixel[x,y].blue = bSum/bSum
```

В примере хорошо видно, что чем больше изображение и сложнее фильтр, тем больше придется выполнить вычислений.

Этот алгоритм при написании программы можно (и нужно) оптимизировать. Например, вес матрицы в нашем случае совер-

шенно не обязательно считать каждый раз, поскольку эта величина статическая.

В результате применения такого фильтра изображение будет размытым, например, так делают для маскировки «склейки» фотографий, размывая края объектов.

Еще один пример такого преобразования — осветление изображения. Для этого достаточно представить цвета точек в модели HLS и увеличить значение компонента Light. После этого цвет пересчитывается снова в RGB и отображается.

Аналогично действуют фильтры обычного размытия, повышения резкости, выявления краев и т. п.

Использование слоев, фильтров, применение преобразований по шаблонам — основной современный метод обработки самых различных изображений.

**Хранение изображений.** Одна из существенных задач, возникающих при обработке изображений, — это их хранение. Если использовать прямой способ сохранения, то изображения станут занимать чрезмерно много места на носителях, и, что в современных условиях существеннее, потребуют намного больше времени при пересылке[1]. Для сокращения объема изображений применяют уже упоминавшиеся нами методы сжатия информации, т. е. ее кодирования, уменьшающего размер.

Все методы сжатия делят на две большие группы: *алгоритмы сжатия без потерь информации* и *с потерями*.

Первая группа методов применяется тогда, когда исходное изображение нужно восстановить без каких-либо искажений: например, при подготовке качественной печати, в научной графике и т. д.

Самый простой пример алгоритма сжатия без потерь — метод RLE (от англ. *Run-length encoding*), кодирование путем учета длин повторов. Принцип его работы очень прост — записывается цвет точки и количество его повторов до другого цвета.

Если длинных последовательностей много, то результат будет очевидно лучше, если повторов нет, то размер изображения даже вырастет.

[1] Речь идет не только о передаче данных в сетях, но и о записи изображения на носитель, например, при выполнении съемки серии фотографий.

В современных системах используют чаще всего более сложные алгоритмы, построенные на поиске повторяющихся последовательностей, а не просто цветов. Такие методы эффективны при сжатии чертежей, контурных рисунков, текста и т. п.

Очевидно, что для изображений с малым количеством повторов (практически все фотографии) такие методы будут неэффективны — цвета рядом хоть чуть-чуть, но различаются.

Поэтому метод RLE в настоящее время используется редко.

Вторая группа методов получила название сжатия с потерями: изображение сжимается этими методами с искажением, т. е. потерей части информации. Несмотря на формальные потери, в реальных условиях человек воспримет это искажение как допустимое (а иногда вообще не заметит).

В таких методах сжатия используется несколько приемов:

1) некоторые компоненты цвета человек воспринимает хуже других, поэтому можно отвести на них меньше битов — всё равно разница будет малозаметна;
2) если разница в яркости между соседними пикселями невелика, то их цвета можно приблизить друг к другу — разница будет не очень заметна.

В наиболее популярном алгоритме сжатия изображений с потерями JPEG изображение разбивается на квадраты размером 8 × 8 пикселей и внутри этих квадратов выполняется несколько преобразований.

При выполнении сжатия с потерями, как правило, можно задать коэффициент сжатия — и чем выше этот коэффициент, тем меньше будет сжато изображение, но тем лучше будет его качество при восстановлении. Сильно сжатое изображение будет «рассыпаться» на квадраты.

**Фотореалистичные изображения. Моделирование физического мира.** С ростом вычислительной мощности и доступности элементов памяти, с появлением качественных графических терминалов и устройств вывода была разработана большая группа алгоритмов и программных решений, которые позволяют формировать на экране изображение, представляющее некоторую объемную сцену. Первые такие решения были предназначены для решения задач архитектурного и машиностроительного проектирования.

При формировании такого изображения (статического или динамического) его построение рассматривается в пределах некоторого пространства координат, которое называется **сценой**. Сцена

подразумевает работу в объемном, трехмерном мире, поэтому и направление получило название трехмерной (3-Dimensional, 3D) графики (рис. 1.8).

На сцене размещаются отдельные объекты, составленные из геометрических объемных тел и участков сложных поверхностей (чаще всего для построения применяются так называемые **B-сплайны**). Для формирования изображения и выполнения дальнейших операций поверхности этих тел разбиваются на треугольники — минимальные плоские фигуры и в дальнейшем обрабатываются именно как набор треугольников.

На следующем этапе **мировые** координаты узлов сетки пересчитывают с помощью матричных преобразований в координаты **видовые**, т. е. зависящие от точки зрения на сцену. Положение точки просмотра, как правило, называют положением **камеры**.

