Алексей Поляков
Виталий Брусенцев

bhv®

Методы и алгоритмы компьютерной графики

в примерах на Visual C++

2-е издание

+ CD-ROM

- Основы компьютерной графики
- Работа с векторной и растровой графикой
- Приемы программирования
- Обзор основных графических библиотек
- Использование возможностей библиотеки GDI+

Алексей Поляков
Виталий Брусенцев

Методы и алгоритмы компьютерной графики

в примерах на Visual C++ и C#

2-е издание

Санкт-Петербург

«БХВ-Петербург»

2003

УДК 681.3.068+800.92 Visual C++
ББК 32.973.26-018.1
П54

Поляков А. Ю., Брусенцев В. А.

П54 Методы и алгоритмы компьютерной графики в примерах на Visual C++, 2-е изд., перераб. и доп. — СПб.: БХВ-Петербург, 2003. — 560 с.: ил.

ISBN 5-94157-377-4

В книге последовательно, "от простого к сложному", рассматриваются понятия, алгоритмы и методы компьютерной графики, а также средства программирования. Описаны особенности платформ Windows и .NET, разработка программ в среде Visual C++ с использованием объектно-ориентированного стиля программирования, возможности, предоставляемые библиотекой MFC и архитектурой Document-View, создание многопоточных приложений с MDI-интерфейсом. Подробно анализируются возможности библиотеки нового поколения GDI+: рисование векторных примитивов сложной формы с градиентной заливкой, управление прозрачностью векторных и растровых объектов, поддержка форматов графических файлов (BMP, GIF, TIFF, JPEG и др.), отрисовка растров с наложением альфа-канала, масштабированием, растяжением, искажением и поворотом. К книге прилагается компакт-диск с примерами программ и изображений.

Для студентов и программистов

УДК 681.3.068+800.92 Visual C++
ББК 32.973.26-018.1

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Анатолий Адаменко*
Зав. редакцией	*Григорий Добин*
Редактор	*Дарья Масленникова*
Компьютерная верстка	*Натальи Караваевой*
Корректор	*Елена Самсонович*
Оформление серии	*Via Design*
Дизайн обложки	*Игоря Цырульникова*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 14.08.03.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 45,15.
Тираж 3 000 экз. Заказ №
"БХВ-Петербург", 198005, Санкт-Петербург, Измайловский пр., 29.

Гигиеническое заключение на продукцию, товар № 77.99.02.953.Д.001537.03.02
от 13.03.2002 г. выдано Департаментом ГСЭН Минздрава России.

Отпечатано с готовых диапозитивов
в Академической типографии "Наука" РАН
199034, Санкт-Петербург, 9 линия, 12.

ISBN 5-94157-377-4

Содержание

Структура книги

Книга состоит из 15 глав и тематически может быть разделена на пять частей. Первая часть — изучение основных понятий компьютерной графики и средств программирования (*главы 1–3*). Вторая часть — векторная графика (*главы 4–8*). Третья часть — растровая графика (*главы 9–11*). Четвертая часть — использование библиотеки GDI+ (*главы 12–14*). Пятая часть — назначение графических библиотек (*глава 15*).

Глава 1 посвящена рассмотрению задач, при решении которых используется компьютерная графика. В ней также приводятся основные понятия и определения.

В *главе 2* большое внимание уделено основным средствам, которые будут использованы далее при программировании всех примеров книги. Среди них: особенности программирования "под Windows"; основные понятия и принципы объектно-ориентированного программирования; назначение библиотеки MFC; процесс создания приложения в MS Visual C++.

В *главе 3* рассматривается создание "графического" приложения Painter, в котором пользователь сможет нарисовать свой первый рисунок, сохранить его в файл и вывести на печать.

В *главе 4* более детально рассматриваются используемые программные средства, добавляется новая функциональность в программу Painter, проектируется иерархия классов графических объектов.

Глава 5 посвящена рассмотрению математических основ преобразований на плоскости.

В *главе 6* материал пятой главы находит свое практическое применение, а также рассматривается реализация в программе Painter некоторых функций редактирования создаваемых изображений.

Глава 7 посвящена теории и практике преобразований в трехмерном пространстве. В программе Painter реализуется построение трехмерной поверхности $z = f(x, y)$ и линий уровня на поверхности.

В *главе 8* рассматриваются математические основы и программная реализация построения сплайновых кривых.

Глава 9 посвящена использованию растровых ресурсов (пиктограмм, курсоров, растровых картинок) в приложениях.

В *главе 10* рассматривается растровый формат BMP, в программе Painter реализуется экспорт изображений в BMP-файл.

Глава 11 целиком посвящена работе с растровыми изображениями. В ней рассматривается создание и применение графических фильтров для обработки изображений. Разрабатывается программа, позволяющая загрузить, обработать с помощью цифровых фильтров и сохранить растровые изображения. Программа иллюстрирует разработку многодокументных, многопоточных приложений. Эту программу вы сможете использовать на практике для улучшения качества своего электронного фотоархива, а расширив ее возможности — добиться неповторимых эффектов.

