МИНИСТЕРСТВО ВЫСШЕГО И СРЕДНЕГО СПЕЦИАЛЬНОГО ОБРАЗОВАНИЯ
РЕСПУБЛИКИ УЗБЕКИСТАН

НАЦИОНАЛЬНЫЙ УНИВЕРСИТЕТ УЗБЕКИСТАНА
имени Мирзо Улугбека

ОБЩИЙ КУРС КОМПЬЮТЕРНОЙ ГРАФИКИ

Для студентов механико-математического факультета

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС

Область знаний: 500000 – «Инженерные, обрабатывающие и строительные отрасли»

Направление: 5521900 –Информатика и информационные технологии

Общее количество часов	99,2
Лекции	24
Практические занятия	24
Самостоятельная работа	40
Консультации	1,2
Рейтинг	10

Ташкент-2011

Рабочий план и рейтинговые показатели обсуждены и утверждены на заседании кафедры «Информатики и прикладного программирования» протоколом № 1. 28 августа 2011 года.

Разработаны на основании учебного плана и типовой программы специальности «Информатика и информационные технологии».

Составители:
доц. Варламова Л.П. _____

Зав. кафедрой: д. ф-м.н., проф. Б.Ф.Абдурахимов _____

Рабочий план и рейтинговые показатели предмета обсуждены и утверждены на заседании факультетского совета Механико-математического факультета протоколом № 1. 28 августа 2011 года.

Председатель Совета факультета:
проф. **Б. А. Шоимкулов** _____

Предисловие

Целью курса «Компьютерная графика» является изучение теоретических основ компьютерной графики, связанных с вводом, хранением и обработкой цифровых изображений в памяти компьютеров, базовых методов и алгоритмов, использующихся в большинстве современных систем обработки растровых и векторных изображений, ознакомление с системами компьютерной графики различного назначения. Рассматриваются теоретические и прикладные вопросы применения современных систем компьютерной графики.

СОДЕРЖАНИЕ

Требования к уровню освоения содержания дисциплины

В результате изучения дисциплины студенты должны:

1. освоить основные понятия, связанные с вводом, представлением, хранением и обработкой изображений в памяти компьютера. Изучить математические основы компьютерной графики, основные методы и алгоритмы растровой и векторной графики, знать наиболее распространенные графические форматы и цветовые модели, познакомиться с программным обеспечением систем КГ различного назначения.

2. уметь применять на практике алгоритмы компьютерной графики.

3. познакомиться и освоить работу с системами растровой и векторной графики (на примере одного из стандартных пакетов для каждого вида графики).

4. Знать принцип работы устройств ввода и отображения визуальной информации.

Рабочая программа

Введение в компьютерную графику, общие понятия.

Цель, задачи и структура курса. Предмет компьютерной графики. Роль компьютерной графики, сферы применения, назначение. Основные понятия: ввод, отображение и хранение изображений, понятие о разрешении, дискретизация и квантовании. Современные аппаратные средства растровой графики. Типы изображений. Представление изображений в памяти компьютера. Организация хранения изображений. Графические форматы. Алгоритмы сжатия. Описание цвета. Цветовые модели: аддитивные, субтрактивные, перцепционные.

Методы обработки изображений

Основные этапы цифровой обработки и анализа изображений.

Математическое описание растровых изображений.

Методы обработки растровых изображений. Понятие поэлементных, локальных и глобальных операторов обработки изображений.

Гистограмма изображения. Простейшие геометрические преобразования. Улучшение изображений. Изменение контраста. Сглаживание шумов. Видоизменение гистограмм. Подчеркивание границ. Выделение контуров (границ). Методы улучшения изображений.

Методы классификации изображений. Понятие об управляемой и неуправляемой классификации. Понятие об обучающих выборках или эталонах на изображении. Обнаружение объектов на изображении.

Векторная графика. Элементы векторной графики. Линии. Кривые Безье. Узлы (Опорные точки). Примитивы. Математические основы векторной графики.

Системы компьютерной графики

Обзор систем компьютерной графики.

Photoshop – пакет обработки растровых изображений. Назначение и основные возможности пакета Photoshop. Окно программы. Основные пункты меню. Инструменты и палитры. Средства редактирования изображений. Средства выделения Геометрические преобразования. Каналы. Наложение изображений с помощью средства Слои. Средства фильтрации изображений. Цветовые модели и регулировка цвета. Графические форматы, используемые в пакете Photoshop.

Corel Draw – пакет обработки векторной графики. Назначение и основные возможности. Основное меню. Инструменты пакета. Создание и преобразование объектов. Эффекты. Форматы векторной графики.

Распределение по часам

	Тема	Лек.	Прак
1неделя	Введение в предмет компьютерной графики. Роль компьютерной графики, сферы применения, назначение. Основные понятия: ввод и хранение изображений, понятие о разрешении. Современные аппаратные средства растровой графики. Типы изображений. Знакомство с функциями пакета обработки растровых изображений PhotoShop	2	2
2неделя	Организация хранения изображений. Графические форматы. Алгоритмы сжатия. Поддержка графических форматов пакетом PhotoShop.	4	2
	Описание цвета. Цветовые модели: аддитивные, субтрактивные,		

3 неделя	перцепционные. Алгоритмы перехода из одной модели в другую. Поддержка цветовых моделей пакетом PhotoShop.	4	2
4неделя 5 неделя 6 неделя	Хранение изображений в памяти компьютера. Графические форматы для растровых изображений. Методы обработки растровых изображений Основные этапы цифровой обработки и анализа изображений. Математическое описание растровых изображений. Методы обработки растровых изображений. Понятие поэлементных, локальных и глобальных операторов обработки изображений. Гистограмма изображения. Видоизменение гистограмм. Простейшие геометрические преобразования. Методы улучшения изображений. Изменение контраста. Сглаживание шумов. Подчеркивание границ. Выделение контуров (границ). Окно программы PhotoShop. Основные пункты меню. Инструменты и палитры. Средства редактирования изображений. Средства выделения Геометрические преобразования. Каналы. Наложение изображений с помощью средства Слои. Средства фильтрации изображений. Цветовые модели и регулировка цвета. Графические форматы, используемые в пакете Photoshop.	2 2 4	2 2 2 4
7неделя	Corel Draw – пакет обработки векторной графики. Назначение и основные возможности. Основное меню. Инструменты пакета.	2	2 2
8 неделя	Создание и преобразование объектов. Эффекты.	2	2
9 неделя	Форматы векторной графики.	2	2
	Всего	24	24

Таблица контролей за один семестр

ПК №1	ПК №2	ИК	ТК			Всего
			ТК1	ТК2	ТК3	
14	14	30	14	14	14	100

Лекция 1: Общее введение в компьютерную графику
Предмет и область применения компьютерной графики

Компьютерная графика - это область информатики, которая охватывает все стороны формирования изображений с помощью компьютера. Появившись в 1950-х годах, она поначалу давала возможность выводить лишь несколько десятков отрезков на экране. В наши дни средства компьютерной графики позволяют создавать реалистические изображения, не уступающие фотографическим снимкам. Создано разнообразное аппаратное и программное обеспечение для получения изображений самого различного вида и назначения - от простых чертежей до реалистических образов естественных объектов. Компьютерная графика используется практически во всех научных и инженерных дисциплинах для наглядности восприятия и передачи информации. Применение ее для подготовки демонстрационных слайдов уже считается нормой. Трехмерные изображения используются в медицине (компьютерная томография), картографии, полиграфии, геофизике, ядерной физике и других областях. Телевидение и другие отрасли индустрии развлечений используют анимационные средства компьютерной графики (компьютерные игры, фильмы). Общепринятой практикой считается также использование компьютерного моделирования при обучении пилотов и представителей других профессий (тренажеры). Знание основ компьютерной графики сейчас необходимо и инженеру, и ученому.

Конечным результатом применения средств компьютерной графики является изображение, которое может использоваться для различных целей. Поскольку наибольшее количество информации человек получает с помощью зрения, уже в древние времена появились схемы и карты, используемые при строительстве, в географии и в астрономии.

Современная компьютерная графика - это достаточно сложная, основательно проработанная и разнообразная научно-техническая дисциплина. Некоторые ее разделы, такие как геометрические преобразования, способы описания кривых и поверхностей, к настоящему времени уже исследованы достаточно полно. Ряд областей продолжает активно развиваться: методы растрового сканирования, удаление невидимых линий и поверхностей, моделирование цвета и освещенности, текстурирование, создание эффекта прозрачности и полупрозрачности и др.

Сфера применения компьютерной графики включает четыре основных области.

1. Отображение информации

Проблема представления накопленной информации (например, данных о климатических изменениях за продолжительный период, о динамике популяций животного мира, об экологическом состоянии различных регионов и т.п.) лучше всего может быть решена посредством графического отображения.

Ни одна из областей современной науки не обходится без графического представления информации. Помимо визуализации результатов экспериментов и анализа данных натурных наблюдений существует обширная область математического моделирования процессов и явлений, которая просто немыслима без графического вывода. Например, описать процессы, протекающие в атмосфере или океане, без соответствующих наглядных картин течений или полей температуры практически невозможно. В геологии в результате обработки трехмерных натурных данных можно получить геометрию пластов, залегающих на большой глубине.

В медицине в настоящее время широко используются методы диагностики, использующие компьютерную визуализацию внутренних органов человека. Томография (в частности, ультразвуковое исследование) позволяет получить трехмерную информацию, которая затем подвергается математической обработке и выводится на экран. Помимо этого применяется и двумерная графика: энцефалограммы, миограммы, выводимые на экран компьютера или графопостроитель.

2. Проектирование

В строительстве и технике чертежи давно представляют собой основу проектирования новых сооружений или изделий. Процесс проектирования с необходимостью является итеративным, т.е. конструктор перебирает множество вариантов с целью выбора оптимального по каким-либо параметрам. Не последнюю роль в этом играют требования заказчика, который не всегда четко представляет себе конечную цель и технические возможности. Построение предварительных макетов - достаточно долгое и дорогое дело. Сегодня существуют развитые программные средства автоматизации проектно-конструкторских работ (САПР), позволяющие быстро создавать чертежи объектов, выполнять прочностные расчеты и т.п. Они дают возможность не только изобразить проекции изделия, но и рассмотреть его в объемном виде с различных сторон. Такие средства также чрезвычайно полезны для дизайнеров интерьера, ландшафта.

3. Моделирование

Под моделированием в данном случае понимается имитация различного рода ситуаций, возникающих, например, при полете самолета или космического аппарата, движении автомобиля и т.п. В английском языке это лучше всего передается термином simulation. Но моделирование используется не только при создании различного рода тренажеров. В телевизионной рекламе, в научно-популярных и других фильмах теперь синтезируются движущиеся объекты, визуально мало уступающие тем, которые могут быть получены с помощью кинокамеры. Кроме того, компьютерная графика предоставила киноиндустрии возможности создания спецэффектов, которые в прежние годы были попросту невозможны. В последние годы широко распространилась еще одна сфера применения компьютерной графики - создание виртуальной реальности.

4. Графический пользовательский интерфейс

На раннем этапе использования дисплеев как одного из устройств компьютерного вывода информации диалог "человек-компьютер" в основном осуществлялся в алфавитно-цифровом виде. Теперь же практически все системы программирования применяют графический интерфейс. Особенно впечатляюще выглядят разработки в области сети Internet. Существует множество различных программ-браузеров, реализующих в том или ином виде средства общения в сети, без которых доступ к ней трудно себе представить. Эти программы работают в различных операционных средах, но реализуют, по существу, одни и те же функции, включающие окна, баннеры, анимацию и т.д.

В современной компьютерной графике можно выделить следующие основные направления: изобразительная компьютерная графика, обработка и анализ изображений, анализ сцен (перцептивная компьютерная графика), компьютерная графика для научных абстракций (когнитивная компьютерная графика, т.е. графика, способствующая познанию).

Изобразительная компьютерная графика своим предметом имеет синтезированные изображения. Основные виды задач, которые она решает, сводятся к следующим:

построение модели объекта и формирование изображения;

преобразование модели и изображения;

идентификация объекта и получение требуемой информации.

Обработка и анализ изображений касаются в основном дискретного (цифрового) представления фотографий и других изображений. Средства компьютерной графики здесь используются для:

повышения качества изображения;

оценки изображения - определения формы, местоположения, размеров и других параметров требуемых объектов;

распознавания образов - выделения и классификации свойств объектов (при обработке аэрокосмических снимков, вводе чертежей, в системах навигации, обнаружения и наведения).

Анализ сцен связан с исследованием абстрактных моделей графических объектов и взаимосвязей между ними. Объекты могут быть как синтезированными, так и выделенными на фотоснимках. К таким задачам относятся, например, моделирование "машинного зрения" (роботы), анализ рентгеновских снимков с выделением и отслеживанием интересующего объекта (внутреннего органа), разработка систем видеонаблюдения.

Когнитивная компьютерная графика - только формирующееся новое направление, пока еще недостаточно четко очерченное. Это - компьютерная графика для научных абстракций, способствующая рождению нового научного знания. Технической основой для нее являются мощные ЭВМ и высокопроизводительные средства визуализации.

Одним из наиболее ранних примеров использования когнитивной компьютерной графики является работа Ч.Страуса "Неожиданное применение ЭВМ в чистой математике" (ТИИЭР, т. 62, № 4, 1974, с.96-99). В ней показано, как для анализа сложных алгебраических кривых используется "n-мерная" доска на основе графического терминала. Пользуясь устройствами ввода, математик может легко получать геометрические изображения результатов направленного изменения параметров исследуемой зависимости. Он может также легко управлять текущими значениями параметров, "углубляя тем самым свое понимание роли вариаций этих параметров". В результате получено "несколько новых теорем и определены направления дальнейших исследований".

В настоящем курсе предполагается рассмотреть следующие вопросы:

представление изображения в компьютерной графике;

способы подготовки изображения к визуализации;

методы вывода изображения на экран;

методы работы с изображением;

методы вычислительной геометрии.

Краткая история

В этом кратком историческом очерке мы будем упоминать алгоритмы и методы, с которыми читатель сможет познакомиться в данном курсе. Эти предварительные упоминания не должны смущать при первом прочтении. По завершении курса можно вернуться к этому разделу и пройти его заново.

Компьютерная графика в начальный период своего возникновения была далеко не столь эффектной, какой она стала в настоящие дни. В те годы компьютеры находились на ранней стадии развития и были способны воспроизводить только самые простые контуры (линии). Идея компьютерной графики не сразу была подхвачена, но ее возможности быстро росли, и постепенно она стала занимать одну из важнейших позиций в информационных технологиях.

Первой официально признанной попыткой использования дисплея для вывода изображения из ЭВМ явилось создание в Массачусетском технологическом университете машины Whirlwind-I в 1950 г. Таким образом, возникновение компьютерной графики можно отнести к 1950-м годам. Сам же термин "компьютерная графика" придумал в 1960 г. сотрудник компании Boeing У. Феттер.

Первое реальное применение компьютерной графики связывают с именем Дж. Уитни. Он занимался кинопроизводством в 50-60-х годах и впервые использовал компьютер для создания титров к кинофильму.

Следующим шагом в своем развитии компьютерная графика обязана Айвэну Сазерленду, который в 1961 г., еще будучи студентом, создал программу рисования, названную им Sketchpad (альбом для рисования). Программа использовала световое перо для рисования простейших фигур на экране. Полученные картинки можно было сохранять и восстанавливать. В этой программе был расширен круг основных графических примитивов, в частности, помимо линий и точек был введен прямоугольник, который задавался своими размерами и расположением.

Первоначально компьютерная графика была векторной, т.е. изображение формировалось из тонких линий. Эта особенность была связана с технической реализацией компьютерных дисплеев. В дальнейшем более широкое применение получила растровая графика, основанная на представлении изображения на экране в виде матрицы однородных элементов (пикселей).

В том же 1961 г. студент Стив Рассел создал первую компьютерную видеоигру Spacewar ("Звездная война"), а научный сотрудник Bell Labs Эдвард Зэджек создал анимацию "Simulation of a two-giro gravity control system".

В связи с успехами в области компьютерной графики крупные корпорации начали проявлять к ней интерес, что в свою очередь стимулировало прогресс в области ее технической поддержки.

Университет штата Юта становится центром исследований в области компьютерной графики благодаря Д.Эвансу и А.Сазерленду, которые в это время были самыми заметными фигурами в этой области. Позднее их круг стал быстро расширяться. Учеником Сазерленда стал Э.Кэтмул, будущий создатель алгоритма удаления невидимых поверхностей с использованием Z-буфера (1978). Здесь же работали Дж.Варнок, автор алгоритма удаления невидимых граней на основе разбиения области (1969) и основатель Adobe System (1982), Дж.Кларк, будущий основатель компании Silicon Graphics (1982). Все эти исследователи очень сильно продвинули алгоритмическую сторону компьютерной графики.

В том же 1971 г. Гольдштейн и Нагель впервые реализовали метод трассировки лучей с использованием логических операций для формирования трехмерных изображений.

В 1970-е годы произошел резкий скачок в развитии вычислительной техники благодаря изобретению микропроцессора, в результате чего началась миниатюризация компьютеров и быстрый рост их производительности. И в это же время начинает интенсивно развиваться индустрия компьютерных игр. Одновременно компьютерная графика начинает широко использоваться на телевидении и в киноиндустрии. Дж.Лукас создает отделение компьютерной графики на Lucasfilm.

В 1977 г. появляется новый журнал "Computer Graphics World".

В середине 1970-х годов графика продолжает развиваться в сторону все большей реалистичности изображений. Э.Кэтмул в 1974 г. создает первые алгоритмы текстурирования криволинейных поверхностей. В 1975 г. появляется упомянутый ранее метод закрашивания Фонга. В 1977 г. Дж.Блин предлагает алгоритмы реалистического изображения шероховатых поверхностей (микрорельефов); Ф.Кроу разрабатывает методы устранения ступенчатого эффекта при изображении

контуров (антиэлайзинг). Дж.Брезенхем создает эффективные алгоритмы построения растровых образов отрезков, окружностей и эллипсов. Уровень развития вычислительной техники к этому времени уже позволил использовать "жадные" алгоритмы, требующие больших объемов памяти, и в 1978 г. Кэтмул предлагает метод Z-буфера, в котором используется область памяти для хранения информации о "глубине" каждого пикселя экранного изображения. В этом же году Сайрус и Бэк развивают алгоритмы клиппирования (отсечения) линий. А в 1979 г. Кэй и Гринберг впервые реализуют изображение полупрозрачной поверхности.

В 1980 г. Т.Уиттед разрабатывает общие принципы трассировки лучей, включающие отражение, преломление, затенение и методы антиэлайзинга. В 1984 г. группой исследователей (Горэл, Торрэнс, Гринберг и др.) была предложена модель излучательности, одновременно развиваются методы прямоугольного клиппирования областей.

В 1980-е годы появляется целый ряд компаний, занимающихся прикладными разработками в области компьютерной графики. В 1982 г. Дж.Кларк создает Silicon Graphics, тогда же возникает Ray Tracing Corporation, Adobe System, в 1986 г. компания Pixar отпочковывается от Lukasfilm.

В эти годы компьютерная графика уже прочно внедряется в киноиндустрию, развиваются приложения к инженерным дисциплинам. В 1990-е годы в связи с возникновением сети Internet у компьютерной графики появляется еще одна сфера приложения.

Здесь перечислены далеко не все серьезные шаги на пути развития графики, но более подробное знакомство с ее историей требует достаточно хорошего представления о теории и алгоритмах этой дисциплины, поэтому мы ограничиваемся лишь кратким обзором. Нетрудно заметить, что приоритет в развитии данного направления в информационных технологиях достаточно прочно удерживают американские исследователи. Но и в отечественной науке тоже были свои разработки, среди которых можно назвать ряд технических реализаций дисплеев, выполненных в разные годы:

1968, ВЦ АН СССР, машина БЭСМ-6, вероятно, первый отечественный растровый дисплей с видеопамятью на магнитном барабане;

1972, Институт автоматики и электрометрии (ИАиЭ), векторный дисплей "Символ";

1973, ИАиЭ, векторный дисплей "Дельта";

1977, ИАиЭ, векторный дисплей ЭПГ-400;

1982, Киев, НИИ периферийного оборудования, векторный дисплей СМ-7316, 4096 символов, разрешение 2048?2048;

1979-1984, Институт прикладной физики, серия растровых цветных полутоновых дисплеев "Гамма". Последние дисплеи данной серии имели таблицу цветности, поддерживали окна, плавное масштабирование.

Таким образом, в процессе развития компьютерной графики можно выделить несколько этапов.

В 1960-1970-е годы она формировалась как научная дисциплина. В это время разрабатывались основные методы и алгоритмы: отсечение, растровая развертка графических примитивов, закраска узорами, реалистическое изображение пространственных сцен (удаление невидимых линий и граней, трассировка лучей, излучающие поверхности), моделирование освещенности.

В 1980-е графика развивается более как прикладная дисциплина. Разрабатываются методы ее применения в самых различных областях человеческой деятельности.

В 1990-е годы методы компьютерной графики становятся основным средством организации диалога "человек-компьютер" и остаются таковыми по настоящее время.

Технические средства поддержки компьютерной графики

Развитие компьютерной графики во многом обусловлено развитием технических средств ее поддержки. Прежде всего это устройства вывода, каковыми являются дисплеи. В настоящее время существует несколько типов дисплеев, использующих электронно-лучевую трубку, а также дисплеи на жидкокристаллических индикаторах и другие их виды. Нас интересуют главным образом функциональные возможности дисплеев, поэтому мы не будем касаться их внутреннего устройства и электронных схем.

Возникновение компьютерной графики, как уже говорилось ранее, можно отнести к 50-м годам. Дисплейная графика на первом этапе своего развития использовала электронно-лучевые трубки (ЭЛТ) с произвольным сканированием луча для вывода в виде изображения информации из ЭВМ. С эксперимента в Массачусетском технологическом институте начался этап развития векторных дисплеев (дисплеев с произвольным сканированием луча).

Самым простым из устройств на ЭЛТ является дисплей на запоминающей трубке с прямым копированием изображения. Запоминающая трубка обладает свойством длительного времени

послесвечения: изображение остается видимым в течение длительного времени (до одного часа). При выводе изображения интенсивность электронного луча увеличивают до уровня, при котором происходит запоминание следа луча на люминофоре. Сложность изображения практически не ограничена. Стирание происходит путем подачи на всю трубку специального напряжения, при котором свечение исчезает, и эта процедура занимает приблизительно 0,5 с. Поэтому изображения, полученные на экране, нельзя стереть частично, а стало быть, динамические изображения или анимация на таком дисплее невозможны. Дисплей на запоминающей трубке является векторным, или дисплеем с произвольным сканированием, т.е. он позволяет провести отрезок из одной адресуемой точки в любую другую. Его достаточно легко программировать, но уровень интерактивности у него ниже, чем у ряда дисплеев других типов ввиду низкой скорости и плохих характеристик стирания.

Следующий тип - это векторные дисплеи с регенерацией изображения. При перемещении луча по экрану в точке, на которую попал луч, возбуждается свечение люминофора экрана. Это свечение достаточно быстро прекращается при перемещении луча в другую позицию (обычное время послесвечения - менее 0,1 с). Поэтому, для того чтобы изображение было постоянно видимым, приходится его "перерисовывать" (регенерировать изображение) 50 или 25 раз в секунду. Необходимость регенерации изображения требует сохранения его описания в специально выделенной памяти, называемой памятью регенерации. Само описание изображения называется дисплейным файлом. Понятно, что такой дисплей требует достаточно быстрого процессора для обработки дисплейного файла и управления перемещением луча по экрану.

Обычно серийные векторные дисплеи успевали 50 раз в секунду строить только около 3000–4000 отрезков. При большем числе отрезков изображение начинает мерцать, так как отрезки, построенные в начале очередного цикла, полностью гаснут к тому моменту, когда будут строиться последние.

Другим недостатком векторных дисплеев является малое число градаций по яркости (обычно от двух до четырех). Были разработаны, но не нашли широкого применения двух- и трехцветные ЭЛТ, также обеспечивавшие несколько градаций яркости.

В векторных дисплеях легко стереть любой элемент изображения - достаточно при очередном цикле построения удалить стираемый элемент из дисплейного файла.

Текстовый диалог поддерживается с помощью алфавитно-цифровой клавиатуры. Косвенный графический диалог, как и во всех остальных дисплеях, осуществляется перемещением перекрестия (курсора) по экрану с помощью тех или иных средств управления перекрестием - координатных колес, управляющего рычага (джойстика), трекбола (шаровой рукоятки), планшета и т.д. Отличительной чертой векторных дисплеев является возможность непосредственного графического диалога, заключающаяся в простом указании с помощью светового пера объектов на экране (линий, символов и т.д.).

Векторные дисплеи обычно подключаются к ЭВМ высокоскоростными каналами связи. Первые серийные векторные дисплеи за рубежом появились в конце 1960-х годов.

Прогресс в технологии микроэлектроники привел к тому, что с середины 1970-х годов преимущественное распространение получили дисплеи с растровым сканированием луча. Растровое устройство можно рассматривать как матрицу дискретных точек (пикселей), каждая из которых может быть подсвечена. Таким образом, оно является точечно- рисующим устройством. Поэтому любой изображаемый на экране дисплея отрезок строится с помощью последовательности точек, аппроксимирующих идеальную траекторию отрезка, подобно тому, как можно строить изображение по клеткам на клетчатом листке бумаги. При этом отрезок получается прямым только в случаях, когда он горизонтален, вертикален или направлен под углом 45° к горизонтали. Все другие отрезки выглядят как последовательность "ступенек" (ступенчатый эффект).

При построении изображения в растровых графических устройствах используется буфер кадра, представляющий собой большой непрерывный участок памяти компьютера. Для каждой точки в растре отводится как минимум один бит памяти. Буфер кадра сам по себе не является устройством вывода, он лишь используется для хранения рисунка. Наиболее часто в качестве устройства вывода, используемого с буфером кадра, выступает видеомонитор.

Чтобы понять принципы работы растровых дисплеев, мы рассмотрим в общих чертах устройство цветной растровой электронно-лучевой трубки. Изображение на экране получается с помощью сфокусированного электронного луча, который, попадая на экран, покрытый люминофором, дает яркое цветовое пятно. Луч в растровом дисплее может отклоняться только в строго определенные позиции на экране, образующие своеобразную мозаику. Люминофорное

покрытие тоже не непрерывно, а представляет собой множество близко расположенных мельчайших точек, куда может позиционироваться луч. Дисплей, формирующий черно-белые изображения, имеет одну электронную пушку, и ее луч высвечивает однотонные цветовые пятна. В цветной ЭЛТ находятся три электронных пушки, по одной на каждый основной цвет: красный, зеленый и синий. Электронные пушки часто объединены в треугольный блок, соответствующий треугольным блокам красного, зеленого и синего люминофоров на экране. Электронные лучи от каждой из пушек, проходя через специальную теневую маску, попадают точно на пятно своего люминофора. Изменение интенсивности каждого из трех лучей позволяет получить не только три основных цвета, но и цвета, получаемые при их смешении в разных пропорциях, что дает очень большое количество цветов для каждого пикселя экрана.

Дисплеи на жидкокристаллических индикаторах работают аналогично индикаторам в электронных часах, но, конечно, изображение состоит не из нескольких крупных сегментов, а из большого числа отдельно управляемых точек. Эти дисплеи имеют наименьшие габариты и энергопотребление, поэтому широко используются в портативных компьютерах. Они имеют как преимущества, так и недостатки по сравнению с дисплеями на ЭЛТ. Хотя исторически такой способ вывода изображения появился раньше, чем растровый дисплей с ЭЛТ, но быстро развиваться он начал значительно позднее. Эти дисплеи также являются растровыми устройствами (их тоже можно представить как матрицу элементов - жидких кристаллов).

Существуют и другие виды дисплеев, например плазменная панель, но мы не будем их касаться, поскольку они также являются растровыми, а техническая реализация не является предметом нашего курса. Важно то, что рассматриваемые нами алгоритмы разработаны для растровых графических дисплеев, а общие принципы работы этих устройств нам понятны.

Помимо дисплеев, в качестве устройств вывода изображений используются плоттеры (графопостроители), предназначенные для вывода графической информации на бумагу. Ранние графические пакеты были ориентированы именно на модель перьевого плоттера, формирующего изображение с помощью пера. Перо может перемещаться вдоль двух направляющих, соответствующих двум координатным осям, причем оно может находиться в двух состояниях - поднятом и опущенном. В поднятом состоянии оно просто перемещается над поверхностью бумаги, а в опущенном оставляет на бумаге линии, формирующие изображение. Таким образом, плоттер стоит ближе к векторным дисплеям, но отличается от них тем, что стирать выводимые изображения невозможно. Поэтому для них изображение сначала полностью формируется в памяти компьютера, а затем выводится.

Кроме того, следует упомянуть принтеры, выводящие изображение на бумагу или пленку. Изображение, получаемое с помощью современных принтеров, также формируется как точечное (растровое), но, как правило, с лучшим разрешением, чем экранное. Как и в случае с графопостроителем, стереть изображение или его часть невозможно.

Теперь сделаем небольшой обзор устройств ввода информации, позволяющих решать различные задачи компьютерной графики, не вдаваясь в детали физических принципов их работы. Эти устройства позволяют организовать диалог "человек-компьютер", а особенности конструкции каждого устройства позволяют ему специализироваться на выполнении определенного круга задач. Нас они интересуют именно как логические устройства, т.е. с точки зрения выполняемых ими функций.

Первую группу устройств, с помощью которых пользователь может указать позицию на экране, назовем устройствами указания (pointing device): мышь, трекбол (trackball), световое перо (lightpen), джойстик (joystick), спейсбол (spaceball). Практически все устройства этой группы оснащены парой или несколькими кнопками, которые позволяют сформировать и передать в компьютер какие-либо сигналы или прерывания.

Рис. 1.1. Мышь

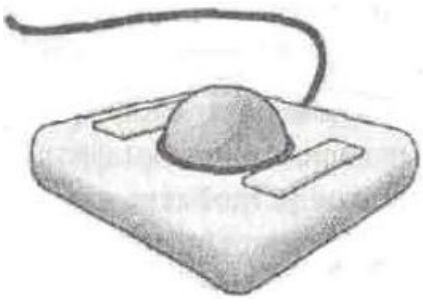


Рис. 1.2. Трекбол

Мышь (рис. 1.1) и трекбол (рис. 1.2) похожи не только по назначению, но часто и по конструкции. В механической мыши и трекболе вращение шарика преобразуется с помощью пары преобразователей в сигналы, передаваемые в компьютер. Преобразователи измеряют вращение относительно двух взаимно перпендикулярных осей. Существует очень много модификаций устройств этих групп. В оптической мыши используются не механические, а оптические чувствительные элементы для измерения перемещения: измеряется расстояние путем подсчета штрихов на специальной подложке. Маленькие трекболы широко применяются в портативных компьютерах, где их встраивают прямо в клавиатуру.

В некоторые клавиатуры встраиваются приборы, чувствительные к давлению, которые выполняют те же функции, что и мышь или трекбол, но при этом в них отсутствуют подвижные элементы. Преобразователи в таких устройствах измеряют величину давления на небольшой выпуклый набалдашник, размещенный между двумя кнопками в средней части клавиатуры. Они, как и трекбол, используются преимущественно в портативных компьютерах.

Выходные сигналы мыши или трекбола можно рассматривать как две независимые величины и преобразовывать их в координаты положения на двумерной плоскости экрана или в какой-либо другой системе координат. Считанные с устройства значения можно сразу же использовать для управления специальной отметкой (курсором) на экране.

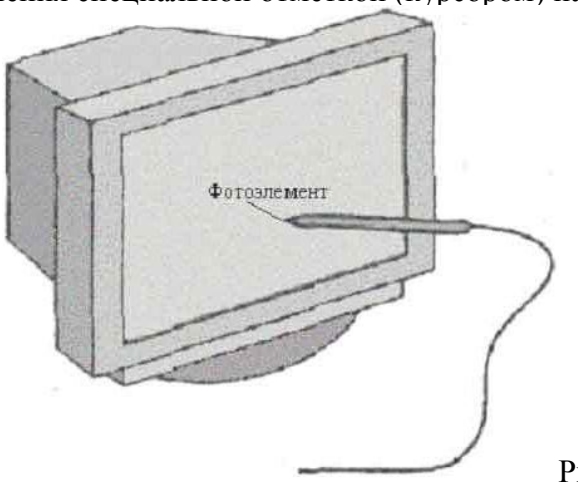


Рис. 1.3. Световое перо

Ветераном среди устройств ввода в компьютерной графике является устройство, названное при его создании световым пером. Впервые оно появилось в уже упомянутом проекте А.Сазерленда Sketchpad. Световое перо содержит фоточувствительный элемент (рис. 1.3), который при приближении к экрану воспринимает излучение, порождаемое при столкновении электронов с люминофорным покрытием экрана. Если мощность светового импульса превышает определенный порог, фоточувствительный элемент формирует импульс, который передается в компьютер. Анализируя смещение по времени этого импульса относительно начала цикла регенерации, компьютер может точно определить координаты той точки экрана, возбуждение которой "высветило" фотоэлемент. Таким образом, в распоряжении пользователя оказывается устройство непосредственного указания, работающее напрямую с изображением на экране. В настоящее время это устройство уже практически вышло из употребления: оно вытеснено более простым и надежным - мышью.

Еще одно устройство, которое достаточно активно используется в мультимедийных приложениях, а также в различного рода компьютерных тренажерах - джойстик (рис. 1.4). Перемещение джойстика в двух взаимно перпендикулярных направлениях воспринимается преобразователями, интерпретируется как вектор скорости, а полученные значения используются для управления положением маркера на экране. Обработка сигнала выполняется таким образом, что неподвижный джойстик в каком- либо промежуточном положении не изменяет положения маркера, а чем дальше джойстик отклонен от начального положения, тем быстрее маркер перемещается по экрану. Таким образом, джойстик играет роль устройства ввода с переменной чувствительностью. Другое достоинство джойстика - наличие силовой обратной связи, обеспеченной наличием разного рода пружин. При этом пользователь чувствует, что чем дальше отклонен джойстик, тем большее усилие требуется для его дальнейшего движения. Это как раз те свойства, которые нужны при работе с разного рода симуляторами, а также в компьютерных играх.

Рис. 1.4. Джойстик

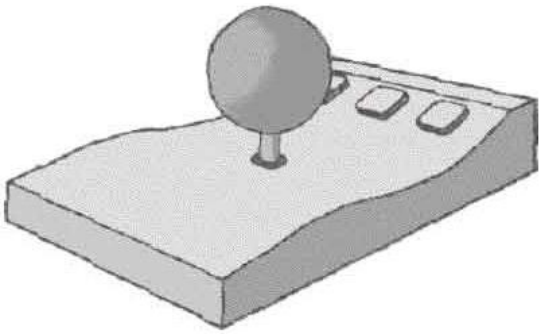


Рис. 1.5. Спейсбол

Спейсбол - это "трехмерное" устройство ввода. Хотя и существуют различные конструкции таких устройств, они все еще не получили широкого распространения, поскольку проигрывают популярным двумерным устройствам как по стоимости, так и по техническим характеристикам. Спейсбол похож на джойстик, но отличается от него тем, что он имеет вид закрепленного на рукоятке шара, причем рукоятка в этой конструкции неподвижна (рис. 1.5). Шар имеет датчики давления, которые измеряют усилие, прикладываемое пользователем. Шар может измерять не только составляющие усилия в трех основных направлениях (сверху вниз, от себя или на себя, влево-вправо), но и вращение относительно трех осей. Таким образом, это устройство способно передавать в компьютер шесть независимых параметров (т. е. имеет шесть степеней свободы), характеризующих как поступательное движение, так и вращение.

Существуют и другие трехмерные системы измерения и ввода, использующие самые современные технологии, например лазерные. В системах виртуальной реальности используются более сложные устройства, позволяющие динамически отслеживать положение и ориентацию пользователя. Для приложений, связанных с современной робототехникой и моделированием виртуальной реальности, иногда требуются устройства, обладающие еще большим числом степеней свободы, чем спейсбол. В последнее время появились новые разработки в этом направлении, в частности - перчатки с системой датчиков, которые способны улавливать движения отдельных частей руки человека (рис. 1.6).

Рис. 1.6. Перчатка для ввода данных

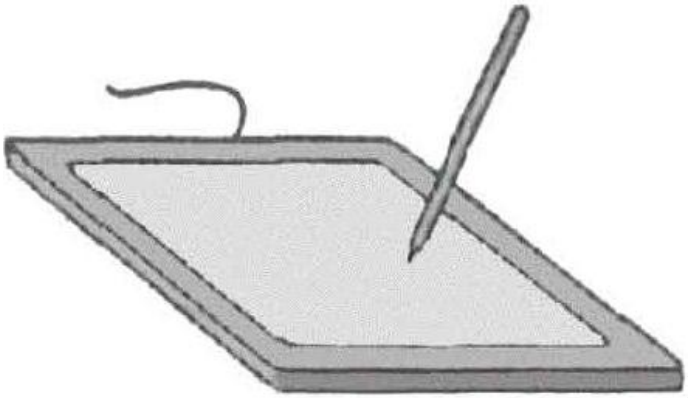


Рис. 1.7. Планшет

При использовании мыши или трекбола анализируется относительное положение устройства. Если переместить указатель на экране каким-либо способом в другое место, не вращая при этом шарик мыши или трекбола, то дальнейшие сигналы будут смещать указатель относительно новой позиции. Можно также аккуратно переместить мышь без вращения шарика и это не приведет к перемещению курсора на экране. Абсолютные координаты устройства не считываются обрабатывающей программой. Но при вводе в компьютер графиков прикладной программе зачастую требуются абсолютные координаты устройства ввода. Такую возможность обеспечивают разного рода планшеты (рис. 1.7). В планшете применяется, как правило, ортогональная сетка проводов, расположенная под его поверхностью. Положение пера определяется через электромагнитное взаимодействие сигналов, проходящих от проводов к щупу. Иногда в качестве планшета используются чувствительные к прикосновению прозрачные экраны, которые наносятся на поверхность ЭЛТ. Небольшие экраны такого типа размещаются иногда на клавиатуре портативных компьютеров. Чувствительные панели можно использовать в режимах как абсолютных, так и относительных координат.

Для растрового ввода изображений используются сканеры, позволяющие не только ввести образ в компьютер, но и произвести их обработку и документирование. Одна из важных областей применения сканеров - ввод текстов. При этом обработка введенного изображения выполняется программным обеспечением распознавания текстов, которое в настоящее время стало уже достаточно развитым. В САПР сканеры используются для автоматизации ввода ранее подготовленной конструкторской документации. В этом случае проблема заключается в том, что данные от сканера представлены в растровой, а не векторной форме, и требуется выполнение обратного преобразования "растр-вектор". Эта задача очень сложна: необходимо распознавать различные изображения и тексты, в том числе рукописные, учитывать, что линия может при сканировании не только получить различную ширину на разных участках, но и оказаться разорванной и т.д. Для решения этой задачи средств одной лишь компьютерной графики недостаточно: необходимо привлечение и других дисциплин.

