1. Цели и задачи компьютерной графики [1,2,3].
2. Растровая и векторная визуализация изображений. Основные характеристики растровых изображений [1,2].
3. Аддитивная цветовая модель RGB. Кодирование цвета. Палитра[1,2].
4. Графические форматы. Работа с растровыми изображениями (форматы DDB, DIB, bmp – файлы) [1,2,3].
5. Понятие об инкрементных алгоритмах растеризации. Растровое представление отрезка (вывод алгоритма)
6. Стиль линии (перо). Алгоритмы вывода линий [1,2]
7. Стиль заполнения, кисть, текстура [1,2]
8. ***Геометрические основы компьютерной графики***. Системы координат (декартова, полярная, сферическая) и векторы. Скалярное и векторное произведение векторов [1,2,3].
9. ***Геометрические основы компьютерной графики***. Уравнение прямой, проходящей через две точки (**на плоскости**). Параметрическое уравнение прямой. Уравнение отрезка прямой. Задача о взаимном расположении двух отрезков [1].
10. ***Геометрические основы компьютерной графики***. Уравнение прямой, проходящей через две точки (**в пространстве**). Параметрическое уравнение прямой. Задача о взаимном расположении отрезка прямой и плоскости (пересекает, не пересекает, параллелен плоскости, лежит в плоскости) [1].
11. ***Геометрические основы компьютерной графики***. Вычисление координат вектора нормали к поверхности.
12. Мировые и экранные координаты. Алгоритм пересчета мировых 2D-координат в оконные. Вывод и реализация (Листинг функции SpaceToWindow)
13. Физическая и логическая системы координат. Режимы отображения (без настройки параметров). Алгоритм преобразование координат в GDI и его связь с алгоритмом пересчета мировых координат в оконные[1]
14. Режимы отображения и настройка их параметров (функции класса CDC MFC). Функция **SetMyMode**, назначение, параметры и реализация (листинг) [1].
15. ***Аффинные преобразования на плоскости***. Преобразования системы координат: смещение, растяжение-сжатие, поворот.[1,2]
16. ***Аффинные преобразования на плоскости.*** Преобразования объектов: смещение, растяжение-сжатие, поворот [1,2].
17. ***Аффинные преобразования в пространстве.*** Преобразования системы координат: смещение, растяжение-сжатие, повороты вокруг осей координат [1,2].
18. ***Аффинные преобразования в пространстве***. Преобразования объектов: смещение, растяжение-сжатие, повороты вокруг осей координат [1,2].
19. Основные типы проекций. Видовая система координат (вывод матрицы преобразования) [1].
20. Перспективные преобразования (вывод матрицы преобразования). Схема пересчета координат при переходе от мировых 3D-координат к экранным [1].
21. Модели описания поверхностей [1,2]
22. Каркасная визуализация трехмерных изображений. Принцип удаления невидимых граней для выпуклого многогранника. Пример реализации по лабораторной работе. (Листинг функции по лабораторной работе). [1]
23. Изображение поверхности z=f(x,y). Метод сортировки граней по глубине (алгоритм художника). Исходные данные и последовательность вычислений. Пример реализации метода по лабораторной работе. (Листинг функции по лабораторной работе). [1]
24. ***Закрашивание поверхностей***. Модели отражения света [1,2]
25. ***Закрашивание поверхностей***. Вычисление нормалей и углов отражения. (Листинг функции для пирамиды по лабораторной работе)
26. Метод закрашивания Гуро [1,2].
27. **Цели и задачи компьютерной графики**

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

В компьютерной графике можно выделить несколько основных направлений:

 Визуализация научных данных Большинство современных математических программных пакетов (Maple, MatLab) имеют средства для отображения графиков, поверхностей и трехмерных тел, построенных на основе какихлибо расчетов. Кроме того, графическая информация может активно использоваться в самом процессе вычислений. Визуализация позволяет представить большой объем данных в удобной для анализа форме и широко используется при обработке результатов различных измерений и вычислений.

 Геометрическое проектирование и моделирование Это направление компьютерной графики связано с решением задач начертательной геометрии – построением чертежей, эскизов, объемных изображений с помощью программных систем, получивших название CAD-системы (от английского ComputerAided Design), например AutoCAD. Существует большое количество специализированных CAD-систем в машиностроении, архитектуре и т. д.