После формирования **каркаса** («проволочной сетки») выполняется **закрашивание** — придание поверхностям объектов некоторых свойств. Свойства поверхности в первую очередь определяют-

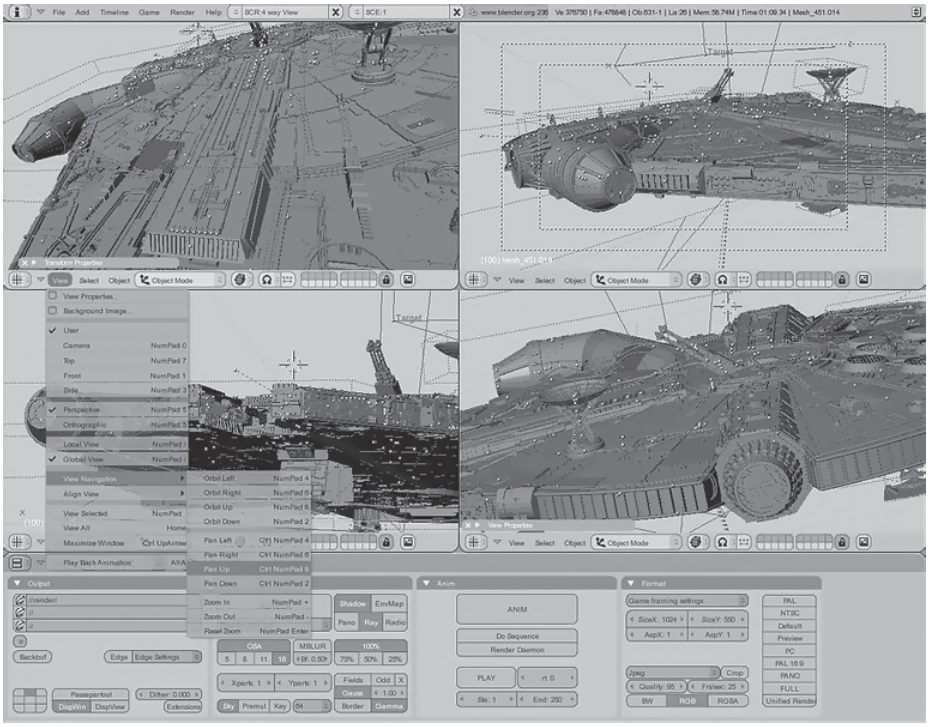


**Рис. 1.8.** 3D-моделирование в системе Blender

ся ее световыми характеристиками — светимостью, отражающей, поглощающей и рассеивающей способностью. Этот набор характеристик позволяет определить материал, поверхность которого моделируется, — металл, пластик, стекло и т. п. Материалы прозрачные и полупрозрачные обладают еще рядом характеристик.

Как правило, во время выполнения этой процедуры выполняется и **отсечение невидимых поверхностей**. Существует много методов выполнения такого отсечения, но самым популярным стал метод **Z-буфера**, когда создается массив чисел, обозначающий «глубину» — расстояние от точки на экране до первой непрозрачной точки. Следующие точки поверхности будут обработаны только тогда, когда их глубина будет меньше — и тогда координата Z уменьшится. Мощность этого метода напрямую зависит от максимально возможного значения удаленности точки сцены от экрана, т. е. от количества битов на точку в буфере.

Выполнение указанных операций позволяет создать так называемые **твердотельные модели** объектов, но реалистичным это изображение не будет. Для формирования реалистичного изображения на сцене размещаются **источники света** и выполняется **расчет освещенности** каждой точки видимых поверхностей (рис. 1.9).

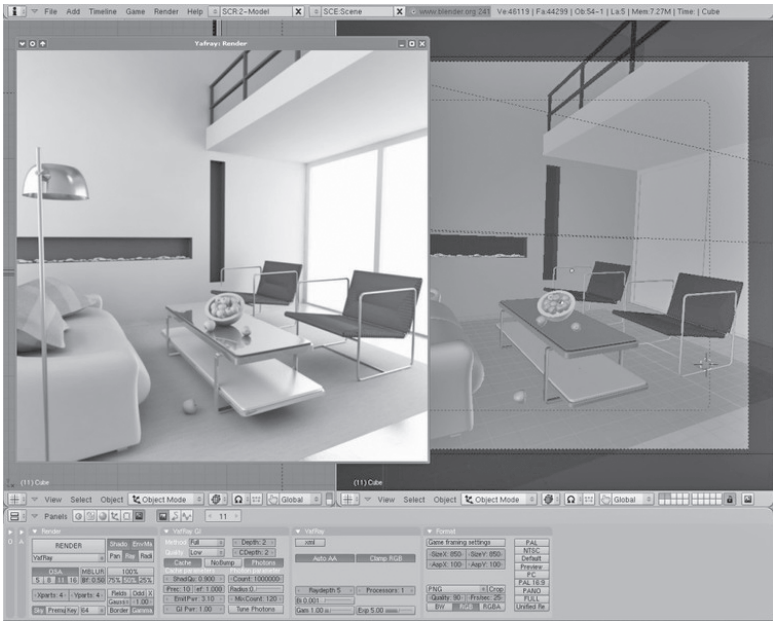


**Рис. 1.9.** Расчет освещенности в системе Blender

Для придания реалистичности поверхность объектов «обтягивается» **текстурой** — изображением (процедурой, формирующей его), определяющим нюансы их внешнего вида. Процедура называется наложением текстуры. Во время наложения текстуры применяются методы растяжения и сглаживания — так называемая фильтрация. Пример: упоминаемая в описании видеокарт **анизотропная фильтрация**, не зависящая от направления преобразования текстуры.