В *главе 12* читателю сообщаются необходимые сведения о назначении и структуре библиотеки GDI+, новых возможностях для работы с графикой, создании программ на Visual C++ и на Visual C# с применением GDI+.

Глава 13 дает читателю достаточно полное представление о работе с растрами средствами GDI+, классе Bitmap и поддержке стандартных графических форматов, выводе растров на экран, создании анимации и специальных эффектов.

Глава 14 посвящена возможностям GDI+ при работе с векторной графикой. Рассматриваются темы: векторные примитивы и геометрия GDI+; графические объекты и их состояние; вывод различных векторных примитивов — программа Clock; регионы и траектории; использование метафайлов; системы координат.

В *главе 15* рассматривается назначение библиотек OpenGL и DirectX, приводится пример использования OpenGL для визуализации трехмерной сцены
и вывода на экран растрового изображения.

Моему брату

Предисловие к первому изданию

Эта книга представляет собой курс основ компьютерной графики. Здесь изложены алгоритмы и методы решения многих задач, возникающих при работе с векторными и растровыми изображениями. Основное внимание уделено вопросам прикладного программирования с использованием языка Microsoft Visual C++ и MFC. Материал книги накоплен за несколько лет чтения курса лекций "Компьютерная графика" в Томском государственном университете систем управления и радиоэлектроники (ТУСУР), а также в результате практической работы над научными и коммерческими проектами.

Основная цель книги — показать, как можно запрограммировать ту или иную задачу компьютерной графики: построить сплайновую кривую, нарисовать поверхность, отобразить на этой поверхности линии уровня, управлять несколькими объектами-фигурами в программе, сохранить изображение, считать растровое изображение с диска, обработать и записать обратно на диск, распечатать изображение на принтере.

Другая задача — продемонстрировать различные пути и способы достижения одинаковых результатов. Поэтому при изучении материала книги старайтесь находить новые, более удачные варианты решения рассмотренных задач.

Весь теоретический материал книги иллюстрируется рабочими примерами программ. В качестве языка программирования выбран C++, поскольку он представляется наиболее подходящим для решения серьезных задач компьютерной графики. В примерах применяется объектно-ориентированный стиль программирования (ООП). Используя стиль ООП, легко представить программу в виде отдельных частей (модулей, деталей), взаимодействующих между собой, если же вы склонны к философствованию, то можно мыслить о программе, как о сообществе неких индивидуумов, которые могут рождаться, обретать какие-то свойства, общаться между собой и умирать.

Каждая программа, описанная в книге, представляет собой законченное приложение. Законченное в том смысле, что его можно откомпилировать,

запустить, посмотреть, что оно делает, сохранить результаты работы. Для работы с примерами вам потребуется компьютер с операционной системой MS Windows 95 (или более поздней версией) и среда разработки MS Visual C++ 6.0.

Поскольку данная книга является учебником по компьютерной графике, а не справочным пособием по Windows API или GDI, то в ней, как правило, не приводится подробного описания параметров вызова API-функций или полного описания методов классов MFC. Эти сведения могут быть легко получены из электронной библиотеки Microsoft Developer Network (MSDN) Library или специальной литературы.

На кого рассчитана эта книга

Книга предназначена студентам и преподавателям, специализирующимся в области информатики и вычислительной техники.

Предполагается, что читатель уже имеет определенный опыт программирования. Однако изложение в книге ведется "от простого к сложному", основные этапы создания программ описаны достаточно подробно. Для понимания теоретического материала требуются начальные знания из области линейной алгебры и тригонометрии. Необходимым условием для успешного усвоения практической части книги является знание языка программирования C++ и знакомство с основными идеями объектно-ориентированного программирования.

Благодарности

Я благодарен преподавателям и студентам кафедры компьютерных систем управления и проектирования ТУСУР за участие в обсуждении многих рассмотренных в книге тем. Хочу выразить отдельную признательность моему научному руководителю, кандидату технических наук Леониду Ивановичу Бабаку. Спасибо студентам Елене Завадской и Сергею Цибенко за участие в деле построения поверхностей и линий уровня.

Я очень признателен руководству компании Элекард **http://www.elecard.com** за предоставленное в мое распоряжение необходимое оборудование, а своим коллегам по фирме Moonlight Russia — за ценное общение.

Выражаю глубокую благодарность за поддержку своим родителям Анне Афанасьевне и Юрию Антониновичу.

Особое спасибо жене Марине за терпение, заботу и вдохновение.

Алексей Поляков

Отзывы читателей на первое издание книги "Методы и алгоритмы компьютерной графики на Visual C++"

"Мне очень понравилась ваша книга, все так здорово и доступно описано, простым человеческим языком (если можно так выразиться :)"
Николай Кириченко

n-Y-c@nwgsm.ru
ООО "Русская коллекция", Оператор ФНА, Предпечатная подготовка
Санкт-Петербург, Россия

"С большим удовольствием прочитал Вашу книгу "Методы и алгоритмы компьютерной графики...". Книга на самом деле замечательная. ... У меня такое субъективное впечатление, что наконец пошла новая волна в отечественном (в какой-то степени отечество у нас общее) творчестве.
...Я думаю категориями конкретных задач. У меня не академический интерес. Я работаю для промышленности и... хочу видеть в книге живые ситуации и думать — ага, а в моем случае это будет так-то и так-то. Пишите дальше, у Вас здорово получается".