Все вышеперечисленные устройства ввода с точки зрения передачи информации прикладным программам следует рассматривать как логические. Функционирование систем ввода характеризуется тем, какую информацию устройство передает в программу, когда и как оно передает эту информацию. Эти вопросы становятся особенно существенными при разработке пользовательского интерфейса.

Основной объем информации об окружающем мире человек воспринимает зрительно (визуально). Более 90 процентов информации, воспринимаемой человеком, составляет именно визуальная информация. Исторически первым видом вводимой и обрабатываемой в компьютерах информации была алфавитно-цифровая. Однако большой объем задач, решаемых с использованием визуальной информации и диспропорция между скоростями поступления и обработки видеоинформации привели к тому, что в начале 60-х годов были начаты работы по автоматизации ее обработки.

Компьютерная графика в настоящее время сформировалась как наука об аппаратном и программном обеспечении для разнообразных изображений от простых чертежей до реалистичных образов естественных объектов. Компьютерная графика используется почти во всех научных и инженерных дисциплинах для наглядности восприятия и передачи информации. Применяется в медицине, рекламном бизнесе, индустрии развлечений и т. д. Основным объектом обработки, а в некоторых случаях конечным продуктом в компьютерной графике является изображение. Это изображение может использоваться в различных сферах, например, оно может быть техническим

чертежом, иллюстрацией с изображением детали в руководстве по эксплуатации, простой диаграммой, архитектурным видом предполагаемой конструкции или проектным заданием, рекламной иллюстрацией или кадром из мультфильма.

В очень многих случаях данные представляются в виде изображений: графиков, карт, чертежей, цветных и черно-белых полутоновых снимков и т.д. Изображения получают в диапазоне частот видимого цвета, ультразвуковом, инфракрасном, рентгеновском, акустическом, ультрафиолетовом, с помощью гамма лучей, ядерного магнитного резонанса.

Средства их формирования и регистрации отличаются большим разнообразием. Для обработки и анализа изображений были созданы аналоговые, аналого-цифровые, телевизионные, оптико-механические устройства и системы. Однако по мере развития компьютерных технологий особая роль в технике обработки изображений стала принадлежать компьютерам. Использование компьютеров для обработки и анализа изображений стало новой обширной областью их применения. Компьютерная (ее называют еще цифровой) обработка изображений позволяет быстро и с меньшими, по сравнению с экспериментальным макетированием, затратами моделировать любые методы обработки изображений. При этом обеспечиваются точность, надежность, воспроизводимость результатов, возможность контроля процесса обработки на любой промежуточной стадии, гибкость в отношении типа и характера решаемых задач.

Области применения цифровой обработки и анализа изображений.

Цифровая обработка и анализ изображений проникают почти во все области деятельности человека, поэтому развитие методов и вычислительных средств обработки и анализа изображений привело к образованию новой отрасли науки и техники, имеющей большую перспективу.

Особое развитие цифровая обработки изображений получила в связи с появлением и широким внедрением во все сферы деятельности человека персональных компьютеров. Методы цифровой обработки изображений играют значительную роль в научных исследованиях и в производственных процессах. К областям применения цифровой обработки и анализа изображений относятся медицина, биология, геология, геодезия, картография. Обработка спутниковой информации вообще не может обойтись без использования методов и средств цифровой обработки изображений. Примерами использования этих методов могут служить видеотелефонная связь, передача изображений по сети Интернет, цифровая передача изображений с космических летательных аппаратов, коррекция их искажений, повышение четкости изображений, автоматический анализ характера местности и исследование природных ресурсов по фотоснимкам, передаваемым со спутников Земли. (см. таб.1)

Примеры применения методов обработки изображений и решаемых задач.

Отрасль науки и техники	Вид изображения	Решаемые задачи
1	2	3
Астрономия	Астронегативы со снимками планет и звезд, звездные архивы, каталоги на фото -носителях.	Определение координат звезд, звездных скоплений, расстояний между звездами. Составление баз данных.
Геология	Аэрокосмические снимки (в том числе спектрозональные), микроскопические снимки шлифов минералов и пород. Карты, в основном цветные	Определение линеаментов и кольцевых структур для поиска новых и оценки запасов старых месторождений полезных ископаемых, параметров кристаллов. Составление фотокарт. Анализ и статобработка карт.
Биология(в том числе молекулярная биология, цитология, генетика, микробиология, вирусология и т.д.)	Микроскопические и электронно-микроскопические снимки срезов тканей, клеток растительного и животного происхождения	Определение параметров клеток, хромосом, тканей органов и т.д. Получение численных характеристик объектов на снимках.
Медицина	Снимки, полученные с	Исследование, анализ

(медицинская биология, клиническая медицина, фармакология, эпидемиология)	микроскопов, электронных микроскопов, рентгенограммы, томограммы, МРТ-снимки т.д.	состояния отдельных органов, клеток, получение численных характеристик, морфометрический анализ.
Геофизика, Геодезия и картография География	Аэрокосмические снимки, карты различного назначения	Дешифрирование снимков для составления карт, диаграмм, фотокарт, составление баз картографических данных и т.д.
Металлургия	Снимки шлифов и сплавов, порошков металлов	Изучение структуры металлов, получение количественных характеристик.
Машиностроение	Графические изображения, особенно чертежи, схемы.	Автоматизация технологической подготовки производства, эффективное использование машинной графики.
Робототехника	Изображение различных реальных объектов в системах технического зрения робототехнических комплексов. Различные изображения тканей, деталей, комплектующих и пр.	Распознавание объектов по их изображениям, определение дефектов, подсчет, сортировка деталей и т.д.
Физика	Изображение траекторий различных частиц, интерференционные и дифракционные картины	Вычисление параметров частиц, определение харак-теристик газодинамических и элетронных процессов
Лесное хозяйство	Аэрокосмические снимки, карты	Картирование лесов, определение зараженности лесов, ревизия запасов
Сельское хозяйство	Аэрокосмические и микроскопические снимки	Сельхоздешифрирование аэрокосмических снимков, прогноз урожая, картиро-вание почв
Водное хозяйство	Аэрокосмические снимки и карты	Картирование водохозяйственных систем, контроль водопользования, определение площадей орошения, охрана вод
Мониторинг окружающей среды	Аэрокосмические снимки, карты	Распознавание зон загрязнения, определение их параметров. Определение результатов стихийных бедствий, обновление карт различного назначения.
Телевидение, кино, фототелеграфия, реклама, индустрия компьютерных игр	Телевизионные изображения, анимационные картинки, 2-х и 3-х мерная графика	Разработка эффективных методов кодирования, моделирование систем передачи данных, разработка анимационных эффектов.
Архитектура Градостроительство Дизайн	Планы, макеты, аэрокосмические снимки	Составление схем, планов, макетов отдельных зданий, районов, городов
Архивное дело	Фотографии, тексты, картины	Улучшение качества, вос-становление, реставрация, размножение уникальных исторических документов
Криминалистика	Изображения отпечатков пальцев, подписей и пр.	Анализ отпечатков пальцев, определение количест-венных

		характеристик.

Приведенный перечень задач предполагает наличие специализированных систем обработки, в составе которых, кроме чисто графических средств, есть и средства вычислительные, измерительные и т.д. Для некоторых из указанных задач были разработаны специализированные графические системы, особенно это касается задач обработки спутниковой информации, компьютерных томографов пр. Перечень можно было бы продолжать долго.

В настоящее время появилось новое, очень интересное приложение компьютерной графики - виртуальная реальность.

По телевидению часто можно видеть передачи иллюстрирующие приложения компьютерной графики в автоматизации проектирования (были передачи об автоматизированном проектировании самолетов, автомобилей), много передач об автоматизации производства с различными робототехническими системами.

Передачи о мире бизнеса практически не обходятся без показа различной дисплейной техники и ее использования.

Что касается искусства, то достаточно упомянуть, что один из самых крупных первых суперкомпьютерных центров мира находился на студии Уолта Диснея и использовался для подготовки мультфильмов. Всем известно, что многие "жутики" и боевики также готовились с широким использованием средств компьютерной графики для подготовки высокореалистичных сцен.

Применение компьютерной графики в средствах массовой информации мы видим ежедневно, как в виде различных заставок и телеэффектов на экране, так и в виде газет, при подготовке многих из которых используется электронная верстка на компьютере.

С компьютерными играми, отнимающими не только время досуга, конечно же знаком каждый.

Виртуальная реальность - новейшее направление приложений компьютерной графики, позволяющее имитировать окружающую действительность с новым уровнем взаимодействия человек-ЭВМ. Основа систем виртуальной реальности - высокопроизводительная графическая рабочая станция, обладающая достаточным быстродействием и изобразительными возможностями для формирования высокореалистичных цветных полутоновых изображений. Устройства отображения в различных системах могут быть самыми различными - от обычных мониторов высокого разрешения до экранов во всю стену, используемых в имитаторах боевых действий, или же стереоскопических систем отображения, в том числе и в виде специальных очков, вмонтированных в шлем, надеваемый на голову.

Одно из важнейших отличий систем виртуальной реальности от других систем отображения - наличие средств воздействия не только на зрение, но и на другие органы чувств. В первую очередь это системы стереозвука, имитирующие требуемое распределение и интенсивности источников звука в пространстве. Наиболее дорогие системы обеспечивают воздействие и на осязание за счет использования специальных шлемов, перчаток и костюмов, которые за счет встроенных в них устройств не только определяют положение головы, направление взгляда, положение рук, пальцев, тела, но и имитируют прикосновения, сопротивление или "податливость" ручек и т.д. Можно почувствовать прикосновение к объекту существующему лишь в памяти компьютера! Осталось симитировать запахи.

В настоящее время системы виртуальной реальности очень дороги. Самые дешевые стоят около 20 тыс долларов, более совершенные системы - около 100 тысяч долларов.

Вопросы и упражнения

1. Назовите четыре основные области применения компьютерной графики.
2. Каковы основные направления развития компьютерной графики? Какие задачи они решают?
3. Где и когда впервые был использован дисплей в качестве устройства вывода ЭВМ?
4. Кем и когда была разработана первая интерактивная программа для рисования?
5. Назовите основных разработчиков методов закрашивания гладких поверхностей.
6. Кто является автором ряда алгоритмов построения растровых образов различных геометрических объектов?
7. Назовите авторов алгоритмов удаления невидимых линий.
8. В чем состоит основное различие между дисплеями с произвольным сканированием и растровым сканированием?

9. Чем отличается дисплей на запоминающей трубке от векторного дисплея с регенерацией изображения?
10. Каковы основные принципы работы цветной растровой электронно-лучевой трубки?
11. Как работает перьевой плоттер?
12. Назовите основные устройства ввода, использующиеся в компьютерной графике.
13. Какие из устройств ввода дают возможность работать в абсолютных координатах?
14. Перечислите области применения сканеров.

ЛЕКЦИЯ 2: ГРАФИЧЕСКИЕ ФОРМАТЫ

Любая информация, хранящаяся в файле, - это последовательность байт. Каждый байт может принимать значение от 0 до 255 (28 - 1). Способ записи информации с помощью последовательности байт и называют **форматом файла.** То есть, **графический формат**- это способ записи графической информации. Способ представления изображения оказывает влияние на возможности его редактирования, печати, на объем занимаемой памяти.

Огромное количество разнообразных графических форматов объясняется тем. что фирмы-разработчики в разное время придумывали для своих нужд то, что им удобно, поэтому до сих пор не существует какого-то одного стандартного формата.

Графи́ческий форма́т — это способ записи графической информации. Графические форматы файлов предназначены для хранения изображений, таких как фотографии и рисунки.

Графические форматы делятся на векторные и растровые.

Векторный формат

Существуют два основных способа кодирования графической информации: векторный и растровый. При **векторном,** на котором мы сейчас не будем подробно останавливаться, рисунок представляется в виде комбинации простых геометрических фигур - - точек, отрезков прямых и кривых, окружностей, прямоугольников и т. п. При этом для полного описания рисунка необходимо знать вид и базовые координаты каждой фигуры, например, координаты двух концов отрезка, координаты центра и диаметр окружности и т. д. Этот способ кодирования идеально подходит для рисунков, которые легко представить в виде комбинации простейших фигур, например, для технических чертежей.

Растровый формат

Растровый формат характеризуется тем, что все изображение по вертикали и горизонтали разбивается на достаточно мелкие прямоугольники - - так называемые элементы изображения, или **пикселы** (от английского ***pixel*** - - *picture element*).

В файле, содержащем растровую графику, хранится информация о цвете каждого пиксела данного изображения. Чем меньше прямоугольники, на которые разбивается изображение, тем больше разрешение ***(resolution)***, то есть, тем более мелкие детали можно закодировать в таком графическом файле.

Размер ***(size)*** изображения, хранящегося в файле, задается в виде числа пикселов по горизонтали ***(width)*** и вертикали ***(height)***. Для примера, оптимальное разрешение 15 - дюймового монитора, как правило, составляет 1024x768.

Глубина цвета

Кроме размера изображения, важной является информация о количестве цветов, закодированных в файле. Цвет каждого пиксела кодируется определенным числом бит ***(bit)***, то есть элементарных единиц информации, с которыми может иметь дело компьютер. Каждый бит может принимать два значения - - 1 или 0. В зависимости от того, сколько бит отведено для цвета каждого пиксела, возможно кодирование различного числа цветов. Нетрудно сообразить, что если для кодировки отвести лишь один бит, то каждый пиксел может быть либо белым (значение 1), либо черным (значение 0). Такое изображение называют монохромным ***(monochrome)***.

Далее, если для кодировки отвести четыре бита, то можно закодировать 24 = 16 различных цветов, отвечающих комбинациям бит от 0000 до 1111. Если отвести 8 бит - - то такой рисунок может содержать 28 = 256 различных цветов (от 00000000 до 11111111), 16 бит - - 216 = 65 536 различных цветов (так называемый ***High Color***). И, наконец, если отвести 24 бита, то потенциально рисунок может содержать 224 = 16 777 216 различных цветов и оттенков - - вполне достаточно даже для самого взыскательного художника! В последнем случае кодировка называется *24 - bit* ***True Color***. Следует обратить внимание на слово "потенциально": даже если в файле и отводится 24 бита на каждый пиксел, это еще не означает, что вы действительно сможете насладиться такой богатой палитрой - - ведь технические возможности мониторов ограничены.

Наиболее распространенным способом кодирования цвета является модель **RGB**. При этом способе кодирования любой цвет представляется в виде комбинации трех цветов: красного ***(Red)***, зеленого ***(Green)*** и синего ***(Blue)***, взятых с разной интенсивностью. Интенсивность каждого из трех цветов - - это один байт (т. е. число в диапазоне от 0 до 255), который хорошо представляется двумя 16 - ричными цифрами (числом от 00 до FF). Таким образом, цвет удобно записывать тремя парами 16 - ричных цифр, как это принято, например, в HTML - документах.

Пример.

В языке гипертекстовой разметки документов HTML цвета можно задавать так: черный - - 000000, белый - - FFFFFF, желтый - - FFFF00 и т. д.; чтобы получить более темный желтый цвет, надо одинаково уменьшить интенсивности красного и зеленого - - A7A700.

Чем больше значение байта цветовой составляющей, тем ярче этот цвет. При наложении одной составляющей на другую яркость суммарного цвета также увеличивается.

Файлы с расширениями BMP, PCX, GIF, TIF и JPG содержат растровые графические изображения. Расширение в имени файла говорит о том, в каком формате хранится информация. Например, расширение BMP обозначает BMP-файл, поддерживаемый в системах Windows и OS/2 (BMP - сокращение от bitmap, т.е. битовый, растровый); TIF - сокращение от TIFF или Tagged Image File Format. Это только два представителя большого семейства форматов, используемых на персональных компьютерах.

Каждый из этих форматов по-разному хранит графическую информацию, и каждый из них разрабатывался под конкретные цели. Формат GIF (Graphics Interchange File - файл графического обмена), например, был придуман для того, чтобы уместить как можно больше информации в ограниченном пространстве с целью уменьшить время получения файлов для пользователей сети CompuServe. Формат PCX изначально был придуман для хранения черно-белых графических файлов, создаваемых ранней версией программы раскраски PC Paintbrush для компьютеров IBM PC. Сейчас формат PCX применим и для цветных графических файлов.

Форматы растровой графики

GIF (Graphics Interchange Format)

Формат GIF был разработан в 1987 году компьютерной информационной службой CompuServe. Сейчас этот формат является наиболее используемым форматом в сети Интернет.

Преимущества

1. Малый размер, который достигается ограниченной цветовой гаммой - не более 256 цветов.
2. Прозрачный фон.
3. Анимация.

И еще один немаловажный фактор, то что алгоритм сжатия LZW формата GIF запатентован. Владельцем патента с 1994 года является фирма Unisys, и она начала брать плату с разработчиков, использующих формат GIF.

Структура файла GIF зависит от версии GIF-спецификации, которой соответствует файл. В настоящее время используются две версии, GIF87a и GIF89a. Первая из них проще. Независимо от номера версии, файл GIF начинается с 13-байт заголовка, содержащего сигнатуру, которая идентифицирует этот файл в качестве GIF-файла, номер версии GIF и другую информацию. Если файл хранит всего одно изображение, вслед за заголовком обычно располагается общая таблица цветов, определяющая цвета изображения. Если в файле хранится несколько изображений (формат GIF, аналогично TIFF, позволяет в одном файле кодировать два и больше изображений), то вместо общей таблицы цветов каждое изображение сопровождается локальной таблицей цветов.

В файле GIF87a вслед за заголовком и общей таблицей цветов размещается изображение, которое может быть первым из нескольких располагаемых подряд изображений. Каждое изображение состоит из 10-байт описателя изображения, расположенной вслед за ним локальной таблицы цветов и битов растрового массива. Для повышения эффективности использования памяти данные растрового массива сжимаются с помощью алгоритма LZW.

Файлы GIF89a имеют аналогичную структуру, но они могут содержать факультативные блоки расширения с дополнительной информацией о каждом изображении. В спецификации GIF89a определены четыре типа блоков расширения. Это блоки расширения для управления графикой, которые описывают, как изображение должно выводиться на экран (например, накладывается ли оно на предыдущее изображение подобно диапозитиву или просто заменяет его); блоки расширения с обычным текстом, содержащие текст, отображаемый вместе с графикой; блоки расширения для комментария, содержащие комментарии в коде ASCII; и блоки расширения прикладных программ, в которых хранится информация, принадлежащая только создавшей этот файл программе. Блоки расширения могут находиться практически в любом месте файла после общей таблицы цветов.

Основные достоинства GIF заключаются в широком распространении этого формата и его компактности. Но ему присущи два достаточно серьезных недостатка. Один из них состоит в том, что в изображениях, хранящихся в виде GIF-файла, не может быть использовано более 256 цветов. Второй, возможно, еще более серьезный, заключается в том, что разработчики программ, использующие в них форматы GIF, должны иметь лицензионное соглашение с CompuServe и вносить плату за каждый экземпляр программы; такая ценовая политика была принята CompuServe

после того, как Unisys объявила, что начнет добиваться соблюдения своих прав собственности и потребовала от тех, кто пользуется алгоритмом сжатия LZW, вносить лицензионные платежи. Возникшее в результате этого запутанное юридическое положение тормозит внедрение программистами в свои графические программы средств для работы с файлами GIF.

PNG (Portable Network Graphics)

Плод сообщества независимых программистов - ответная реакция на переход популярнейшего формата GIF в разряд коммерческих продуктов. Формат PNG делает почти все, что и формат GIF, за исключением анимации. Формат PNG (Portable Network Graphic - переносимый сетевой формат, произносится "пинг") был разработан для замены GIF, чтобы обойти юридические препятствия, стоящие на пути использования GIF-файлов. PNG унаследовал многие возможности GIF и, кроме того, он позволяет хранить изображения с истинными цветами. Еще более важно, что он сжимает информацию растрового массива в соответствии с вариантом пользующегося высокой репутацией алгоритма сжатия LZ77 (предшественника LZW), которым любой может пользоваться бесплатно.

Преимущества

1. Лучшие сжатие данных - сжимает растровые изображения не только по горизонтали, но и по вертикали
2. поддерживает цветные фотографические изображения вплоть до 48-битных включительно
3. 256 уровней прозрачности

Размер картинки в формате PNG будет меньше, чем у GIF. Но самые мелкие мелочи получатся легче у GIF, потому что в файле изображения PNG около 1 Кб занимает описание палитры цветов, что порой бывает сопоставимо с размером самого изображения.

PNG-24

Формат, аналогичный PNG-8, но использующий 24-битную палитру цвета.

JPG, JPEG, JFIF (JPEG File Interchange Format)

Для поиска лучшего способа сжатия изображений фотографического качества, две организации по стандартизации – International Telecommunications Union (ITU) и International Organization for Standartization (ISO) – создали Joint Photographic Experts Group (JPEG). Использует сжатие с "потерями" (lossy compression). При таком сжатии удаляется та информация, которая несущественна для восприятия изображения.

WBMP (WAP BitMap)

Монохромные (двухцветные) изображения. Максимальный размер картинки не должен превосходить ограничений на размер карты – 1,5 Кбайт.

BMP(BitMap)

Самый простой растровый формат BMP является родным форматом Windows. В BMP данные о цвете хранятся только в модели RGB, то есть этот формат создан для использования на экране. Формат файла BMP (сокращенно от BitMaP) - это "родной" формат растровой графики для Windows, поскольку он наиболее близко соответствует внутреннему формату Windows, в котором эта система хранит свои растровые массивы. Для имени файла, представленного в BMP-формате, чаще всего используется расширение BMP, хотя некоторые файлы имеют расширение RLE, означающее run length encoding (кодирование длины серий). Расширение RLE имени файла обычно указывает на то, что произведено сжатие растровой информации файла одним из двух способов сжатия RLE, которые допустимы для файлов BMP-формата.

В файлах BMP информация о цвете каждого пиксела кодируется 1, 4, 8, 16 или 24 бит(бит/пиксел). Числом бит/пиксел, называемым также глубиной представления цвета, определяется максимальное число цветов в изображении. Изображение при глубине 1 бит/пиксел может иметь всего два цвета, а при глубине 24 бит/пиксел - более 16 млн. различных цветов.

На приведенной схеме показана структура типичного BMP-файла, содержащего 256-цветное изображение (с глубиной 8 бит/пиксел). Файл разбит на четыре основные раздела: заголовок файла растровой графики, информационный заголовок растрового массива, таблица цветов и собственно данные растрового массива. Заголовок файла растровой графики содержит информацию о файле, в том числе адрес, с которого начинается область данных растрового массива. В информационном заголовоке растрового массива содержатся сведения об изображении, хранящемся в файле, например, его высоте и ширине в пикселах. В таблице цветов представлены значения основных цветов RGB (красный, зеленый, синий) для используемых в изображении цветов. Программы, считывающие и отображающие BMP-файлы, в случае использования видеоадаптеров, которые не позволяют отображать более 256 цветов, для точной цветопередачи могут программно устанавливать такие значения RGB в цветовых палитрах адаптеров.

Структура файла BMP
Заголовок файла растровой графики (14 байт)
Сигнатура файла BMP (2 байт)
Размер файла (4 байт)
Не используется (2 байт)
Не используется (2 байт)
Местонахождение данных растрового массива (4 байт)
Информационный заголовок растрового массива (40 байт)
Длина этого заголовка (4 байт)
Ширина изображения (4 байт)
Высота изображения (4 байт)
Число цветовых плоскостей (2 байт)
Бит/пиксел (2 байт)
Метод сжатия (4 байт)
Длина растрового массива (4 байт)
Горизонтальное разрешение (4 байт)
Вертикальное разрешение (4 байт)
Число цветов изображения (4 байт)
Число основных цветов (4 байт)
Таблица цветов (длина изменяется от 8 до 1024 байт)
Собственно данные растрового массива (длина переменная)

Формат собственно данных растрового массива в файле BMP зависит от числа бит, используемых для кодирования данных о цвете каждого пиксела. При 256-цветном изображении каждый пиксел в той части файла, где содержатся собственно данные растрового массива, описывается одним байтом (8 бит). Это описание пиксела не представляет значений цветов RGB, а служит указателем для входа в таблицу цветов файла. Таким образом, если в качестве первого значения цвета RGB в таблице цветов файла BMP хранится R/G/B=255/0/0, то значению пиксела 0 в растровом массиве будет поставлен в соответствие ярко-красный цвет. Значения пикселов хранятся в порядке их расположения слева направо, начиная (как правило) с нижней строки изображения. Таким образом, в 256-цветном BMP-файле первый байт данных растрового массива представляет собой индекс для цвета пиксела, находящегося в нижнем левом углу изображения; второй байт представляет индекс для цвета соседнего справа пиксела и т. д. Если число байт в каждой строке нечетно, то к каждой строке добавляется дополнительный байт, чтобы выровнять данные растрового массива по 16-бит границам.

Не все файлы BMP имеют структуру, подобную показанной на схеме. Например, файлы BMP с глубиной 16 и 24 бит/пиксел не имеют таблиц цветов; в этих файлах значения пикселов растрового массива непосредственно характеризуют значения цветов RGB. Также могут различаться внутренние форматы хранения отдельных разделов файла. Например, информация растрового массива в некоторых 16 и 256-цветных BMP-файлах может сжиматься посредством алгоритма RLE, который заменяет последовательности идентичных пикселов изображения на лексемы, определяющие число пикселов в последовательности и их цвет. В Windows допускается работа с BMP-файлами стиля OS/2, в которых используются различные форматы информационного заголовка растрового массива и таблицы цветов.

PCX (PCExchange)

Изображения в формате PCX можно посмотреть большинством программ под DOS. Как и BMP, этот формат в значительной мере устарел и поддерживается современными графическими программами исключительно для совместимости с антикварным софтом.

Файлы изображений, которые используются в продуктах семейства PC Paintbrush и FRIEZE, были разработаны фирмой Zsoft. Расширение файла DOS - PCX, тип формата растровый. Открывать или импортировать файлы PCX могут почти все графические приложения для персональных компьютеров. Цветовые возможности 1, 2, 4, 8 или 24- битовый цвет, никаких оттенков серого. Всегда применяется сжатие ROB. У этого формата есть ряд недостатков: 1) не поддерживает изображения с оттенками серого или таблицы коррекции шкалы серого; 2) не поддерживает цвета CMYK или другие системы отличные от RGB; 3) многочисленные варианты, особенно при работе с цветами, могут делать работу с файлом невозможным; 4) неудобная схема сжатия в действительности может увеличивать размеры некоторых файлов.

Среди положительных сторон формата PCX можно назвать то, что он хорош для изображений, которые: 1)создают ограниченную палитру цветов (лучше всего 16 или 256); 2) не являются изображениями отсканированными или фотографиями, которые плохо сжимаются; 3)должны быть читаемыми всеми приложениями персонального компьютера.

Файлы изображения PCX начинается с заголовка длиной 128 байт. Затем идут закодированные графические данные. При кодировании используется простой алгоритм, основанный на методе длинных серий. Если в файле запоминается несколько цветовых слоев, каждая строка изображения запоминается по цветовым слоям. Согласно документации Zsoft, это выполняется по приведенной ниже схеме (R - красный слой, G - зеленый слой, B - синий слой, I - слой интенсивности)

Строка изображения 0:

RRR...

GGG...
BBB...
III...
Строка изображения 1:
RRR...
GGG...
BBB...
III...
(и т.д.)

Замечание о некорректности документа. Запоминание по слоям проводится, как правило, для 16-цветных изображений EGA. При стандартной палитре EGA, которая устанавливается по умолчанию BIOS'ом, нулевой слой видео памяти содержит СИНЮЮ компоненту цвета, а не красную. Если же палитра отлична от стандартной, то говорить о том, что слои видео памяти, соотносятся с компонентами цвета вообще затруднительно.

Метод кодирования состоит в следующем:
ДЛЯ каждого байта X, прочитанного из файла. ЕСЛИ оба старших бита X равны 1, то <повторитель> = значению, хранящемуся в 6 младших битах X <данные> = находятся в следующем байте за X. ИНАЧЕ <повторитель> = 1 <данные> = X
Поскольку для насыщения данного алгоритма требуется в среднем 25% неповторяющихся данных и, по меньшей мере, наличие смещения между повторяющимися данными, то размер получаемого файла, как правило, оказывается приемлемым.

Таблица 2. Формат заголовка PCX

Смещение	Обозначение	Длина	Описание / комментарий
0	Manufacturer	1	Постоянный флаг 10 = ZSoft .PCX
1	Version	1	Информация о версии: 0 = Версия 2.5 2 = Версия 2.8 с информацией о палитре 3 = Версия 2.8 без информации о палитре 5 = Версия 3.0
2	Encoding	1	1 = PCX кодирование длинными сериями
3	Bits per pixel	1	Число бит на пиксел в слое
4	Window	8	Размеры изображения (Xmin, Ymin) – (Xmax, Ymax) в пикселах включительно
12	Hres	2	Горизонтальное разрешение создающего устройства
14	Vres	2	Вертикальное разрешение создающего устройства
16	Colormap	48	Набор цветовой палитры (см. далее текст)
64	Reserved	1	
65	NPlanes	1	Число цветовых слоев
66	Bytes per Line	2	Число байт на строку в цветовом слое (для PCX-файлов всегда должно

			быть четным)
68	Palette Info	2	Как интерпретировать палитру: 1 = цветная/черно-белая, 2 = градации серого
70	Filler	58	Заполняется нулями до конца заголовка

Все переменные длины 2 являются целыми.

Декодирование файлов в формате PCX

Сначала определяется размер изображения, для этого вычисляют [XSIZE = Xmax - Xmin + 1] и [YSIZE = Ymax - Ymin + 1]. Затем вычисляют, сколько байтов требуется для сохранения одной несжатой строки развертки изображения:

TotalBytes = NPlanes * BytesPerLine

Т.к. всегда используется целое число байтов, возможно существование неиспользуемых данных в конце каждой строки развертки. TotalBytes показывает, сколько памяти должно быть доступно для декодирования каждой строки развертки, включая неиспользуемую информацию на правом конце каждой строки.

Далее выполняется собственно декодирование, читается первый байт данных из файла. Если два старших бита этого байта равны 1, оставшиеся шесть битов показывают, сколько раз следует повторить следующий байт из файла. Если это не так, то этот байт сам является данными с повторителем равным 1. Далее продолжается декодирование до конца строки, ведя подсчет количества байтов, переданных в буфер вывода. В конце ка-ждой строки развертки имеет место остановка алгоритма кодирования, но ее не существует при переходе от одного слоя к другому. Когда строка сформирована полностью, в конце каждого слоя внутри строки возможно наличие лишних данных. Для нахождения этого остатка используются значения XSIZE и YSIZE. Если данные являются многослойными, то BytesPerLine показывает, где заканчивается каждый слой внутри строки развертки.

Продолжается декодирование оставшихся строк. В файле возможно наличие лишних строк с округлением на 8 или 16 строк.

Замечание

В конце каждой строки развертки предусматривается остановка алгоритма кодирова-ния, т.е. предполагается, что каждая строка развертки кодируется независимо. Но это не всегда так (файлы в этом формате пишет не только ZSoft Corporation !). Конец каждой строки лучше все-таки фиксировать по заполнении ее буфера. Это, безусловно, чуть усложнит программу декодирования, но сделает ее более гибкой.

Описание информации о палитре

Информация о 16-цветной палитре (EGA/VGA). Информация о палитре запоминается в одном из двух различных форматов. В стандартном формате RGB (IBM EGA, IBM VGA) данные запоминаются в 16 тройках. Каждая тройка состоит из 3 байтов со значениями красного (Red), зеленого (Green) и синего (Blue) цветов. Значения могут находиться в диапазоне 0-255, и поэтому необходима их интерпретация в формат используемого оборудования. Например, на IBM EGA существуют 4 возможных уровня RGB для каждого цвета. Поскольку 256/4 = 64, то ниже приведен список соответствия цветовых значений и уровней:

Значение	Уровень
0-63	0
64-127	1
128-192	2
193-254	3

Информация о 256-цветной палитре VGA. 256-цветная палитра форматируется и интерпретируется точно так же, как 16-цветная. Палитра (число цветов * 3 байта длины) добавляется в конец PCX файла и ей предшествует байт с десятичным значением 12. Для определения палитры VGA BIOS достаточно разделить прочитанные значения цветов на 4.

Для доступа к 256-цветной палитре следует:

1. Прочитать в заголовке поле Version. Если оно равно 5, палитра должна быть. Или прочитать в заголовке поле Bits per pixel. Если оно равно 8, 256-цветная палитра должна быть.

2. Прочитать конец файла и отсчитать назад 769 байт. Найденное вами значение должно равняться 12, что указывает на присутствие 256-цветной палитры.

В пакете программ PCX Programmer's Toolkit фирмы Genus Microprogramming принят другой способ хранения 256-цветной палитры. Отличие состоит в том, что палитре предшествует байт с десятичным значением 10 (вместо 12), а значения цветовых компонент занимают младшие 6 битов в каждом из байтов цветовой тройки (т.е. изменяются от 0 до 63, как это принято в VGA BIOS).

TIFF, TIF (Tagged Image File Format)

Изначально разработан компанией Aldus для своего графического редактора PhotoStyler. Как универсальный формат для хранения растровых изображений, TIFF достаточно широко используется, в первую очередь, в издательских системах, требующих изображения наилучшего качества. Благодаря своей совместимости с большинством профессионального ПО для обработки изображений, формат TIFF очень удобен при переносе изображений между компьютерами различных типов (например, с PC на Mac и обратно). Если PCX - один из самых простых для декодирования форматов растровой графики, то TIFF (Tagged Image File Format, формат файлов изображения, снабженных тегами) - один из самых сложных. Файлы TIFF имеют расширение TIFF. Каждый файл начинается 8-байт заголовком файла изображения (IFH), важнейший элемент которого - каталог файла изображения (Image File Directory, IFD) - служит указателем к структуре данных. IFD представляет собой таблицу для идентификации одной или нескольких порций данных переменной длины, называемых тегами; теги хранят информацию об изображении. В спецификации формата файлов TIFF определено более 70 различных типов тегов. Например, тег одного типа хранит информацию о ширине изображения в пикселах, другого - информацию о его высоте. В теге третьего типа хранится таблица цветов (при необходимости), а тег четвертого типа содержит сами данные растрового массива. Изображение, закодированное в файле TIFF, полностью определяется его тегами, и этот формат файла легко расширяется, поскольку для придания файлу дополнительных свойств достаточно лишь определить дополнительные типы тегов.

Так что же делает TIFF столь сложным? С одной стороны, составление программ, различающих все типы тегов, - это непростое дело. В большинстве программ для чтения файлов TIFF реализуется только подмножество тегов, именно поэтому созданный одной программой файл TIFF иногда не может быть прочитан другой. Кроме того, программы, создающие файлы TIFF, могут определять собственные типы тегов, имеющие смысл только для них. Программы чтения файлов TIFF могут пропускать непонятные для них теги, но всегда существует опасность, что это повлияет на внешний вид изображения.

Еще одна сложность заключается в том, что файл TIFF может содержать несколько изображений, каждому из которых сопутствуют собственный IFD и набор тегов. Данные растрового массива в файле TIFF могут сжиматься с использованием любого из нескольких методов, поэтому в надежной программе для чтения файлов TIFF должны быть средства распаковки RLE, LZW (LempelZivWelch) и несколько других. Ситуацию еще больше ухудшает то обстоятельство, что пользование программами распаковки LZW должно осуществляться в соответствии с лицензионным соглашением с фирмой Unisys Corp. на право пользования алгоритмом LZW и часто за плату. В результате даже самые лучшие программы считывания TIFF нередко "сдаются", когда сталкиваются со сжатым по методу LZW изображением.

Несмотря на свою сложность, файловый формат TIFF остается одним из лучших для передачи растровых массивов с одной платформы на другую благодаря своей универсальности, позволяющей кодировать в двоичном виде практически любое изображение без потери его визуальных или каких-либо иных атрибутов.

PSD (Photoshop)

Формат Adobe Photoshop, отличается возможностью хранения слоев (layers). Удобен только для обработки в Photoshop и для хранения исходника для редактирования в будущем.

RAW (RAW Image Data)

Формат разработан для цифровых фотоаппаратов. Это точная копия картинки, запечатленной на матрице во время съемки, представляет из себя три фотографии, снятые в красных, синих и зеленых цветах.
Расширения RAW-файлов у разных производителей могут отличаться, и их далеко не всегда получается открыть с помощью программ для обработки изображений.
Хотя если камера поддерживает сохранение RAW, то, как правило, к ней в комплекте прилагается какая-нибудь программа для обработки файлов этого формата.
В настоящее время корпорацией Adobe предложен формат DNG (Digital Negative Specification), который создан для того, чтобы облегчить жизнь производителям средств для работы с графикой.

Некоторые компании (Leica и Pentax) уже включили DNG в свои камеры, однако большинство поставщиков камер всё-таки продолжают использовать свои форматы.

Расширения		формата	RAW
.dng	-	Adobe	(универсальный)
.crw	.cr2	-	Canon
.raf	-		Fuji
.kdc	-		Kodak
.mrw	-		Minolta
.nef	-		Nikon
.orf	-		Olympus
.ptx	.pef	-	Pentax
.x3f	-		Sigma

.arw - Sony

EPS (EncapsulatedPostScript)

Отдельного обсуждения достоин формат EPS (EncapsulatedPostScript). Он представляет собой описание изображения на языке PostScript, предпочтительном для полиграфических целей. В рамках данного формата возможно хранение векторной и растровой графики, шрифтов, контуров обтравки, кривых калибровок и т. д. Как и сам язык PostScript, формат EPS является универсальным форматом описания не только растровых и векторных изображений, но и текстовой информации.

Если изображение в формате EPScодержит и векторные данные, то при его импортировании Photoshop растрирует их. Требуемые параметры изображения задаются в диалоговом окне **Rasterize Generic EPS Format** (Растеризация формата EPS).

В полях **Width** (Ширина) и **Height** (Высота) задаются геометрические размеры растрированного изображения, а в поле **Resolution** (Разрешение) — его разрешение. Флажок **Constrain Proportions** (Сохранять пропорции) фиксирует соотношение сторон, предотвращая искажения. Если он установлен, то достаточно ввести только один из размеров, а второй будет вычислен автоматически. В списке **Mode** (Режим) устанавливается тип изображения. Это может быть полутоновое изображение или полноцветное в моделях RGB, Lab или CMYK. Четкие контуры векторных объектов в EPS-файлах при растрировании требуют сглаживания, иначе на них могут проявиться неровности, "зазубрины". Сглаживание включается флажком **Antialiased** (Сглаживание).

Изображение в формате EPS можно не только открыть как новый документ, но и поместить в существующий с помощью команды **Place** (Поместить) из меню **File** (Файл). В этом случае оно тоже растрируется Photoshop, но без указания цветовой модели и разрешения. Очевидно, что в указании этих параметров нет необходимости, поскольку модель и разрешение полностью определяются документом, в который вставляется EPS-изображение. Единственное, что нужно задать при размещении, — это размер импортируемого рисунка. Photoshop позволяет это сделать интерактивно, что гораздо удобнее численного ввода в диалоговом окне.