После определения всех параметров необходимо выполнить процедуру формирования изображения, т. е. расчет цвета точек на экране. Процедура обсчета называется **рендерингом**. Во время выполнения такого расчета необходимо определить свет, попадающий на каждую точку модели, с учетом того, что он может отражаться, что поверхность может закрывать другие участки от этого источника и т. п.

Для расчета освещенности применяются два основных метода. Первый — метод **обратной трассировки луча**. При использовании этого метода рассчитывается траектория тех лучей, которые в итоге попадают в пиксели экрана по обратному ходу. Расчет ведется отдельно по каждому из цветовых каналов, поскольку свет разного спектра ведет себя на разных поверхностях по-разному.

Второй метод — **метод излучательности**. Он предусматривает расчет интегральной светимости всех участков, попадающих в кадр, и обмен светом между ними.

На полученном изображении учитываются заданные характеристики камеры — средства просмотра.

Таким образом, в результате большого количества вычислений появляется возможность создавать изображения, мало отличающиеся по общему впечатлению от фотографий. Для уменьшения количества вычислений стараются уменьшить число объектов и там, где это возможно, заменить расчет фотографией, например, при формировании фона изображения.

После разработки методов формирования статического изображения следующим шагом в развитии технологий трехмерной реалистичной графики стали возможности ее анимации — движения и покадрового изменения сцены. Первоначально с таким объемом расчетов справлялись только суперкомпьютеры, и именно они использовались для создания первых трехмерных анимационных роликов.

Позже были разработаны специально предназначенные для обсчета и формирования изображений специальные платы расширения, взаимодействовавшие с видеокартами, — **3D-акселераторы**, позволившие в упрощенной форме выполнять такое формирование в реальном масштабе времени, что и используется в современных компьютерных играх. Фактически сейчас даже обычные видеокарты включают в себя такие средства и являются своеобразными мини-компьютерами узкого назначения.

При моделировании и проектировании различных объектов в процессе создания игр, съемок фильмов, разработки тренажеров у задачи формирования реалистичного изображения появляется еще один существенный аспект — моделирование не просто движения и изменения объектов, а моделирование их поведения, соответствующего физическим принципам окружающего мира.

Такое направление, с учетом применения всевозможных аппаратных средств передачи воздействий внешнего мира и повышения эффекта присутствия, получило название **виртуальной реальности**.

Для реализации такой реалистичности создаются специальные методы расчета параметров и преобразования объектов — изменения прозрачности воды при ее движении, расчет поведения и внешнего вида огня, взрывов, столкновения объектов и т. д. Такие расчеты носят достаточно сложный характер, и для их реализации в современных программах предложен целый ряд методов.

Один из них — это обработка и использование **шейдеров** — процедур, изменяющих освещенность (или точное положение) в ключевых точках по некоторому алгоритму. Это позволяет создавать эффекты «светящегося облака», «взрыва», повысить реалистичность сложных объектов и т. д.

Появились и стандартизируются интерфейсы работы с «физической» составляющей формирования изображения, что позволяет повысить скорость и точность таких расчетов, а значит и реалистичность создаваемой модели мира.

Трехмерная графика — одно из самых зрелищных и коммерчески успешных направлений развития информационных технологий, часто ее называют одним из основных стимулов развития аппаратного обеспечения.

Средства трехмерной графики активно применяются в архитектуре, машиностроении, научных работах, при съемке кинофильмов, в компьютерных играх, обучении.

### Вопросы и задания

1. Предложите матрицу фильтра, который сдвинет точки растра на 3 пикселя вперед и вниз, не изменяя их цветов.
2. Используя предлагаемое в параграфе описание алгоритма Брезенхема для рисования линии, определите класс его сложности.
3. Сколько места займет на диске растровое изображение размером 10 × 15 дюймов с разрешением 300 точек на дюйм без применения сжатия, если в нем используется цвет TrueColor? Если для представления цвета используется палитра на 16 цветов?
4. Каков порядок и количество действий при применении фильтра Гауссова размытия с предложенной матрицей к растру размером 10 × 15 дюймов и разрешением 300 точек на дюйм?
5. Как увеличится количество действий в предыдущем задании, если разрешение увеличится до 600 точек на дюйм?