Дмитрий Удовицкий
info@techno-sys.com
ООО "Технос", директор.
Разработка систем автоматизированного проектирования и технологической подготовки производства.
г. Николаев, Украина

"Я просто хотел сказать спасибо за книжку "Комп. граф. в примерах на VC++". Очень помогло. Было просто не найти материала, MSDN слишком велика, чтобы там что-то отыскать :)"

Дмитрий Алексеевич Лубенский,
Department of Computer Science, программист, проект "Music Recognition"
Joensuu, Finland.

Предисловие ко второму изданию

Выход второго издания книги означает, что книги первого издания нашли своего читателя, а это радует. Со времени первого издания книги прошел всего лишь год, а базовые принципы мироустройства меняются не так быстро, поэтому материал, изложенный в книге, не утратил своей актуальности. Положительные отзывы читателей свидельствуют о том, что выбранный подход, заключающийся в опробировании всего теоритического материала на практике, оправдывает себя.

В то же время прогресс не стоит на месте. Это нашло отражение в том, что в книгу добавлены три главы, в которых рассмотрены перспективные новинки в технологии программирования: графическая библиотека **GDI+**, платформа **.NET Framework** и язык разработки **C#**. Поскольку книга посвящена компьютерной графике, то основное внимание уделено изучению библиотеки **GDI+**. Надо отметить, что **GDI+** — это действительно библиотека нового поколения с существенно расширенными возможностями, которые позволяют рядовым программистам достичь уровня графики и функциональности, сравнимого с продуктами класса CorelDRAW! и Adobe Photoshop. Вот лишь некоторые новшества и достоинства новой библиотеки:

- **Поддержка популярных форматов графических файлов:** необычайно приятное новшество для всех программистов, имеющих дело с разными графическими форматами. Поддерживаются форматы **BMP**, **GIF**, **TIFF**, **JPEG**, **Exi** (расширение **TIFF** и **JPEG** для цифровых фотокамер), **PNG**, **ICO**, **WMF** и **EMF**. Декодеры различных форматов выполнены с учетом их специфики, так что возможно, например, отобразить анимационный **GIF** или добавить комментарий к **TIFF**-файлу. Загруженный, созданный или модифицированный файл может быть сохранен на диск в одном из подходящих форматов.
- **Работа с растрами:** теперь можно практически все! Поддерживается отрисовка растров с наложением внешнего альфа-канала, масштабированием, растяжением, искажением и поворотом растров. При этом можно установить

режимы отображения отдельных пикселов — от простого переноса до префильтрации (наилучшее качество изображения). Стало возможным рисовать векторные примитивы, залитые текстурами.

- **Градиентная закраска:** позволяет заливать сложные фигуры оттенками с различными законами распределения цвета, рисовать векторные примитивы (например, линии) с градиентной окраской.
- **Поддержка прозрачности:** можно создавать кисти и растры с прозрачными и полупрозрачными областями, заливать области полупрозрачным цветом, назначать Color Key для растрового изображения и работать с его альфа-каналом, а также рисовать полупрозрачные (!) векторные примитивы и текст.

Причем благодаря объектно-ориентированному интерфейсу и новому дизайну графических функций/объектов использовать **GDI+** просто и удобно. На момент написания этих строк авторам не известно о наличии других книг на русском языке, в которых бы рассматривалась тема **GDI+**.

Материал книги первого издания также подвергся некоторым изменениям. Во-первых, благодаря откликам читателей удалось исправить ряд допущенных ошибок. Во-вторых, существенной переработке подверглась *глава 11* — материал дополнился расмотрением фильтров для удаления шума с растровых изображений, структура примера переработана так, чтобы обеспечить многопоточное выполнение программы. Использование дополнительных потоков для преобразований позволяет одновременно выполнять продолжительные операции в нескольких изображениях и при этом не терять контроль над выполнением программы.

Хочется еще раз подчеркнуть, что основная идея книги — дать читателю широкое представление о возможностях компьютерной графики и показать практичесие приемы решения ряда задач. При таком подходе, конечно, трудно претендовать на полноту изложения, в частности, такие интересные библиотеки как Open GL и Direct X в книге рассмотрены весьма кратко. Тем не менее мы уверены, что книга получилась интересной и полезной.

Обратная связь

Если у вас возникнут вопросы, замечания, пожелания, критика и предложения, пожалуйста, направляйте их в "книгу жалоб и предложений" по адресу **graphics@elecard.net.ru**. Мы будем рады, если вы поделитесь с нами своими впечатлениями. Ответы на часто задаваемые вопросы будут публиковаться по адресу **http://www.elecard.com/graphics**.

Благодарности

Прежде всего, хочу поблагодарить всех читателей, приславших свои отзывы на первое издание книги. Добрые слова и поддержка послужили стимулом для дальнейшего развития, а замечания — совершенствования. Особенно я хотел бы поблагодарить за отзывы и комментарии Дмитрия Удовицкого (г. Николаев, Украина), Николая Кириченко (Санкт-Петербург), Дмитрия Лубенского (Joensuu, Finland), Романа Худеева (компания Alparysoft, Томск).