Команда **Place** (Поместить) открывает стандартное диалоговое окно, в котором необходимо выбрать загружаемый файл. Затем модуль импорта интерпретирует содержимое файла и размещает рисунок посередине документа в том размере, который определен в исходном EPS-файле. Чтобы скорректировать размер, если это необходимо, воспользуйтесь манипуляторами на габаритной рамке рисунка. Когда вы придадите ему требуемый размер, сделайте двойной щелчок мышью внутри рамки. Photoshop растеризует рисунок и поместит его на отдельном слое.

При сохранении файлов в формате EPS тоже необходимо указать несколько важных параметров. Для этого предназначено диалоговое окно **EPS Options**(Параметры EPS), изображенное на рис. 2.1.

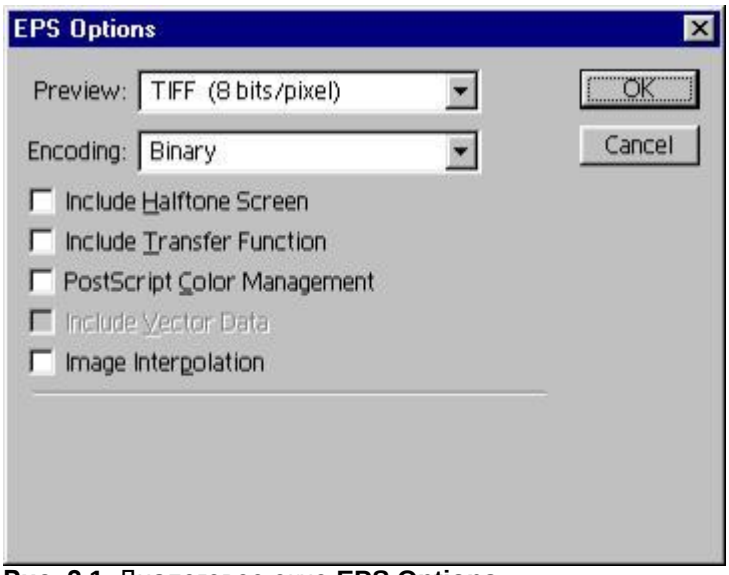


Рис. 2.1. Диалоговое окно **EPS Options**

В списке **Preview** (Просмотр) выберите глубину цвета растрового TIFF-изображения, включаемого в EPS-файл для просмотра его содержимого в издательских системах.

В списке **Encoding** (Кодировка) задайте способ кодирования: ASCII, Binary или JPEG. Наиболее универсальным вариантом является ASCII. Двоичное кодирование существенно сокращает размер файла, но в ущерб совместимости с другими приложениями и устройствами печати. Безопасно использовать двоичное кодирование можно только на компьютерах Macintosh. Наиболее компактные файлы, как и следовало бы ожидать, дает сжатие по алгоритму JPEG. Для него вы можете выбрать в том же списке одну из четырех градаций фактора качества.

Флажок **Include Halftone Screen** (Включить параметры растра) включает сохранение параметров растрирования изображения. В общем случае этого делать не надо.

Флажок **Include Transfer Function** (Включить функцию передачи) внедряет в сохраненный файл функцию переноса для принтера. Если вы используете систему управления цветом или просто не вводили функцию переноса *(см. гл. 10),* то сбросьте флажок.

Внедрение в EPS-файл цветовых профилей устанавливается флажком **PostScript Color Management** (Управление цветом PostScript). Учтите, что только PostScriptLevel 3 поддерживает управление цветом для изображений в модели CMYK.

Флажок **Include Vector Data** (Включать векторные данные) заставляет Photoshop помещать в файл EPS векторные представления фигур и шрифтов. При повторном открытии EPS-файла в программе Photoshop эти элементы, тем не менее, все равно будут растрирование

Флажок **Image Interpolation** (Интерполяция изображения) позволяет несколько улучшить качество печати изображений с низким разрешением за счет сглаживания.

Большинство ведущих специалистов-графиков, имеющих дело с алгоритмом LZW, сталкиваются с аналогичными юридическими проблемами при использовании популярного межплатформенного формата файлов растровой графики GIF (Graphics Interchange Format - формат обмена графическими данными, произносится "джиф"), разработанного компанией CompuServe. Обычно для имени файлов GIF используется расширение GIF, и тысячи таких файлов можно получить в CompuServe.

JPEG (Joint Photographic Experts Group)

Формат файла JPEG (Joint Photographic Experts Group - Объединенная экспертная группа по фотографии, произносится "джейпег) был разработан компанией C-Cube Microsystems как эффективный метод хранения изображений с большой глубиной цвета, например, получаемых при сканировании фотографий с многочисленными едва уловимыми (а иногда и неуловимыми) оттенками цвета. Самое большое отличие формата JPEG от других рассмотренных здесь форматов состоит в том, что в JPEG используется алгоритм сжатия с потерями (а не алгоритм без потерь) информации. Алгоритм сжатия без потерь так сохраняет информацию об изображении, что распакованное изображение в точности соответствует оригиналу. При сжатии с потерями приносится в жертву часть информации об изображении, чтобы достичь большего коэффициента сжатия. Распакованное изображение JPEG редко соответствует оригиналу абсолютно точно, но очень часто эти различия столь незначительны, что их едва можно (если вообще можно) обнаружить.

Процесс сжатия изображения JPEG достаточно сложен и часто для достижения приемлемой производительности требует специальной аппаратуры. Вначале изображение разбивается на квадратные блоки со стороной размером 8 пиксел. Затем производится сжатие каждого блока отдельно за три шага. На первом шаге с помощью формулы дискретного косинусоидального преобразования фуры (DCT) производится преобразование блока 8x8 с информацией о пикселах в матрицу 8x8 амплитудных значений, отражающих различные частоты (скорости изменения цвета) в изображении. На втором шаге значения матрицы амплитуд делятся на значения матрицы квантования, которая смещена так, чтобы отфильтровать амплитуды, незначительно влияющие на общий вид изображения. На третьем и последнем шаге квантованная матрица амплитуд сжимается с использованием алгоритма сжатия без потерь.

Поскольку в квантованной матрице отсутствует значительная доля высокочастотной информации, имеющейся в исходной матрице, первая часто сжимается до половины своего первоначального размера или даже еще больше. Реальные фотографические изображения часто совсем невозможно сжать с помощью методов сжатия без потерь, поэтому 50%-ное сжатие следует признать достаточно хорошим. С другой стороны, применяя методы сжатия без потерь, можно сжимать некоторые изображения на 90%. Такие изображения плохо подходят для сжатия методом JPEG.

При сжатии методом JPEG потери информации происходят на втором шаге процесса. Чем больше значения в матрице квантования, тем больше отбрасывается информации из изображения и тем более плотно сжимается изображение. Компромисс состоит в том, что более высокие значения квантования приводят к худшему качеству изображения. При формировании изображения JPEG пользователь устанавливает показатель качества, величине которого "управляет" значениями матрицы квантования. Оптимальные показатели качества, обеспечивающие лучший баланс между коэффициентом сжатия и качеством изображения, различны для разных изображений и обычно могут быть найдены только методом проб и ошибок.

Форматы векторной графики

CDR (CorelDRAW)

Формат популярного CorelDRAW, являющимся неоспоримым лидером в классе векторных графических редакторов на платформе PC. Имея сравнительно невысокую устойчивость и проблемы с совместимостью файлов разных версий формата.

AI (Adobe Illustrator)

Являясь частью семейства Adobe, поддерживают практически все программы, так или иначе связанные с векторной графикой. Лучший посредник при передаче изображений из одной программы в другую, с PC на Macintosh и наоборот. Отличается наибольшей стабильностью и совместимостью с языком PostScript, на который ориентируются практически все издательско-полиграфические приложения.

WMF (Windows Metafile)

Еще один родной формат Windows, на сей раз векторный. Понимается практически всеми программами Windows, так или иначе связанными с векторной графикой.

EMF (Enhaced Metafile)

Подобный WMF.

Другие форматы

SWF (ShokWaveFlash)

Формат Flash, продукт компании «Macromedia», позволяющий разрабатывать интерактивные мультимедийные приложения. Сфера использования Flash различна, это могут быть игры, веб-сайты, CD презентации, баннеры и просто мультфильмы. При создании продукта можно использовать медиа, звуковые и графические файлы, можно создавать интерактивные интерфейсы и полноценные веб-приложения с использованием PHP и XML.

SVG (Scalable Vector Graphics)

Стандарт, рекомендованый World Wide Web Consortium для описания с помощью XML markup двумерной векторной и комбинированной векторно-растровой графики.
В браузере SVG-графика отрисовывается с помощью растровых механизмов. Поддержка полупрозрачностей в каждом слое, градиенты линейные, градиенты радиальные, визуальные эффекты (тени, отмывки, блестящие поверхности, текстуры (фактуры), паттерны любой конструкции, символы любой сложности).
SVG - это формат для двухмерной векторной графики - так определено в спецификации, но с

помощью добавления скрипта (а именно JavaScript) внутрь SVG файла можно создавать трехмерные анимированные изображения.
В SVG может быть встроено растровое изображение, к которому как и к любому другому объекту в SVG может быть применена трансформация, прозрачность и т.д.

ICO (Icon)

Иконка, в Интернете используется как символ сайта, логотип. Например, сейчас вы видите красный квадратик в адресной строке. Если вы добавите страничк нашго сайта в избранное (favorit), радом со ссылкой появится наша иконка, которая поможет быстро визуально находит ссылку на сайт. Собственно, это и есть главное предназначение иконки в Интернете.

EPS (Encapsulated PostScript)

Самый надежным и универсальным способом сохранения данных. Он использует упрощенную версию PostScript не может содержать в одном файле более одной страницы, не сохраняет ряд установок для принтера. Как и в файлы печати PostScript, в EPS записывают конечный вариант работы, хотя такие программы, как Adobe Illustrator и Adobe Photoshop могут использовать его как рабочий. EPS предназначен для передачи векторов и растра в издательские системы, создается почти всеми программами, работающими с графикой.

Лекция 3: Цвет в компьютерной графике

О ПРИРОДЕ СВЕТА И ЦВЕТА

Из курса элементарной физики известно, что белый свет представляет собой смесь цветов. Если пропустить луч белого света через призму, то он разложится на цветной спектр, который называется видимым. Цвета этого спектра условно классифицируют как красный, оранжевый, желтый, зеленый, голубой, синий, фиолетовый. Каждый из цветов представляет собой электромагнитное излучение определенного диапазона длин волн, как это приведено в таблице 3.

Табл.3 Спектральный состав видимого цвета.

Цвет	Длина волны (мм)
Красный	620-700
Оранжевый	590-620
Желтый	540-690
Зеленый	500-540
Голубой	470-500
Синий	430-470
Фиолетовый	400-430

Для нашего глаза каждый кусочек этого спектра обладает уникальными свойствами, которые называются *цветом.* Разные длины волн могут иметь разную интенсивность. Та длина волны, которая имеет максимальную интенсивность, называется *доминирующей.* Она и определяет в основном окраску цвета.

Свет имеет двойственную природу, обладая свойствами волны и частицы. Корпускулы света, называемые фотонами, излучаются источником света в виде волн, распространяющихся с постоянной скоростью порядка 300000 км/сек. Волны имеют гребни и впадины. В качестве характеристики световых волн используют расстояние между двумя гребнями, которая называется длиной волны. Измеряется длина волны в метрах или ангстремах (1 ангстрем = 10^{-8} м.). В качестве второй характеристики используют амплитуду, которая определяется расстоянием между гребнем и впадиной. Амплитуда определяет интенсивность света.

Свет как физическое явление представляет собой поток электромагнитных волн различной длины и амплитуды. Глаз человека, будучи сложной оптической системой, воспринимает эти волны в диапазоне длин приблизительно от 350 до 780 нм. Свет воспринимается либо непосредственно от источника, например, от осветительных приборов, либо как отраженный от поверхностей объектов или преломленный при прохождении сквозь прозрачные и полупрозрачные объекты. Цвет - это характеристика восприятия глазом электромагнитных волн разной длины, поскольку именно длина волны определяет для глаза видимый цвет. Амплитуда, определяющая энергию волны

(пропорциональную квадрату амплитуды), отвечает за яркость цвета. Таким образом, само понятие цвета является особенностью человеческого "видения" окружающей среды.

УСТРОЙСТВО ГЛАЗА

.

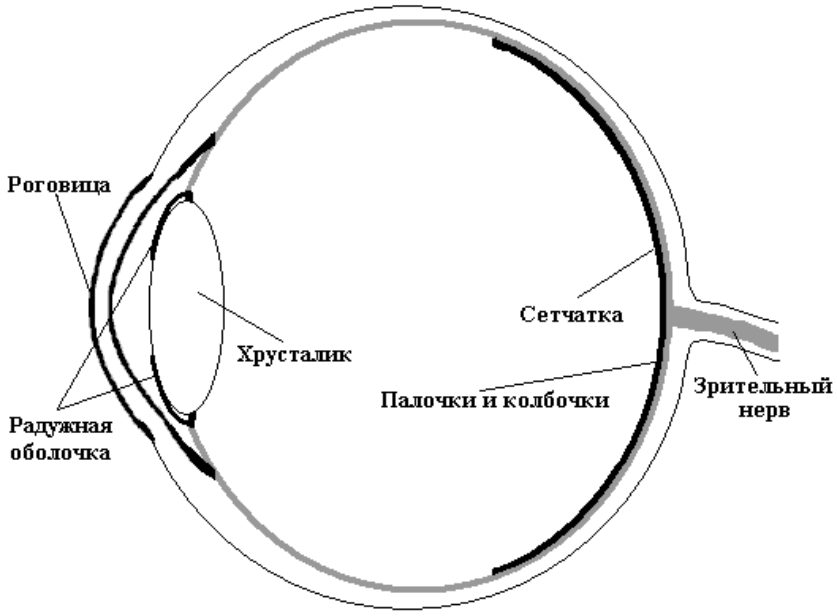


Рис. 3.1. Глаз человека

На рис. 3.1 схематически изображен глаз человека. Фоторецепторы, расположенные на поверхности сетчатки, играют роль приемников света. Хрусталик - это своеобразная линза, формирующая изображение, а радужная оболочка исполняет роль диафрагмы, регулируя количество света, пропускаемого внутрь глаза. Чувствительные клетки глаза неодинаково реагируют на волны различной длины. Интенсивность света есть мера энергии света, воздействующего на глаз, а яркость - это мера восприятия глазом этого воздействия. Интегральная кривая спектральной чувствительности глаза приведена на рис. 3.2; это стандартная кривая Международной комиссии по освещению (МКО, или CIE - Comission International de l'Eclairage).

Фоторецепторы подразделяются на два вида: палочки и колбочки. Палочки являются высокочувствительными элементами и работают в условиях слабого освещения. Они нечувствительны к длине волны и поэтому не "различают" цвета. Колбочки же, наоборот, обладают узкой спектральной кривой и "различают" цвета. Палочек существует только один тип, а колбочки подразделяются на три вида, каждый из которых чувствителен к определенному диапазону длин волн (длинные, средние или короткие.) Чувствительность их также различна.

На рис. 3.3 представлены кривые чувствительности колбочек для всех трех видов. Видно, что наибольшей чувствительностью обладают колбочки, воспринимающие цвета зеленого спектра, немного слабее - "красные" колбочки и существенно слабее - "синие".

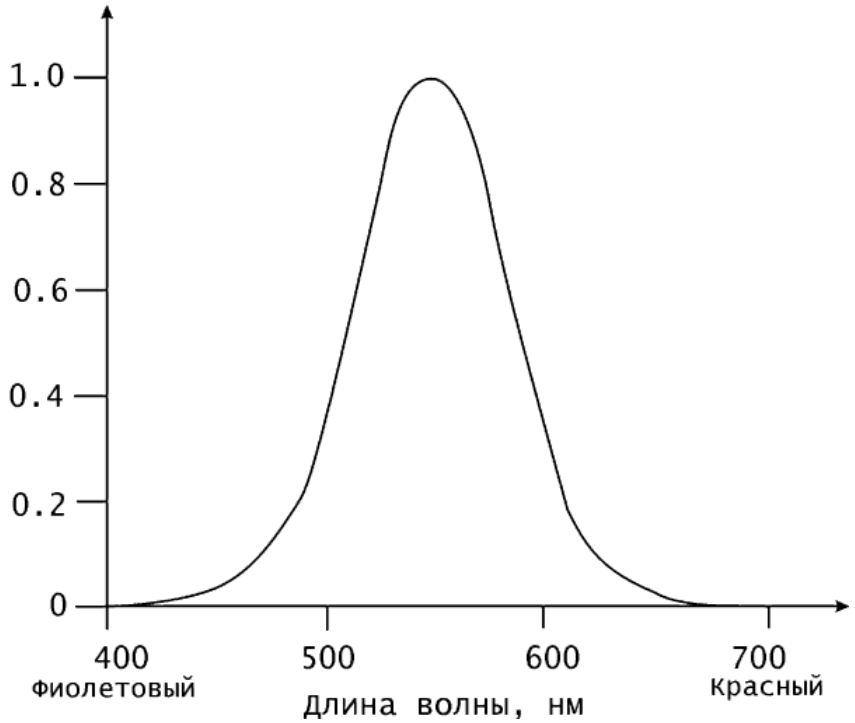


Рис. 3.2. Интегральная кривая спектральной чувствительности глаза

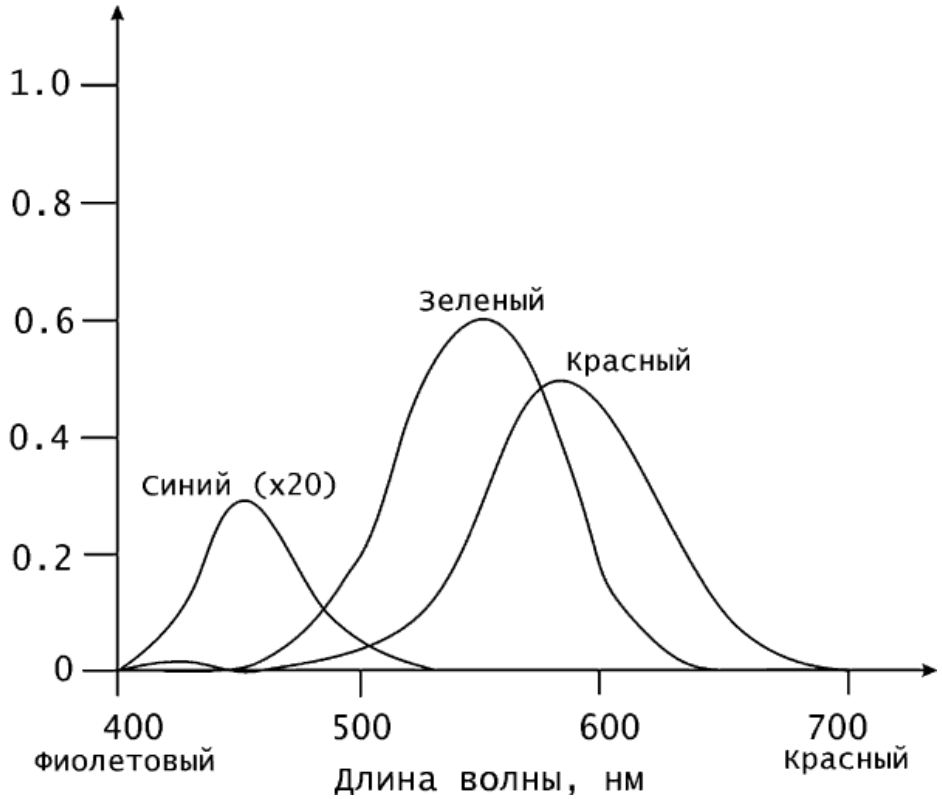


Рис. 3.3. Кривые чувствительности различных рецепторов

Таким образом, если функция $C(\lambda)$ характеризует спектральное разложение светового излучения от некоторого источника (рис. 3.4), т. е. распределение интенсивности по длинам волн, то три типа колбочек будут посылать в мозг сигналы R, G, B (красный, зеленый, синий), мощность которых определяется интегральными соотношениями

$$R = \int C(\lambda) S_R(\lambda) d\lambda,$$

$$G = \int C(\lambda) S_G(\lambda) d\lambda,$$

$$B = \int C(\lambda) S_B(\lambda) d\lambda,$$

где S_R, S_G, S_B - функции чувствительности соответствующих типов колбочек.

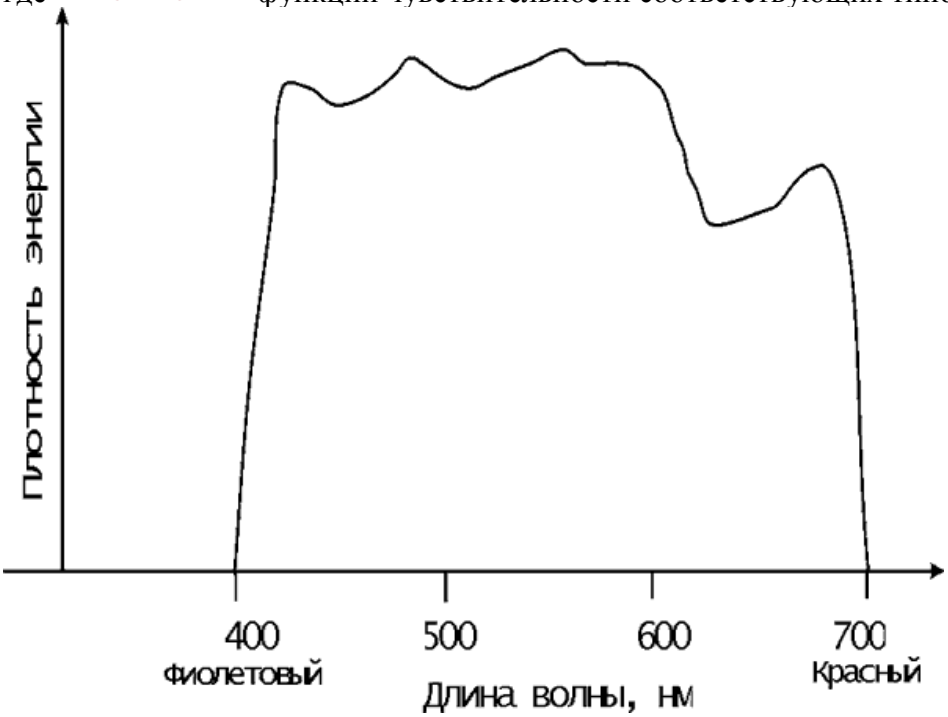


Рис. 3.4. Характерная спектральная кривая

Если воспринимаемый свет содержит все видимые длины волн в приблизительно равных количествах, то он называется ахроматическим и при максимальной интенсивности воспринимается как белый, а при более низких интенсивностях - как оттенки серого цвета. Интенсивность отраженного света удобно рассматривать в диапазоне от 0 до 1, и тогда нулевое значение будет соответствовать черному цвету. Если же свет содержит длины волн в неравных пропорциях, то он является хроматическим. Объект, отражающий свет, воспринимается как цветной, если он отражает или пропускает свет в узком диапазоне длин волн. Точно так же и источник света воспринимается как цветной, если он испускает волны в узком диапазоне длин. При освещении цветной поверхности цветным источником света могут получаться довольно разнообразные цветовые эффекты. Цветовой график МКО

Трехмерная природа восприятия цвета позволяет отображать его в прямоугольной системе координат. Любой цвет можно изобразить в виде вектора, компонентами которого являются относительные веса красного, зеленого и синего цветов, вычисленные по формулам

$$r = \frac{R}{R+G+B}, \quad g = \frac{G}{R+G+B}, \quad b = \frac{B}{R+G+B}.$$

Поскольку эти координаты в сумме всегда составляют единицу, а каждая из координат лежит в диапазоне от 0 до 1, то все представленные таким образом точки пространства будут лежать в одной плоскости, причем только в треугольнике, отсекаемом от нее положительным октантом системы координат (рис. 3.5а). Ясно, что при таком представлении все множество точек этого треугольника можно описать с помощью двух координат, так как третья выражается через них посредством соотношения $b = 1 - r - g.$

Таким образом, мы переходим к двумерному представлению области, т.е. к проекции области на плоскость (рис. 3.5б).

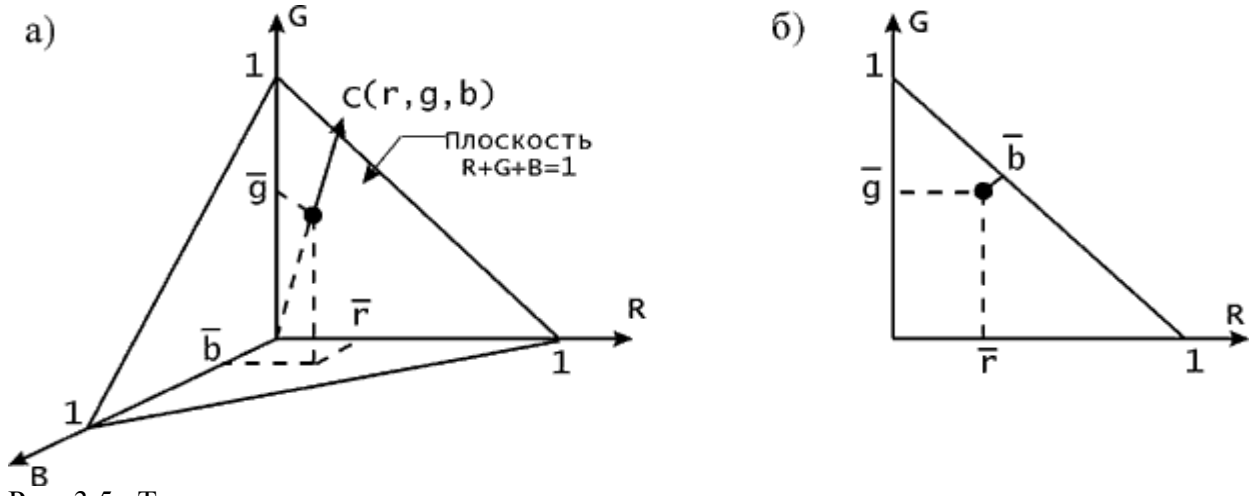


Рис. 3.5. Трехмерное цветовое пространство

С использованием такого преобразования в 1931 г. были выработаны международные стандарты определения и измерения цветов. Основой стандарта стал так называемый двумерный цветовой график МКО. Поскольку, как показали физические эксперименты, сложением трех основных цветов можно получить не все возможные цветовые оттенки, то в качестве базисных были выбраны другие параметры, полученные на основе исследования стандартных реакций глаза на свет. Эти параметры - X, Y, Z - являются чисто теоретическими, поскольку построены с использованием отрицательных значений основных составляющих цвета. Треугольник основных цветов был построен так, чтобы охватывать весь спектр видимого света. Кроме того, равное количество всех трех гипотетических цветов в сумме дает белый цвет. Координаты цветности строятся так же, как и в приведенной выше формуле:

$$x = \frac{X}{X+Y+Z}, \quad y = \frac{Y}{X+Y+Z}, \quad z = \frac{Z}{X+Y+Z}, \quad x+y+z=1$$

При проекции этого треугольника на плоскость получается цветовой график МКО. Но координаты цветности определяют только относительные количества основных цветов, не задавая яркости результирующего цвета. Яркость можно задать координатой Y, а X, Z определить исходя из величин (x, y, Y), по формулам

$$X = \frac{Y}{y}x, \quad Z = \frac{Y}{y}(1-x-y).$$

Цветовой график МКО приведен на рис. 3.6. Область, ограниченная кривой, охватывает весь видимый спектр, а сама кривая называется линией спектральных цветностей. Числа, проставленные на рисунке, означают длину волны в соответствующей точке. Точка C, соответствующая полуденному освещению при сплошной облачности, принята в качестве опорного белого цвета.

Цветовой график удобен для целого ряда задач. Например, с его помощью можно получить дополнительный цвет: для этого надо провести луч от данного цвета через опорную точку до пересечения с другой стороной кривой (цвета являются дополнительными друг к другу, если при сложении их в соответствующей пропорции получается белый цвет). Для определения доминирующей длины волны какого-либо цвета также проводится луч из опорной точки до пересечения с данным цветом и продолжается до пересечения с ближайшей точкой линии цветностей.

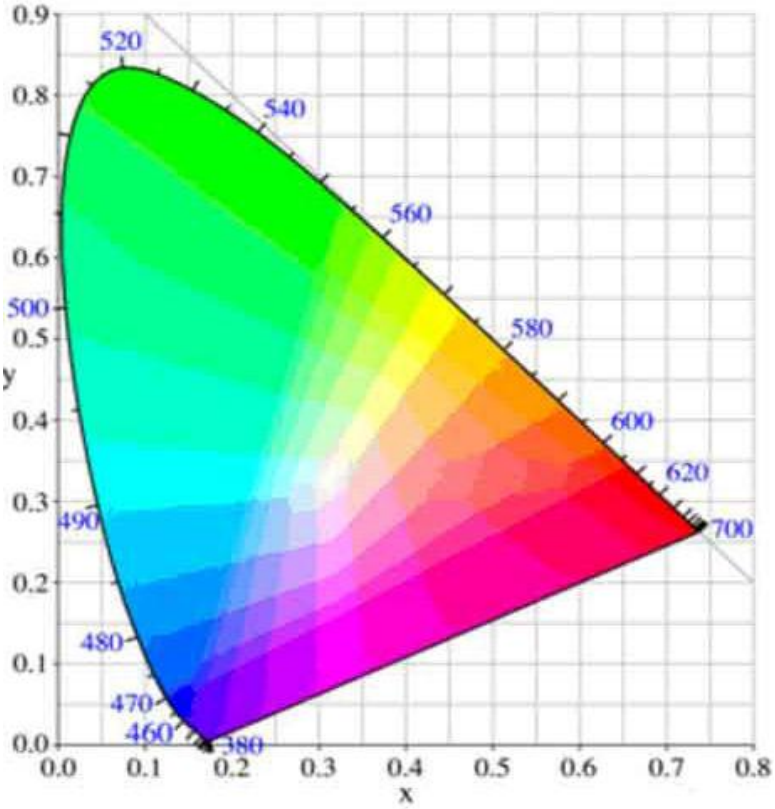


Рис. 3.6. Цветовой график МКО. На контуре указаны длины волн в нанометрах

Для смешения двух цветов используются законы Грассмана. Пусть два цвета заданы на графике МКО координатами $D_1 = (x_1, y_1, Y_1)$ и $D_2 = (x_2, y_2, Y_2)$. Тогда смешение их дает цвет $D_{12} = (x_1 + x_2, y_1 + y_2, z_1 + z_2)$. Если ввести обозначения $t_1 = \frac{Y_1}{y_1}, t_2 = \frac{Y_2}{y_2}$, то получим координаты цветности смеси

$$x_{12} = \frac{x_1 t_1 + x_2 t_2}{t_1 + t_2}, \quad y_{12} = \frac{y_1 t_1 + y_2 t_2}{t_1 + t_2}, \quad Y_{12} = Y_1 + Y_2.$$

Координаты МКО являются точным стандартом определения цвета. Но в различных областях, имеющих дело с цветом, есть свой подход к его моделированию. В частности, может использоваться другой набор основных цветов. Компьютерная графика опирается на систему RGB, поэтому представляет интерес переход между этими двумя наборами цветов (иными словами, преобразование координат цветности).

Понятие цвета

Цвет чрезвычайно важен в компьютерной графике как средство усиления зрительного впечатления и повышения информационной насыщенности изображения. Ощущение цвета формируется человеческим мозгом в результате анализа светового потока, попадающего на сетчатку глаза от излучающих или отражающих объектов. Считается, что цветовые рецепторы (колбочки) подразделяются на три группы, каждая из которых воспринимает только единственный цвет — красный, зеленый или синий. Нарушения в работе любой из групп приводит к явлению *дальтонизма* — искаженного восприятия цвета.

Световой поток формируется излучениями, представляющими собой комбинацию трех «чистых» спектральных цветов (красный, зеленый, синий — КЗС) и их производных (в англоязычной литературе используют аббревиатуру *RGB — Red, Green, Blue).* Для излучающих объектов характерно *аддитивное цветовоспроизведение* (световые излучения суммируются), для отражающих объектов — *субтрактивное цветовоспроизведение* (световые излучения вычитаются). Примером объекта первого типа является электронно-лучевая трубка монитора, второго типа — полиграфический отпечаток.

Физические характеристики светового потока определяются параметрами *мощности, яркости* и *освещенности.* Визуальные параметры ощущения цвета характеризуются *светлотой,* то есть различимостью участков, сильнее или слабее отражающих свет. Минимальную разницу между яркостью различимых по светлоте объектов называют *порогом.* Величина порога пропорциональна логарифму отношения яркостей. Последовательность оптических характеристик объекта

(расположенная по возрастанию или убыванию), выраженная в оптических плотностях или логарифмах яркостей, составляет *градацию* и является важнейшим инструментом для анализа и обработки изображения.

Для точного цветовоспроизведения изображения на экране монитора важным является понятие *цветовой температуры,* В классической физике считается, что любое тело с температурой, отличной от 0 градусов *по шкале Кельвина,* испускает излучение. С повышением температуры спектр излучения смещается от инфракрасного до ультрафиолетового диапазона, проходя через оптический.

Для идеального черного тела легко находится зависимость между длиной волны излучения и температурой тела. На основе этого закона, например, была дистанционно вычислена температура Солнца — около 6500 К. Для целей правильного цветовоспроизведения характерна обратная задача. То есть, монитор с выставленной цветовой температурой 6500 К должен максимально точно воспроизвести спектр излучения идеального черного тела, нагретого до такой же степени. Таким образом, стандартные значения цветовых температур используют в качестве всеобщего эталона, обеспечивающего одинаковое цветовоспроизведение на разных излучающих устройствах.

На практике зрение человека непрерывно подстраивается под спектр, характерный для цветовой температуры источника излучения. Например, на улице в яркий солнечный день цветовая температура составляет около 7000 К. Если с улицы зайти в помещение, освещенное только лампами накаливания (цветовая температура около 2800 К), то в первый момент свет ламп покажется желтым, белый лист бумаги тоже приобретет желтый оттенок. Затем происходит адаптация зрения к новому соотношению КЗС, характерному для цветовой температуры 2800 К, свет лампы и лист бумаги будут восприниматься как белые.

Насыщенность цвета показывает, насколько данный цвет отличается от монохроматического («чистого») излучения того же цветового тона. В компьютерной графике за единицу принимается насыщенность цветов спектральных излучений.

Ахроматические цвета (белый, серый, черный) характеризуется только светлотой. *Хроматические цвета* имеют параметры насыщенности, светлоты и цветового тона.

СПОСОБЫ ОПИСАНИЯ ЦВЕТА

В компьютерной графике применяют понятие *цветового разрешения* (другое название — *глубина цвета).* Оно определяет метод кодирования цветовой информации для ее воспроизведения на экране монитора. Для отображения черно-белого изображения достаточно двух бит (белый и черный цвета). Восьмиразрядное кодирование позволяет отобразить 256 градаций цветового тона. Два байта (16 бит) определяют 65 536 оттенков (такой режим называют *High Color).* При 24-разрядном способе кодирования возможно определить более 16,5 миллионов цветов (режим называют *True Color).*

С практической точки зрения цветовому разрешению монитора близко понятие *цветового охвата.* Под ним подразумевается диапазон цветов, который можно воспроизвести с помощью того или иного устройства вывода (монитор, принтер, печатная машина и прочие).

В соответствии с принципами формирования изображения аддитивным или суб-трактивным методами разработаны способы разделения цветового оттенка на составляющие компоненты, называемые *цветовыми моделями.* В компьютерной графике в основном применяют модели *RGB* и *HSB* (для создания и обработки аддитивных изображений) и *CMYK* (для печати копии изображения на полиграфическом оборудовании).

Цветовые модели расположены в трехмерной системе координат, образующей *цветовое пространство,* так как из *законов Грассмана* следует, что цвет можно выразить точкой в трехмерном пространстве.

Вопросы и упражнения

1. Расположите в убывающем порядке чувствительность рецепторов глаза к цветам: красный, зеленый, синий.
2. Что такое хроматический спектр?
3. Что такое ахроматический спектр?
4. Как осуществляется проекция трехмерного цветового пространства на плоскость?
5. Чем отличается цветовой график МКО от треугольной проекционной области цветового пространства?
6. Что такое дополнительный цвет?

Лекция 4:Цветовые модели

ЦВЕТОВЫЕ МОДЕЛИ, СИСТЕМЫ СООТВЕТСТВИЯ ЦВЕТОВ И РЕЖИМЫ

Субъективность в восприятии цвета при обработке изображений крайне нежелательна. Для обеспечения одинакового воспроизведения одного и того же цвета видеомониторами, принтерами и сканерами разных фирм-изготовителей необходимо наличие объективных измерительных систем, позволяющих установить однозначное определение цветовых координат. Для этих целей разработаны специальные средства, включающие:

- цветовые модели;
- системы соответствия цветов;
- цветовые режимы.

Цветовые модели

В основе создания цветовых моделей лежит использование универсальных языков, позволяющих реализовать способы точного описания цвета с помощью стандартных математических выражений. Без их помощи было бы невозможно выполнить ни один из этапов обработки цифровых изображений, включая сканирование, редактирование и печать.

В современных компьютерных программах манипуляции с цветом осуществляются с помощью *цветовых моделей* и *режимов.*

Цветовые модели (или *цветовые пространства)* предоставляют средства для концептуального и количественного описания цвета.

Режим — это способ реализации определенной цветовой модели в рамках конкретной графической программы.

Понятие цветовой модели

Цветовые модели (color model) используются для математического описания определенных цветовых областей спектра. Большинство компьютерных цветовых моделей основано на использовании трех основных цветов, что соответствует восприятию цвета человеческим глазом. Каждому основному цвету присваивается определенное значение цифрового кода, после чего все остальные цвета определяются как комбинации основных цветов. Именно такой подход используют художники при создании картины на базе ограниченной палитры цветов.

Несмотря на то, что цветовые модели позволяют представить цвет математически, такое представление всегда будет казаться несовершенным в силу отличия от нашего восприятия. Однако они удобны при использовании в компьютерных программах для однозначного определения выводимого цвета. Так, если послать на монитор цветовой сигнал R255 G000 B255, то на любом хорошо откалиброванном мониторе теоретически должен появиться один и тот же цвет (в данном случае пурпурный). Независимо оттого, что лежит в ее основе, любая модель должна удовлетворять трем требованиям:

1. реализовывать определения цвета некоторым стандартным способом, не зависящим от возможностей какого-либо конкретного устройства.
2. Точно задавать диапазон воспроизводимых цветов, поскольку ни одно множество цветов не является бесконечным.
3. Учитывать механизм восприятия цветов: излучение или отражение. Современные графические пакеты располагают развитым интерфейсом для выбора нужной цветовой модели и цвета внутри нее. Далее в этом разделе будет подробно рассмотрено большинство цветовых моделей, используемых в современных графических пакетах.