## § 3 Визуализация

- *Визуализация числовых данных*
- *Карты и геоинформационные системы*
- *Схемы*

Числовая и текстовая форма представления информация, как мы видели, предоставляет огромные возможности по ее хранению и обработке. Целый ряд задач никаким другим способом не может быть решен. Тем не менее вполне очевидно и то, что восприятие информации в этих формах требует от любого человека некоторых усилий. Фактически это усилия на получение (чтение) данных, их перекодирование и связывание специфического содержания с реальностью (осмысление). Осмыслить и понять двоичные данные по очевидным причинам невозможно.

В целом ряде случаев объем данных таков, что их чтение и осознание вообще выходят за рамки человеческих возможностей[1].

Скорость восприятия человеком ситуации, очевидно, важна даже в тех случаях, когда в принципе такая возможность сохраняется — для ускорения принятия решений, для снижения количества ошибок, просто для удобства.

Поэтому одним из важных направлений применения компьютерной графики является ***визуализация*** — представление в виде изображения изначально не графических данных.

Первое, с чего начинается визуализация, это с выбора основного средства отображения. Рассмотрим эти средства.

1. *Пиктограммы* (*иконки*) (рис. 1.10). Пиктограммы — это небольшие условные изображения, обозначающие какое-либо понятие. Пиктограмма выбирается таким образом, чтобы при взгляде на нее сразу становилось понятно, о чем идет речь.
2. *Изображение* (рис. 1.11 и рис. IV на цветной вклейке). Если есть возможность, то удобно продемонстрировать данные изображением самого объекта — реального или условного. Такое изображение делается максимально реалистичным.
3. *Диаграмма* (рис. 1.12). Это плоское изображение, позволяющее сопоставить данные.
4. *Визуальная метафора* (рис. 1.13). В этом случае процесс или объект описывается с помощью метафоры, т. е. в переносном значении, например как путешествие.
5. *Карта* (*граф*). На карте (часто условной) соотношения между объектами демонстрируются положением и связями.
6. *Схема*. Это отображение информации в виде набора связанных объектов, условно обозначающих элементы.

Средства отображения вполне могут сочетаться на одном изображении, если это необходимо.

[1] Большой адронный коллайдер во время эксперимента вырабатывает поток данных, поступающий со скоростью свыше 54 терабайтов в секунду. Бóльшая часть этого потока отсекается электронными схемами внутри самого детектора, и на выходе получается около 300 мегабайт в секунду.


**Рис. 1.10.** Пиктограммы (иконки)[1]

**Рис. 1.11.** Визуализация сложных математических объектов

[1] С сайта http://www.visual-literacy.org

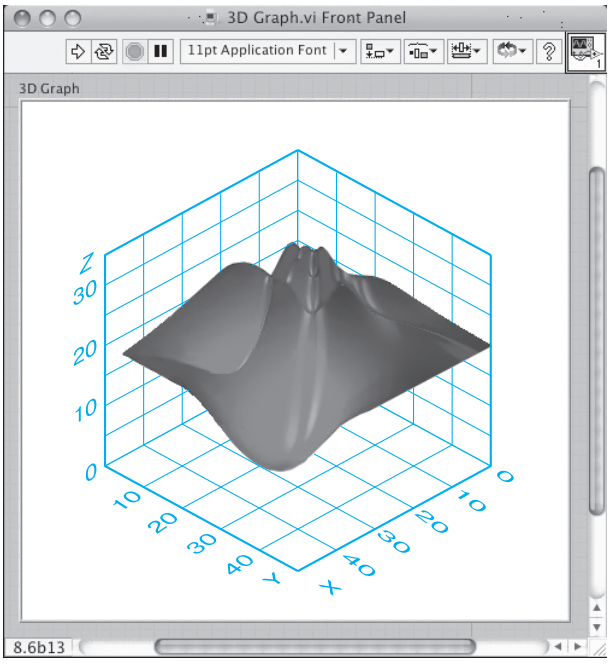



**Рис. 1.12.** Трехмерный график — один из видов диаграммы

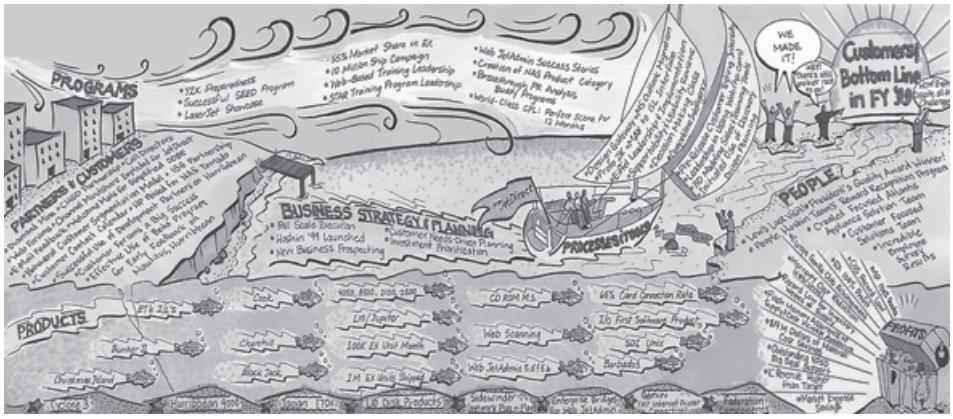



**Рис. 1.13.** Визуальная метафора

По сути визуализация — процесс создания изображения. При этом мы можем использовать несколько общих средств:

1) форма и размер объектов;
2) взаимное расположение объектов;
3) цвета и текстуры объектов и фона.