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

 Изобразительное искусство К этому направлению можно отнести разнообразную графическую рекламу: от текстовых транспарантов и фирменных знаков до компьютерных видеофильмов, обработку фотографий, создание рисунков, мультипликацию и т. д. В качестве примера популярных программ из этой области компьютерной графики можно назвать Adobe Photoshop (обработка растровых изображений), CorelDRAW (создание векторной графики), 3DS Мах (трехмерное моделирование).

 Виртуальная реальность Реальность, даже виртуальная, подразумевает воздействия на всю совокупность органов чувств человека, в первую очередь на его зрение. К компьютерной графике можно отнести задачи моделирования внешнего мира в различных приложениях: от компьютерных игр до тренажеров.

 Цифровое видео Все более широкое распространение получают анимированные изображения, записанные в цифровом формате. Это прежде всего фильмы, передаваемые через компьютерные сети, а также видеодиски, цифровое, кабельное и спутниковое телевидение.

1. **Растровая и векторная визуализация изображений. Основные характеристики растровых изображений**

Растр граф устройства: дисплей, телевизор, принтер.

Векторные: дисплей, плоттер.

Векторизация- преобразование из растрового в векторное описание.

Растеризация- наоборот.

Растровая визуализация основывается на формировании изобр на экране или бумаге в виде совокупности отдельных точек (пикселов). Вместе пикселы образуют растр.

Векторная визуализация основывается на формировании изобр на экране или бумаге рисованием линий (векторов)- прямых или кривых. Качество вект визуализации для вект устройств обуславливается точностью вывода и номенклатурой базовых граф примиивов- линий, дуг, кругов, эллипсов и др.

Дроминирует растровый способ, т.к. больше распространены растровые дисплеи и принтеры.

Недостаток растр уст-в— дискретность изображения.

Недостатки век тут-в— проблемы при сплошном заполнении фигур, меньшее кол-во цветов, меньшая скорость (в сравнении с растр ус-ми).

В зависимости от расположения пикселов в пространстве различают типы растра: квадратный, прямоугольный, гексагональный и иные типы.

Геометрические хар-ки растра.

Разрешающая способность характеризует расстояние между соседними пикселами. РС измеряют кол-вом пиксело на единицу длины. Наиб распространена dpi(dot per inch)- кол-во пикселов в одном дюйме длины(2.54 см).

Размер растра обычно измеряется кол-вом пикселов по горизонтали и вертикали.

Форма пикселов растра определяется особенностями устройства граф вывода.

Кол-во цветов (глубина цвета)—одна из важн хар-к растра.

Глаз человека различает 350000 цветов.

Классификация изображений:

-двухцветные(бинарные)—1 бит на пиксел. Черно-белое из.

-полутоновые—градации серого или иного цвета. Наприм, 256 градаци1 (1 байт на пиксел).

-цветные из. От 2 бит на пиксел и выше. Глубина цвета 16 бит на пиксел (65536 ЦВЕТОВ) получила название High Color, 24 бит\п —True Color. В комп граф системах используют и большую глубину цвета—32,48 бит\п.

Разрешающая способность растра: dpi=25.4/dP

3. **Аддитивная цветовая модель RGB. Кодирование цвета. Палитра**

Для того чтобы компьютер имел возможность работать с цветными изображениями, необходимо представлять цвета в виде чисел – кодировать цвет. Способ кодирования зависит от цветовой модели и формата числовых данных в компьютере.

Чтобы оцифровать цвет, его необходимо измерить. Наука, которая изучает цвет и его измерения, называется колориметрией. Она описывает общие закономерности цветового восприятия света человеком. Одними из основных законов колориметрии являются законы смешивания цветов. Эти законы в наиболее полном виде были сформулированы в 1855 году немецким математиком Германом Грассманом:

 закон трехмерности – любой цвет может быть представлен комбинацией трех основных цветов;

 закон непрерывности – к любому цвету можно подобрать бесконечно близкий;

 закон аддитивности – цвет смеси зависит только от цвета составляющих.

Первый закон означает, что для любого заданного цвета (Color) можно записать такое цветовое уравнение, выражающее линейную зависимость цветов:

Color = k1Color1 + k2Color2 + k3Color3,

где Color1, Color2, Color3 – некоторые базисные, линейно независимые цвета, коэффициенты k1 , k2 , k3 указывают количество соответствующего смешиваемого цвета.

Линейная независимость цветов означает, что ни один из них не может быть выражен взвешенной суммой (линейной комбинацией) двух других.

Цветовая модель RGB

За основные три цвета приняты красный (Red), зеленый (Green), синий (Blue). В модели RGB любой цвет (Color) получается в результате сложения основных цветов.