Большое спасибо Алексею Соколову (компания Darim, Томск) за то, что он взял на себя труд прочитать книгу и сделал ценные замечания.

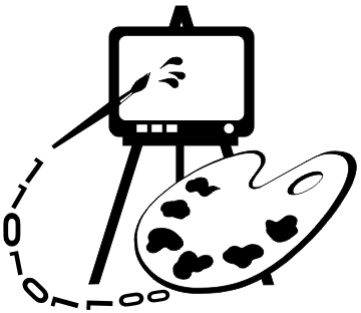
Хочу еще раз выразить свою признательность коллегам по фирме Moonlight Cordless Ltd. (Russian developer center), а также сотрудникам компании Elecard за творческую и дружескую атмосферу (чему в немалой степени способствует зеленый чай нашего доктора Ромы Позднякова), ну и, конечно, за обмен опытом и просвещение сослуживцев, в чем особо преуспели Андрей Поздняков, Петр Губанов и Александр Иванов.

В значительной степени второе издание книги состоялось благодаря Виталию Брусенцеву, который написал три замечательных главы про GDI+.

Хочется также поблагодарить создателей и авторов сайта **www.rsdn.ru** — Russian Software Developer Network за разработку и поддержку прекрасного ресурса, в котором можно найти статьи и форумы по широкому кругу вопросов программирования, в том числе посвященных темам компьютерной графики: GDI, GDI+, OpenGL, DirectX. Большое спасибо работникам издательства "БХВ-Петербург" и нашему редактору Дарье Масленниковой (отловившей массу ошибок) — именно их нелегкий труд превращает электронный текст в книжки с глянцевой обложкой.

Но Самое Большое Спасибо — моим жене и сыну, благодаря им я вообще что-то делаю :-).

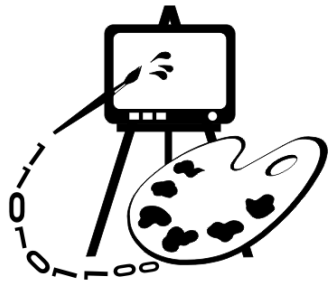
Алексей Поляков



Часть I

ОСНОВЫ КОМПЬЮТЕРНОЙ ГРАФИКИ И СРЕДСТВ ПРОГРАММИРОВАНИЯ

Глава 1



Основные понятия компьютерной графики

В данной главе рассматриваются:

- цели и задачи компьютерной графики;
- основные понятия и определения.

1.1. Цели и задачи компьютерной графики

Понятие "компьютерная графика" объединяет довольно широкий круг операций по обработке графической информации с помощью компьютера. Причем наблюдается явная тенденция "компьютеризации" изображений, циркулирующих в обществе. Стали обыденностью термины "цифровое фото" и "видео". Западные кинорежиссеры давно уже пытаются испугать нас ужасами будущего, захваченного компьютерными монстрами, подсовывающими людям виртуальный суррогат вместо прекрасной реальности. В виртуальных буднях грядущего компьютерной графике отводится огромная роль. Это связано с тем, что, по мнению ученых, исследующих проблемы мозга, зрительная система в иерархии мозговых структур человека занимает особое место. С восприятием и обработкой визуальной информации непосредственно связано примерно 20% мозга человека. Благодаря зрению мы получаем по разным оценкам от 70 до 90% сведений об окружающем мире. Следовательно, образный мир компьютерной графики является одним из глубинных проявлений человеческой природы.

В компьютерной графике можно выделить несколько основных направлений.

- Визуализация научных (расчетных или экспериментальных) данных. Большинство современных математических программных пакетов (например, Maple, MatLab, MathCAD) имеют средства для отображения графиков, поверхностей и трехмерных тел, построенных на основе каких-либо расчетов. Кроме того, графическая информация может активно ис-

пользоваться в самом процессе вычислений. Например, в системе Image, разработанной на кафедре компьютерных систем управления и проектирования Томского университета систем управления и радиоэлектроники, визуальные образы, выводимые на экран, являются основой для решения математических и проектных задач. Визуализация позволяет представить большой объем данных в удобной для анализа форме и широко используется при обработке результатов различных измерений и вычислений.

- Геометрическое проектирование и моделирование. Это направление компьютерной графики связано с решением задач начертательной геометрии — построением чертежей, эскизов, объемных изображений с помощью программных систем, получивших название CAD-системы (от английского Computer-Aided Design), например AutoCAD.

  Существует большое количество специализированных CAD-систем в машиностроении, архитектуре и т. д.

- Распознавание образов. Способность распознавать абстрактные образы считают одним из важнейших факторов, определившим развитие мыслительных способностей человека, выделив его из животного мира[1]. Задача распознавания и классификации графической информации является одной из ключевых и при создании искусственного интеллекта. Уже в наши дни компьютеры распознают образы повсеместно (системы идентификации футбольных хулиганов у входа на стадион; анализ аэро- и космических фотоснимков; системы сортировки, наведения и т. д.). Возможно, самый известный пример распознавания образов — сканирование и перевод "фотографии" текста в набор отдельных символов, формирующих слова. Такую операцию позволяет выполнить программное обеспечение многих современных сканеров. Кроме того, существуют специализированные программы распознавания текста, например FineReader.