ЦВЕТОВАЯ МОДЕЛЬ CIE LAB

В 1920 году была разработана цветовая пространственная модель *CIELab (Communication Internationale de I'Edairage — международная комиссия по освещению. L,a,b* — обозначения осей координат в этой системе). Система является аппаратно независимой и потому часто применяется для переноса данных между устройствами, В модели *CIELab* любой цвет определяется светлотой (Z,) и хроматическими компонентами: параметром *а,* изменяющимся в диапазоне от зеленого до красного, и параметром *Ь,* изменяющимся в диапазоне от синего до желтого. Цветовой охват модели *CIELab* значительно превосходит возможности мониторов и печатных устройств, поэтому перед выводом изображения, представленного в этой модели, его приходится преобразовывать. Данная модель была разработана для согласования цветных фотохимических процессов с полиграфическими. Сегодня она является принятым по умолчанию стандартом для программы Adobe Photoshop.

ЦВЕТОВАЯ МОДЕЛЬ RGB

Цветовая модель *RGB* является аддитивной, то есть любой цвет представляет собой сочетание

в различной пропорции трех основных цветов — красного *(Red),* зеленого *(Green),* синего *(Blue).* Она служит основой при создании и обработке компьютерной графики, предназначенной для электронного воспроизведения (на мониторе, телевизоре). При наложении одного компонента основного цвета на другой яркость суммарного излучения увеличивается. Совмещение трех компонентов дает ахроматический серый цвет, который при увеличении яркости приближается к белому цвету. При 256 градационных уровнях тона черному цвету соответствуют нулевые значения *RGB, а* белому — максимальные, с координатами (255,255,255).

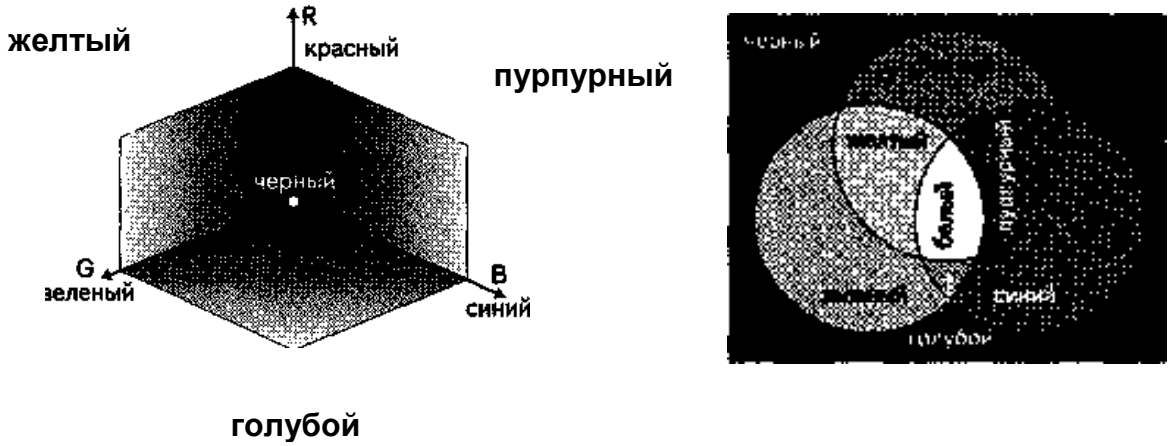


Рис. 4.1. Аддитивная цветовая модель RGB

ЦВЕТОВАЯ МОДЕЛЬ HSB

Цветовая модель *HSB* разработана с максимальным учетом особенностей восприятия цвета человеком. Она построена на основе цветового круга Манселла. Цвет описывается тремя компонентами: оттенком *(Hue),* насыщенностью *{Saturation)* и яркостью *{Brightness).* Значение цвета выбирается как вектор, исходящий из центра окружности. Точка в центре соответствует белому цвету, а точки по периметру окружности — чистым спектральным цветам. Направление вектора задается в градусах и определяет цветовой оттенок. Длина вектора определяет насыщенность цвета. На отдельной оси, называемой *ахроматической,* задается яркость, при этом нулевая точка соответствует черному цвету. Цветовой охват модели *HSB* перекрывает все известные значения реальных цветов.

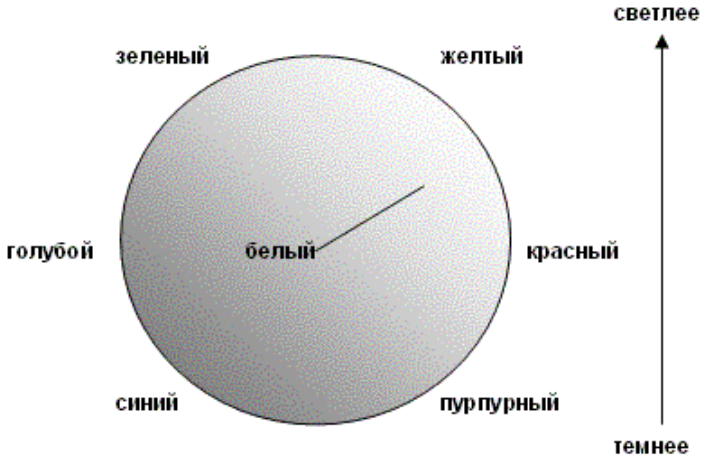


Рис. 4.2. Цветовая модель HSB

Модель *HSB* принято использовать при создании изображений на компьютере с имитацией приемов работы и инструментария художников. Существуют специальные программы, имитирующие кисти, перья, карандаши. Обеспечивается имитация работы с красками и различными полотнами. После создания изображения его рекомендуется преобразовать в другую цветовую

модель, в зависимости от предполагаемого способа публикации.

Цветовая модель CMYK, цветоделение

Цветовая модель *CMYK* относится к субтрактивным, и ее используют при подготовке публикаций к печати. Цветовыми компонентами *CMY* служат цвета, полученные вычитанием основных из белого:

голубой (cyan) = белый - красный = зеленый + синий;

пурпурный (magenta) = белый - зеленый = красный + синий;

желтый (yellow) = белый - синий = красный + зеленый.

Такой метод соответствует физической сущности восприятия отраженных от печатных оригиналов лучей. Голубой, пурпурный и желтый цвета называются *дополнительными,* потому что они дополняют основные цвета до белого. Отсюда вытекает и главная проблема цветовой модели *CMY* — наложение друг на друга дополнительных цветов на практике не дает чистого черного цвета. Поэтому в цветовую модель был включен компонент чистого черного цвета. Так появилась четвертая буква в аббревиатуре цветовой модели *CMYK (Cyan, Magenta, Yellow, blacK).*

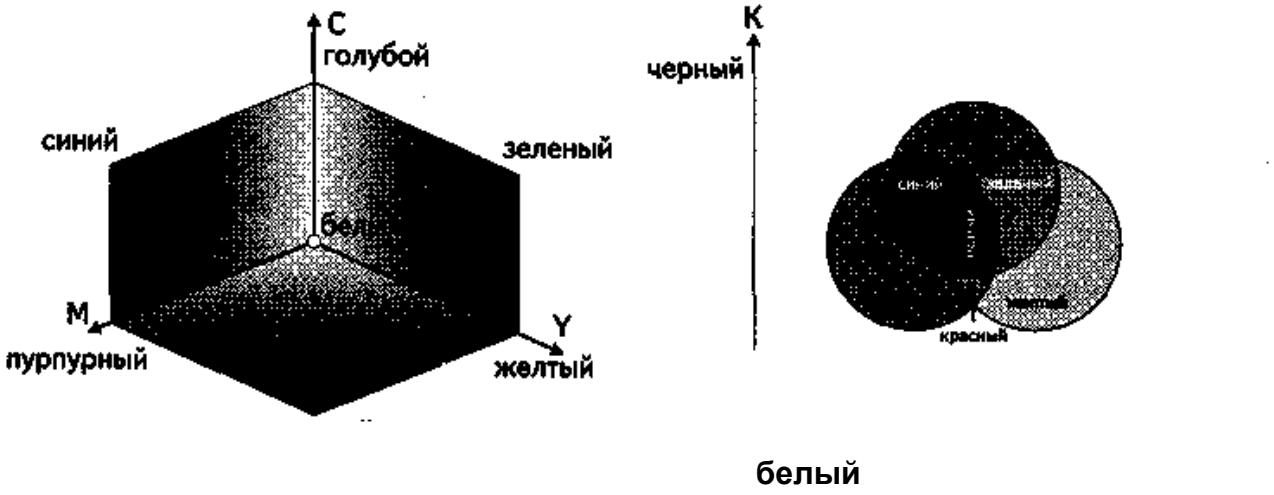


Рис. 4.3. Цветовая модель CMYK

Для печати на полиграфическом оборудовании цветное компьютерное изображение необходимо разделить на составляющие, соответствующие компонентам цветовой модели *CMYK.* Этот процесс называют *цветоделением.* В итоге получают четыре отдельных изображения, содержащих одноцветное содержимое каждого компонента в оригинале. Затем в типографии с форм, созданных на основе цветоделенных пленок, печатают многоцветное изображение, получаемое наложением цветов *CMYK.*

Цветовая палитра

Электронная *и/зетовая палитра* в компьютерной графике по предназначению подобна палитре художника, но включает гораздо большее число цветов. Электронная палитра состоит из определенного числа ячеек, каждая из которых содержит отдельный цветовой тон. Конкретная цветовая палитра соотносится с определенной цветовой моделью, так как ее цвета созданы на основе цветового пространства этой модели. Но если в цветовой модели возможно воспроизвести любой из описываемых ею цветов, цветовая палитра содержит ограниченный набор цветов, называемых *стандартными.*

Примером стандартных цветовых палитр являются наборы фирмы Pantone, ориентированные на полиграфическую публикацию изображений. Программы создания и обработки компьютерной графики, как правило, предоставляют на выбор несколько цветовых палитр в цветовых моделях *RGB, HSB, CIELab, CMYK.*

Состав цветовых палитр *RGB* зависит от выбранного цветового разрешения — 24, 16 или 8 бит. В последнем случае цветовая палитра называется *индексной,* потому что каждый цветовой оттенок кодируется одним числом, которое выражает не цвет пиксела, а *индекс* (номер) цвета. Таким образом, к файлу цветного изображения, созданного в индексной палитре, должна быть приложена сама палитра, так как программе обработки компьютерной графики неизвестно, какая именно палитра была использована.

Изображения, подготавливаемые для публикации в Интернете, принято создавать в так называемой *безопасной палитре* цветов. Она является вариантом рассмотренной выше индексной палитры. Но так как файлы изображений в Web-графике должны иметь минимальный размер, необходимо было отказаться от включения в их состав индексной палитры. Для этого была принята единая фиксированная палитра цветов; названная *«безопасной*,* то есть обеспечивающей правильное отображение цветов на любых устройствах (в программах), поддерживающих единую палитру. Безопасная палитра содержит всего 216 цветов, что связано с ограничениями, накладываемыми требованиями совместимости с компьютерами, не относящимися к классу *IBM PC.*

Способы описания цвета

В большинстве цветовых моделей для описания цвета используется трехмерная система координат. Она образует цветовое пространство, в котором цвет можно представить в виде точки с тремя координатами. Для оперирования цветом в трехмерном пространстве Г. Грассман вывел следующие три закона.

1. *Трехмерность природы цвета.* Глаз реагирует на три различные цветовые со- ставляющие. Примеры:

• красный, зеленый и синий цвета;

• цветовой тон (доминирующая длина волны), насыщенность (чистота) и яр- кость (светлость).

. *2. Четыре цвета всегда линейно зависимы,* то есть: **cC = rR+ gG+ bB,**

где c, r, g, b≠0 — весовые коэффициенты для каждой из составляющих цвета. Для смеси двух цветов $(cC)_1$, и $(cC)_2$, имеет место равенство:

$(cC)_1$, + $(cC)_2$, = $(rR)_1$, + $(gG)_1$, + $(bB)_1$, + $(rR)_2$, + $(gG)_2$, + $(bB)_2$,

которое свидетельствует о том, что цвет смеси излучений **C** зависит только от их цвета, но не от спектрального состава.

Следствие: если цвет C1 равен цвету C и цвет C2 тоже равен цвету C, то следует, что цвет C1 равен цвету C2 независимо от структуры спектров энергии цветов C, C1и C2.

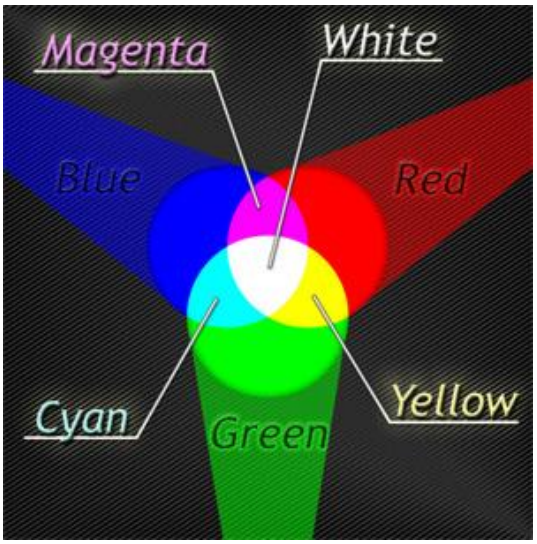
3. *Цветовое пространство непрерывно.* Если в смеси *трех* цветов один непрерыв-
но изменяется, а другие остаются постоянными, то цвет смеси будет меняться непрерывно.

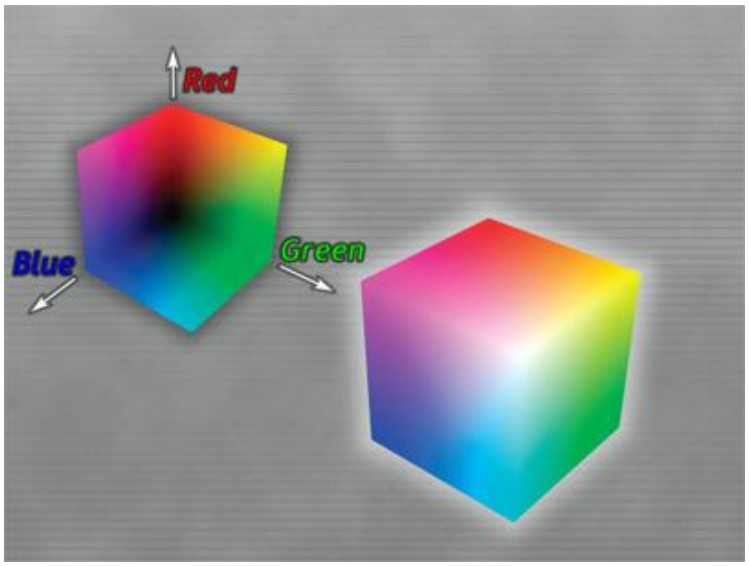
Цветовая Модель RGB

В модели RGB цвета описываются с помощью сложения трёх цветовых пучков - красного (Red), зелёного (Green), и синего (Blue). Их также называют цветовыми каналами модели RGB. При их попарном сложении получаются жёлтый (Yellow), голубой (Cyan), и светло-пурпурный (Magenta) цвета. При сложении всех трёх получается белый (White) цвет:

Каждый из базовых цветов может принимать интенсивность в диапазоне от 0 до 255. Полное количество цветов, представляемых этой моделью равно 256*256*256 = 16 777 216. Чёрный цвет получается, если интенсивность всех базовых цветов равна нулю. Белый цвет получается при их максимальной интенсивности (255), то есть, RGB - аддитивная модель.

Можно представить модель RGB как 3 прожектора (красный, зелёный и синий), которые светят на экран. Регулируя яркость таких прожекторов можно получить желаемый цвет на экране. Именно на таком принципе работают современные проекторы.



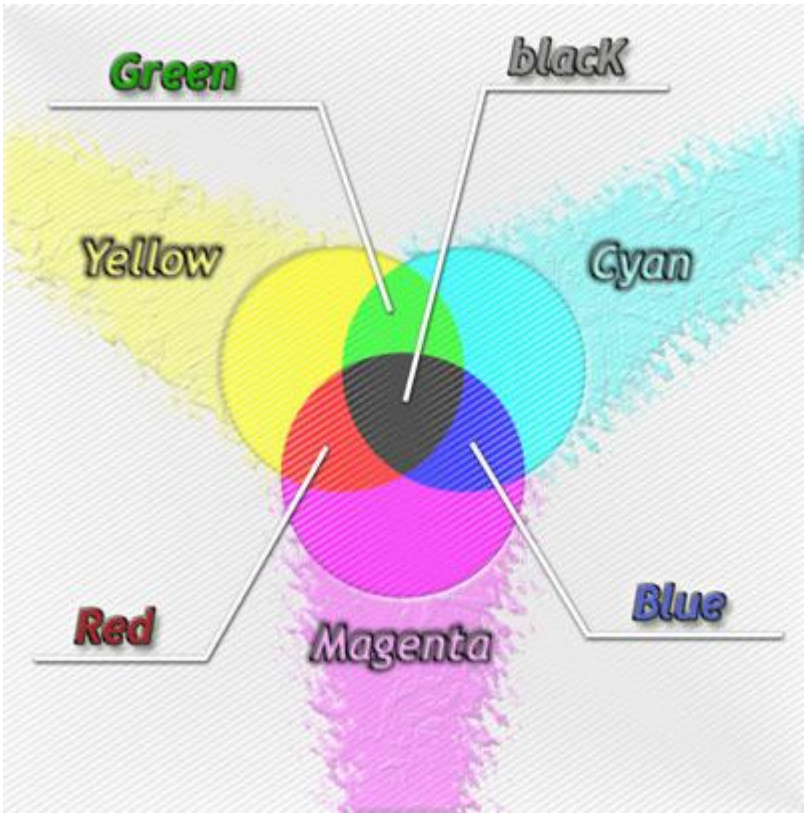


Модель RGB хороша для описания цветов, отображаемых мониторами и сканерами, ведь именно в них цвет получается путём смешения световых пучков. Она также используется для описания цветов на страницах Интернет в специальном шестнадцатеричном виде (#RRGGBB).

Цветовая Модель CMYK

В модели CMYK, в отличие от RGB, цвета смешиваются как краски: при отсутствии красок виден белый лист, после смешения всех красок максимальной интенсивности получается чёрный цвет. Стало быть, CMYK - субтрактивная модель. Базовые цвета CMY являются дополняющими к базовым цветам RGB, поэтому первые три базовых цвета модели CMYK можно получить путём вычитания из белого цветов RGB.

Цвета в CMYK не такие чистые, как в RGB. Базовые цвета модели CMYK - голубой (Cyan), светло-пурпурный (Magenta) и жёлтый (Yellow). При их попарном смешивании получаются фиолетовато-синий (Blue), оранжево-красный (Red) и синевато-зелёный (Green) цвета. При смешивании этих трёх базовых цветов получается грязно-серый цвет, а не чёрный (как если смешать такие же цветные краски на листе, получится грязь). Поэтому помимо трёх базовых цветов в этой модели есть также чёрный (Black) цвет. Это значит, что в CMYK используется 4 цветовых канала. Каждый из четырёх цветов может принимать значения от 0% до 100%.



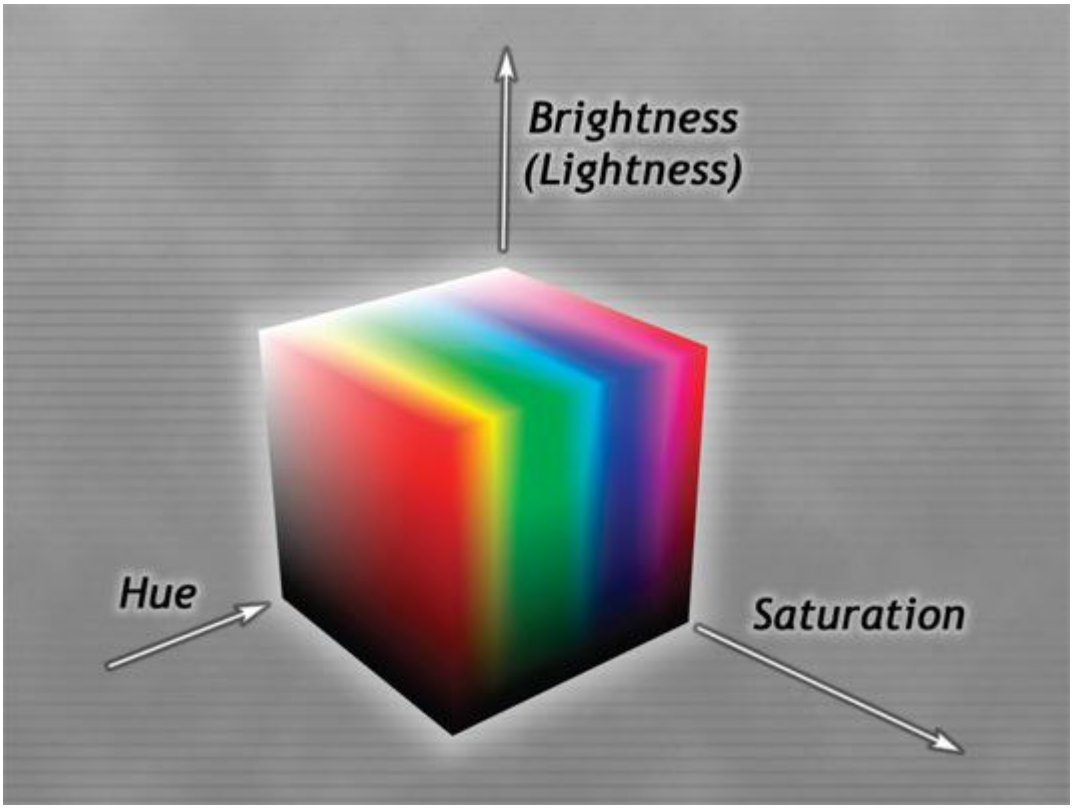
Несмотря на то, что в CMYK способна принимать 100*100*100*100 = 100 млн. значений, её цветовой диапазон меньше, чем у RGB (16 млн.) из-за перекрытия областей чёрной и цветной составляющей. Можно сказать, что в диапазон CMYK входят все тусклые цвета из RGB, а яркие - не все (особенно мало ярких синих, а также красных). Поэтому, если Вы работаете в CMYK, то некоторые цвета модели RGB не будут работать. Чтобы цвета моделей лучше соответствовали друг другу, требуется настроить так называемые цветовые профили (ICC profiles) для ваших устройств - монитора, принтера, сканера и тому прочих.

Модель CMYK следует использовать, если изображение предназначено для печати, именно таким образом цвета описываются в печатающих устройствах. Более того, если Вы делаете изображение в RGB, при печати оно всё равно преобразуется в CMYK. При этом могут возникнуть значительные цветовые искажения. Если же Вы не собираетесь печатать свои творения (или не уверены), то Вам всё же рекомендуется работать с изображением в модели RGB.

Цветовая Модель HSB

Модель HSB основывается не на базовых цветах, а на более естественных для восприятия понятиях: оттенок (Hue), насыщенность (Saturation) и яркость (Brightness) - всего 3 канала (яркость иногда называют не Brightness, а Lightness, тогда название модели не HSB, а HSL).

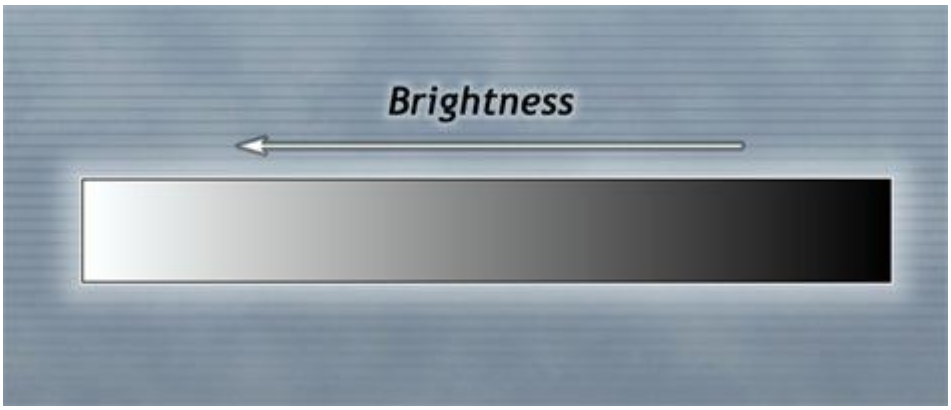
Снижение насыщенности аналогично добавлению белой краски на палитру, так же как снижение яркости - добавлению чёрной. Оттенок может принимать значения от 0° до 360°, а насыщенность и яркость - от 0% до 100%:



Модель HSB интуитивно лучше понятна, чем RGB или CMYK. Работая в ней легче найти нужный цвет. Графические файлы в изучаемых в учебнике программах не создаются в цветовом режиме HSB, он служит лишь для удобства. Все цвета из этой модели автоматически переводятся в рабочую модель (обычно RGB или CMYK). При этом количество цветов, доступных модели HSB определяется количеством цветов рабочей модели (за счёт того, что значения насыщенности и яркости могут быть выражены в долях процентов, а не только целыми значениями процентов).

Чёрно-белый режим (Grayscale)

Режим Grayscale - чёрно-белый, в нём есть информация только об уровне чёрного цвета. Ноль соответствует белому цвету, 255 - чёрному, остальное - градации серого:



Единственный канал Grayscale соответствует чёрно-белому изображению. Помимо чёрно-белых изображений Grayscale также используется при создании различных эффектов, масок

прозрачности, альфа каналов и при работе с выделением. Также, любой цветовой канал может быть представлен, как чёрно-белый.

Цветовые каналы (Color Channels) Каждую цветовую модель можно разложить на составляющие. В модели RGB, например можно выделить уровень красного, зелёного или синего цвета, приравняв остальные к нулю. Эти составляющие в каждой модели называются каналами.

В аддитивной модели выделение канала аналогично пропусканию света через соответствующий фильтр, то есть изображение канала R модели RGB будет похоже на то, что вы увидите, смотря через красное стекло.

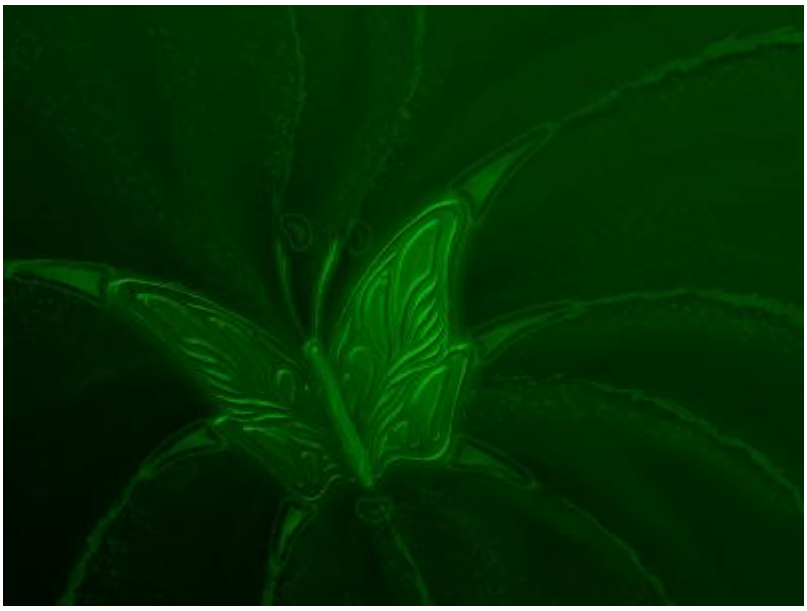
В субтрактивной модели CMYK каждый канал соответствуют количеству краски, затрачиваемой принтером на его печать, то есть если в вашем струйном принтере засохнет пурпурная и жёлтая краска и кончится чёрная, то вы увидите примерно то же изображение, что и в канале C.

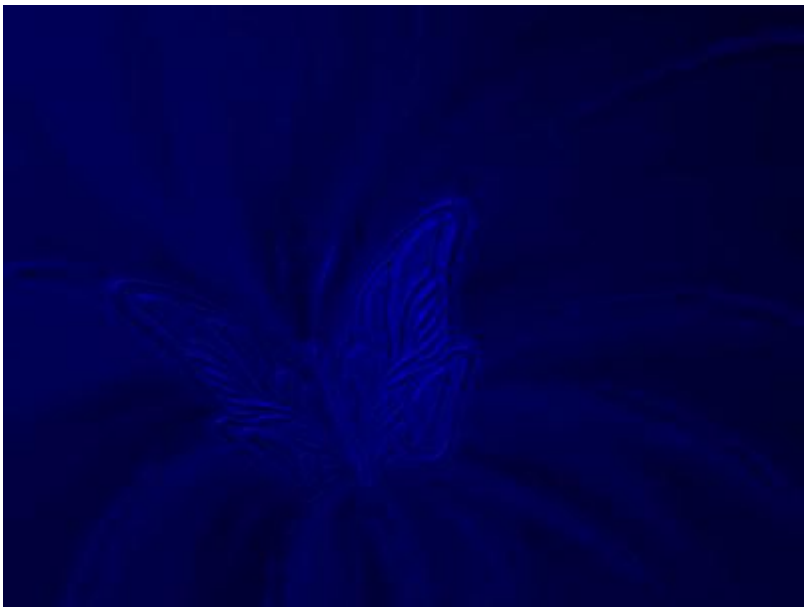
В некоторых графических форматах есть возможность добавления дополнительных каналов. Например, часто можно добавить альфа-канал, определяющий прозрачность изображения или отдельных слоёв и объектов (обычно 256 уровней прозрачности).

Каналы могут быть использованы для цветовой коррекции изображений и для создания различных спецэффектов. Каналы в основном используются при работе с растровой графикой.

Многие трюки в современных фильмах сделаны при помощи каналов. Например, чтобы создать эффект полёта, человека помещают в специально окрашенную зелёную комнату, и подвешивают за зелёные верёвки. Затем изображение человека можно легко вырезать, используя зелёный канал, и поместить на нужном фоне.

Цветовые каналы RGB





Цветовые каналы CMYK

Глубина цвета (Color Depth)

Глубина цвета - постоянная величина для данного изображения. Она определяет количество возможных комбинаций интенсивностей различных каналов в изображении как некоторую степень двойки. Например, для полноцветного изображения RGB количество цветов равно 256*256*256 = 16777216 = 224. Тогда говорят, что глубина цвета равна 24 бит. Если к нему ещё добавить альфа-канал (256 уровней прозрачности), то получится 232 - 32х-битный цвет. При этом количество возможных цветов остаётся равным 224. Для чёрно-белого изображения в режиме Grayscale глубина цвета равна 8 бит (28 = 256).

Фактически, для растровой графики, глубина цвета определяет количество информации, необходимой для описания цвета одного пикселя (если не применяется сжатие).

Форматы графических файлов

Формат графического файла определяет тип графической информации - растровый или векторный, алгоритм сжатия, глубину цвета хранимой информации, возможность записи анимации и звука, а кроме этого в нём может храниться дополнительная информация, используемая графическими редакторами (например, информация о слоях). Каждый формат соответствует определённому расширению файла.

Форматы растровых графических файлов

BMP - это формат графического редактора Paint. В нём не применяется сжатие. Он хорошо подходит для хранения очень маленьких изображений - таких как иконки на рабочем столе. Большие же файлы в этом формате занимают слишком много места.

GIF - формат, использующий алгоритм сжатия без потерь информации LZW. Максимальная глубина цвета - 8 бит (256 цветов). В нём также есть возможность записи анимации. Поддерживает прозрачность пикселей (двухуровневая – полная прозрачность, либо полная непрозрачность). Данный формат широко применяется при создании Web-страниц. Его выгодно применять для изображений с малым количеством цветов и резкими границами (например, для текстовых изображений).

PNG - разработан с целью заменить формат GIF. Использует алгоритм сжатия Deflate без потерь информации (усовершенствованный LZW). Максимальная глубина цвета - 48 бит. Поддерживает каналы градиентных масок прозрачности (256 уровней прозрачности). PNG - относительно новый формат, и поэтому ещё не очень распространён. В основном используется в Web-дизайне. К сожалению, даже в некоторых современных браузерах (таких, как Internet Explorer 6) отсутствует поддержка прозрачности PNG и поэтому не рекомендуется использовать прозрачные PNG изображения на Web-страницах.

JPEG(JPG) - формат, использующий алгоритм сжатия с потерями информации, который позволяет уменьшить размер файла в сотни раз. Глубина цвета - 24 бит. Не поддерживается прозрачность пикселей. При сильном сжатии в области резких границ появляются дефекты. Формат JPEG хорошо применять для сжатия полноцветных фотографий. Учитывая то, что при повторном сжатии происходит дальнейшее ухудшение качества, рекомендуется сохранять в JPEG только конечный результат работы. JPEG широко применяется при создании Web-страниц, а также для хранения больших коллекций фотографий.

TIFF - формат, специально разработанный для сканированных изображений. Может использовать алгоритм сжатия без потерь информации LZW. Позволяет сохранять информацию о слоях, цветовых профилях(ICC-профилях) и каналах масок. Поддерживает все цветовые модели. Аппаратно независим. Используется в издательских системах, а также для переноса графической информации между различными платформами (с PC на Macintosh, например).

PSD - формат графического редактора Adobe Photoshop. Использует алгоритм сжатия без потерь информации RLE. Позволяет сохранять всю информацию, создаваемую в этой программе. Кроме этого, в связи с популярностью Photoshop, данный формат поддерживается практически всеми современными редакторами компьютерной графики. Его удобно использовать для сохранения промежуточного результата при работе в Photoshop и других растровых редакторах.

RIFF - формат графического редактора Corel Painter. Позволяет сохранять всю информацию, создаваемую в этой программе. Его следует использовать для сохранения промежуточного результата при работе в Painter.

Форматы векторных и смешанных графических файлов

AI - это формат векторного редактора Adobe Illustrator. Позволяет сохранять всю информацию, создаваемую в этой программе. Его можно импортировать практически в любой графический редактор, а также во многие растровые, например Photoshop. При открытии в растровом редакторе документ растеризуется.

CDR - формат векторного редактора Corel Draw. Поддерживается меньшим количеством программ, чем AI.

EPS - смешанный графический формат, может содержать информацию о растровой и векторной графике. Поддерживает все цветовые модели. Этот формат понимает большинство современных (PostScript) принтеров, поэтому он широко применяется для печати изображений. Его понимает подавляющее большинство современных графических редакторов. Он также используется для обмена данными (через буфер обмена) в программах фирмы Adobe. Файлы в формате EPS имеют большой объем (по сравнению с TIFF, например).

PDF - смешанный формат, предназначенный для передачи информации по сети. В основном используется для создания электронных версий книг и статей. Может содержать текстовую, растровую и векторную графическую и звуковую информацию, видео, а также гиперссылки. При этом различные типы информации по-разному сжимаются. Готовые документы просматриваются с помощью Adobe Acrobat Reader.

SWF - формат векторной анимации. В нём можно хранить несложную векторную графику, растровую графику, анимацию, а также звук. Формат SWF недоступен для редактирования, только для просмотра с помощью программы Flash Player или Web-браузера. Анимация создаётся с помощью программы Macromedia Flash, также имеется возможность экспорта в этот формат из большинства редакторов векторной графики. Формат SWF аппаратно независим.

FLA - внутренний формат программы Macromedia Flash. В нём можно сохранять промежуточные результаты, для просмотра требуется преобразование в формат SWF. Поддерживает язык сценариев ActionScript, что даёт возможность создания сложных интерактивных фильмов и Web-страниц.

О сжатии растровой графики

Иногда характеристики растрового изображения записывают в такой форме: 1024x768x24. Это означает, что ширина изображения равна 1024 пикселям, высота - 768 и глубина цвета равна 24. 1024x768 - рабочее разрешение для 15 - 17 дюймовых мониторов. Несложно догадаться, что размер несжатого изображения с такими параметрами будет равен 1024*768*24 = 18874368 байт. Это более 18 мегабайт - слишком жирно для одной картинки, особенно если требуется хранить несколько тысяч таких картинок - это не так уж много по компьютерным меркам. Вот почему компьютерную графику используют почти всегда в сжатом виде.

RLE (Run Length Encoding) - метод сжатия, заключающийся в поиске последовательностей одинаковых пикселей в сточках растрового изображения ("красный, красный, ..., красный" записывается как "N красных").

LZW (Lempel-Ziv-Welch) - более сложный метод, ищет повторяющиеся фразы - одинаковые последовательности пикселей разного цвета. Каждой фразе ставится в соответствие некоторый код, при расшифровке файла код замещается исходной фразой.

При сжатии файлов формата JPEG (с потерей качества) изображение разбивается на участки 8x8 пикселей, и в каждом участке их значение усредняется. Усреднённое значение располагается в левом верхнем углу блока, остальное место занимается меньшими по яркости пикселями. Затем большинство пикселей обнуляются. При расшифровке нулевые пиксели получают одинаковый цвет. Затем к изображению применяется алгоритм Хаффмана.

Алгоритм Хаффмана основан на теории вероятности. Сначала элементы изображения (пиксели) сортируются по частоте встречаемости. Затем из них строится кодовое дерево Хаффмана. Каждому элементу сопоставляется кодовое слово. При стремлении размера изображения к бесконечности достигается максимальность сжатия. Этот алгоритм также используется в архиваторах.

Сжатие применяется и для векторной графики, но здесь уже нет таких простых закономерностей, так как форматы векторных файлов достаточно сильно различаются по содержанию.

Графические редакторы

Графические редакторы – программы, в которых создаётся и редактируется графика. Их, также как и графику, разделяют на растровые и векторные. Но на самом деле чисто растровых или векторных редакторов практически нет. В растровых редакторах, например, есть возможность редактирования текста, который, в свою очередь состоит из векторных объектов - букв. В векторных редакторах обычно можно помещать в документ растровые изображения.

Adobe Photoshop - самый популярный на данный момент редактор растровой графики. Используется для коррекции и ретуширования растровых изображений, а также создания графических эффектов.

Adobe Illustrator - векторный графический редактор. Кроме обычной векторной графики позволяет применять растровые эффекты к векторным объектам, при каждой трансформации которых происходит пересчёт и создание нового изображения и, следовательно, не ухудшается качество.

Corel Painter - редактор растровой графики, имитирующий настоящие художественные инструменты и материалы. Также есть и специальные инструменты, которых у обычного художника нет. Для работы с ним желательно иметь графический планшет. Есть возможность создания графических скриптов.

Corel Draw - ещё один векторный редактор. В Corel Draw есть множество возможностей, которых нет в Adobe Illustrator и наоборот. Поддерживает Visual Basic For Applications.

Macromedia Flash - редактор векторной графики и анимации. В нём также можно создавать мультимедийные презентации, Web-страницы, игры и т.д. и т.п. Имеет встроенный язык сценариев ActionScript.

Цветовые модели RGB и CMY

Цветовые модели, используемые в компьютерной графике, - это средства описания цветов в определенном диапазоне.