Мы рассмотрим несколько типовых решений, часто используемых в разных областях.

**Визуализация числовых данных** — очень распространенная задача, поскольку воспринимать большой объем такой информации человеку особенно трудно. Для решения этой задачи чаще всего применяют различные диаграммы.

***Диаграммой*** (от греч. Διαγραμμα (*diagramma*) — условное изображение, чертеж, рисунок) называют плоское схематическое изображение какого-либо объекта изучения. Даже когда речь идет о трехмерной диаграмме, на самом деле мы все равно будем создавать изображение на плоскости (например, на плоскости экрана).

Основной целью создания диаграммы является сравнение — сопоставление значений. Возможных видов сравнений не так много.

1. *Покомпонентное.* Этот вид сравнения сопоставляет состояние нескольких элементов, например, доли продажи разных товаров в прибыли предприятия.
2. *Позиционное.* В этом случае мы сравниваем позиции, т. е. взаимное расположение, например место по порядку.
3. *Временно́е.* В этом случае мы сравниваем состояния объектов во времени и можем ответить на вопрос «как они развиваются?».
4. *Частотное.* Это сравнение применяется для того, чтобы понять, какие объекты встречаются чаще.
5. *Корреляционное.* Сравнивая колебания нескольких параметров, мы можем на схеме показать их связь.

Один из самых частых случаев применения диаграмм — визуализация статистических сведений. Основных видов таких диаграмм пять.

1. *Круговая диаграмма* (рис. 1.14) — обозначение объектов как секторов круга.
2. *Гистограмма* (рис. 1.15) — столбчатая диаграмма.

3. *Линейчатая (полосная) диаграмма.*
4. *График* (рис. 1.16) — отображение зависимых данных в виде непрерывных линий (одна независимая переменная) или поверхностей (две независимых переменных).
5. *Точечная диаграмма* (рис. 1.17). Отображение данных в виде точек на плоскости или в пространстве, позволяющее оценить их группировку или предположить зависисимость.

У каждого из этих видов множество разновидностей, но в целом перечисленные пять видов — основа для всех остальных. Первые три вида применяют для сравнений ограниченного числа элементов — ***категорий***. Четвертый и пятый используют в тех случаях, когда сравниваются результаты по общей непрерывной переменной, например времени.

Очень часто тип диаграммы выбирается неверно. Рекомендуется использовать при выборе типа диаграммы следующий подход.

1. В случае **покомпонентного** сравнения выбирайте круговую диаграмму. В этой диаграмме мы с одной стороны можем сравнить размеры объектов, с другой — нам не требуется накладывать их друг на друга. Цвет служит средством отличия одного элемента от другого. Часто допускают ошибку — сравнивают компоненты с помощью гистограммы, тогда визуально гораздо сложнее сопоставить, например, первый и последний объекты в ряду.

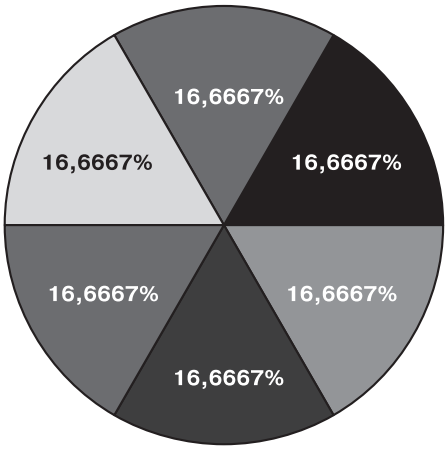



**Рис. 1.14.** Круговая диаграмма

2. При позиционном сравнении лучше всего использовать гистограмму. Задача позиционного сравнения — показать, во-первых, место по порядку, а во-вторых, величину отрыва. Иногда элементы ставят не по порядку, чтобы подчеркнуть различие некоторых из них.