- Изобразительное искусство. К этому направлению можно отнести разнообразную графическую рекламу: от текстовых транспарантов и фирменных знаков до компьютерных видеофильмов, обработку фотографий, создание рисунков, мультипликацию и т. д. В качестве примера популярных

[1] "Символические изображения, будь то животные, нарисованные на стенах пещер, или лунные календари, вырезанные на кости животных, ассоциируются исключительно с нашим видом. Возможно, в результате произошедшего небольшого щелчка в устройстве нашего мозга вкупе с некоторым "переключением проводов" наши предки обрели тот тип мышления, который был недоступен неандертальцам, — а именно способность к распознаванию абстрактных символов. И как только этот скрытый интеллектуальный потенциал начинал использоваться вследствие изобретения способов символьной коммуникации — произошедшего, скажем, 35 000 лет назад, его обладатели должны были быстро выйти на совершенно новый уровень технологий". — Майкл Л. Ротсчайльд. Биономика.

программ из этой области компьютерной графики можно назвать Adobe Photoshop (обработка растровых изображений), CorelDRAW (создание векторной графики), 3DS Max (трехмерное моделирование).

- Виртуальная реальность. Реальность, даже виртуальная, подразумевает воздействия на всю совокупность органов чувств человека, в первую очередь на его зрение. К компьютерной графике можно отнести задачи моделирования внешнего мира в различных приложениях: от компьютерных игр до тренажеров. Кроме того, не стоит забывать о компьютерах-злодеях, которые используют виртуальную реальность для захвата мира. Поэтому надо изучать компьютерную графику, чтобы не дать себя провести :-).
- Цифровое видео. Все более широкое распространение получают анимированные изображения, записанные в цифровом формате. Это прежде всего фильмы, передаваемые через компьютерные сети, а также видеодиски Digital Video Disk (DVD), цифровое, кабельное и спутниковое телевидение.

Приведенная классификация сфер применения компьютерной графики является во многом условной. Возможно, найдутся задачи, которые нельзя отнести ни к одному из обозначенных направлений.

1.2. Основные понятия и определения

1.2.1. Графический формат

Графическим форматом называют порядок (структуру), согласно которому данные, описывающие изображение, записаны в файле.

Графические данные обычно разделяются на два класса: *векторные* и *растровые*. Изображения, в зависимости от типа описывающих их данных, называются векторными или растровыми.

Векторные данные используются для представления прямых, многоугольников, кривых и т. д. с помощью определенных в числовом виде *базовых* (опорных, контрольных, ключевых) точек. Программа, обрабатывающая векторные данные, воспроизводит линии посредством соединения базовых точек. Вместе с информацией о базовых точках хранятся атрибуты (цвет, толщина и другие параметры линий) и набор правил (соглашений) вывода (рисования). Пример векторного изображения приведен на рис. 1.1.

Растровые данные представляют собой набор числовых значений, определяющих яркость и цвет отдельных *пикселов*. Пикселами (или пикселями — от английского *pixel*) называются минимальные элементы (цветные точки), из которых формируется растровое изображение. Термин "растр" в компьютерной графике и полиграфии имеет несколько отличающиеся значения.

Далее под растром будем понимать массив пикселов (массив числовых значений). Для обозначения массива пикселов часто используется термин *bitmap* (битовая карта). В *bitmap* каждому пикселу отводится определенное число битов (одинаковое для всех пикселов изображения). Это число называется *битовой глубиной* пиксела или *цветовой глубиной* изображения, т. к. от количества битов, отводимых на один пиксел, зависит количество цветов изображения. Наиболее часто используется цветовая глубина 1, 2, 4, 8, 15, 16, 24 и 32 бита.

Рис. 1.1. Векторный рисунок

Источниками растровых данных могут быть программы, формирующие изображение на растровом экране и различного рода устройства для ввода изображений (сканеры, цифровые камеры и др.).

Пример растрового изображения приведен на рис. 1.2.

Рис. 1.2. Растровый рисунок

Типы форматов графических файлов определяются способом хранения и типом графических данных. Наиболее широко используются растровый, векторный и метафайловый форматы.

Векторный формат наиболее удобен для хранения изображений, которые можно разложить на простые геометрические фигуры (например, чертежи или текст). Векторные файлы содержат математические описания элементов изображения. Наиболее распространенные векторные форматы: AutoCAD DXF и Microsoft SYLK.

Растровый формат используется для хранения растровых данных. Файлы такого типа особенно хорошо подходят для хранения изображений реального мира, например оцифрованных фотографий. Растровые файлы содержат битовую карту изображения и ее спецификацию. Наиболее распространенные растровые форматы: BMP, TIFF, GIF, PCX, JPEG.

Метафайловый формат позволяет хранить в одном файле и векторные, и растровые данные. Примером такого формата являются файлы CorelDRAW — CDR.