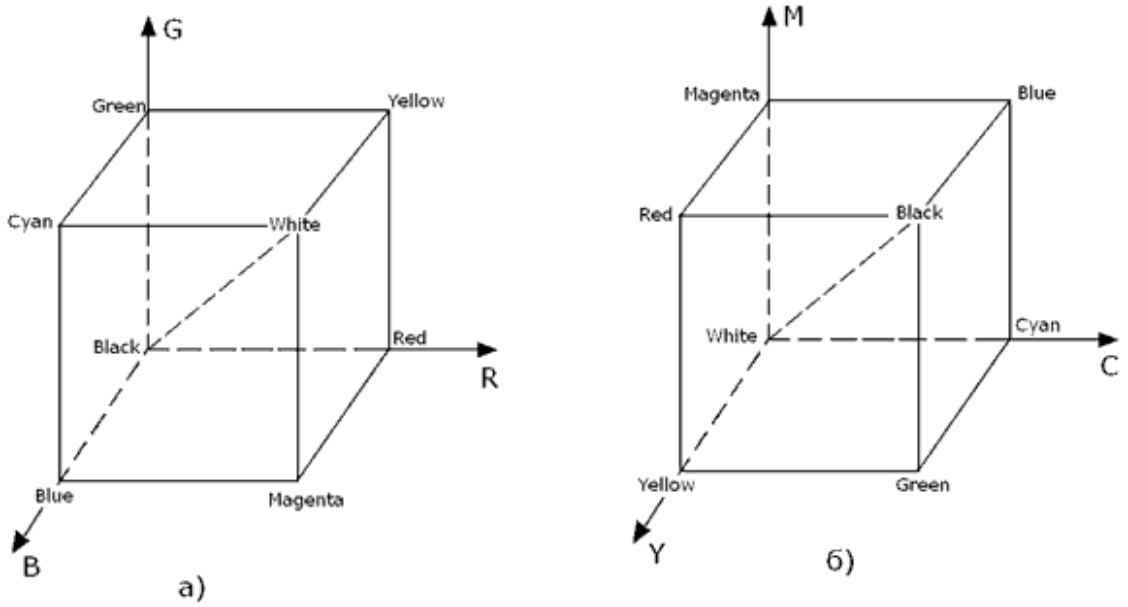


Рис. 4.7. Цветовой куб для моделей RGB и CMY

На основе описанных выше физических представлений в компьютерной графике была принята так называемая аддитивная цветовая модель, использующая три первичных составляющих цвета. Эта модель предполагает, что любой цвет можно рассматривать как взвешенную сумму трех основных цветов. Проиллюстрировать ее можно на примере освещения сцены с помощью трех прожекторов разного цвета. Каждый прожектор управляется независимо, и путем изменения мощности каждого из них можно воспроизвести практически все цвета. В модели RGB цвет можно представить в виде вектора в трехмерной системе координат с началом отсчета в точке (0,0,0). Максимальное значение каждой из компонент вектора примем за 1. Тогда вектор (1,1,1) соответствует белому цвету. Все цветовые векторы, таким образом, заключены внутри единичного куба, называемого цветовым кубом (рис. 2.7а).

Другая модель смешения цветов - субстрактивная цветовая модель, или модель CMY, использующая в качестве первичных составляющих цвета Cyan, Magenta, Yellow (голубой, пурпурный, желтый), которые являются дополнительными к Red, Green, Blue. В этой модели оттенки цвета получаются путем "вычитания" из падающего света волн определенной длины. Этот подход нуждается в пояснении. В этой системе координат вектор (0,0,0) соответствует белому цвету, а вектор (1,1,1) - черному. Соответствующий цветовой куб представлен на рис. 2.7б.

Связь между значениями (R,G,B) и (C,M,Y) для одного и того же цвета выражается формулой

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} C \\ M \\ Y \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

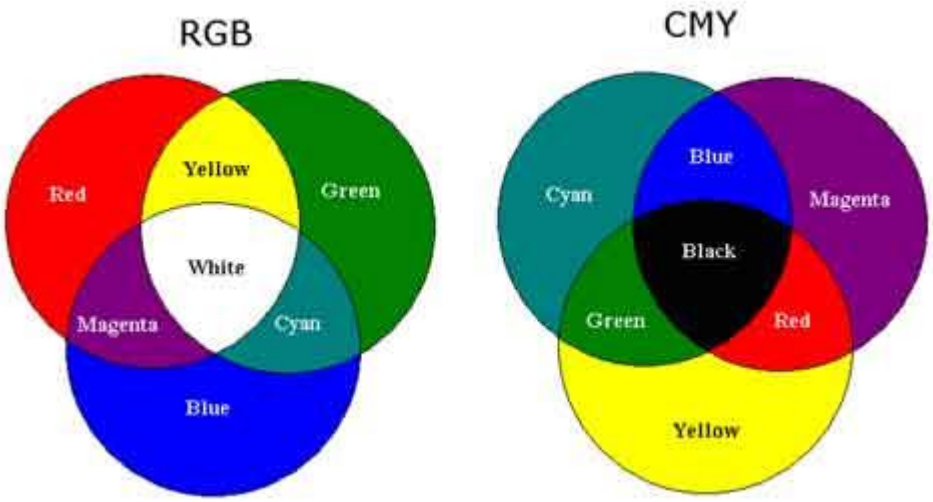


Рис. 4.8. Схема смешения цветов для моделей RGB и CMY

Цвета одной модели являются дополнительными к цветам другой (дополнительный цвет - это цвет, результатом смешения которого с данным является белый). Схема смешения цветов для двух моделей представлена на рис. 2.8. Пример субстрактивного формирования оттенков показан на рис. 2.9. При освещении падающим белым светом в слое голубой (Cyan) краски из спектра белого цвета поглощается (вычитается) красная часть как дополнительный цвет, затем из оставшегося света в слое пурпурной (Magenta) краски поглощается зеленая часть спектра, и, наконец, от белой поверхности отражается синий цвет, который мы и видим. Таким образом, смешение голубого и пурпурного цветов дает в итоге синий цвет.

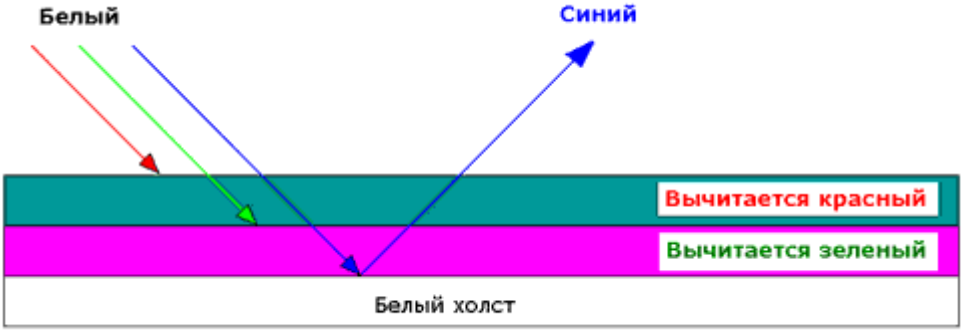


Рис. 4.9. Субстрактивное формирование оттенков

Растровые дисплеи, как правило, используют аппаратно- ориентированную модель цветов RGB. Существуют также дисплеи с таблицей цветности, представляющей собой матрицу, каждый элемент которой - некоторый цвет (вектор RGB). В таких дисплеях значения кодов пикселей, заносимые в видеопамять, представляют собой индексы матрицы цветности. При отображении некоторого пикселя на экран по значению кода выбирается элемент таблицы цветности, содержащий тройку значений R, G, B. Эта тройка и передается на монитор для задания цвета пикселя на экране.

В полноцветных дисплеях для каждого пикселя в видеопамять заносится тройка значений R, G, B. В этом случае для отображения пикселя из видеопамяти непосредственно выбираются значения R, G, B, которые и передаются на монитор (но могут и передаваться в таблицу цветности).

В моделях RGB и CMY легко задавать яркости для одного из основных цветов, но довольно затруднительно задать оттенок с требуемым цветовым тоном и насыщенностью, соответствующим какому-либо образцу цвета. В различного рода графических редакторах эта задача чаще всего решается с помощью интерактивного выбора из палитры цветов и формированием цветов в палитре путем подбора значений координат до получения требуемого визуального результата. Иногда такая палитра наглядно отображает выбор вектора из цветового куба: сначала посредством одного движка выбирается цветовая плоскость, а затем на этой плоскости выбирается конкретная точка. Но и таким методом не сразу удается достигнуть желаемого эффекта, поскольку не так просто выбрать правильную цветовую плоскость.

Цветовые модели HSV и HLS

Приведенные модели не охватывают всего диапазона видимого цвета, поскольку их цветовой охват - это лишь треугольник на графике МКО, вершинам которого соответствуют базовые цвета. Они являются аппаратно ориентированными, т.е. соответствуют технической реализации цвета в устройствах графического вывода. Но психофизиологическое восприятие света определяется не интенсивностью трех первичных цветов, а цветовым тоном, насыщенностью и светлотой. Цветовой тон позволяет различать цвета, насыщенность задает степень "разбавления" чистого тона белым цветом, а светлота - это интенсивность света в целом. Поэтому для адекватного нашему восприятию подбора оттенков более удобными являются модели, в числе параметров которых присутствует тон (Hue). Этот параметр принято измерять углом, отсчитываемым вокруг вертикальной оси. При этом красному цвету соответствует угол 0°, зеленому - 120°, синему - 240°, а дополняющие друг друга цвета расположены один напротив другого, т.е. угол между ними составляет 180°. Цвета CMY расположены посредине между составляющими их компонентами RGB. Существует две модели, использующие этот параметр.

Модель HSV (Hue, Saturation, Value, или тон, насыщенность, количество света) можно представить в виде световой шестигранной пирамиды (рис. 2.10), по оси которой откладывается значение V, а расстояние от оси до боковой грани в горизонтальном сечении соответствует параметру S (за диапазон изменения этих величин принимается интервал от нуля до единицы). Значение S равно единице, если точка лежит на боковой грани пирамиды. Шестиугольник, лежащий в основании пирамиды, представляет собой проекцию цветового куба в направлении его главной диагонали (рис. 2.11).

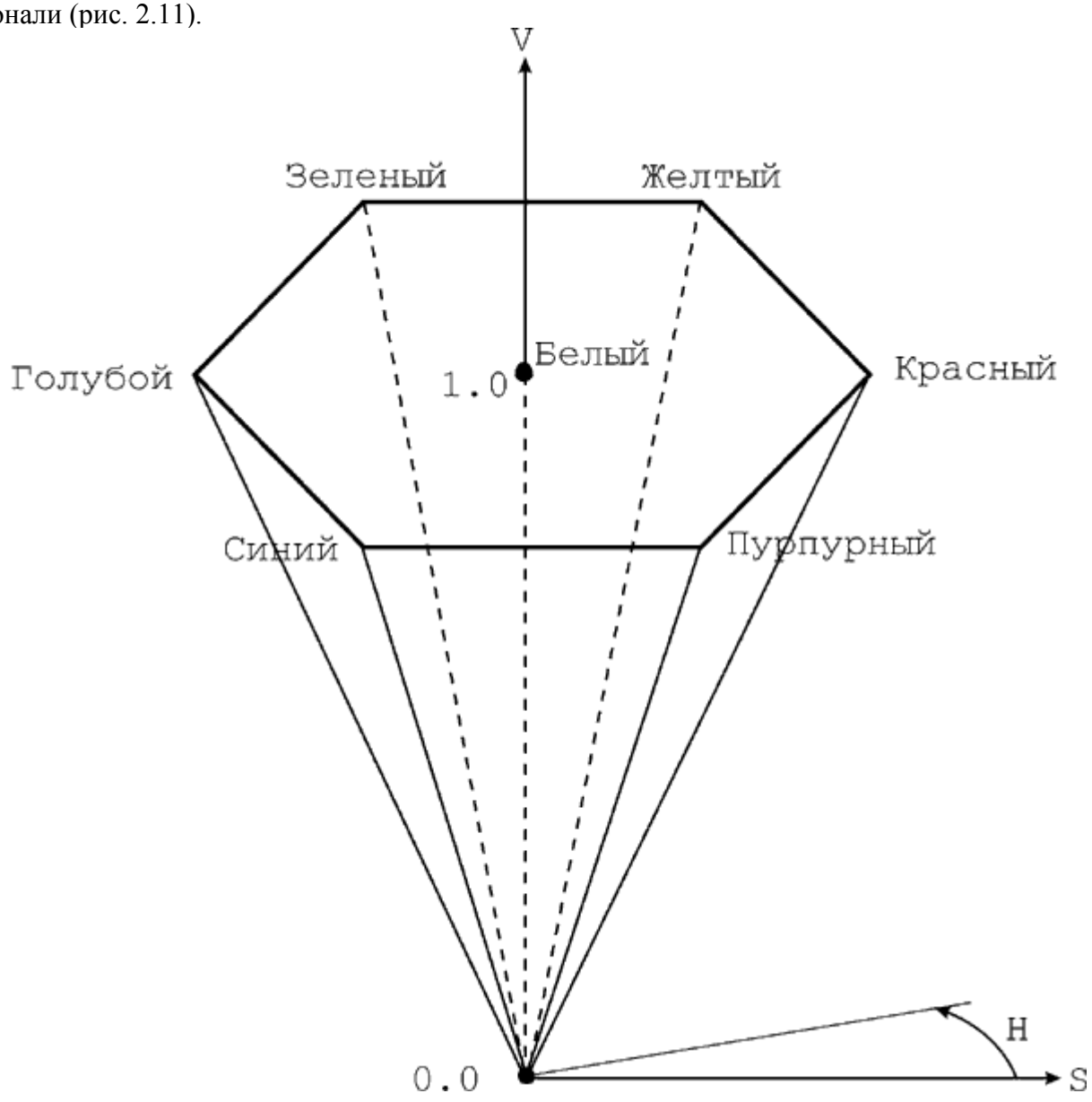


Рис. 4.10. Цветовое пространство HSV

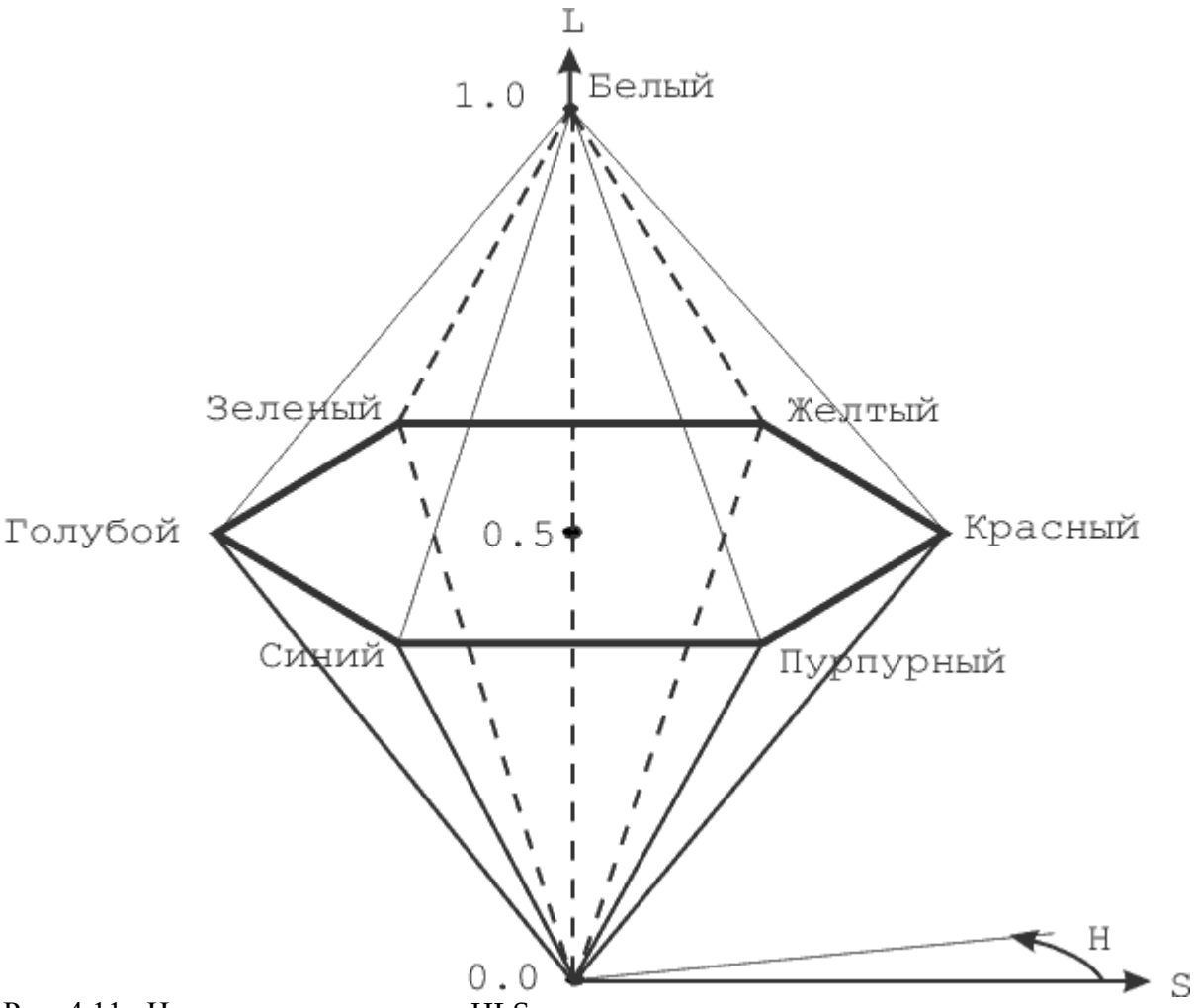


Рис. 4.11. Цветовое пространство HLS

Преобразование цветового пространства HSV в RGB осуществляется непосредственно с помощью геометрических соотношений между шестигранной пирамидой и кубом.

Цветовая модель HLS (Hue, Lightness, Saturation, или тон, светлота, насыщенность) является расширением модели HSV. Здесь цветовое пространство уже представляется в виде двойной пирамиды (рис. 2.11), в которой по вертикальной оси откладывается L (светлота), а остальные два параметра задаются так же, как и в предыдущей модели. В литературе эти пирамиды иногда называют шестигранным конусом.

На рис. 2.12 и 2.13 приведены блок-схемы преобразования моделей HSV и HLS в модель RGB. Алгоритмы обратного преобразования предлагаются читателю в качестве упражнения.

В первом алгоритме используется функция Ent, означающая целую часть числа. Кроме того, используется операция присваивания для векторов. Константа ndf (сокращенное от выражения "not defined") используется при входе в алгоритм для того, чтобы выяснить, задано ли значение переменной H. Например, по соглашению ndf может быть некоторым отрицательным значением, так как тон - это всегда положительная величина. Во втором алгоритме применяется вспомогательная функция Value (H, M1, M2) для вычисления значения компоненты R, G или B в зависимости от ситуации.

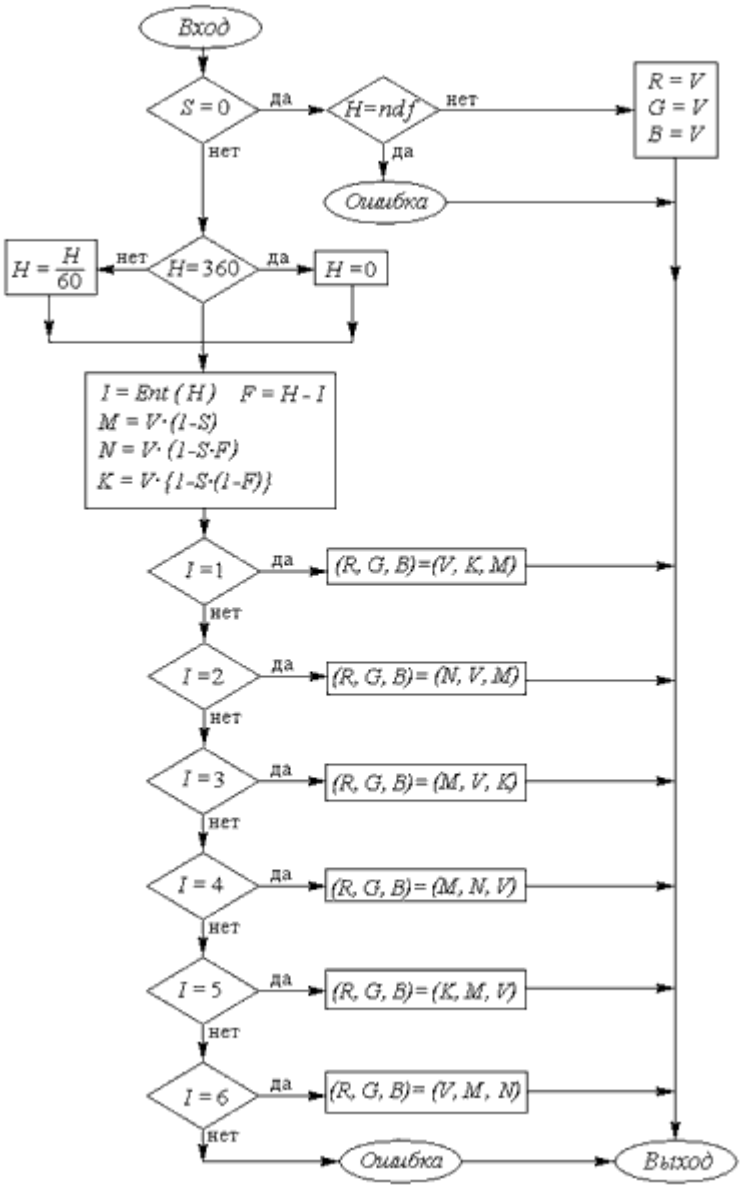


Рис. 4.12. Преобразование модели HSV в RGB

Алгоритм преобразования:
Приведение H к заданному диапазону:
  Пока H<0 H=H+360
  Пока H>360 H=H-360
Определение координат
  Если H<60 то Value=M1+(M2-M1)*H/60
  Если 60<=H<180 то Value=M2
  Если 180<=H<240 то Value=M1+(M2-M1)*(240-H)/60
  Если 240<=H то Value=M1

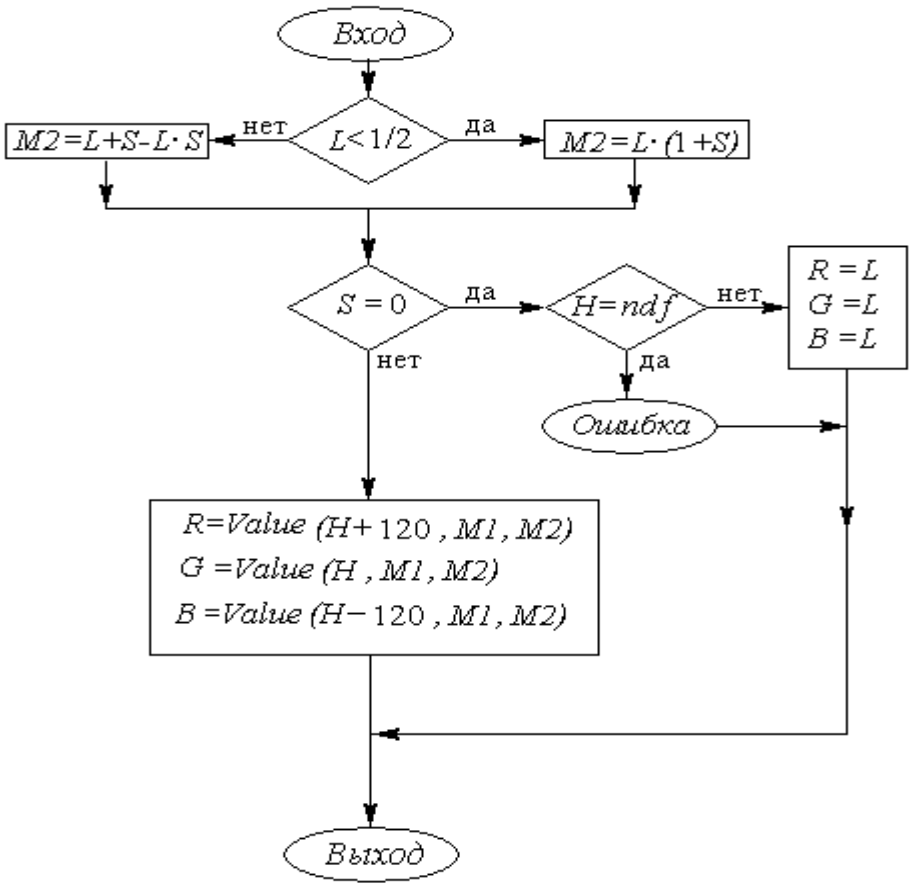


Рис. 4.13. Преобразование модели HLS в RGB

Пространство CIE Luv

Один из существенных минусов цветового пространства XYZ — это то, что оно не является перцептивно (визуально) равномерным и не может использоваться для вычисления цветовых расстояний. Поэтому CIE (МКО) продолжила разработку перцептивно равномерного пространства. Целью комитета CIE было создание повторяемой системы стандартов цветопередачи для производителей красок, чернил, пигментов и других красителей. Самая важная функция этих стандартов — предоставить универсальную схему, в рамках которой можно было бы устанавливать соответствие цветов.

В результате было создано цветовое пространство CIE Luv, позволяющее определить различение цветов для человека с "усредненным" зрением, (т.е. различные люди неодинаково воспринимают разницу между цветами). Свое название пространство получило благодаря его компонентам L, u и v. Параметр L соответствует яркости цвета, u отвечает за переход от зеленого к красному (при увеличении), а при увеличении параметра v происходит переход от синего к фиолетовому. Если u и v равны 0, то, меняя L, получаем цвета, являющиеся градациями серого.

Это цветовое пространство было разработано для количественного измерения различия двух цветов. CIE были проведены исследования с участием большого числа людей, результатом чего явилось создание пространства Luv. Измерения проводились в "хороших" условиях (достаточное освещение и неяркий монотонный фон); перед испытуемым находились два листа бумаги, окрашенных соответственно двумя цветами, и он должен был дать ответ, насколько, по его мнению, различаются эти цвета. В случае реальных изображений мы должны находить различия между цветами на более сложном фоне, при этом не всегда при хорошем освещении (например, слишком ярком). Но освещение зависит и от помещения, и от времени суток, и от того, под каким углом находится поверхность к источнику света.

Переход из RGB в Luv осуществляется следующим образом. Сначала нормируем R, G, B:

$$\begin{pmatrix} R^* \\ G^* \\ B^* \end{pmatrix} = \begin{pmatrix} R/255 \\ G/255 \\ B/255 \end{pmatrix}$$

Далее совершаем преобразование пространства RGB в XYZ:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} 0.412453 & 0.35758 & 0.180423 \\ 0.212671 & 0.71516 & 0.072169 \\ 0.019334 & 0.119193 & 0.950227 \end{pmatrix} \times \begin{pmatrix} R^* \\ G^* \\ B^* \end{pmatrix}$$

Цветовое пространство CIE Luv — непрерывное однородное преобразование пространства CIE XYZ, описываемое следующими формулами:

$$L = \begin{cases} 116 \times \sqrt[3]{\frac{Y}{Y_n}} - 16, & \frac{Y}{Y_n} > 0.008856 \\ 903.3 \times \frac{Y}{Y_n}, & \frac{Y}{Y_n} \le 0.008856 \end{cases}$$

$$u' = \frac{4X}{X + 15Y + 3Z} = \frac{4x}{-2x + 12y + 3}, \quad v' = \frac{9Y}{X + 15Y + 3Z} = \frac{9y}{-2x + 12y + 3}$$

$$u = 13L(u' - u_n), \quad v = 13L(v' - v_n)$$

Для определения параметров Y_n, u_n и v_n, вводится понятие белой точки (white point). Белая точка - это пара параметров цветности (x, y), определяющая эталон белого цвета для различных источников света. CIE составила таблицу белых точек для источников света разной яркости. При этом значение компоненты Y белой точки в XYZ нормализовано до 100 (в приведенных выше формулах Y_n как раз соответствует нормализованной Y компоненте). Параметры u_n и v_n вычисляются по тем же формулам, что u' и v', в которых используются значения x и y для белой точки.

Как уже упоминалось выше, компонента L соответствует яркости цвета, а из формул видно, что L пропорциональна кубическому корню из компоненты Y пространства XYZ. Однако существует мнение, что человеческому восприятию больше соответствует корень второй степени из освещенности. Так, например, в цветовом пространстве Lab параметр L вычисляется с использованием квадратного корня.

Немного о свойствах величин L, u, v:

L меняется от 0 до 100;

u, v лежат в пределах -200, 200;

u отвечает за переход от зеленого к красному (при увеличении u);

v отвечает за переход от синего к фиолетовому (при увеличении v);

если u и v равны 0, меняя L, получаем изображение, содержащее градации серого (grayscale).

Наконец, самое важное, к чему мы стремились, переходя в это пространство. Нам заданы два цвета - L_1, u_1, v_1 и L_2, u_2, v_2. Как определить расстояние между цветами, то есть насколько человек заметил бы различие между ними? Оказывается, оно задается евклидовой нормой

$$D = \sqrt{(L_1 - L_2)^2 + (u_1 - u_2)^2 + (v_1 - v_2)^2}$$

При расстоянии между двумя цветами $D > 5$ большинство людей уже замечают различие, при $D > 10$ оно заметно всем. В этом и состоит главное достоинство этого пространства. Оно учитывает восприятие цветов человеком, и различие между цветами определяется очень простой формулой. Необходимо заметить, что эта формула применима в определенных условиях: освещение, фон не должны мешать и отвлекать.

Одновременно с разработкой CIE Luv было также разработано перцептивно равномерное цветовое пространство CIE Lab. Из этих двух моделей более широко применяется модель CIE Lab. Структура цветового пространства Lab основана на той теории, что цвет не может быть одновременно зеленым и красным или желтым и синим (рис. 2.14). Следовательно, для описания атрибутов "красный/зеленый" и "желтый/синий" можно воспользоваться одними и теми же значениями. Формулы перехода от пространства XYZ к пространству Lab осуществляется следующим образом:

$$L = \begin{cases} 116 \cdot [(Y/Y_n)^{1/3}] - 16 & \text{если } (Y/Y_n) > 0.008856 \quad a = 500 \cdot [f(X/X_n) - f(Y/Y_n)] \\ 903.3 \cdot Y/Y_n & \text{если } (Y/Y_n) \le 0.008856 \quad b = 200 \cdot [f(Y/Y_n) - f(Z/Z_n)] \end{cases}$$

$$\text{где } f(t) = \begin{cases} t^{1/3} & \text{если } (Y/Y_n) > 0.008856 \\ 7.787 \cdot t + 16/116 & \text{если } (Y/Y_n) \le 0.008856 \end{cases}$$

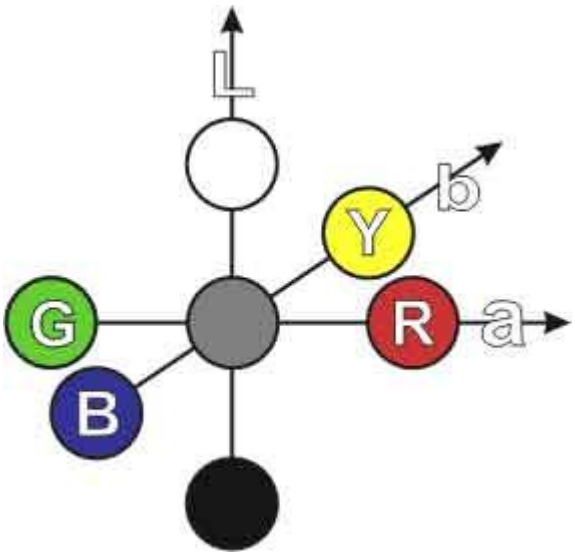


Рис. 4.14. Представление цвета в пространстве CIE Lab

Рис. 4.15. Видимое стандартным наблюдателем пространство Lab

Каждая цветовая модель, помимо преимуществ, также имеет и свои индивидуальные недостатки. Существуют и другие модели, которые здесь не рассматриваются.

Представление цвета в машинной графике

Понятие цвета возникает при описании восприятия глазами человека электромагнитных волн в определенном диапазоне частот (длина волны λ от 400 нм (фиолетовый) до 700 нм (красный). Таким образом, самым общим описанием светового потока может служить его спектральная функция I(λ). Свет называется **монохроматическим** (не путать с монохромными дисплеями, рассматриваемыми в следующей лекции), если его спектр состоит из одного значения α; математически $I_\alpha(\lambda) = c \cdot \delta(\lambda - \alpha)$), где c - яркость. Понятно, что описание цвета путем описания функции в большинстве случаев слишком громоздко, хотя иногда и применяется. К тому же, оно является избыточным, если подробнее рассмотреть, как глаз человека воспринимает свет. На сетчатке глаза находятся два типа рецепторов: палочки и колбочки. Палочки реагируют на степень яркости (или интенсивность) падающего света, а колбочки отвечают за различение цветов; при этом колбочки

резко теряют свою чувствительность в темноте (в отличие от палочек), поэтому все объекты начинают казаться серыми. Колбочки бывают трех видов (их часто обозначают **S**, **M** и **L**)[1)], и их кривые относительной чувствительности представлены на рис. 1.3. Пики на кривых чувствительности отвечают красному, зеленому и синему цветам. При этом следует заметить, что восприимчивость к синему цвету значительно ниже, чем к двум другим. Также важным свойством восприятия света человеком является его линейность: при освещении двумя источниками света (со спектральными функциями $I_1(\lambda)$, $I_2(\lambda)$) человек воспринимает их как один со спектральной функцией, равной сумме $I(\lambda) = I_1(\lambda)+I_2(\lambda)$. Этот факт называется **законом Грассмана**. Благодаря ему можно строить сравнительно простую теорию цветовосприятия.

Так как области восприятия для разных типов колбочек перекрываются, то возникают **метамеры**, - потоки волн с разными спектральными характеристиками, но воспринимаемые как имеющие один и тот же цвет.

Цветовая модель RGB

Из рассмотренной выше модели человеческого зрения вытекает, что достаточно обоснованной является цветовая модель **RGB** (от англ. Red, Green, Blue - красный, зеленый, голубой), в которой спектральная функция представляется как сумма кривых чувствительности для каждого типа колбочек с неотрицательными весовыми коэффициентами (обычно их нормируют от 0 до 1), которые так и обозначаются - R, G и B. Эта модель характеризуется свойством аддитивности (мы складываем цвета для получения новых). К примеру, спектральные функции:

- черного цвета: $f_{black} = 0$, (R,G,B) = (0,0,0);
- фиолетового цвета $f_{violet} = f_{red} + f_{blue}$, (R,G,B) = (1,0,1);
- белого цвета $f_{white} = f_{red} + f_{green} + f_{blue}$, (R,G,B) = (1,1,1).

Цветовая система CIE XYZ и диаграмма цветности CIE

Международный стандарт представления цвета **CIE XYZ** был принят в 1931 году Международной комиссией по освещению (CIE - фр. Commission Internationale de l'Eclairage), В нем определяются три базисные функции $\rho_X(\lambda)$, $\rho_Y(\lambda)$, $\rho_Z(\lambda)$, зависящие от длины волны, линейные комбинации которых с неотрицательными коэффициентами (X, Y и Z) позволяют получить все видимые человеком цвета.

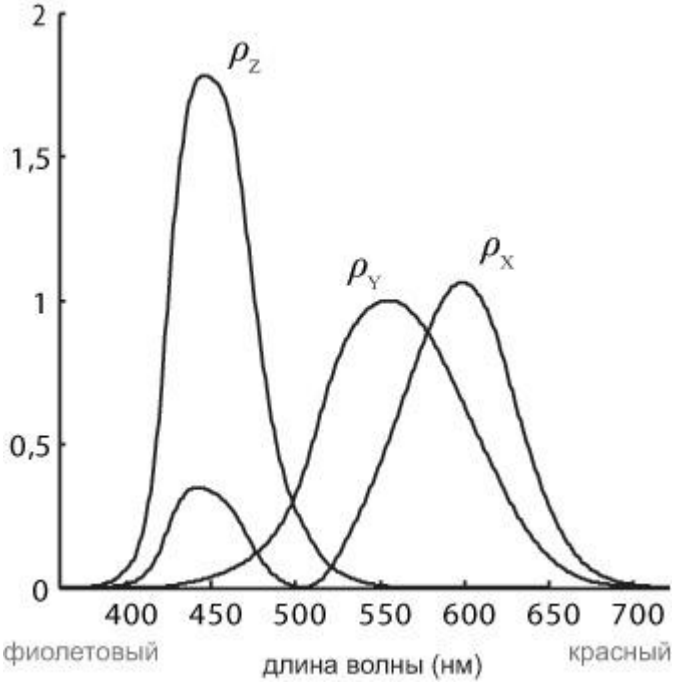


Рис. 4.16. Функции представления цвета для CIE XYZ.

Математически можно записать получение коэффициентов так:

$$X = k \int I(\lambda)\rho_X(\lambda)d\lambda,$$
$$Y = k \int I(\lambda)\rho_Y(\lambda)d\lambda,$$
$$Z = k \int I(\lambda)\rho_Z(\lambda)d\lambda,$$

где $I(\lambda)$ - спектральная функция распределения для представляемого цвета, а k - масштабный коэффициент, выбираемый исходя из того, какой цвет принимается за белый и в каком диапазоне должны лежать значения Y.

$$k = \frac{Y_{max}}{\int I_{бел}(\lambda)\rho_Y(\lambda)d\lambda}$$

где $I_{бел}(\lambda)$ - спектральная функция распределения для выбранного эталона белого цвета. Функция $\rho_Y(\lambda)$ соответствует относительному восприятию интенсивности света палочками.

Если рассмотреть значения X,Y,Z как координаты в трехмерном евклидовом пространстве, то видимые цвета образуют криволинейный конус в первом квадранте (см. рис. 4.17).

Рассмотрим **значения цветности** (англ. chromacity values) x, y, z, которые определяются из X,Y,Z следующим образом:

$$x = \frac{X}{X+Y+Z},$$
$$y = \frac{Y}{X+Y+Z},$$
$$z = \frac{Z}{X+Y+Z}. \quad (1.1)$$

Они вводятся для описания только цветовых свойств света, безотносительно его энергии, и зависят только от основной длины волны и насыщенности. Таким образом, если опять же поместить эти точки в трехмерное евклидово пространство, то они будут как раз лежать на плоскости X + Y + Z = 1 (она также показана на рис. 4.17). Проекция этой плоскости на Oxy называется **диаграммой цветности CIE** (см. рис. 4.18).

Эта диаграмма весьма полезна и наглядна и широко используется. Цвета, расположенные на границе проекции, являются монохроматическими. При смешении базисных цветов можно получить все цвета, находящиеся в их выпуклой оболочке на диаграмме цветности. Этим как раз и объясняется, что с помощью трех базовых цветов R,G,B (да и любых других) мы не можем получить все видимые цвета.

Введем понятие **точки белого** (англ. white point). Это точка на диаграмме цветности, соответствующая измеренным координатам белого цвета. Она может варьироваться в зависимости от того, какой источник цвета принимается за белый. В исходной модели CIE XYZ весовые функции были специально подобраны так, чтобы дневному свету солнца соответствовала точка (x, y, z) = (1/3, 1/3, 1/3). Другие точки белого применяются для компенсации условий съемки, например при освещении флуоресцентными лампами, или свойств оборудования. В фотографии это связано с так называемым нахождением баланса белого.