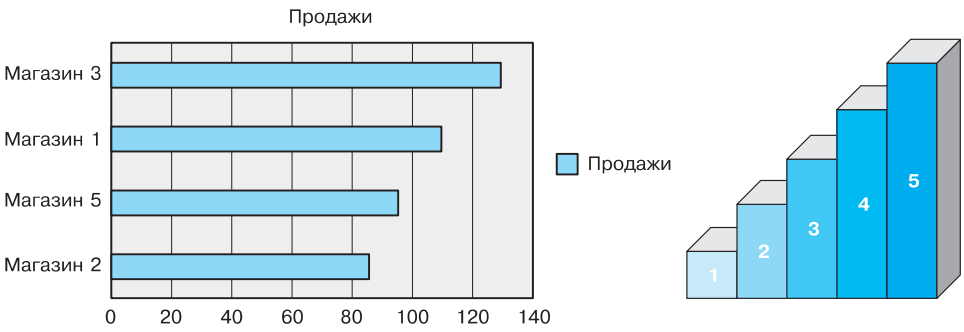



**Рис. 1.15.** Гистограммы

3. Временно́е сравнение применяется для сравнения величин, изменяющихся во времени. Сравнивать их можно либо на гистограмме (если количество замеров ограничено), либо на графике (если промежуточные значения имеют смысл).


**Рис. 1.16.** График

4. Частотное сравнение чаще всего применяется, чтобы сопоставить величины, измеренные по номинальной шкале. Если изменяемая переменная имеет промежуточные значения, то можно построить график.

5. Корреляционное сравнение применяется для выявления связи двух величин. Для этого можно использовать точечную диаграмму. Если количество величин конечно, то сравнить их можно на двойной линейчатой диаграмме. Связанные величины должны меняться связанно (например, при увеличении одной стороны увеличивается и другая). Если связи нет, изменения будут независимыми. Если же величины непрерывны, мы можем отметить все имеющиеся точки на плоскости и оценить их расположение.


**Рис. 1.17.** Точечная диаграмма

Перечислим некоторые типовые ошибки при подготовке диаграмм.

1. *Диаграмма, подготовленная без учета способа просмотра.* При демонстрации для аудитории на экране, например, не имеет смысла показывать гистограмму на 60 категорий. Ничего не будет видно. А при просмотре данных на экране чаще всего не имеет смысла строить диаграмму на три категории — и так все представимо.
2. *Нарушение масштаба.* Особенно часто возникает на графиках. Если исключить нулевую точку, то растяжение и сжатие изменяют степень изменения графиков — в результате неверно оцениваются тенденции.
3. *Перегруженная диаграмма* — в попытке уместить и передать больше информации на диаграмме отражают значения множества параметров, в результате чего разобрать что-либо трудно.

**Карты и геоинформационные системы.** Исторически различные карты и схемы, показывающие взаимное расположение

объектов, — самый старый способ визуализации данных. Картография развивалась несколько тысяч лет, и в ее рамках разработано много высокоточных и информативных способов представления различной информации, связанной с определенной территорией (рис. 1.18 и рис. V на цветной вклейке).

Создание и отображение и самих карт, и информации напрямую связано с расчетами, а связанной с местом информации всегда много (от расположения домов и коммуникаций до изменения температуры). Это все привело к появлению целого класса информационных систем, позволяющих такую информацию вводить, отображать и обрабатывать, — *геоинформационных*.

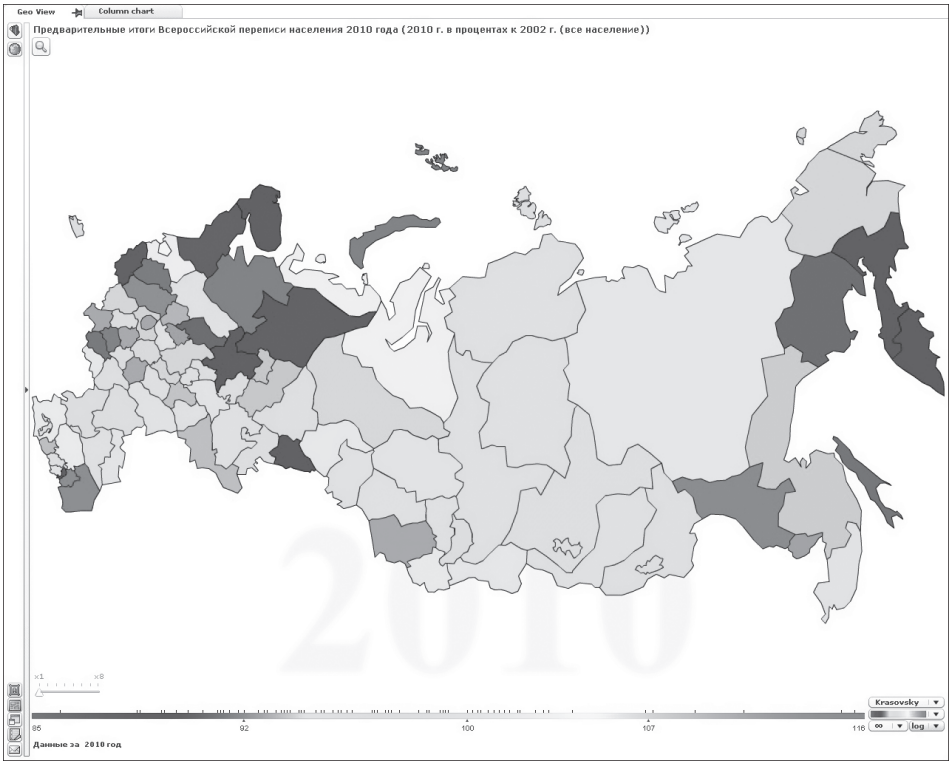



**Рис. 1.18.** Тепловая карта[1] — один из видов компонентной диаграммы

[1] Карта подготовлена на сайте datapult.info.