Кроме того, существуют файловые форматы для хранения мультипликации (видеоинформации), мультимедиа-форматы (одновременно хранят звуковую, видео- и графическую информацию), гипертекстовые (позволяют хранить не только текст, но и связи-переходы внутри него) и гипермедиа (гипертекст плюс графическая и видеоинформация) форматы, форматы трехмерных сцен, форматы шрифтов и т. д.

1.2.2. Элементы графического файла

Графические файлы состоят из последовательности данных или структур данных, называемых *файловыми элементами* или *элементами данных*. Эти элементы подразделяются на три категории: поля, теги и потоки.

- *Поле* — это структура данных в графическом файле, имеющая фиксированный размер. Для определения положения поля в файле обычно задают либо абсолютное смещение от начала или конца файла, либо смещение относительно другого поля.
- *Тег* — это структура данных, размер и позиция которой изменяются от файла к файлу. Позиция тега может задаваться абсолютно, либо относительно другого файлового элемента. Теги могут содержать в себе другие теги или наборы связанных полей.
- *Поток* — набор данных, предназначенный для последовательного чтения. В отличие от полей и тегов поток не обеспечивает быстрого доступа к нужным данным, т. к. их положение в файле определяется в процессе чтения.

Как правило, в графических файлах применяются комбинации этих элементов данных.

1.2.3. Преобразование форматов

Часто возникает задача преобразования формата данных, связанная с необходимостью обмена изображениями между различными программами. Для преобразования данных существуют специализированные программы. Кроме того, распространенные графические редакторы позволяют читать и сохранять изображения в различных форматах. Поэтому преобразование растровых данных из одного формата в другой обычно не представляет сложности. Другое дело — преобразование векторных изображений в растровые. При таком преобразовании неизбежно теряется часть информации, т. к. происходит переход от "идеального" математического описания рисунка к его дискретному (растровому) представлению. Обратное преобразование из растрового формата в векторный вообще является нетривиальной задачей, связанной с распознаванием образов.

1.2.4. Сжатие данных

Сжатие — процесс уменьшения физического размера блока данных. Так как изображения, как правило, описываются большим количеством данных, файлы изображений имеют большой размер. Поэтому графические данные часто подвергаются сжатию. Обычно каждый формат графического файла поддерживает какой-либо из методов (алгоритмов) сжатия. В большинстве случаев сжатие заключается в замене избыточной информации на ее более компактную форму. Сжатие бывает физическое и логическое. Различие между физическим и логическим сжатием заключается в методе получения более компактной формы данных. Физическое сжатие данных выполняется без учета содержащейся в них информации. Логическое сжатие, напротив, основано на логическом анализе информации. Примером логического сжатия может служить преобразование строки "Союз Советских Социалистических Республик" в аббревиатуру "СССР". Для графических данных логическое сжатие не применяется.

Методы сжатия бывают *с потерями* и *без потерь*. Когда данные сжимаются, а затем восстанавливаются (распаковываются), причем полученные данные полностью соответствуют исходной информации, то говорят, что имело место сжатие без потерь. То есть при методе сжатия без потерь не должно происходить какого-либо изменения данных.

Методы сжатия с потерями предусматривают отбрасывание некоторой части данных изображения для достижения большей степени сжатия.

Некоторые наиболее распространенные методы сжатия.

- Упаковка пикселов. Метод заключается в компактной записи пикселов с глубиной 1, 2 и 4 бита компактно в 8-битовые байты соответственно по 8, 4 и 2 штуки.

- Групповое кодирование (Run-Length Encoding, RLE) — является общим алгоритмом кодирования и применяется в таких растровых форматах, как BMP, TIFF, PCX.
- Алгоритм Lempel-Ziv-Welch (LZW) — применяется в форматах GIF, TIFF.
- Алгоритм JPEG, разработанный объединенной экспертной группой по фотографии, включает в себя целый набор методов сжатия. Базовая реализация JPEG применяет схему преобразования изображения по алгоритму дискретных косинус-преобразований с последующим кодированием методом Хаффмана.
- Фрактальное сжатие — математический процесс, используемый для кодирования растровых изображений в совокупность математических данных, которые описывают фрактальные (похожие, повторяющиеся) свойства изображения.

Сжатие в основном применяется к данным растровых изображений. В растровых файлах сжимаются только данные изображения, другие же данные (заголовок файла, таблица цветов и т. п.) всегда остаются несжатыми.

Векторные файлы сжимаются редко. Это связано с тем, что векторные форматы сами по себе хранят данные в очень компактной форме и сжатие не дает ощутимого эффекта.

Важно уяснить, что алгоритмы сжатия не задают какой-либо файловый формат, а определяют только способ кодирования данных.

1.2.5. Пикселы и цвет

Различают физические и логические пикселы.

Физические пикселы — реальные точки, отображаемые на устройстве вывода — наименьшие элементы на поверхности отображения, которыми можно манипулировать. При выводе на экран или принтер один физический пиксел обычно формируется из нескольких более мелких цветовых точек. Например, один цветной пиксел на мониторе формируется из трех более мелких точек красного, зеленого и синего цветов, яркость которых и определяет цвет пиксела.