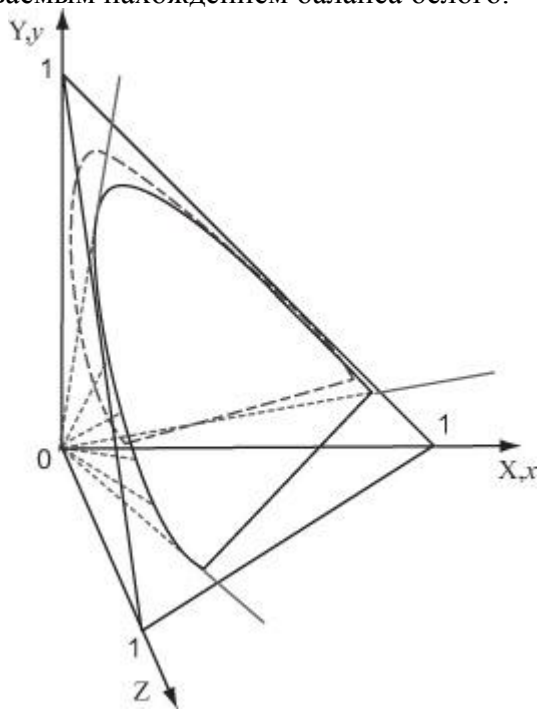


Рис. 4.17. Конус видимых цветов в трехмерном пространстве в модели CIE XYZ.

Важной характеристикой как цветовых моделей, так и конкретных устройств, отображающих цветную информацию, является **Цветовая гамма** (англ. Color gamut) - подмножество цветов, воспроизводимое в условиях конкретной цветовой модели или для конкретного устройства цветового отображения.

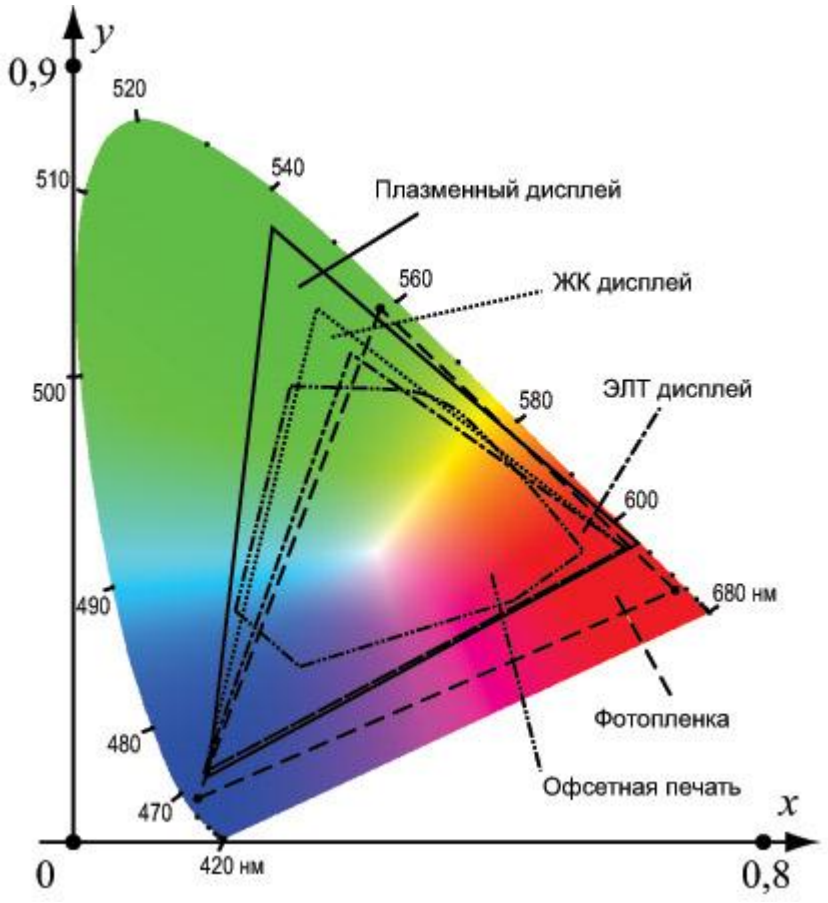


Рис. 4.18. Диаграмма цветности CIE с приблизительными цветовыми гаммами для разных классов устройств.

Корректно отображать цветовую гамму как некоторое подмножество в конусе видимых цветов Можно также ограничиться проекцией на диаграмму цветности, но при этом не учитывается диапазон яркости. На рис. 4.18 представлены некоторые типичные цветовые гаммы, которые позволяют судить о полноте охвата отображаемых цветов разными устройствами.

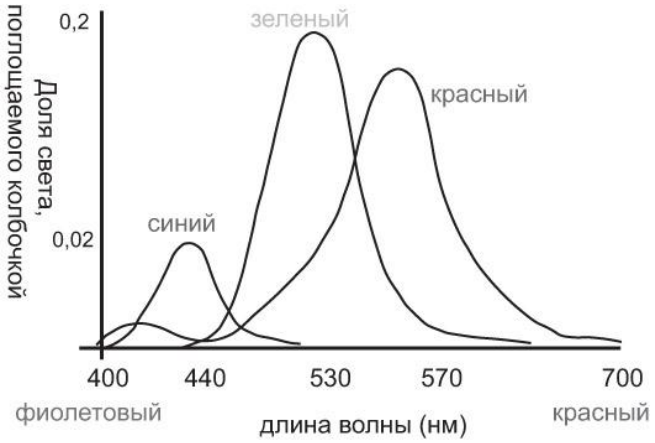


Рис. 4.19. Относительная чувствительность колбочек.

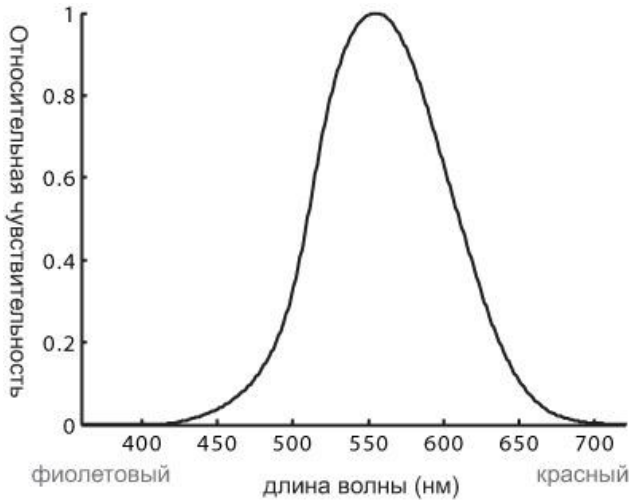


Рис. 4.20. Относительное восприятие интенсивности света палочками.

Если представить эти коэффициенты как координаты в трехмерном евклидовом пространстве и каждой точке сопоставить соответствующий цвет, получим наглядное изображение пространства **RGB**.

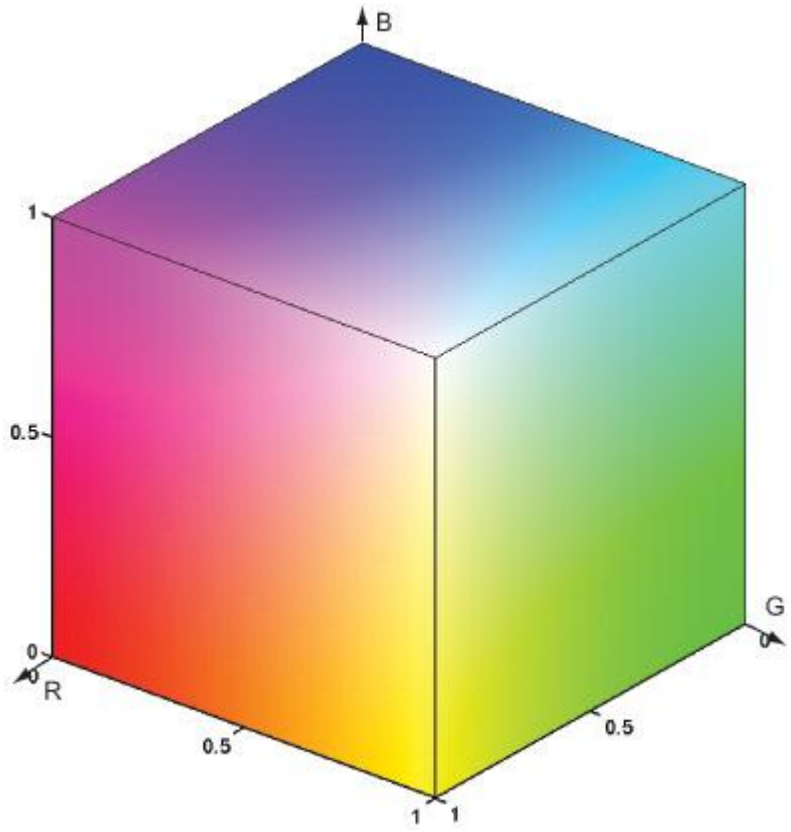


Рис. 4.21. Цветовая модель RGB.

Эта модель является в настоящее время самой распространенной. В то же время ей присущ важный недостаток: не все цвета, видимые человеком, представимы в этой модели. В конце 1920-х годов В.Д. Райтом и Дж. Гилдом были проведены эксперименты, в которых наблюдателю предлагалось каждому монохроматическому цвету[2)] фиксированной яркости в видимом диапазоне сопоставить цвет, составленный из смеси основных цветов R, G и B с некоторыми весами, регулируемыми наблюдателем. Оказалось, что для некоторых цветов необходимо было добавить отдельно яркости испытуемого света и одного из базисных цветов (был выбран R), с тем чтобы получить одинаковое восприятие. Это соответствует отрицательному весу R-компоненты. Такой эффект связан с тем, что волны из видимого диапазона воздействуют сразу на все типы колбочек и не всегда возможно ограничиться положительными коэффициентами для представления некоторых цветов из видимого спектра. К счастью, доля воспроизводимых цветов значительно больше, чем доля не представимых в этой модели цветов. Модель, с помощью которой можно представить все

цвета из спектра, ограничиваясь неотрицательными коэффициентами, представлена в следующем подразделе.

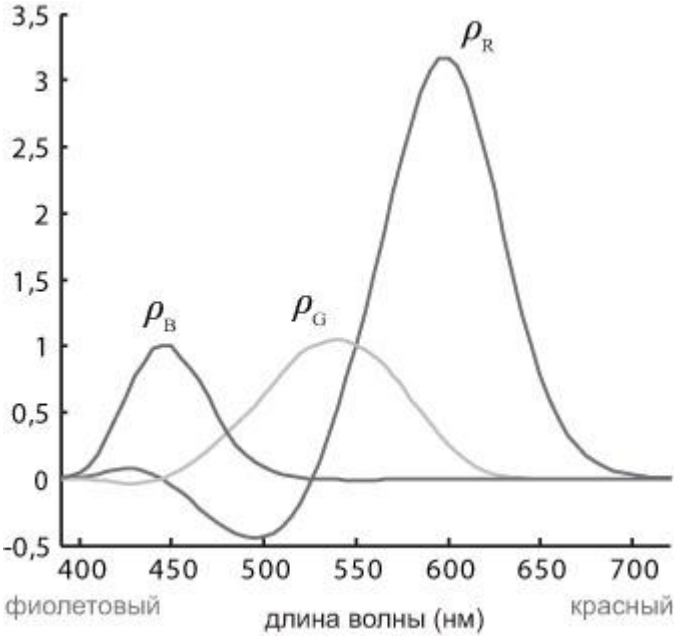


Рис. 4.22. Функции представления цвета для RGB.

Преобразования между CIE XYZ и RGB

Цветовое пространство RGB, как и CIE XYZ, является трехмерным и аддитивным. Поэтому преобразования между двумя этими пространствами описываются матрицами 3 × 3; достаточно задать координаты базисных цветов R, G и B в системе CIE XYZ. Обычно удобно это делать, отдельно задавая цветовую информацию точками (x, y) на диаграмме цветности и яркостной компонентой Y. Если цвет задан таким образом (x, y, Y), то из формул (1.1) следует, что

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} \frac{Y}{y}x \\ Y \\ \frac{Y}{y}(1-x-y) \end{pmatrix}. \quad (1.2)$$

Тогда, если базисные RGB-цвета заданы как (x_R, y_R, Y_R), (x_G, y_G, Y_G), (x_B, y_B, Y_B), получаем следующую формулу преобразования:

```
zR=1-xR-yR;
zG=1-xG-yG;
zB=1-xB-yB;
```

Листинг 1.1. Переход от RGB к CIE XYZ (html, txt)

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{bmatrix} \frac{Y_R}{y_R}x_R & \frac{Y_G}{y_G}x_G & \frac{Y_B}{y_B}x_B \\ Y_R & Y_G & Y_B \\ \frac{Y_R}{y_R}z_R & \frac{Y_G}{y_G}z_G & \frac{Y_B}{y_B}z_B \end{bmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix};$$

Все представимые в модели RGB цвета образуют параллелепипед в CIE XYZ (см. рис. 1.10), пересечение которого с плоскостью X + Y + Z = 1, спроецированное на диаграмму цветности и дает цветовую гамму данной модели.

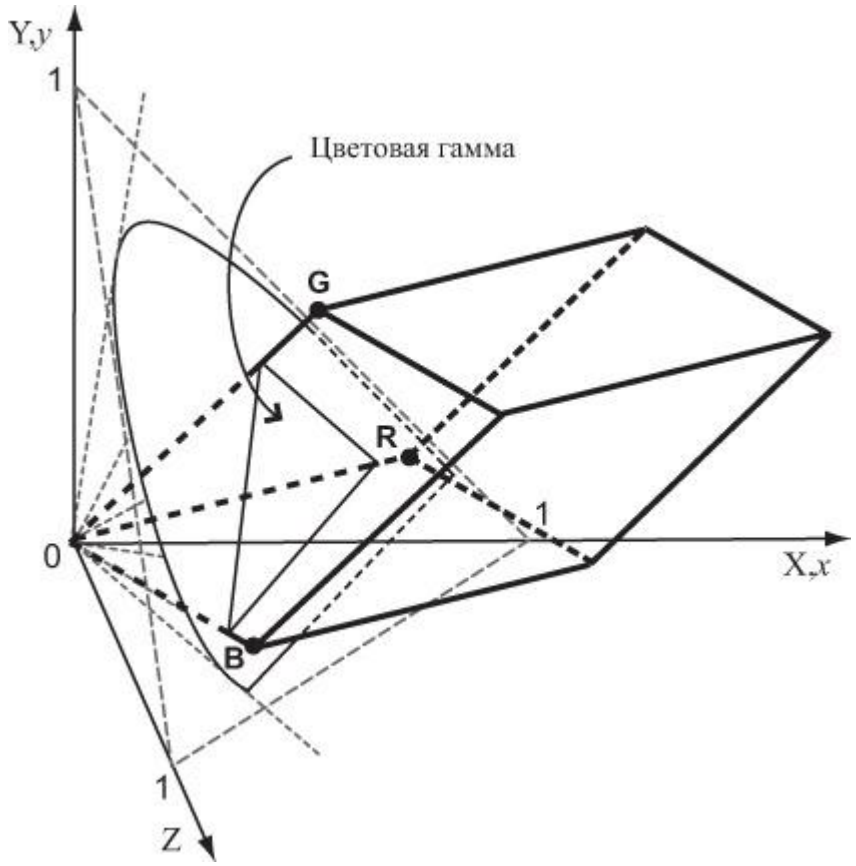


Рис. 4.23. RGB-параллелепипед в пространстве CIE XYZ.

Таким образом, в зависимости от координат (x, y, Y) для базисных цветов в конкретном устройстве (фотоаппарате, мониторе, проекторе и т.п.) одним и тем же значениям (R,G,B) соответствуют разные цвета.

Цветовые пространства, в которых каждому набору цветовых компонент соответствует физически единственный цвет, называются **абсолютными цветовыми пространствами**. Таким пространством является как раз CIE XYZ. Если мы также однозначно зафиксируем (x, y, Y) для базисных RGB-цветов, то получим абсолютное RGB-пространство. Такие стандартные пространства играют важную роль в обеспечении одинакового отображения одного и того же изображения на разных устройствах. Для корректного отображения на конкретном устройстве изображение надо перевести из абсолютного пространства в цветовое пространство для данного устройства. Для осуществления подобного преобразования программным путем информация о характеристиках устройства хранится в сопоставленном ему специальном файле. Стандарт на такие файлы был разработан ICC (англ. International Color Consortium), поэтому они получили название профилей ICC.

Наиболее широко распространенным абсолютным RGB-пространством является модель sRGB (хорошо отражает характеристики цифровых фотокамер любительского уровня), также были созданы Adobe RGB, AdobeWide Gamut RGB и ProPhoto RGB, каждая последующая с все более широкой цветовой гаммой для представления максимальной части цветов видимого спектра. В ProPhoto RGB "базисные цвета" R, G и B ради этого даже находятся за рамками зоны видимых цветов. Для представления цветов в таком широком диапазоне рекомендуется использовать повышенную точность с 16 или более бит/канал. Все эти модели представлены на рис. 4.24.

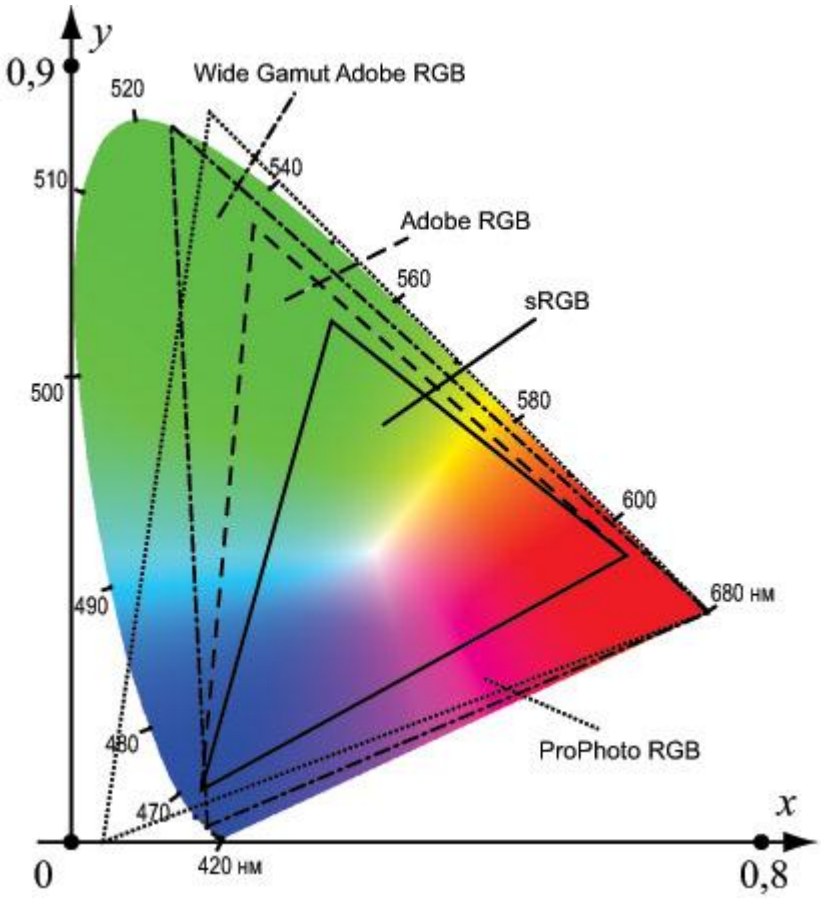


Рис. 4.24. Диаграмма цветности CIE с цветовыми гаммами для абсолютных пространств RGB.

Цветовые модели CIE L*u*v* и CIE L*a*b*

У модели CIE XYZ все же есть существенный недостаток - неоднородность восприятия изменения цвета относительно расстояния на диаграмме цветности. В идеале хотелось бы, чтобы одинаковые расстояния между точками, соответствующими цветам на диаграмме цветности, соответствовали приблизительно одинаковому восприятию человеком отличий между этими парами цветов. Именно для этих целей CIE в 1976 году предложила модель L*u*v*. L* (от англ. Lightness) в этой модели соответствует яркости, скорректированной с учетом особенностей человеческого восприятия. Получающаяся диаграмма цветности представлена на рис. 4.25. Эта модель рекомендуется для представления света от источников.

Определим функцию F(s) как

$$F(s) = \begin{cases} 7,787s + 16/116; & 0 \le s < 0,008856 \\ s^{1/3}; & s \ge 0,008856 \end{cases} \quad (1.3)$$

и определим

```
u' = 4X/(X + 15Y + 3Z);
v' = 9Y/(X + 15Y + 3Z);
```

Пусть точка белого имеет координаты (X_w, Y_w, Z_w), тогда получим следующий алгоритм преобразования:

```
L* = 116F(Y/Yw) - 16;
u* = 13L*(u' - u'w);
v* = 13L*(v' - v'w);
```

Листинг 1.2. Переход от XYZ к L*u*v* (html, txt)

В обратную сторону:

```
u' = u*/(13L*) + u'w;
v' = v*/(13L*) + v'w;
Y = F-1((L* + 16)/116)Yw;
X = 9Yu'/4v';
Z = (4X - 15v'Y - v'X)/3u';
```

Листинг 1.3. Переход от L*u*v* к XYZ (html, txt)

Также CIE в 1976 году с той же целью предложила и другую похожую модель L*a*b*, которая получила несколько более широкое распространение [21]. Эта модель рекомендуется для представления отраженного света.

В ней используется та же функция F(s), определенная в (1.3), и L* имеет то же самое значение, что в L*u*v*.

```
L* = 116F(Y/Yw) - 16;
a* = 500[F(X/Xw) - F(Y/Yw)];
b* = 200[F(Y/Yw) - F(Z/Zw)];
```

Листинг 1.4. Переход от XYZ к L*a*b* (html, txt)

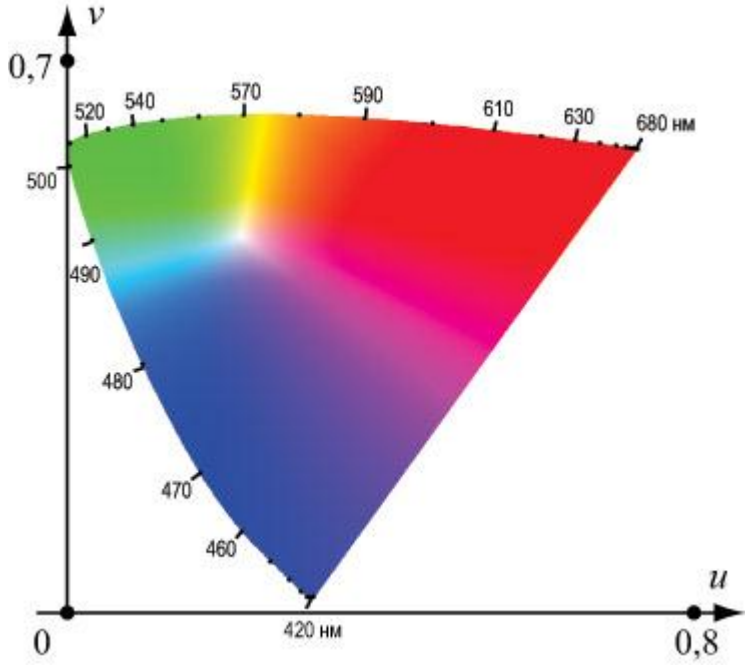


Рис. 4.25. Диаграмма цветности для CIE L*u*v*.

Преобразования между CIE XYZ и RGB

Цветовое пространство RGB, как и CIE XYZ, является трехмерным и аддитивным. Поэтому преобразования между двумя этими пространствами описываются матрицами 3 × 3; достаточно задать координаты базисных цветов R, G и B в системе CIE XYZ. Обычно удобно это делать, отдельно задавая цветовую информацию точками (x, y) на диаграмме цветности и яркостной компонентой Y. Если цвет задан таким образом (x, y, Y), то из формул (1.1) следует, что

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} \frac{Y}{y}x \\ Y \\ \frac{Y}{y}(1-x-y) \end{pmatrix}. \quad (1.2)$$

Тогда, если базисные RGB-цвета заданы как (x_R, y_R, Y_R), (x_G, y_G, Y_G), (x_B, y_B, Y_B), получаем следующую формулу преобразования:

```
zR=1-xR-yR;
zG=1-xG-yG;
zB=1-xB-yB;
```

Листинг 1.1. Переход от RGB к CIE XYZ (html, txt)

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{bmatrix} \frac{Y_R}{y_R}x_R & \frac{Y_G}{y_G}x_G & \frac{Y_B}{y_B}x_B \\ Y_R & Y_G & Y_B \\ \frac{Y_R}{y_R}z_R & \frac{Y_G}{y_G}z_G & \frac{Y_B}{y_B}z_B \end{bmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix};$$

Все представимые в модели RGB цвета образуют параллелепипед в CIE XYZ (см. рис. 1.10), пересечение которого с плоскостью X + Y + Z = 1, спроецированное на диаграмму цветности и дает цветовую гамму данной модели.

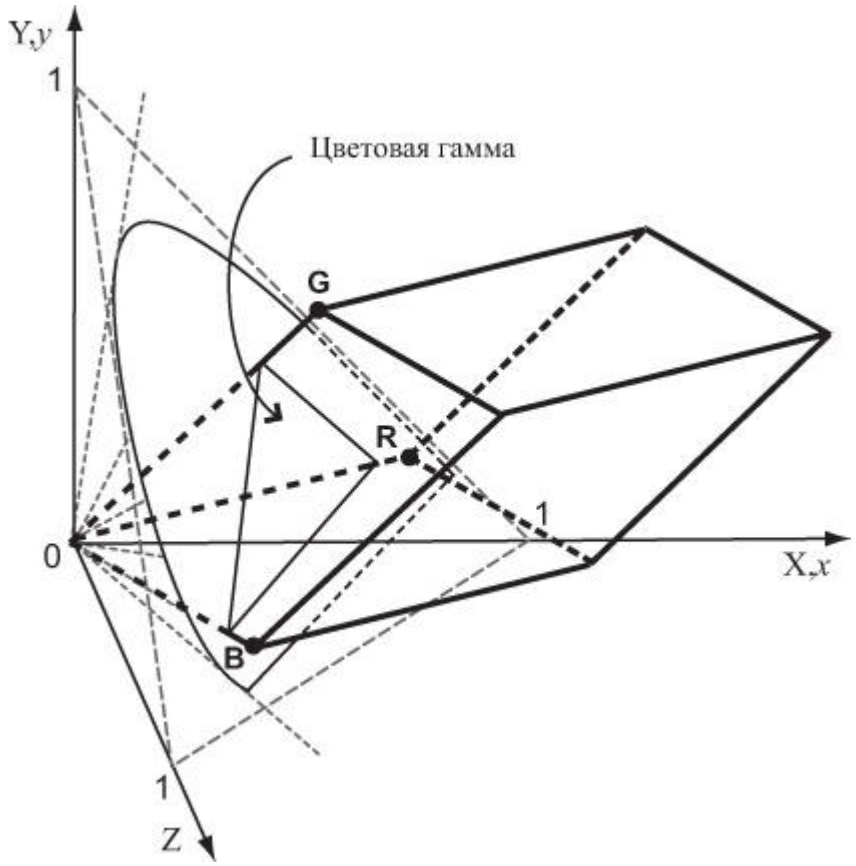


Рис. 4.26. RGB-параллелепипед в пространстве CIE XYZ.

Таким образом, в зависимости от координат (x, y, Y) для базисных цветов в конкретном устройстве (фотоаппарате, мониторе, проекторе и т.п.) одним и тем же значениям (R,G,B) соответствуют разные цвета.

Цветовые пространства, в которых каждому набору цветовых компонент соответствует физически единственный цвет, называются **абсолютными цветовыми пространствами**. Таким пространством является как раз CIE XYZ. Если мы также однозначно зафиксируем (x, y, Y) для базисных RGB-цветов, то получим абсолютное RGB-пространство. Такие стандартные пространства играют важную роль в обеспечении одинакового отображения одного и того же изображения на разных устройствах. Для корректного отображения на конкретном устройстве изображение надо перевести из абсолютного пространства в цветовое пространство для данного устройства. Для осуществления подобного преобразования программным путем информация о характеристиках устройства хранится в сопоставленном ему специальном файле. Стандарт на такие файлы был разработан ICC (англ. International Color Consortium), поэтому они получили название профилей ICC.

Наиболее широко распространенным абсолютным RGB-пространством является модель sRGB (хорошо отражает характеристики цифровых фотокамер любительского уровня), также были созданы Adobe RGB, AdobeWide Gamut RGB и ProPhoto RGB, каждая последующая с все более широкой цветовой гаммой для представления максимальной части цветов видимого спектра. В ProPhoto RGB "базисные цвета" R, G и B ради этого даже находятся за рамками зоны видимых цветов. Для представления цветов в таком широком диапазоне рекомендуется использовать повышенную точность с 16 или более бит/канал. Все эти модели представлены на рис. 4.27.

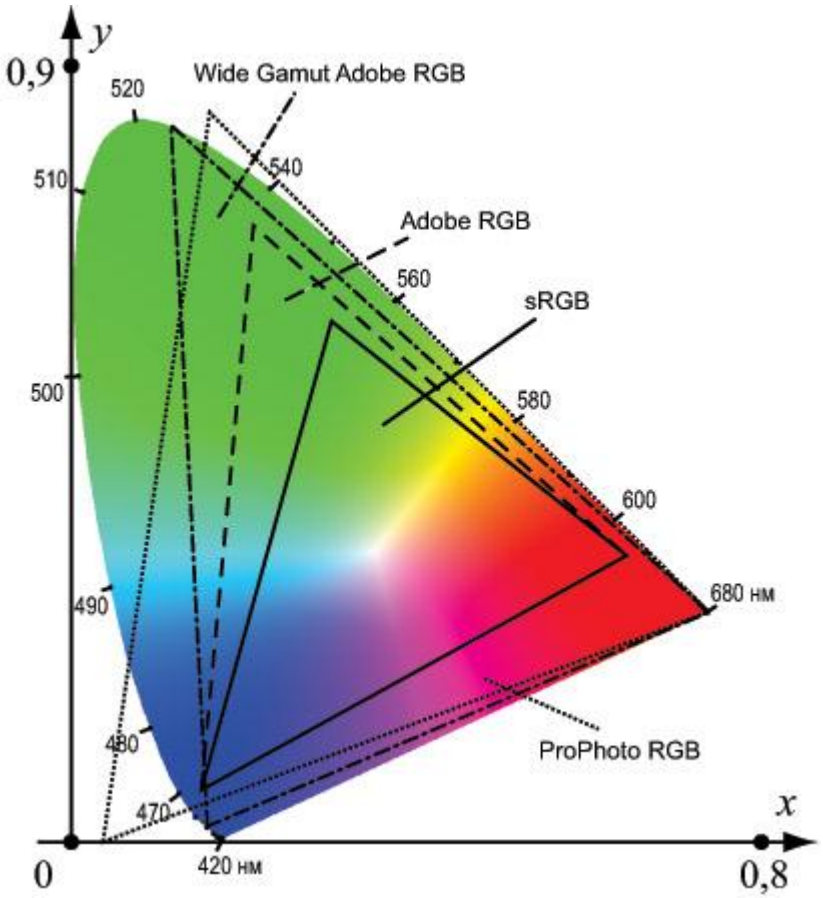


Рис. 4.27. Диаграмма цветности CIE с цветовыми гаммами для абсолютных пространств RGB.

Цветовые модели CIE L*u*v* и CIE L*a*b*

У модели CIE XYZ все же есть существенный недостаток - неоднородность восприятия изменения цвета относительно расстояния на диаграмме цветности. В идеале хотелось бы, чтобы одинаковые расстояния между точками, соответствующими цветам на диаграмме цветности, соответствовали приблизительно одинаковому восприятию человеком отличий между этими парами цветов. Именно для этих целей CIE в 1976 году предложила модель L*u*v*. L* (от англ. Lightness) в этой модели соответствует яркости, скорректированной с учетом особенностей человеческого восприятия [21]. Получающаяся диаграмма цветности представлена на рис. 1.12. Эта модель рекомендуется для представления света от источников.

Определим функцию F(s) как

$$F(s) = \begin{cases} 7,787s + 16/116; & 0 \le s < 0,008856 \\ s^{1/3}; & s \ge 0,008856 \end{cases} \quad (1.3)$$

и определим

```
u' = 4X/(X + 15Y + 3Z);
v' = 9Y/(X + 15Y + 3Z);
```

Пусть точка белого имеет координаты (X_w, Y_w, Z_w), тогда получим следующий алгоритм преобразования:

```
L* = 116F(Y/Yw) - 16;
u* = 13L*(u' - u'w);
v* = 13L*(v' - v'w);
```

Листинг 1.2. Переход от XYZ к L*u*v* (html, txt)

В обратную сторону:

```
u' = u*/(13L*) + u'w;
v' = v*/(13L*) + v'w;
Y = F-1((L* + 16)/116)Yw;
X = 9Yu'/4v';
Z = (4X - 15v'Y - v'X)/3u';
```

Листинг 1.3. Переход от L*u*v* к XYZ (html, txt)

Также CIE в 1976 году с той же целью предложила и другую похожую модель L*a*b*, которая получила несколько более широкое распространение [21]. Эта модель рекомендуется для представления отраженного света.

В ней используется та же функция F(s), определенная в (1.3), и L* имеет то же самое значение, что в L*u*v*.

$$L^* = 116F(Y/Y_w) - 16;$$
$$a^* = 500[F(X/X_w) - F(Y/Y_w)];$$
$$b^* = 200[F(Y/Y_w) - F(Z/Z_w)];$$

Листинг 1.4. Переход от XYZ к L*a*b* (html, txt)

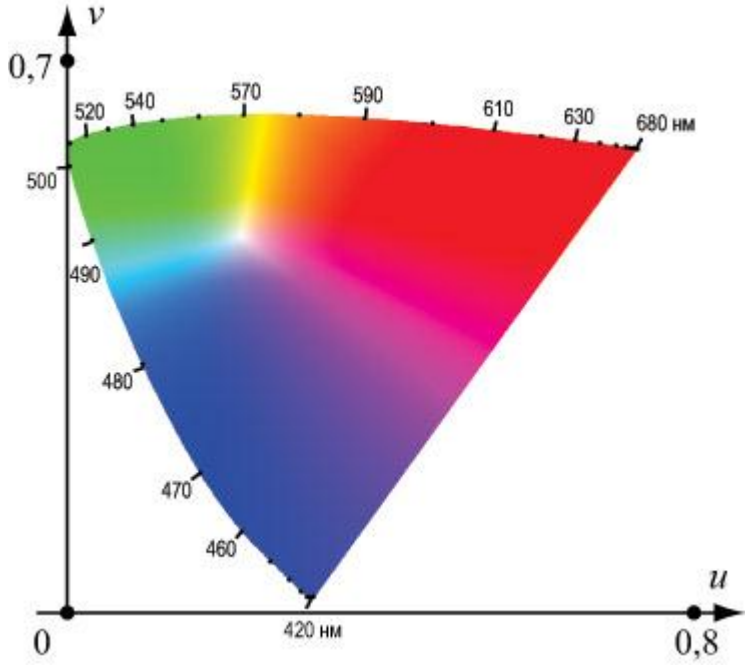


Рис. 4.28. Диаграмма цветности для CIE L*u*v*.

Цветовая модель HLS

Цветовая модель HLS (от англ. Hue, Lightness, Saturation - тон, светлота, насыщенность) схожа с моделью HSV. Снова рассмотрим цилиндрические координаты в трехмерном евклидовом пространстве, H - угол в горизонтальной плоскость от оси Ox, S - радиус в горизонтальной плоскости (расстояние до оси Oz), L - высота (по оси Oz). Все цветовое пространство представляет из себя две соединенные основаниями шестигранные пирамиды (см. рис. 4.27). На рис. 4.29 для наглядности вырезан один из шести секторов.

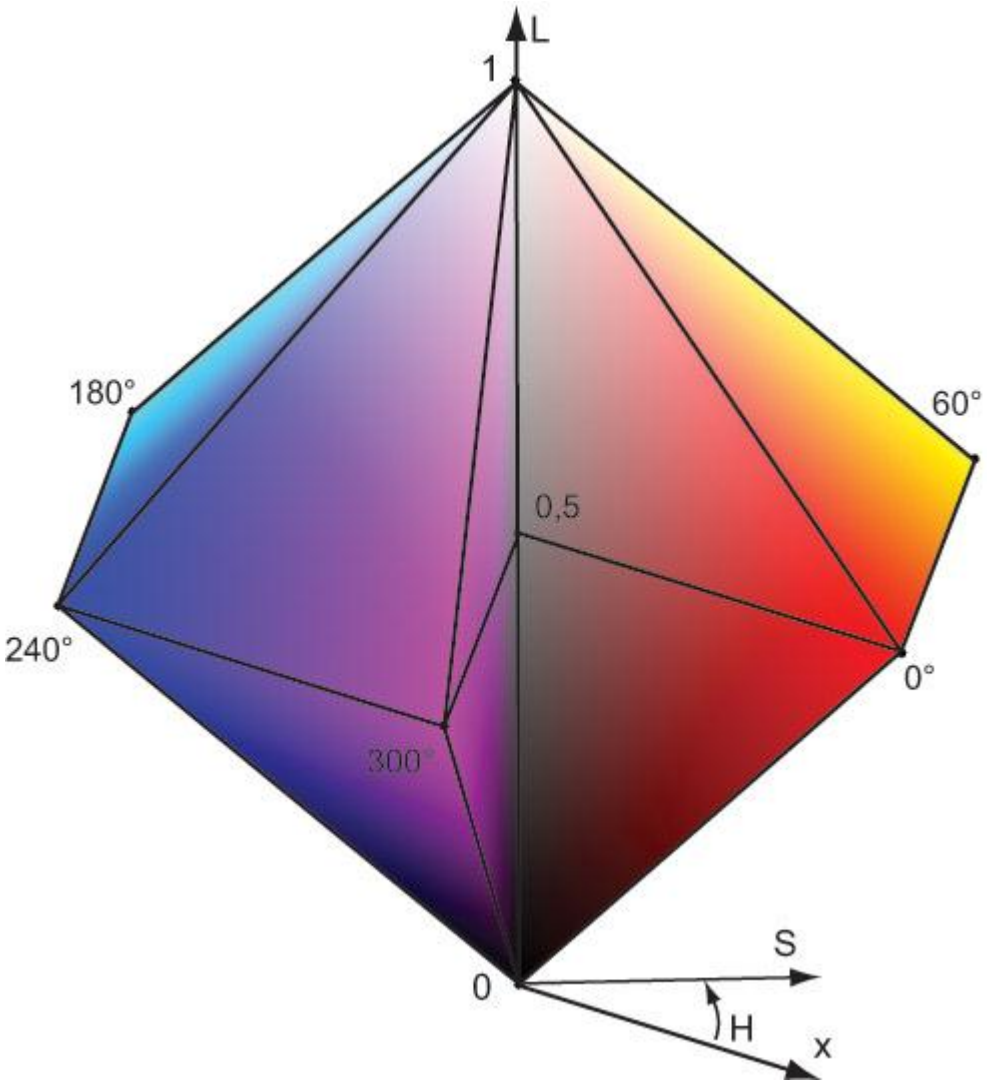


Рис. 4.29. Цветовая модель HLS.