Например, на рис. 1.19 показана фотокарта земной поверхности с отображаемой на ней погодой.

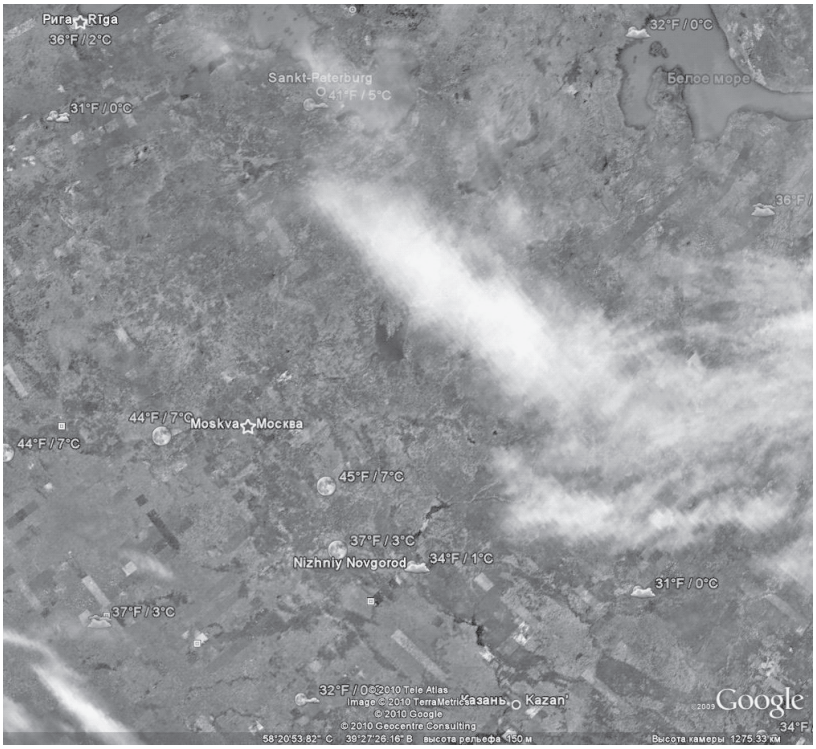



**Рис. 1.19.** Изображение Google Earth

Геоинформационные системы — чрезвычайно мощный и удобный инструмент визуализации реальных данных, позволяющий не просто показать данные, но и оценить их местоположение, взаимное влияние и т. п.

**Схемы.** Еще один популярный и эффективный способ визуализации — схемы.

Схемы применяются в самых разных областях — от техники до финансов и позволяют быстро выделить суть взаимоотношений.

Для некоторых видов схем существуют заранее обусловленные наборы знаков, определенных стандартами. Применение таких стандартизированных схем позволяет использовать их для обмена сведениями между людьми без дополнительного согласования.

Например:

1) блок-схема процесса выпуска книги в издательстве;
2) дерево устранения ошибок.

Основа схемы — это набор блоков разной формы, связанных между собой. Схемы применяются в тех случаях, когда надо показать структуру некоторого информационного объекта. Приведем пример схемы, предназначенной для организации и визуализации процесса мышления, — так называемой карты памяти (карты знаний, MindMap) (рис. 1.20).

Карты знаний строятся от общего центра (ядра). От центра отходят варианты решения (пути), классы и т. п. Каждый из этих классов может развиваться дальше.

Рисовать такую схему можно (и даже рекомендуется) вручную, используя большой лист бумаги и разноцветные фломастеры, но это потребует много времени и материалов.

Использование компьютеризированных средств позволяет организовать работу нескольких человек, временно убирать с экрана ненужные блоки и связи, менять оформление карты.