Логические пикселы подобны чертям на кончике иглы[1] — они имеют местоположение и цвет, но не занимают физического пространства (то есть нельзя вычислить высоту и ширину такого пиксела). По сути логический пиксел — это всего лишь некоторое число, которое задает его цвет. Положение же логического пиксела (его координаты) определяется его местом в карте изображения и количеством пикселов на единицу измерения (разрешением).

[1] "Сколько чертей уместится на кончике иглы" — известный схоластический спор.

При отображении логических пикселов на физическом экране происходит преобразование численных данных, характеризующих яркость и цвет логических пикселов, в интенсивность свечения физических пикселов. При этом никак не обойтись без учета размера и расположения физических пикселов.

Количество возможных цветов пиксела напрямую связано с отводимым для него количеством битов и равно 2^n, где n — количество битов. Изображения, каждому пикселу которого отводится один бит, называют монохромными — двухцветными. Такие изображения вполне подходят для чертежей и текста. Изображения с глубиной цвета 24 бита и более называют *truecolor* (истинные цвета). Каждый пиксел такого изображения может принимать один из более 16 миллионов цветов. Считается, что этого вполне достаточно, чтобы приемлемо отобразить окружающую нас действительность.

При отображении цветов, заданных для логических пикселов на устройстве визуализации, может возникнуть проблема согласования цветов. Например, если устройство вывода способно отобразить до 16 миллионов цветов, а изображение имеет глубину цвета 8 бит (256 цветов), то проблем с его отображением не будет. Если же все наоборот, то программе визуализации придется потрудиться, выполняя преобразование цветов изображения соответственно возможностям устройства вывода. В последнем случае неизбежна потеря части данных и, как следствие, снижение качества изображения. Кроме того, возможно возникновение всякого рода побочных эффектов (муар, вторичные контуры — артефакты, одним словом).

1.2.6. Палитры цветов

Палитра цветов, называемая также *картой* или *таблицей цветов*, представляет собой одномерный массив цветовых величин. С помощью палитры цвета задаются косвенно, посредством указания их позиции в массиве. При использовании палитры цветов сведения о цветах пикселов записаны в файле в виде последовательности индексов. Использование палитр во многих случаях позволяет значительно сократить объем растровых данных.

Наиболее часто используются палитры из 16 и 256 цветов, соответственно глубина цвета составляет 4 и 8 битов, но могут быть и палитры других размеров.

Например, каждый пиксел изображения с глубиной цвета в 4 бита может иметь 16 цветов (2^4). Эти 16 цветов определены в палитре, которая обычно включается в один файл с растровыми данными. Каждое пикселное значение рассматривается как индекс в этой палитре и содержит одно из значений от 0 до 15. Значения цветов в палитре задаются с максимально возможной точностью. Обычно элемент палитры занимает 24 бита (3 байта). Таким образом, элемент палитры может задавать один из 16 777 216 цветов (2^{24}). Программа, осуществляющая визуализацию изображения, читает из файла растровые данные — индексы в таблице цветов и использует соответствующие

им цвета для окрашивания пикселов экрана (рис. 1.3). Палитры разных изображений чаще всего различаются.

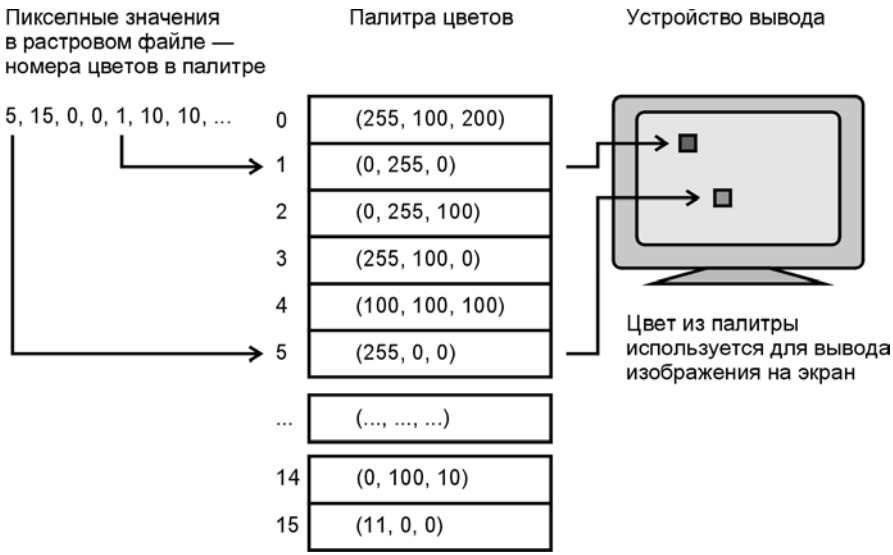


Рис. 1.3. Определение цвета с помощью палитры

1.2.7. Цвет. Цветовые модели

Для описания цветов применяют несколько различных математических систем (моделей). Ни одна из существующих систем представления цвета не является наилучшей. Для разных целей и задач служат различные системы.