Как видно на рис. 4.29, эта модель получена из HSV вытягиванием вдоль вертикальной оси. Понятия H и S остались теми же, только по вертикальной оси теперь L вместо V. Концептуальное различие состоит в том, что в этой модели считается, что движение от чистых цветов (у которых L = 0,5, S = 1) как в направлении белого, так и черного (а не только черного, как в HSV) одинаково приводит к уменьшению информации в H (вплоть до того, что в вершинах H не определено (как впрочем, и на всей вертикальной оси S = 0)) и сужению диапазона S.

Алгоритмы преобразования из RGB в HLS и обратно приведены ниже.

Листинг 1.9. Переход от RGB к HLS (html, txt)

Листинг 1.10. Переход от HLS к RGB (html, txt)

Цветовые модели Y**

Существует несколько тесно связанных цветовых моделей, которые объединяет то, что в них используется явное разделение информации о яркости и цвете. Компонента Y соответствует одноименной компоненте в модели CIE XYZ и отвечает за яркость. Такие модели находят широкое применение в телевизионных стандартах, так как исторически необходима была совместимость с черно-белыми телевизорами, которые принимали только сигнал, соответствующий Y. Также они применяются в некоторых алгоритмах обработки и сжатия изображений и видео.

Цветовые модели YUV, YPbPr и YCbCr

Рассмотрим цветовую модель YUV. U и V отвечают за цветовую информацию и определяются через преобразование из RGB:

```
Y  =  0,299R + 0,587G + 0,114B;
U  =  0,492(B - Y)
   = -0,147R - 0,289G + 0,436B;
V  =  0,877(R - Y)
   =  0,615R - 0,515G + 0,100B;
```

Листинг 1.11. Переход от RGB к YUV (html, txt)

Модель YUV применяется в телевизионной системе PAL.

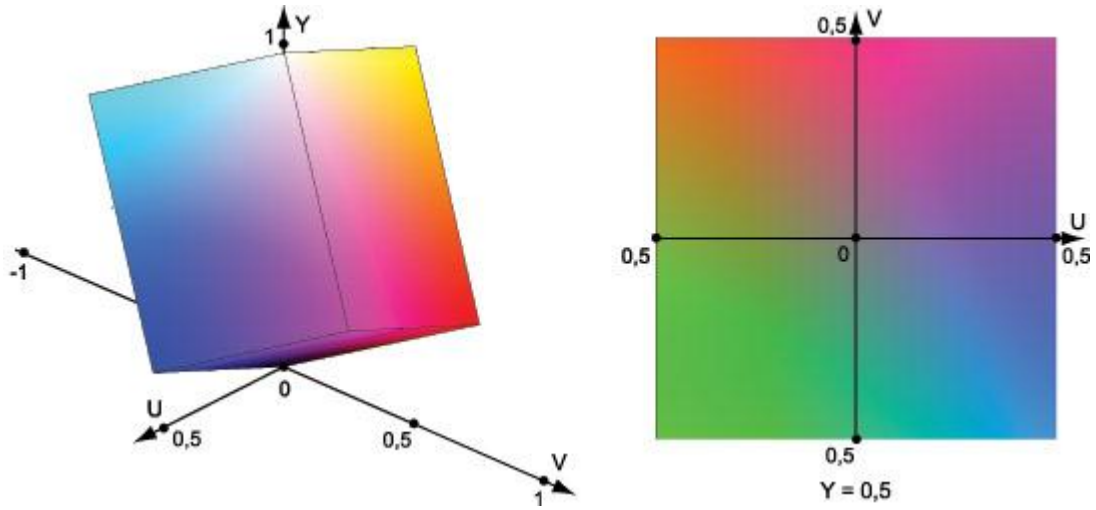


Рис. 4.30. RGB-куб в пространстве YUV, диаграмма UV при Y = 0,5.

Цветовые модели YCbCr и YPbPr являются вариациями YUV с другими весами для U и V (им соответствуют Cb/Pb и Cr/Pr). YPbPr применяется для описания аналоговых сигналов (преимущественно в телевидении), а YCbCr - для цифровых. Для их определения используются два коэффициента: Kb и Kr. Тогда преобразование из RGB в YPbPr описывается так:

Переход от RGB к YPbPr

$$Y = Kr \cdot R + (1 - Kr - Kb) \cdot G + Kb \cdot B;$$
$$Pb = \frac{1}{2} \cdot \frac{1}{1 - Kb} \cdot (B - Y);$$
$$Pr = \frac{1}{2} \cdot \frac{1}{1 - Kr} \cdot (R - Y);$$

Выбор Kb и Kr зависит от того, какая RGB-модель используется (это в свою очередь зависит от воспроизводящего оборудования). Обычно берется, как и выше, Kb = 0, 114; Kr = 0, 299. В последнее время также используют Kb = 0, 0722; Kr = 0, 2126, что лучше отражает характеристики современных устройств отображения.

Из приведенных выше формул следует что при $R, G, B \in [0, 1]$имеем следующие диапазоны $Y \in [0; 1]$; $Pb, Pr \in [-0,5; 0,5]$. Для цифрового представления эти формулы видоизменяют для получения только положительных дискретных коэффициентов в диапазонах

$$Y \in [minY, maxY], Cb, Cr \in [minC, maxC], minY, maxY, minC, maxC \in N \cup \{0\} :$$

Переход от RGB к YCbCr

$$Kg = 1 - Kr - Kb;$$
$$Y = minY + (maxY - minY) \cdot (Kr \cdot R + Kg \cdot G + Kb \cdot B);$$
$$Cb = \frac{minC + maxC}{2} +$$
$$\frac{maxC - minC}{2} \cdot \frac{1}{1 - Kb} \cdot (-Kr \cdot R - Kg \cdot G + (1 - Kb) \cdot B);$$
$$Cr = \frac{minC + maxC}{2} +$$
$$\frac{maxC - minC}{2} \cdot \frac{1}{1 - Kr} \cdot ((1 - Kr) \cdot R - Kg \cdot G - Kb \cdot B);$$

В телевидении обычно берут minY = 16, maxY = 235, minC = 16, maxC = 240. В стандарте сжатия изображений JPEG (см. раздел 14.4) используется полный 8-битный диапазон: minY = 0, maxY = 255, minC = 0, maxC = 255.

Цветовая модель YIQ

Цветовая модель YIQ применялась в телевизионной системе NTSC (I - от англ. in-phase, Q - от англ. quadrature; происходят от особенностей систем декодирования). Она тесно связана с моделью YUV, так как переход от YUV к YIQ является поворотом в плоскости UV = IQ на $33°$.

```
Y = 0,299R + 0,587G + 0,114B
I = 0,735(R - Y) - 0,268(B - Y)
  = 0,596R - 0,274G + 0,321B
Q = 0,478(R - Y) + 0,413(B - Y)
  = 0,211R - 0, 523G + 0,311B
```

Листинг 1.14. Переход от RGB к YIQ (html, txt)

Обратные преобразования для всех моделей получаются в результате применения обратной матрицы преобразования.

Вопросы и упражнения

7. Расположите в убывающем порядке чувствительность рецепторов глаза к цветам: красный, зеленый, синий.
8. Что такое хроматический спектр?
9. Что такое ахроматический спектр?
10. Как осуществляется проекция трехмерного цветового пространства на плоскость?
11. Чем отличается цветовой график МКО от треугольной проекционной области цветового пространства?
12. Что такое дополнительный цвет?
13. Что такое аддитивная и субстрактивная цветовые модели? Чем отличаются их цветовые кубы?
14. Что является основой цветовой модели HSV и HLS?
15. Являются ли цветовые модели HSV и HLS аддитивными или субстрактивными?
16. Постройте алгоритм преобразования модели RGB в HSV.
17. Постройте алгоритм преобразования модели RGB в HLS.
18. В чем состоит главное достоинство цветового пространства Luv?
19. В чем состоит главное достоинство цветового пространства Lab?

Лекция 5 Растровая и векторная графика. Понятие растра

Для представления графической информации на двумерной плоскости (например, экране монитора, странице книги и т.п.) в вычислительной технике применяются два основных подхода: растровый и векторный1).

При векторном подходе графическая информация описывается как совокупность неких абстрактных геометрических объектов, таких как прямые, отрезки, кривые, прямоугольники и т.п.

Растровая графика же оперирует изображениями в виде растров. Неформально можно сказать, что растр - это описание изображения на плоскости путем разбиения всей плоскости или ее части на одинаковые квадраты и присвоение каждому квадрату своего цветового (или иного, например, прозрачности, для последующего наложения изображений друг на друга) атрибута. Если таких квадратов имеется конечное число, то получается, что непрерывная цветовая функция изображения приближенно представлена конечной совокупностью значений атрибутов. Иногда понятие растра определяют более широко: как разбиение плоскости (или ее участка) на равные элементы (т.е."замощение"), например шестиугольниками (гексагональный растр). Далее в этой книге расширенное толкование использоваться не будет.

С другой стороны, растр можно рассматривать как кусочно-постоянную аппроксимацию изображения, заданного как цветовая функция на плоскости. Такая точка зрения позволяет применять математический аппарат теории аппроксимации для работы с растровыми изображениями, о чем подробнее будет рассказано далее.

Формально, введем следующие определения:

Растр (англ. raster) - отображение вида

$$f: X \times Y \to 2^{R^2} \times C,$$

где , $X \subset Z$,$Y \subset Z$

2^{R^2} обозначает множество всех подмножеств R^2,

C - множество значений атрибутов (как правило, цвет).

f(i, j) - элемент растра, называемый пикселем (англ. pixel (от picture element)), в русскоязычной литературе иногда также переводится как пиксел);

f(i, j) = (A(i, j),C(i, j)), где

$A(i,j) \subset R^2$ - область пикселя,

$C(i,j) \in C$ - атрибут пикселя (как правило, цвет). Чаще всего мы будем пользоваться следующими двумя видами атрибутов:

C(i, j) = I(i, j) - интенсивность (или яркость) пикселя;
C(i, j) = {R(i, j),G(i, j),B(i, j)} - цветовые атрибуты в цветовой модели RGB

Также иногда будут употребляться матричные обозначения:
Mij = (Aij ,Cij)
Aij может определяться двояко, в зависимости от того, с какой моделью мы хотим работать:
Aij := (i, j) - одна точка. Пример такой модели растра см. на рис. 1.1;
- квадрат. Пример такой модели растра см. на рис. 1.2. На реальных графических устройствах физически пиксели могут быть прямоугольниками, что иногда порождает дополнительные трудности.

В реальности, как правило, X и Y - ограниченные наборы неотрицательных целых чисел; такой растр называется прямоугольным. Для него применимо понятие Аспектовое отношение (англ. aspect ratio) - отношение ширины к высоте растра (|X|/|Y|). Чаще всего такое понятие употребляется в связи с физическими растрами (дисплеями, ПЗС-матрицами фотоаппаратов и т.д.) и записывается в виде простой дроби с ":", например "4:3".

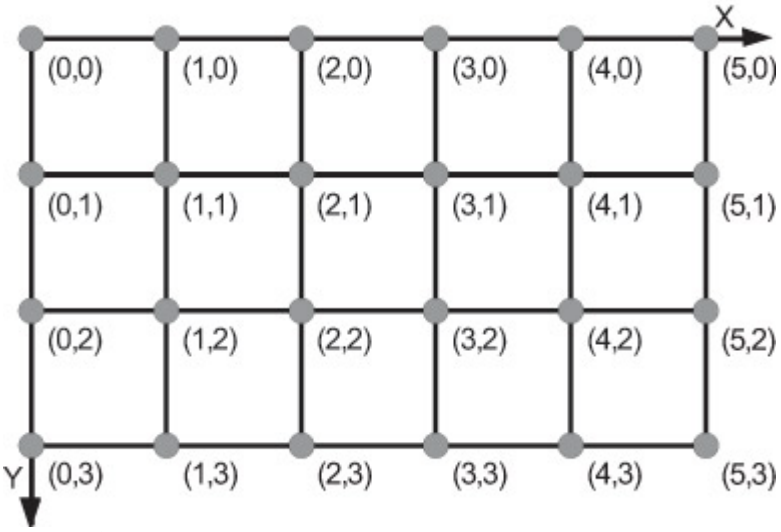


Рис. 5.1. Модель растра первого типа.

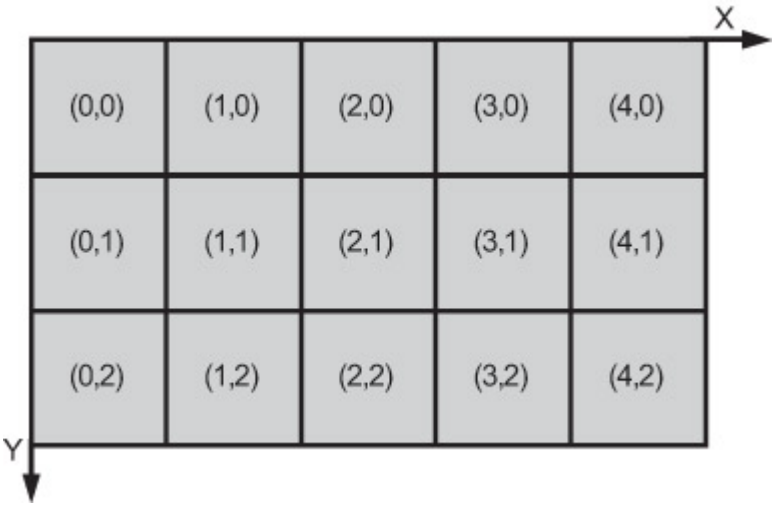


Рис. 5.2. Модель растра второго типа.

Бесконечные растры (когда X и Y неограниченны) бывают удобны для описания алгоритмов, позволяя избежать особых ситуаций. Впрочем, самой сутью некоторых алгоритмов является как раз работа с граничными случаями.

Растровое представление является естественным в тех случаях, когда нам не известна дополнительная информация об изображаемых объектах (например, цифровым фотоаппаратом можно снимать изображения произвольного содержания). В случае же векторного описания примитивами являются более сложные объекты (линии и области, ограниченные линиями), что предполагает априорные знания о структуре изображения. В последнее время проявляется ярко выраженная тенденция к преобладанию устройств ввода-вывода двумерной графической информации, основанных на растровом принципе как более универсальном. Возникающая при

выводе задача отображения геометрических объектов, заданных их математическим описанием (например, координатами концевых точек и цветом для отрезка), на растре, называемая растеризацией, рассмотрена в последующих разделах.

При построении алгоритмов, работающих с изображениями, можно также пользоваться информацией как непосредственно атрибутов пикселей, так и работать с примитивами более высокого порядка. В данной книге в основном рассматриваются алгоритмы первого типа, про которые говорят, что они работают в пространстве изображения (англ. image space), тогда как вторые работают в объектном пространстве (англ. object space) (эти термины чаще употребляются в трехмерной графике).

Введение в растеризацию кривых

Пусть у нас есть некоторая кривая, и мы хотим построить ее изображение на растровой решетке. Возникает вопрос: какие из ближайших пикселей следует закрашивать? В данной и следующей лекциях мы рассмотрим случай построения на монохромном растре, когда возможны только два уровня интенсивности закраски пикселя - "полностью закрашен" или "полностью не закрашен". Если же допустимы несколько уровней интенсивности, то можно растеризовывать более аккуратно, уменьшая эффекты алиасинга (т.е. ступенчатости), см. раздел 7.2.

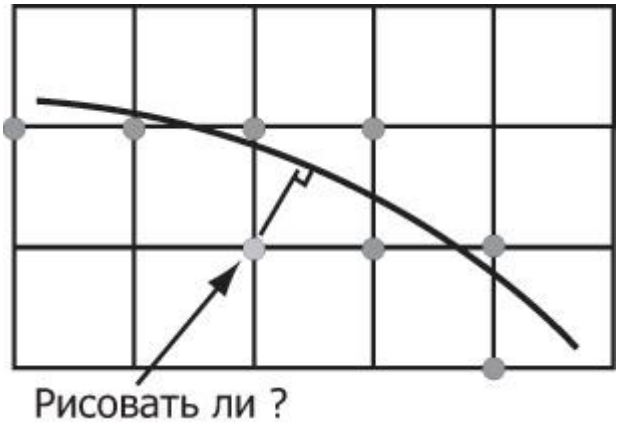
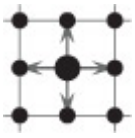


Рис. 5.3. Изображение кривых на растре.

Пусть (x0, y0) - фиксированный пиксель, а (x, y) - некоторый другой пиксель на плоскости. Тогда для определения их близости вводятся следующие понятия:

1. 4-связность



$|x-x0|+|y-y0|=1$

2. 8-связность

$\max\{|x-x0|, |y-y0|\}=1$

В дальнейших рассуждениях расстояние будем считать заданным стандартной евклидовой метрикой1).

Изображение отрезка с целочисленными координатами концов

Пусть наш отрезок - это AB. Перейдем от системы координат Oxy к Ax'y' (см. рис. 3.2, этап 1). Отрезок может лежать в любом из 8 октантов, но всегда существуют симметрии относительно осей, разделяющих эти октанты, симметрии определяются матрицами

$$\begin{pmatrix} \pm 1 & 0 \\ 0 & \pm 1 \end{pmatrix}$$

и

$$\begin{pmatrix} 0 & \pm 1 \\ \pm 1 & 0 \end{pmatrix},$$

позволяющие свести задачу к случаю отрезка, лежащего в первом октанте (пример см. на рис. 5.4, этап 2, в нем матрица имеет вид

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

Назовем такой случай каноническим, в дальнейшем будут рассмотрены алгоритмы для этого случая. В каноническом случае процесс рисования 8-связной линии можно закодировать последовательностью вида: sdssd... (см. рис. 5.5), где

s - горизонтальное смещение;

d - диагональное смещение.

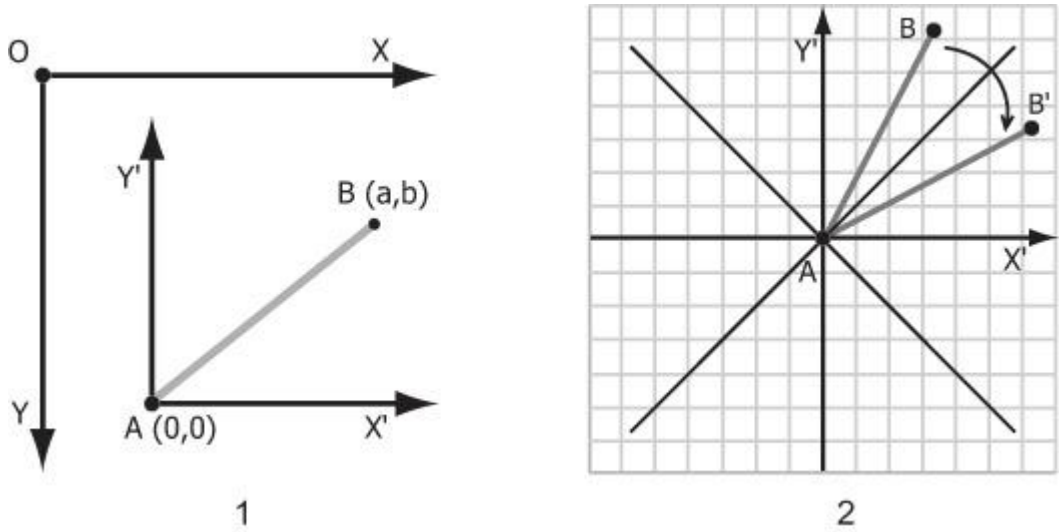


Рис. 5.4. Переход к каноническому случаю в два этапа.

Эквивалентно этой последовательности можно сопоставить бинарный код, где 0 соответствует s, а 1 соответствует d. Такой код для рисования отрезка называется кодом Ротштейна для $\frac{p}{q}$.

Пусть plot(x,y) - функция, закрашивающая точку растра с координатами (x,y).

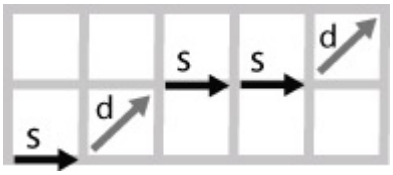
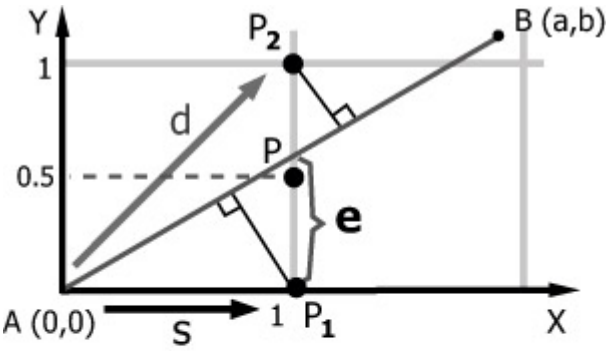


Рис. 5.5. Кодирование закрашивания отрезка (или код Ротштейна).

Цифровой дифференциальный анализатор

Алгоритм Цифровой дифференциальный анализатор (англ. DDA - Digital Differential Analyzer) строит 8-связную линию.

Для начала, пусть P1 = (1, 0); P2 = (1, 1). Для определения того, какой из пикселей, - P1 или P2, - следует закрасить, сравним расстояния до них. В силу подобия треугольников, образованных пересечением рисуемого отрезка, прямой x = 1 и перпендикулярами из P1 и P2 на отрезок (см. рис. 5.6), достаточно сравнить e (ординату пересечения отрезка с прямой x = 1) с $\frac{1}{2}$. Далее, для следующего шага алгоритм работает аналогично с учетом изменения e - ординаты пересечения отрезка со следующей вертикальной прямой $x = k, k \in N$.



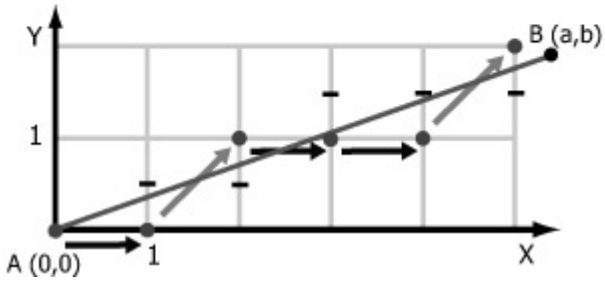


Рис. 5.6. Цифровой дифференциальный анализатор.

```
// Координаты концов отрезка - (0,0) и (a,b)

e = b/a; // Текущая ордината
Δe = b/a; // Приращение ординаты

// (x,y) - Координаты текущей точки
x = 0; y = 0;

while( x < a )
{
    plot(x, y);

    if( e > 1/2 )
    {
        // d : диагональное смещение
        x++; y++;

        // т.к. произошло смещение по y на 1 вверх
        e += Δe - 1;
    }
    else
    {
        // s : горизонтальное смещение
        x++;
        e += Δe;
    }
}
```

Листинг 3.1. Цифровой дифференциальный анализатор (html)

Недостатком данного алгоритма является то, что он работает с числами с плавающей точкой.

Алгоритм Брезенхема

Брезенхем модифицировал алгоритм DDA, чтобы он работал в целых числах. Модифицируем алгоритм следующим образом:

уменьшим везде e на $\frac{1}{2}$, чтобы сравнивать с 0;
домножим e и Δe на 2a: e0 = 2b - a, Δe = 2b

Приходим к следующему алгоритму:

```
// Координаты концов отрезка - (0,0) и (a,b)
e = 2b - a;
ΔeS = 2b;
ΔeD = 2b - 2a;

x = 0; y = 0; // (x,y) - Координаты текущей точки

while(x < a)
{
    plot(x, y);

    if(e > 0)
    { // d : диагональное смещение
        x++; y++;
        e += ΔeD;
    }
    else
    { // s : горизонтальное смещение
        x++;
        e += ΔeS;
    }
}
```

Листинг 3.2. Алгоритм Брезенхема для отрезка (html)

Алгоритм Брезенхема был создан им для вывода отрезков на цифровых инкрементальных графопостроителях, которые могли осуществлять лишь простые единичные сдвиги печатающей головки. Дальнейшая оптимизация может быть произведена, если заметить, что отрезок симметричен относительно прямой, проходящей перпендикулярно ему через его середину; в этом случае можно начинать рисовать сразу с двух концов, что сократит число итераций цикла в алгоритме вдвое.

Алгоритм Кастла-Питвея

Этот алгоритм гораздо менее эффективен с вычислительной точки зрения, чем алгоритм Брезенхема, однако обладает красивой математической структурой. Он основан на идее, схожей с известным алгоритмом Евклида нахождения Наибольшего Общего Делителя двух натуральных чисел [19].

Будем работать со строками, кодирующими последовательность закраски (т.е. состоящими из символов s и d, определенных выше).

Определим две операции над такими строками:

$\oplus$ - конкатенация строк, например

$$"ssds" \oplus "sddd - "ssdssddd"$$

$\sim$ - "переворот" строки, например

$$\sim ("ssdds") = "sddss"$$

```
// Координаты концов отрезка - (0,0) и (a,b)
y = b;
x = a - b;

m1 = "s";
m2 = "d";

while( x \ne y )
{
    if( x > y )
```

```
    {
        x = x - y;
        m2 = m1   ~ m2;
    }
    else
    {
        y = y - x;
        m1 = m2    ~ m1;
    }
}
```

Листинг 3.3. Алгоритм Кастла-Питвея (html, txt)

После завершения работы алгоритма $m = m_2 \oplus \sim m_1$ задает нужную последовательность сдвигов. Доказательство корректности работы этого алгоритма мы опустим ввиду его громоздкости.

Изображение отрезка с нецелочисленными координатами концов

Для отрезка с нецелочисленными координатами концов будем строить соответствующую 4-связную линию на растре.

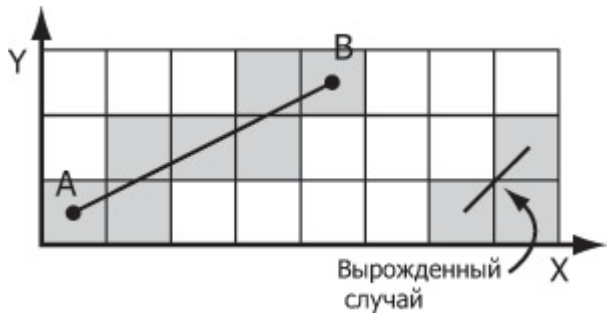


Рис. 5.7. Рисование отрезка с нецелочисленными координатами концов.

Существует два подхода.

1.Округлить координаты концов до целочисленных и воспользоваться алгоритмом для целочисленного случая. Недостаток: может вызывать существенные искажения (особенно в случае отрезков небольшой длины).

2.Перейдем к нашему каноническому случаю, который теперь характеризуется тем, что отрезок лежит в первом октанте, но координаты в этом случае $A(x_A, y_A)$: $x_A \in [0,1), y_A \in [0,1)$. Параметризуем наш отрезок стандартным образом:

$$(x, y) = A + t \cdot \frac{B - A}{c}, t \in [0, c],$$

где A и B - концевые точки, c > 0 - некий масштабный коэффициент. Сделаем с достаточно большим целым числом, чтобы уменьшить ошибки округления. Тогда рассмотрим

$$\Delta h = \frac{c}{x_B - x_A}$$

- приращение t, при сдвиге на 1 пиксель по x;

$$\Delta v = \frac{c}{y_B - y_A}$$

- приращение t, при сдвиге на 1 пиксель по y.

Будем сравнивать текущие значения h и v, а затем, в зависимости от этого, делать шаг по x или y и придавать соответствующие приращения h и v. Алгоритм закончится, когда h или v превысит c.

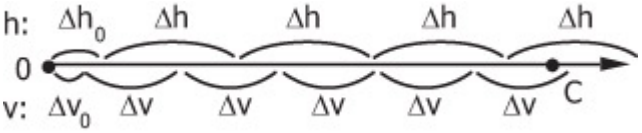


Рис. 5.8. Изменение параметров h и v.

```
x = 0; y = 0; // Канонический случай: начальная точка
// лежит в [0, 1)   [0, 1)

/* Приращения t, соответствующие смещениям от начальной
точки до границ первого пикселя. */

h = Δh * (1 - xA); // Δh0
v = Δv * (1 - yA); // Δv0

while( (h < c) AND (v < c) )
{
    plot(x, y);

    if( h < v )
    {
        // Сдвиг по горизонтали
        x++; h += Δh;
    }
    else if( h > v )
    {
        // Сдвиг по вертикали
        y++; v += Δv;
    }
    else
    {
        // h = v : Вырожденный случай (см. рис. 3.5)
        // рисуем произвольный из двух возможных пикселей,
        // например, верхний:
        plot(x,y+1);
        x++; y++;
        h += Δh; v += Δv;
    }
}
```

Листинг 3.4. Алгоритм отображения отрезка с нецелочисленными координатами концов (html)

Замечание. Приведенный выше алгоритм легко обобщается на n-мерный случай.

Изображение окружностей

Для начала перейдем к канонической системе координат, в которой центр окружности совпадает с началом координат. Тогда можно заметить, что в силу симметрии окружности относительно прямых, разделяющих октанты, достаточно построить растровое представление в одном октанте, а затем с помощью симметрий получить изображения в других октантах (см. рис. 5.9). Будем пользоваться заданием окружности в виде неявной функции: x2 + y2 - R2 = 0.

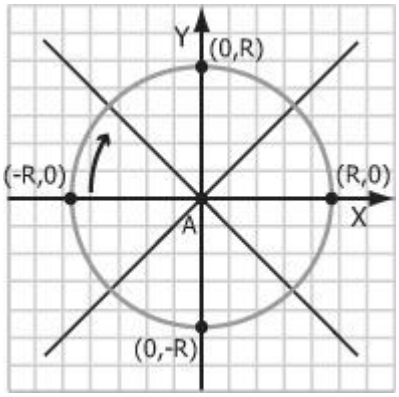


Рис. 5.9. Симметрии при изображении окружности.

Пусть f(x, y) = x2 + y2 - R2. Будем рисовать часть окружности в 4-м октанте, начиная с точки (-R, 0) (см. рис. 3.7, показано стрелкой).

Пусть $R \in N$, тогда $f(x,y) \in Z$ для $x \in Z, y \in Z$. Пусть функция plot8(x, y) отображает на растре все 8 точек, полученных из (x, y) с помощью симметрий.

Алгоритм Брезенхема

Будем рассуждать подобно алгоритму Брезенхема для отрезков (с соответствующими поправками на 4-й октант) [17]. Из двух возможных пикселов в 4-м октанте (соответствующих вертикальному и диагональному смещениям, которые обозначаются аналогично прежним s и d, см. рис. 5.10) будем выбирать тот, расстояние от окружности до которого меньше.

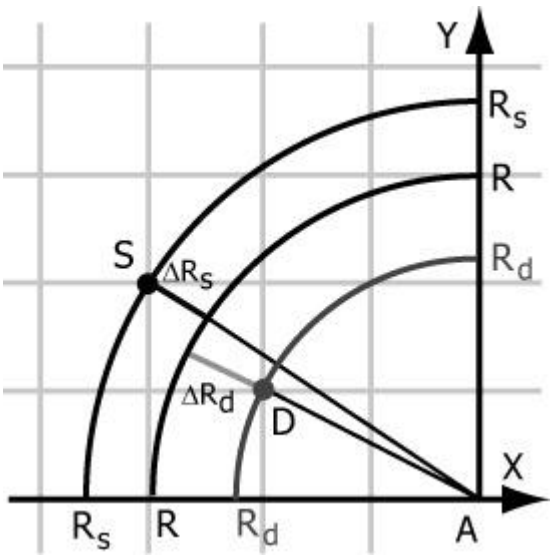


Рис. 5.10. Расстояния до окружности.

Для того чтобы выбрать один из двух возможных пикселей, будем сравнивать расстояния от них до окружности: где расстояние меньше - тот пиксел и будет искомым. В примере на рис. 3.8 сравниваются расстояния от точек S(xs, ys) и D(xd, yd) до окружности с радиусом R. Из евклидовой метрики получаем:

$$\Delta R_s = \sqrt{x_s^2 + y_s^2} - R;$$
$$\Delta R_d = R - \sqrt{x_d^2 + y_d^2}.$$

Но вычисление квадратного корня - вычислительно трудоемкая операция, поэтому при достаточно больших R мы будем заменять сравнение расстояний сравнением приближенных значений их квадратов (см. рис. 3.9):

$$(\Delta R_s)^2 - (\Delta R_d)^2 = x_s^2 + y_s^2 - 2R\sqrt{x_s^2 + y_s^2} + R^2 - x_d^2 - y_d^2 +$$
$$+ 2R\sqrt{x_d^2 + y_d^2} - R^2.$$

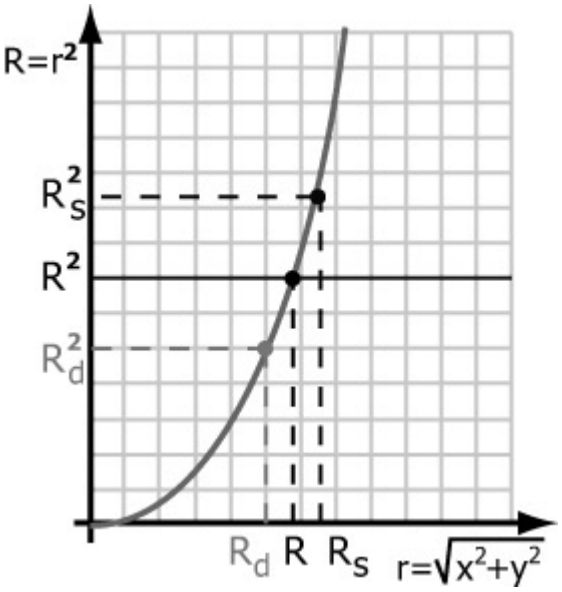


Рис. 5.11. Приближенное сравнение расстояний.

Уменьшим два слагаемых на приблизительно одинаковые величины:
$-2R\sqrt{x_s^2+y_s^2}$ заменим на $-2R\cdot R$,

$+2R\sqrt{x_d^2+y_d^2}$ заменим на $+2\sqrt{x_d^2+y_d^2}\cdot\sqrt{x_d^2+y_d^2}$
получим
$$(\Delta R_s)^2-(\Delta R_d)^2\approx x_s^2+y_s^2-2R\cdot R+R^2-$$
$$-x_d^2-y_d^2+2\sqrt{x_d^2+y_d^2}\cdot\sqrt{x_d^2+y_d^2}-R^2=$$
$$=x_s^2+y_s^2+x_d^2+y_d^2-2R^2.$$
Таким образом, приближенно
D ближе к окружности, чем S $\Leftrightarrow$
$x_s^2+y_s^2+x_d^2+y_d^2-2R^2>0;$
S ближе к окружности, чем D $\Leftrightarrow$
$x_s^2+y_s^2+x_d^2+y_d^2-2R^2<0.$

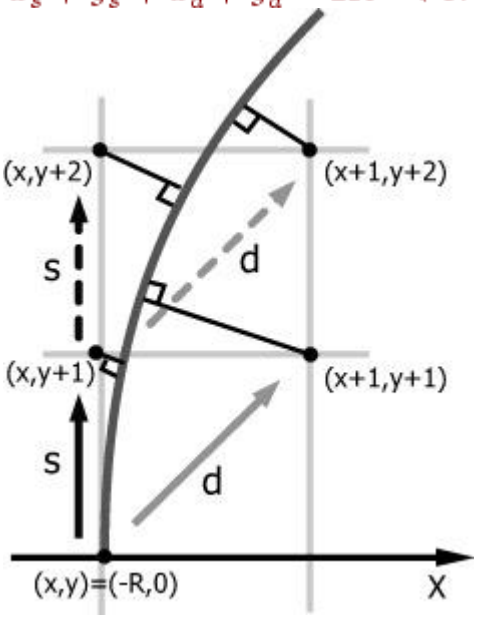


Рис. 5.12. Алгоритм Брезенхема для окружности.

Пусть (x, y) - текущий пиксель. Обозначим

$$F_s = f(x, y+1) = x^2 + (y+1)^2 - R^2 > 0,$$
$$F_d = f(x+1, y+1) = (x+1)^2 + (y+1)^2 - R^2 < 0,$$
$$F = F_s + F_d;$$
$$\Delta F(s) = (\Delta F \text{ при переходе } s\colon (x,y) \to (x, y+1)) =$$
$$= f(x, y+2) + f(x+1, y+2) - f(x, y+1) -$$
$$- f(x+1, y+1)$$
$$= 4y + 6,$$
$$\Delta F(d) = (\Delta F \text{ при переходе } d\colon (x,y) \to (x+1, y+1)) =$$
$$= f(x+1, y+2) + f(x+2, y+2) - f(x, y+1) -$$
$$- f(x+1, y+1)$$
$$= 4x + 4y + 10.$$

Тогда из двух возможных смещений d и s выберем.

1. $F = F_s + F_d < 0 \Leftrightarrow F_s < -F_d$, т.е. (x + 1, y + 1) ближе к окружности, чем (x, y + 1): d: Переходим в (x + 1, y + 1) и придаем соответствующие приращения F, ΔF(s), ΔF(d): F = F +ΔF(d); ΔF(s) = ΔF(s)+4; ΔF(d) = ΔF(d)+8.

2. $F = F_s + F_d < 0 \Leftrightarrow F_s < -F_d$, т.е. (x, y+1) ближе к окружности, чем (x + 1, y + 1): s: Переходим в (x, y + 1) и придаем соответствующие приращения F, ΔF(s), ΔF(d): F = F +ΔF(s); ΔF(s) = ΔF(s)+4; ΔF(d) = ΔF(d)+4.

Если мы начинаем из (-R, 0), то начальные значения будут следующими:

$$F = F_s + F_d = ((-R)^2 + 1^2 - R^2) + ((-R+1)^2 + 1^2 - R^2)$$
$$= 1 + (-2R + 1 + 1) = 3 - 2R,$$
$$\Delta F(s) = 4 \cdot 0 + 6 = 6,$$
$$\Delta F(d) = 4 \cdot (-R) + 4 \cdot 0 + 10 = 10 - 4R.$$

Легко видеть, что в алгоритме все величины, связанные с F, кроме F начального, будут кратны 2. Но, если мы поделим все эти величины на 2 (в дальнейшем значения всех величин уже понимаются в этом смысле), то $F_{\text{нач}} = \frac{1}{2} + 1 - R$. Так как приращения F могут быть только целочисленными, то $F = \frac{1}{2} + T$, где $T \in Z$; т.е. если отнять $\frac{1}{2}$ от всех значений, то знак F не изменится для всех T, кроме T = 0. Для того чтобы результат сравнения остался прежним, будем считать, что F = 0 теперь соответствует смещению s.

x = -r; y = 0;

F = 1-r;
ΔFs = 3;
ΔFd = 5-2*r;

```
while( x + y < 0 )
{
    plot8( x, y );

    if( F > 0 )
    {
        // d: Диагональное смещение
        F += ΔFd;
        x++; y++;
        ΔFs += 2;
        ΔFd += 4;
    }
    else
    {
        // s: Вертикальное смещение
        F += ΔFs;
```

```
            y++;
            ΔFs += 2;
            ΔFd += 2;
        }
    }
```
Листинг 3.5. Алгоритм Брезенхема для окружности (html)

Размерность вычислений этого алгоритма (т.е. отношение максимальных модулей значений величин, с которыми он оперирует к модулям исходных данных (в данном случае R)) равна 2.