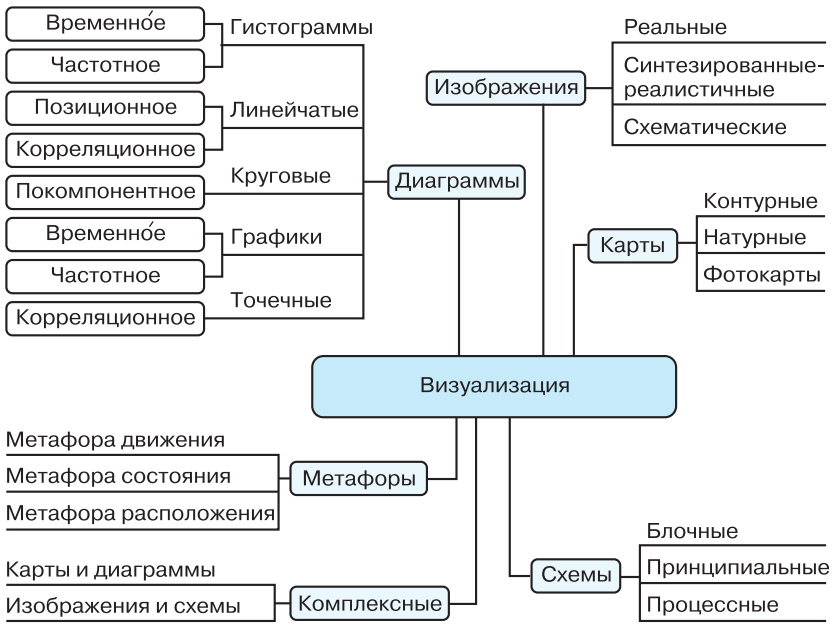



**Рис. 1.20.** Карта знаний этого параграфа

## Вопросы и задания

1. Подготовьте карту знаний какого-либо раздела учебника информатики. Сравните получившийся результат с аналогичными работами одноклассников и сделайте выводы о том, чья карта и почему оказалась наиболее оптимальной.
2. Подготовьте тепловую карту плотности населения регионов России и сопредельных стран. Какие выводы позволяет сделать карта? В каких областях можно использовать полученную информацию?
3. Подготовьте 3D-модель школьного здания в рамках тематического проекта. Какую программу для работы и почему вы будете использовать?
4. Подготовьте визуальное отображение успеваемости в разных классах в зависимости от времени учебного года. Какие средства визуализации предпочтительнее при этом использовать? Сделайте выводы и прокомментируйте полученные результаты.
5. Выполните проект «Графика и визуализация» из задачника-практикума.

## Коротко о главном

Бóльшую часть сведений об окружающем мире человек получает с помощью зрения, поэтому очень важно иметь способ сохранения такого зрительного образа, реального или вымышленного. Для визуального представления информации разработано целое направление информационных технологий, получившее название *компьютерной графики*. Средства компьютерной графики используются всегда, когда нужно представить информацию для зрительного восприятия.

Для хранения и обработки изображений в компьютерной графике предусмотрены два основных способа их представления: в виде матрицы одинаковых прямоугольников (пикселей), окрашенных в разные цвета, — *растровая графика*, или в виде математического описания фигур и параметров их отображения — *векторная графика*.

Общим средством отображения фигур или точек растра для этих средств является цвет. Кодирование цвета выполняется с помощью разных способов его описания — *цветовых моделей*. Чаще других используется аддитивная (складывающая компоненты) модель RGB, субстрактивная (рассматривающая отраженный свет, вычитающая) модель CMYK, а для отражения человеческих представлений — модель HLS.

Растровая графика позволяет сохранить и обработать любое изображение, не выделяя в нем отдельные объекты реального

мира. Таким образом можно работать с фотографиями, рисунками и т. д. При обработке таких изображений часто применяют *слои* (формирование изображения из частей, наложенных друг на друга, с указанием параметров наложения), *фильтры* (средства преобразования цветов точек в некоторой окрестности). Геометрические преобразования растрового изображения часто приводят к ухудшению его качества, поскольку действия выполняются над координатам точек формально.

Растровая графика — основное средство представления изображения при его показе или печати.

Векторная графика применяется тогда, когда можно представить изображение в виде набора объектов. Векторная графика позволяет производить геометрические операции без заметных искажений, поскольку пересчитываются все ключевые параметры. При отображении отдельных элементов требуется выполнить преобразование описания в набор точек растра. Такая операция называется *растеризацией*.

Векторная графика — основа создания и обработки синтезируемых изображений: чертежей, шрифтов, плакатов и т. д.

Рост вычислительных возможностей позволил создать и широко применять средства отображения и обработки не только готовых плоских изображений, но и объемных. Поскольку эти методы рассматривают представление изображения в трех измерениях, то подход получил название трехмерной (3-dimensional, 3D) графики. При построении трехмерного изображения оно представляется в виде описания поверхностей с набором параметров (цвет, фактура, материал и др.) и указания расположения источников света. На основании этого описания строится плоское изображение, для которого рассчитывается цвет каждой точки растра, каким он воспринимался бы из заданной точки через оптическое устройство *камеру*.

Получившееся плоское изображение может быть настолько похожим на фотографию, что их также называют *фотореалистичными изображениями*. Средства 3D-графики активно применяются в играх, научной графике, при подготовке художественных фильмов, визуализации разных предметов.

## Источники

1. *Шикин Е. В.*, *Боресков А. В.* Компьютерная графика. Полигональные модели. — М.: Диалог-Мифи, 2001. — 461 с.
2. *Желязны Джин.* Говори на языке диаграмм. — М.: Манн, Иванов и Фербер, 2007. — 320 с.