В графических файлах обычно используют цветовые модели, основанные на трех основных цветах, которые не могут быть получены смешиванием других цветов. Все множество цветов, которые могут быть получены путем смешивания основных цветов, представляет собой *цветовое пространство*, или *цветовую гамму*.

Цветовые модели могут быть разделены на две категории: *аддитивные* и *субтрактивные*. В аддитивных моделях новые цвета получают путем сложения основных цветов различной интенсивности с черным цветом. Чем больше интенсивность добавляемых цветов, тем ближе к белому результирующий цвет. Белый цвет получается при максимальных значениях интенсивности всех трех основных цветов, черный — при минимальных. Аддитивные модели формирования цвета применяются в самосветящихся устройствах (например, мониторах).

В субтрактивных моделях основные цвета вычитаются из белого. Чем больше интенсивность вычитаемых цветов, тем ближе результат к черному.

Субтрактивные модели применяются при формировании цветных изображений на отражающих носителях, например бумаге.

Наиболее распространенные цветовые модели.

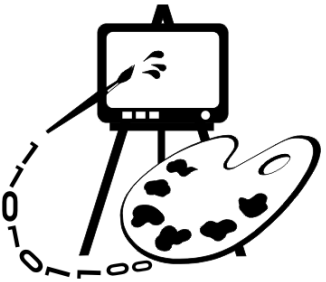
- **Модель RGB** (Red-Green-Blue — красный-зеленый-синий) — модель, наиболее широко используемая в графических форматах. RGB — аддитивная модель. Каждый пиксел представляется в виде трех числовых величин — интенсивностей красного, зеленого и синего цветов. Каждому цвету обычно отводится 8 битов, в которых может быть записано 256 уровней интенсивности. Таким образом, значение (0, 0, 0) представляет черный цвет, а (255, 255, 255) — белый.
- **Модель CMY** (Cyan-Magenta-Yellow — голубой-пурпурный-желтый) — субтрактивная модель, которая применяется для получения цветных изображений на белой поверхности. При освещении изображения, полученного с помощью модели CMY, каждый из основных цветов поглощает дополняющий его цвет: голубой поглощает красный; пурпурный — зеленый; желтый — синий. Теоретически наивысшая интенсивность всех трех основных цветов субтрактивной модели должна обеспечить черный цвет. На практике этого не происходит (так как реальные красители далеки от математических идеалов), поэтому в модель вводят четвертый компонент — черный цвет (Black), обозначаемый буквой "K". В результате получается модель CMYK, широко распространенная в полиграфии. При использовании субтрактивной модели изображение каждого пиксела (цветной точки) изображения состоит из четырех пятен основных цветов.

Существуют и другие модели, не основанные на смешении цветов, например HSV (Hue-Saturation-Value — оттенок-насыщенность-величина). Оттенок — это, по сути, цветовой тон, например, красный, оранжевый, синий и т. д. Насыщенность определяет количество белого в оттенке. Если в полностью насыщенном (100%) оттенке красного не содержится белого, такой оттенок считается чистым. Насыщенность 50% задает более светлый цвет, в нашем примере он будет соответствовать розовому цвету. Величина (яркость) задает интенсивность свечения цвета.

1.3. Заключение

Итак, в этой главе мы коротко рассмотрели основные задачи, при решении которых используются средства компьютерной графики, и дали определения используемым терминам. В следующих главах на практике рассмотрим работу с векторной и растровой графикой. В качестве дополнительной литературы можно порекомендовать [6, 18].

Глава 2



Особенности программирования "под Windows"

В данной главе рассматриваются:

- особенности программирования "под Windows";
- основные понятия и принципы объектно-ориентированного программирования;
- создание приложения в MS Visual C++.

Сегодня, когда операционная система Win32 (имеются в виду все 32-разрядные разновидности Windows) получила широчайшее распространение, и о работе под DOS и Win16 уже стали забывать, программирование под Windows стало обыденностью. Однако не лишним будет подчеркнуть те концепции Windows, благодаря которым она стала столь популярной и которые, наверняка, получат развитие в Win64. Win32 предоставляет программисту такие возможности, о которых лет 10 назад можно было только мечтать. Пожалуй, основными отличиями операционных систем Win32 от их предшественниц является вытесняющая многозадачность и возможность прямой адресации до 4 Гбайт.

Вытесняющая многозадачность означает, что операционная система сама решает, какой из программ предоставить в распоряжение процессор. Каждая программа, отработав некоторое время, автоматически выгружается системой, и управление передается другой задаче. Возможно, кому-то не понравится такой "авторитарный" стиль управления. Зато подобная организация позволяет, во-первых, создать иллюзию одновременного выполнения нескольких программ, и, во-вторых, не допускает возможности полного захвата ресурсов компьютера какой-нибудь сбойной задачей, защищая таким образом систему от "зависания" (но мы-то знаем, что на практике это не всегда так). На мой взгляд, о вытесняющей многозадачности лучше всех сказал Омар Хайям:

Напрасно ты винишь в непостоянстве рок,
Что не в накладе ты, тебе и невдомек.
Когда б он в милостях своих был постоянен,
Ты б очереди ждать своей до смерти мог.