Изображение эллипсов

Прежде всего отметим, что у эллипса, в отличие от окружности, всего 2 оси симметрии, поэтому по точкам придется строить уже 2 октанта (см. рис. 5.13).

Рис. 5.13. Изображение эллипса.

Построение по неявной функции

Будем рассуждать подобно алгоритму Брезенхема для окружностей.

Неявная функция, задающая эллипс, имеет вид

$$\left(\frac{x}{a}\right)^2 + \left(\frac{y}{b}\right)^2 - 1 = 0, a > b$$

$$b^2x^2 + a^2y^2 - a^2b^2 = 0$$

Введем f(x, y) = b2x2 + a2y2 - a2b2.

Аналогично алгоритму для окружности можно сравнивать f для двух возможных вариантов. Подробный вывод оставляем читателю в качестве упражнения.

Построение путем сжатия окружности

Воспользуемся тем, что эллипс с параметрами a, b (пусть a > b) получается из окружности радиуса a сжатием по оси y в a/b раз. Построим алгоритм, который является некой комбинацией алгоритмов Брезенхема для окружности и для отрезка (см. рис. 3.14).

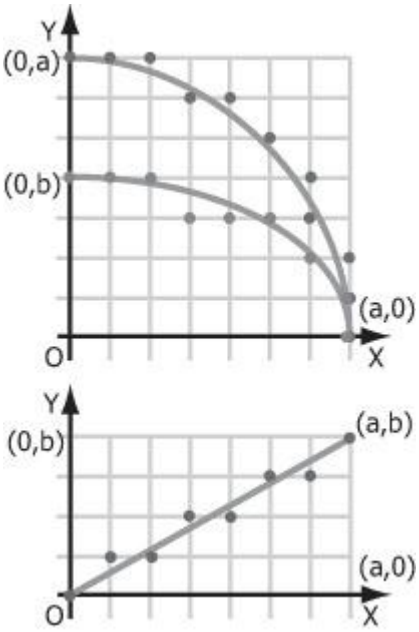


Рис. 5.14. Построение эллипса путем сжатия окружности.

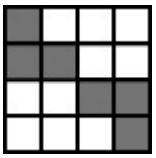


Рис. 5.15. Смешанная связность.

Начнем из точки (a, 0) на окружности и из точки (0, 0) на отрезке. Будем строить эллипс точно так же, как окружность, но смещать текущую точку по y только в том случае, когда такое смещение происходит в текущем шаге уже для отрезка, т.е. построение отрезка как раз и является реализацией сжатия в a/b раз (точнее, его дискретной аппроксимацией). Этот алгоритм тоже имеет недостаток: возможная смешанная связность полученной линии (см. рис. 5.15).

Существуют ли бесконечные растры?

нет, поскольку не существует бесконечных устройств вывода изображения
нет, поскольку не существует бесконечных устройств ввода изображения
да, когда X и Y неограниченны

Сколько точек белого для описания дневного света солнца существует в модели CIE XYZ?

не существует такой точки
такая точка одна с значениями цветности равными 1/3
такая точка одна при X + Y + Z = 1
точек белого бесконечно много

Цветовое пространство CIE XYZ является

профилирующим
трехмерным
аддитивным.

Лекция 6. Основные понятия и определения Растровых изображений

Предположим, что цифровое изображение **f** состоит из подмножества элементов S, обладающих свойством Q, которое устанавливается на основе некоторого описания (яркости, цвета, текстуры и т.д.) и подмножества элементов S, обладающих свойством Q, причем **f=SUS.**

Соседние точки, окрестности

Если точка p с координатами (i, j), обладающая свойством Q, имеет 4-х соприкасающихся по горизонтали и вертикали соседей (i+1, j), (i-1, j), (i, j+1), (i, j+1) и обладающих свойством Q, то эта группа точек называется «четыре непосредственных соседа p» (4-элементная окрестность) , обозначается через $N_4(p)$ f(a)

$N_4(p)$

	, +1	
-1, j	, j	+1, j
	, -1	

$N_D(p)$

-1, j+1		+1, j+1
	, j	
-1, j-1		+1, j-1

$N_8(p)$

-1, j+1	, +1	+1, j+1
-1, j	, j	+1, j
-1, j-1	, -1	+1, j-1

рис. 6а рис. 6б рис.6в

Группа точек, изображенная на рис. 6б, называется «четыре косвенных соседа p»(D-окрестность или диагональная 4-элементная окрестность. Обозначается через $N_D(p)$.

Объединение точек, входящих в $N_4(p)$ и $N_D(p)$ образует «восемь соседей p»(8-элементная окрестность) и обозначается через $N_8(p)$ рис. 1в.

Существуют и другие типы соседства, например «шесть соседей p» (6-элементная окрестность, обозначаемая через $N_6(p)$).

	, +1	+1, +1
-1, j	, j	+1, j
-1, j-1	, -1	

N6(p)

Связности

Две точки p и q, обладающие свойством Q, являются 4-связными, если q относится к группе $N_4(p)$

		+1, +1
	, j	+1, j
-1, j-1		+1, -1

рис. 6.2

Пусть дано изображение, где элементы со свойством Q выделены цветом.

Здесь точка p с координатами i, j имеет одного 4-связного соседа с координатами (i+1, j)

Две точки p и q, обладающие свойством Q, называются D-связными, если q относится к $N_D(p)$

Точка p(i, j) имеет трех D-связных соседей

Два элемента p и q, обладающие свойством Q, являются 8-связными, если q относится к $N_8(p)$. На рис. Точка p имеет xtnshtx 8-связных соседей.

Два элемента p и являются ю-связными (смешанная связь), если

q относится к группе $N_4(p)$

q относится к группе $N_D(p)$

и множество $N_4(p) \cap N_4(q) \neq 0$. Это множество элементов, являющихся 4-мя соседями и к **p** и к **q**

На рис. 2 элемент p(i, j) имеет двух ю-связных соседа с координатами (i+1, j) и (i-1, j-1)

Примыкание

Элемент p примыкает к элементу q, если они связны. Примыкания подразделяется на 4-х, 8- и 10связные.

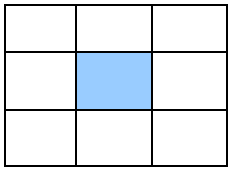
Два подмножества $\mathbf{S_1}$ и $\mathbf{S_2}$ изображения **f** примыкают друг к другу, если несколько элементов из $\mathbf{S_1}$ примыкают к элементам из $\mathbf{S_2}$

Путь от элемента p(x, y) к элементу q(s, t) представляет собой последовательность определенных элементов с координатами $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$, где

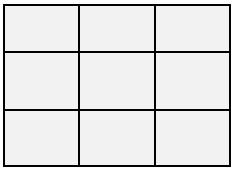
$(x_0, y_0) = (x, y)$

$(x_n, y_n) = (s,t)$

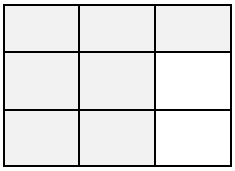
типы пикселей



Изолированный



внутренний



граничный

8-элементная окрестность принадлежит подмножеству S	8-элементная окрестность принадлежит подмножеству S	в 8-элементной окрестности есть элементы принадлеж. S и S

Совокупность граничных элементов называется границей подмножества S.

Совокупность внутренних элементов подмножества S называется внутренностью подмножества S.

Расстояние

Определим для элементов p, q и z соответственно с координатами (x, y), (s, t), (u, v) функцию расстояния или метрику следующим образом:

1. $\rho(p, q) \geq 0$ [$\rho(p, q)=0$, если p=q]
2. $\rho(p, q)= \rho(q, p)$
3. $\rho(p, q) \leq \rho(p, q)+ \rho(q, z)$

Евклидово расстояние между p и qопределяется по формуле

$$\rho_1(p,q) = \sqrt{(x-s)^2 + (y-t)^2}$$

При измерении этого расстояния элементы, имеющие расстояние ≤ некото-рой величине r от (x, y) располагаются в окрестности радиуса r с центром в (x, y).

Расстояние между p и q, определяемое выражением

$\rho_4(p, q)= |x-s| + |y-t|$, называется 4-связным или модульным расстоянием. В этом случае элементы, имеющие расстояние ρ_4, меньшее или равное некоторой величине r от (x, y), образуют ромбовидную структуру с центром в (x, y).

Например, элементы с расстоянием $\rho_4 \leq 3$ от (x, y) – центральный элемент, образуют контуры с равными от центра расстояниями в виде ромба

```
      4
    4 3 4
  4 3 2 3 4
4 3 2 1 2 3 4
  4 3 2 3 4
    4 3 4
      4
```

рис. 6.3.

Построение линий, окружностей, эллипсов

Проще всего начертить линию можно с помощью уравнения y=ax+b. При этом результаты надо округлять до целых, поэтому прямая будет неровная.

Если наоборот, нужно соединить 2 точки с заданными координатами (x1, y1), (x2, y2), то:

a = (y2-y1) / (x2-x1); b = y1 - a*x1;

Уравнение окружности выглядит следующим образом:

x = xc + r*cos(a); y = yc + r*sin(a),

где (xc, yc) - координаты центра окружности, r - радиус, a - угол для текущей точки (x, y).

Можно строить окружность прямо по этому уравнению, задав определенный шаг по углу (a = 0..360 с шагом DA). Однако, если шаг будет слишком мал, окружность за счет округления координат будет неровная и некоторые точки будут высвечиваться по несколько раз. Обычно шаг по углу выбирается равным 1/r радиан, (чаще всего шаг изменения угла должен быть переменным для того, чтобы избежать разрывов или отсутствия изменения координат).

Можно осуществить простой алгоритм аппроксимации отрезками.

Например, координаты 6 отрезков получаются с шагом 60 градусов, затем они соединяются прямыми линиями.

Для быстрого построения используется симметрия окружности (вычисляются координаты точек только 1/8 части окружности - для сегмента от 0 до 45 градусов). Кроме того, можно уйти от операций синуса/косинуса, если выразить координаты следующей точки окружности из предыдущей (рекуррентная формула) :

```
x2 = xc + (x1-xc)  * CA + (y1-yc) * SA;
```

```
y2 = yc + (y1-yc)  * CA - (x1-xc) * SA,  где
SA = sin(DA), CA =  cos(DA).
```

Начальная точка для рекуррентной формулы: x1 = xc, y1 = yc+r.

При построении окружностей следует учитывать, что размеры пикселов по вертикали и горизонтали не совпадают (кроме VGA 640x480) и окружности будут вытягиваться в эллипсы. Для того чтобы избежать этого, нужно вводить по выравнивающие коэффициенты (которые можно определить по GetAspectRatio). Сами же эллипсы описываются уранениями:

x = xc + rx * cos(a); y = yc + ry * sin(a),

где rx, ry - полурадиусы.

Общие принципы построения для них те же, что и для окружностей.

Алгоритм Брезенхема

Алгоритм Брезенхема (Bresenham) был разработан в 1965 году для цифровых графопостроителей, а затем стал использоваться для растровых дисплеев.

Идея алгоритма заключается в том, что одна координата изменяется на единицу, а другая - либо не изменяется, либо изменяется на единицу в зависимости от расположения соответствующей точки от ближайшего узла координатной сетки. Расстояние от точки отрезка до ближайшего узла по соответствующей ортогональной координате называется ошибкой. Алгоритм организован таким образом, что для вычисления второй координаты требуется только определять знак этой ошибки. Величину ошибки Delta можно определить в соответствии со следующим выражением:

Delta = Yуз - Yот ,

где Yуз - координата Y ближайшего узла при X = 1, Yот - координата Y отрезка при X = 1.

Если при X = 1 координата Y точки отрезка равна 1/2, то узлы координатной сетки (1,0) и (1,1) находятся на одинаковом расстоянии от отрезка и в качестве "ближайшего" выбирается узел (1,1). Таким образом, если Yот >= 1/2, то Delta >=0, в противном случае, при Yот < 1/2, Delta < 0. Для организации вычислений удобнее пользоваться не величиной Delta, а другой, определяемой как величина d = Delta - 1/2. При изменении координаты X на 1 величина d меняется на значение углового коэффициента: d = d + dY / dX. На каждом шаге алгоритма производится вычисление координаты X, величины d и выполняется анализ знака d. При этом, если окажется, что d >= 0, то производится увеличение Y на 1, а значение d корректируется путем вычитания из нее 1.

С учетом изложенного алгоритм аппроксимации отрезка в первом октанте можно представить в следующем виде :

```
X := X1; Y : =  Y1;
dX := X2 - X1;
dY := Y2 - Y1;
d : = - 1/2;
while X =< X2 do
 PutPixel (X, Y);
 X := X + 1;
 d := d + dY  / dX;
 if d >= 0 then
 begin
  Y := Y +  1;
  d := d -  1
 end
end while.
```

Представленный алгоритм использует вещественные числа и операцию деления. Оба недостатка можно исключить заменой величины d на другую, равную D = 2 * dX * d. В соответствии с этим арифметические выражения, в которых участвует d, модифицируются путем умножения обеих частей на величину 2 * dX.

Тогда алгоритм принимает вид :

```
X := X1; Y := Y1;
dX := X2 - X1;
dY := Y2 - Y1;
D := - dX;
while X =< X2 do
 PutPixel (X, Y);
 X := X + 1;
 D := D + 2  * dY;
 if D >= 0 then
 begin
```

```
  Y := Y +  1;
  D := D -  2 * dX;
 end
end while.
```

Последний вариант алгоритма можно еще улучшить, если операцию умножения на 2 заменить операцией сдвига влево на один разряд. Кроме того, если вычисление 2 * dX и 2 * dY выполнить перед циклом, то операцию сдвига потребуется выполнить только два раза.

Тогда алгоритм принимает окончательный вид :

```
X := X1; Y := Y1;
dX := X2 - X1;
dY := Y2 - Y1;
D := - dX;
Dx := dX shl 1;
Dy := dY shl 1;
while X =< X2 do
 PutPixel (X, Y);
 X := X + 1;
 D := D + Dy;
 if D >= 0 then
 begin
  Y := Y +  1;
  D := D -  Dx
 end
end while.
```

Оценивая достоинства полученного алгоритма, можно отметить, что он выполняет оптимальную аппроксимацию отрезка, используя при этом целочисленную арифметику и минимальное количество операций сложения и вычитания. Кроме того, алгоритм позволяет использовать его и в остальных октантах плоскости.

В алгоритме Брезенхема для всех октантов линия рассматривается как набор сегментов двух типов: тех, которые расположены диагонально и тех, которые расположены горизонтально или вертикально. Для линий с наклоном больше 1 прямые сегменты вертикальны, <= 1 - горизонтальны. Первая задача алгоритма состоит в вычислении наклона. Затем вычисляется выравнивающий фактор, который следит, чтобы некоторое число прямых сегментов имело большую длину, чем остальные. В цикле по большему отрезку (по x или по y) BX поочередно принимает то положительные, то отрицательные значения, отмечая какой тип сегмента выводится - диагональный или прямой.

В алгоритме проводится линия от (x1, y1) к (x2, y2). Используются некоторые регистры и переменные, назначение которых комментируется.

```
Начало алгоритма.
CX=1, DX=1 - начальные  инкременты по X и по  Y
{ вычисление вертикальной  дистанции }
DI=y2-y1 > 0 ? Если  да, то на шаг 4, иначе  - на след.
DX=-1, DI=-DI (дистанция по  Y должна быть > 0)
DIAG_Y=DX - приращение по  Y для диагональных элементов
{ вычисление горизонтальной  дистанции }
SI=x2-x1 > 0 ? Если  да, то на шаг 7, иначе  - на след.
CX=-1, SI=-SI (дистанция по  X должна быть > 0)
DIAG_X=CX - приращение по  X для диагональных элементов
{ вертикальны или горизонтальны  прямые сегменты }
SI >= DI ? Если  да, то на шаг 9, иначе  шаг 10.
DX=0 - для прямых сегментов  Y не меняется (т.е. они  горизонтальны).
      Переход на шаг  11.
CX=0 - прямые сегменты  вертикальны.
SI <-> DI - обмен  SWAP, чтобы больший отрезок  был в SI.
{ вычисление выравнивающего  фактора }
SHORT=DI - более короткий  отрезок
STR_X=CX, STR_Y=DX - инкременты  для прямых сегментов (один  из них = 0)
STR_COUNT = 2*SHORT
DIAG_COUNT= 2*SHORT - 2*SI
BX=2*SHORT-SI - начальное  значение переключателя
{ подготовка к выводу  линии }

CX=x1, DX=y1 - начальные  координаты
SI=SI+1 - прибавить один  пиксел для конца отрезка
{ вывод линии }
SI=SI-1
SI = 0 ? Если да,  то конец алгоритма - шаг  19.
```

```
Вывод точки с координатами  CX, DX через BIOS или  ПДП.
BX < 0 ? Если  да, то шаг 17, иначе  18.
{ следующий сегмент прямой  - подготовка координат для  вывода }
CX=CX+STR_X
DX=DX+STR_Y
BX=BX+STR_COUNT
переход на шаг 13
{ след. сегмент диагональный  }
CX=CX+DIAG_X
DX=DX+DIAG_Y
BX=BX+DIAG_COUNT
переход на шаг 13
Конец алгоритма.
```

Для примера рассмотрим проведение линии по алгоритму в декартовых координатах из точки (0, 0) в точку (10, 6). Наклон прямой < 1, значит все сегменты либо горизонтальны (STR_X=1, STR_Y=0), либо диагональны (DIAG_X=1, DIAG_Y=1). Больше расстояние по X: SI = 10 (цикл по нему). Выравнивающий фактор: STR_COUNT = 12, DIAG_COUNT = -8. Начальное значение BX=2, следовательно первый сегмент будет диагональный.

Таблица значений по шагам :

```
   BX CX DX       BX CX  DX
1)  2  1  1   6)   2  6   4
2) -6  2  1   7)  -6  7   4
3)  6  3  2   8)   6  8   5
4) -2  4  2   9)  -2  9   5
5) 10  5  3   10) 10 10  6
```

Первая точка будет иметь координаты (0, 0), остальные - как указано в таблице (CX, DX). Нарисуйте получившуюся линию на клетчатой бумаге.

ЗАПОЛНЕНИЕ СПЛОШНЫХ ОБЛАСТЕЙ

Заполнение областей может быть выполнено двумя способами- сканированием строк и затравочным заполнением.

Метод сканирования строк решает задачи определенного порядка сканирования, принадлежности точки внутренней области многоугольника или контура. При затравочном заполнении предполагается наличие некоторой точки, принадлежащей внутренней области (затравочная точка), для которой определяют ее соседние точки и включают ее в список других затравочных точек. Список этих точек анализируется с точки зрения необходимости их закрашивания.

Области могут задаваться внутренне определенным и гранично - определенным способом. Внутренне определенная область состоит из точек только одного цвета или интенсивности. Пикселы, внешние по отношению к точкам внутренне определенной области, имеют отличную от них интенсивность или цвет.

Гранично - определенные области имеют контур, состоящий из пикселов с выделенным значением цвета или интенсивности. Внутренние пикселы области отличны от них по цвету или интенсивности, а внешние могут с ними совпадать.

Внутренне и гранично - определенные области могут быть четырех- и восмисвязными. В четырехсвязных областях любой пиксел достигается движением по четырем взаимно перпендикулярным направлениям, а восьмисвязные- по восьми: горизонтальным, вертикальным и диагональным.

Можно сформировать простейший алгоритм затравочного заполнения четырехсвязной гранично - определенной области в следующем виде.

```
var
 X,Y: integer: { координаты  затравочной точки}
 oldcolor: word: { исходное  значение цвета}
 newcolor: word: { новое  значение цвета}
 framcolor: word: { цвет  границы}
begin
 помещаем пиксел в Stack;
 while (Stack не пуст)  do
 begin
  извлекаем пиксел из  Stack;
  PutPixel (X, Y,  newcolor) ;
  для точек (Xт,  Yт), соседних с (X, Y):
  (X+1, Y); (X,  Y+1); (X-1, Y); (X, Y-1),
```

```
  проверяем условие:
  if GetPixel (Хт,  Ут) = newcolor and
   GetPixel (Хт,  Ут) = framcolor
  then помещаем (Хт,  Ут) в Stack;
 end
end.
```

Аналогично строится алгоритм и для внутренне определенной области. В этом случае меняется только условный оператор, который должен проверять принадлежность соседних точек и наличие неизмененного цвета.

Предложенные алгоритмы легко обобщаются на случай восьмисвязных областей, для которых необходимо выполнять анализ восьми соседних точек.

Процедура затравочного заполнения областей, описанная выше, обладает двумя недостатками: стек заполняется большим количеством повторяющихся точек, что приводит к существенным временным затратам при их анализе и, как следствие, требуется стек большого объема. Однако эти проблемы легко разрешить изящным методом перестановки местами операций помещения точек в стек и их закраски.

Тогда процедура будет записана следующим образом :

```
begin
 PutPixel (X, Y, newcolor)  ;
 помещаем пиксел в Stack;
 while (Stack не пуст)  do
 begin
  извлекаем пиксел (X,  Y) из Stack;
  для точек (Хт,  Ут) , соседних с (X,  Y) :
   (X+ 1,Y); (X,  Y+1); (X-1, Y); (X, Y-1),
  проверяем условие:
  if GetPixel (Хт,  Ут) = newcolor and
   GetPixel (Хт,  Ут) = framcolor
  then
  begin
   PutPixel (Хт,  Ут, newcolor);
   помещаем (Хт,  Ут) в Stack
  end
 end
end.
```

Другой метод повышения эффективности затравочного заполнения областей заключается в организации построчного заполнения непрерывного интервала на сканирующей строке с одной затравочной точкой.

Удаление невидимых линий и поверхностей

Сложность задачи удаления невидимых линий и поверхностей привела к появлению большого числа различных способов ее решения, различных алгоритмов, но наилучшего решения поставленной задачи не существует. Главным недостатком всех алгоритмов является значительный объем вычислений, необходимых для определения удаляемых линий и поверхностей.

В начале реализации любого алгоритма удаления невидимых линий и поверхностей для повышения эффективности его работы обычно проводится сортировка координат объектов синтезируемой сцены. Основная идея сортировки заключается в том, что, чем дальше расположен объект от точки визирования, тем больше вероятность того, что он будет полностью или частично экранироваться одним из объектов, более близких к точке наблюдения.

Алгоритмы удаления невидимых линий и поверхностей классифицируются по способу выбора систем координат или пространства, в котором они работают.

Первый класс - это алгоритмы, работаюшие в объектном пространстве, имеющие дело с физической системой координат (мировые координаты), в которой они описаны.

Второй класс алгоритмов работает в пространстве изображения и имеет дело с системой координат того устройства, на котором эти объекты синтезируются.

* - существует большое число смешанных методов, объединяющих оба подхода

Алгоритмы первого класса используются в тех случаях, когда требуется высокая точность изображения объектов. Синтезируемые в этом случае изображения можно свободно увеличивать (уменьшать) во много раз, сдвигать или поворачивать. Точность вычислений алгоритмов второго класса ограничивается разрешающей способностью экрана. Результаты, полученные в пространстве изображения, а затем увеличенные (уменьшенные) во много раз, не будут соответствовать исходной сцене.

Наиболее часто используются алгоритмы Робертса, Аппеля, Варнока, Вейлера-Азертона, алгоритм, использующий список приоритетов (упорядочений), метод Z-буфера, метод построчного сканирования.

На практике, сравнительный анализ существующих алгоритмов удаления невидимых линий крайне затруднителен. В различных случаях при работе с различными моделями синтезируемого пространства эффективны различные алгоритмы. Даже при работе с одной и той же моделью оказывается, что в зависимости от точки наблюдения следует использовать различные алгоритмы.

Для реализации алгоритмов удаления невидимых линий часто используются алгоритмы нижнего уровня - отсечения отрезка (алгоритм Сазерленда - Кохена) и алгоритм принадлежности точки многоугольнику.

Отсечение нелицевых граней

Рассмотрим задачу удаления невидимых линий для многогранника. Несложно заметить, что если вектор нормали грани составляет с вектором, задающим направление наблюдения, тупой угол, то эта грань заведомо не может быть видна. Тупой угол или нет, определяется знаком скалярного произведения векторов. В случае, когда сцена представляет собой один выпуклый многогранник, удаление нелицевых граней полностью решает проблему удаления невидимых линий (в общем случае позволяет значительно сократить кол-во рассматриваемых граней).

Алгоритм Робертса

Самым первым алгоритмом, предназначенным для удаления невидимых линий был алгоритм Робертса. Сначала в нем отбрасываются все ребра, обе определяющие грани которых являются нелицевыми. Следующим шагом является проверка оставшихся ребер со всеми гранями многогранника на закрывание.

Возможны следующие случаи :

- грань не закрывает ребро;
- грань полностью закрывает ребро;
- грань частично закрывает ребро (в этом случае ребро разбивается на несколько частей, из которых видимыми являются не более двух)

Алгоритм Аппеля

Этот алгоритм основан на понятии количественной невидимости точки, как кол-ва лицевых граней, ее закрывающих. Точка является видимой только в том случае, если ее количественная невидимость = 0.

Метод Z-буфера

Одним из самых простых алгоритмов удаления невидимых граней и поверхностей является метод Z-буфера (буфера глубины). В силу крайней простоты этого метода (OpenGL) часто встречаются его аппаратные реализации.

Сопоставим каждому пикселу (x, y) картинной плоскости его расстояние вдоль направления проектирования z(x, y) - его глубину. Изначально массив глубин инициализируется бесконечностью. Для вывода на картинную плоскость произвольной грани она переводится в свое растровое представление и для каждого пиксела этой грани находится его глубина. В случае, если эта глубина меньше значения глубины, хранящегося в Z-буфере, то этот пиксел рисуется и его глубина заносится в Z-буфер.

Алгоритмы упорядочения

Подход заключается в таком упорядочении граней, чтобы при их выводе в этом порядке получалось корректное изображение. Для этого необходимо, чтобы дальние грани выводились раньше, чем более близкие (используются методы сортировки по глубине и двоичного разбиения пространства).

Метод построчного сканирования

x

Это еще один пример метода, работающего в пространстве картинной плоскости. Все изображение на картинной плоскости можно представить как ряд горизонтальных (вертикальных) линий пикселов. Рассмотрим сечение сцены плоскостью, проходящей через такую линию пикселов и центр проектирования. Пересечением этой плоскости с объектами сцены будет множество непересекающихся (за исключением концов) отрезков, которые и необходимо спроектировать. Задача удаления невидимых линий для такого набора отрезков решается тривиально.

Т.о. исходная задача удаления невидимых граней разбивается на набор гораздо более простых задач. Подобные алгоритмы успешно применяются для создания компьютерных игр (Wolfenstein 3D).

Принципы построения полутоновых изображений

Световая энергия, падающая на поверхность от источника света, может быть поглощена, отражена или пропущена. Количество энергии зависит от длины световой волны. При этом цвет поверхности объекта определяется поглощенными длинами волн.

Разработано множество способов закрашивания гранением, пропорциональное закрашивание, закрашивание по способу Гуро, закрашивание по способу Фонга, закрашивание способом трассировки лучей (лучей зондирования). Эти способы требуют различного количества процессорного времени и, следовательно, обеспечивают различное качество изображения.

Самый простой способ закрашивания называется гранением (faceting). Он требует сравнительно небольших ресурсов компьютера, поскольку предполагает лишь заполнение каждого из многоугольников одним цветом или оттенком. Однако способ слишком примитивен; закрашенные этим способом объекты выглядят не плавными.

Линейное, или пропорциональное закрашивание, предполагает изменение освещенности в пределах каждого многоугольника и благодаря этому позволяет воспроизводить более реалистичные изображения. Здесь яркость и цветовая насыщенность элементов каждого многоугольника плавно меняются в интервале между значениями, вычисленными для его вершин. При этом поверхности воспроизводимых предметов преобретают идеальную сюрреалистическую гладкость, как будто мы видим их при непрямом освещении.

Более реалистические изображения получаются в случае, если яркость и цветовая насыщенность каждого многоугольника плавно меняется не только от угла к углу, но и вдоль его ребер.

Такое закрашивание носит название способа Гуро и осуществляется в четыре этапа:

- вычисление нормалей к поверхности;
- определение нормалей в вершинах путем усреднения нормалей по граням, которым принадлежит данная вершина;
- вычисление интенсивности в вершинах;
- закрашивание многоугольника путем линейной интерполяции значений интенсивности вдоль ребер и между ребрами.

Основной недостаток - эффект полосы Маха : на ребрах смежных многоугольников возникает полоса разрыва непрерывности.

Закрашивание по способу Фонга решает проблемы полосы Маха, поскольку предполагает плавное изменение яркости и насыщенности не только вдоль ребер каждого многоугольника, но и по самой поверхности (вдоль сканирующей строки интерполируется значение вектора нормали, который затем используется в модели освещения для вычисления интенсивности пиксела). При этом даже зеркальные блики на поверностях выглядят вполне правдоподобно (в 3D Studio 4.0 помимо методов Гуро и Фонга добавилось еще закрашивание Metal).

Однако наиболее реалистичные изображения в трехмерной машинной графике обеспечивает закрашивание способом трассировки лучей, которое позволяет учесть такие эффекты, как отражение и преломление, наложение текстур.

Выпустим из каждого источника света пучок лучей во все стороны и мысленно проследим (оттрассируем) дальнейшее распространение каждого из них до тех пор, пока лиюо он не попадет в глаз наблюдателя, либо не покинет сцену. При попадпнии луча на границу объекта выпускаем из точки попадания отраженный и преломленный лучи и отслеживаем их и все порожденные ими лучи. Описанный выше процесс называется прямой трассировкой лучей. В результате его выполнения можно получить изображение сцены, однако он требует огромных вычислительных затрат. Причем, в получаемое изображение вносит вклад лишь небольшая часть трассируемых лучей. Чтобы избежать этого, попытаемся отследить лишь те лучи, которые попадают в глаз наблюдателя. Проследим путь луча, выходящего из глаза наблюдателя и проходящего через соответсвующую точку экрана. Цвет соответствующей точки экрана будет определяться долей световой энергии, попадающей в эту точку и покидающей ее в направлении глаза. Для этого из нее выпускаются лучи в тех направлениях, из которых может прийти энергия. Это, в свою очередь, может привести к определению точек пересечения соответствующих лучей с объектами сцены, выпускания новых лучей и т.д.

Описанный процесс называется обратной трассировкой лучей или просто трассировкой лучей. Именно этот метод и применяется в компьютерной графике.

Для придания более естственного вида сцене желательно иметь возможность менять параметры поверхности (в простейшем случае - цвет).

Существуют разные способы моделирования текстуры :

- проективные текстуры
- процедурные (сплошные) текстуры.

В первом случае образец текстуры проецируется на пов-ть параллельным переносом (плоское проектирование), либо цилиндрическим или сферическим методом. Недостатки проективных текстур - большой объем памяти для хранения образцов текстур, небольшая гибкость и трудность текстурирования объектов сложной формы.

Второй путь не требует больших затрат памяти и работает с объектами любой формы. Но поскольку подобранная функция может зависеть от большого числа переменных, основным недостатком является ее сложность.

Для удаления погрешностей метода трассировки лучей (aliasing, выражающийся в лестничном эффекте и пропадании точек) используется распределенная трассировка, добавляющая высокочастотный шум по методу Монте-Карло. Вообще более сложные задачи - неточечные источники света, нечеткие отражения, глубина резкости тесно связаны с методами оптимизации.

Основным недостатком метода трассировки лучей является то, что с изменением положения наблюдателя необходимо пересчитывать всю сцену (кроме этого неэффективность работы при диффузном отражении). Метод излучательности основан на законе сохранения энергии в замкнутой системе и производит вычисление глобальной освещенности независимо от положения наблюдателя. Все объекты разбиваются на фрагменты и для этих фрагментов составляются уравнения баланса энергии.

Вопросы к итоговой контрольной работе

1. При решении каких задач используется информация в виде изображений
2. Типы изображений
3. Ввод изображений в память компьютера.
4. Что понимается под дискретизацией изображений?
5. Что такое квантование?
6. Что такое разрешение и какие виды разрешения Вы знаете?
7. Что называется пикселем?
8. Как растровое изображение хранится в памяти компьютера?
9. Сколько бит необходимо для хранения разных типов изображений?
10. Что понимается под математической моделью изображения?
11. Что понимается под цветовой моделью?
12. Аддитивные цветовые модели
13. Субтрактивные цветовые модели
14. Перцепционные цветовые модели.
15. Что такое Lab-модель и для чего она используется?
16. Какие цветовые модели поддерживают пакеты Adobe Photoshop и Corel Draw.
17. Виды систем компьютерной графики
18. Что такое гистограмма изображения и как она вычисляется? Программа.
19. Какие виды гистограммных преобразований вы знаете? Для чего они используются.
20. Что понимается под графическим форматом и какие графические форматы вы знаете?
21. Суть алгоритмов сжатия с потерями. JPEG
22. Суть алгоритмов сжатия без потерь. RLE.
23. Опишите формат BMP
24. Опишите формат GIF
25. Опишите формат TIFF
26. Опишите формат EPS
27. Понятие оператора обработки изображений, типы операторов.
28. Что понимается под коррекцией изображений?
29. Что понимается под улучшением изображений?
30. Понятия окрестности, маски, апертуры.
31. Что понимается под контуром в растровой графике? Какие методы выделения контуров Вы знаете?
32. В чем заключаются методы пространственного дифференцирования?

33. Что понимается под сверткой изображения с маской?
34. Сглаживание шумов: усредняющий и медианный фильтры. Программы
35. Локальная фильтрация изображений.
36. Назначение и возможности пакета Adobe Photoshop.
37. Инструменты пакета Adobe Photoshop.
38. Отличительные особенности, достоинства и недостатки векторной графики.
39. Базовые элементы векторной графики.
40. Назначение и возможности пакета Corel Draw.
41. Инструменты пакета Corel Draw.
42. Какие пакеты программ компьютерной графики вы знаете?

ПК

Билет №1
1.При решении каких задач используется информация в виде изображений
2. Что понимается под цветовой моделью?

Билет №2
Типы изображений
Суть алгоритмов сжатия с потерями. JPEG

Билет №3
Что называется пикселем?
Опишите формат EPS

Билет №4
Опишите формат TIFF
Аддитивные цветовые модели

Билет №5
Опишите формат BMP
Субтрактивные цветовые модели

Билет №6
Перцепционные цветовые модели
Опишите формат GIF

Билет №7
Какие цветовые модели поддерживают пакеты Adobe Photoshop и Corel Draw.
Какие цветовые пространства вы можете привести?

Билет №8
Какие пакеты программ компьютерной графики вы знаете?
Цветовая модель HSV

Билет №9
Что понимается под графическим форматом и какие графические форматы вы знаете?
2.Суть алгоритмов сжатия с потерями. JPEG

Билет №10
Что такое гистограмма изображения и как она вычисляется? Программа.
Аддитивные цветовые модели

Билет №11
1.Субтрактивные цветовые модели
2. Модель RGB

Билет №12
1.При решении каких задач, используется информация в виде изображений
2.Опишите трехмерное цветовое пространство

Билет №13
1.Приведите отличия моделей RGB и CMY
2. Опишите формат GIF

Билет №14
1.Цветовая модель HSV
2.Какие системы соответствия цветов вы знаете?
Билет №15
1.Основные графические форматы
2.Опишите цветовое пространство LUV

ИК
Билет №1
Базовые элементы векторной графики.
Что понимается под цветовой моделью?
Сглаживание шумов: усредняющий и медианный фильтры. Программы

Билет №2
Типы изображений
Опишите формат EPS
Локальная фильтрация изображений.

Билет №3
Что называется пикселем?
Опишите формат TIFF
Что понимается под сверткой изображения с маской?

Билет №4
Аддитивные цветовые модели
В чем заключаются методы пространственного дифференцирования?
Суть алгоритмов сжатия с потерями. JPEG

Билет №5
Опишите формат BMP
Субтрактивные цветовые модели
Что понимается под сверткой изображения с маской?

Билет №6
Перцепционные цветовые модели
Опишите формат GIF
Что понимается под коррекцией изображений?

Билет №7
Что понимается под контуром в растровой графике? Какие методы выделения контуров Вы знаете?
Какие цветовые модели поддерживают пакеты Adobe Photoshop и Corel Draw.
При решении каких задач используется информация в виде изображений

Билет №8
Суть алгоритмов сжатия с потерями JPEG
При решении каких задач, используется информация в виде изображений

Что понимается под коррекцией изображений?

Билет №9
Что понимается под графическим форматом и какие графические форматы вы знаете?
Какие пакеты программ компьютерной графики вы знаете?
Что понимается под улучшением изображений?

Билет №10
Понятия окрестности, маски, апертуры
Инструменты пакета Adobe Photoshop.
Суть алгоритмов сжатия без потерь RLE.

Билет №11
Аддитивные цветовые модели
Отличительные особенности, достоинства и недостатки векторной графики.
Что понимается под контуром в растровой графике? Какие методы выделения контуров Вы знаете?

Билет №12
Что понимается под коррекцией изображений?
Суть алгоритмов сжатия без потерь RLE.
Понятие цветового пространства

Билет №13
1.Понятия окрестности, маски, апертуры
2. Что понимается под графическим форматом и какие графические форматы вы знаете?
3. Как Вы понимаете цветовую модель?

Билет №14
1.Что понимается под сверткой изображения с маской?
2. Суть алгоритмов сжатия с потерями JPEG.
3. Цветовое пространство HSV

Билет №15
1.Что понимается под математической моделью изображения
2. Опишите цветовое пространство XYZ
3.Перцепционные цветовые модели

Билет №16
Понятие операторов обработки изображений, типы операторов.
Как производится сглаживание шумов
Приведите и опишите отличия цветовых пространств HSVи HLS

Основная литература:

1. М.Н.Петров, В.П.Молочков Компьютерная графика. Учебник – СПб: Питер,2003 – 736с.

2. Прэтт У. Цифровая обработка изображений: Пер. с англ. – М.: Мир,1982 – кн1.-312с.; кн.2 – 493 с.

3. Садыков С. Цифровая обработка и анализ изображений – Ташкент: НПО «Кибернетика» АН РУз – 1994, 193с.

4. Стразницкас М. Photoshop 5.5 для подготовки Web-графики. –С.-Петебург: Sybex, 2000 – 473с.

5. Симонович С.В. и др. Специальная информатика: Учебное пособие. – М.: АСТ – ПРЕСС: Инфорком-Пресс, 2002. – 480с.

Дополнительная литература

6. www.intuit.ru Курсы по компьютерной графике

7. HELP программы Photoshop.