Для модели RGB каждая из компонент может представляться числами, ограниченными некоторым диапазоном – например, дробными числами от 0 до 1 либо целыми числами от 0 до некоторого максимального значения.

В настоящее время достаточно распространенным является формат True Color, в котором каждая компонента представлена в виде байта, что дает 256 градаций для каждой компоненты: R=0...255, G = 0...255, В = 0...255. Количество цветов составляет 256\*256\*256 = 16.7 млн (224).

Такой способ кодирования цветов можно назвать компонентным. В компьютере коды изображений True Color представляются в виде троек байтов либо упаковываются в длинное целое (четырехбайтное) – 32 бита (так, например, сделано в API Windows):

C = 00000000 bbbbbbbb gggggggg rrrrrrrr.

Цветовая модель RGB применяется для создания графических образов в устройствах, излучающих свет, - мониторах, телевизорах.

1. Графические форматы. Работа с растровыми изображениями (форматы ddb, dib, bmp-файлы)

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

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

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

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

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

В операционной системе Windows используются два формата растровых (битовых) изображений –аппаратно-зависимый DDB (device-dependentbitmap) и аппаратно-независимый DIB(device-independentbitmap).

Согласно определению, данному в документации к SDK, бито-вое изображение DDB есть набор бит в оперативной памяти, который может быть отображен на устройстве вывода(например, выведен на экран видеомонитора или распечатан на принтере). Внутренняя структура изображения DDB жестко привязана к аппарат-ным особенностям устройства вывода. Поэтому представление изображения DDB в оперативной памяти полностью зависит от устройства вывода.

Битовые изображения в формате DDB

Как было сказано выше, битовые изображения в формате DDB являются аппаратно-зависимыми. Поэтому структура изображе-ния в оперативной памяти зависит от особенностей аппаратуры. Как правило, изображения DDB либо загружаются из ресурсов приложения, либо создаются непосредственно в оперативной памя-ти.

Битовые изображения в формате DIB

Изображения DIB, в отличие от изображений DDB, являются аппаратно-независимыми, поэтому без дополнительного преобра-зования их нельзя отображать на экране с помощью функций BitBlt и StretchBlt. В операционной системе Windows битовые изображе-ния хранятся в файлах с расширением имени bmp, при этом ис-пользуется аппаратно-независимый формат DIB.

Палитра отсутствует, если число бит на пиксел равно 24. Также палитра не нужна и для некоторых цветовых форматов 16 и 32 бит на пиксел.

После палитры (если она есть) в файле BMP записывается растр в виде битового (а точнее, байтового массива). В битовом массиве последовательно записываются байты строк растра. Количество байт в строке должно быть кратно четырем, поэтому если количе-ство пикселов по горизонтали не соответствует такому условию, то справа в каждую строку дописывается некоторое число битов (вы-равнивание строк на границу двойного слова).

1. Понятие об инкрементных алгоритмах растеризации. Растровое представление отрезка (вывод алгоритма)

**Брезенхэм** предложил подход, позволяющий разрабатывать так называемые **инкрементные** алгоритмы растеризации [2]. Основной целью для разработки таких алгоритмов было построение циклов вычисления координат на основе только целочисленных операций сложения/вычитаниябез использования умножения и деления. Инкрементные алгоритмы выполняются как последовательное вычисление координат соседних пикселов путем добавления приращений координат.

Разработаны инкрементные алгоритмы для вывода отрезка, окружности, эллипса.

**Алгоритм вывода прямой линии**

Пусть на растре заданы две точкии  с целочисленными координатами

Запишем уравнение прямой, проходящей через эти точки



или

 (4.1)

где

,,,



















Будем рассматривать случай дляи .

Пусть точка  уже поставлена. Необходимо принять решение о том, какие координаты должна иметь точка:  или.

Запишем выражение для *y*-координаты математической прямой для :



Вычислим разности

,

,



.

Тогда



Выполним заменув





или



Полагая, что ,получим

,(4.2)

где

(4.3)

(4.4)

Так как , то



Вычитая (4.2) из (4.4), получим

.

Но, тогда.

При этом

.

Поэтому

.

Вычислим начальное значение .

Из (4.2) получаем

(4.5)

Значение  соответствует точке .

Точка  по условию удовлетворяет уравнению прямой (4.1).

Отсюда

.

Подставляя полученное значение  в выражение (4.3)для, получим



.

Подставляя полученное выражение для в выражение (4.5) для , получим

.

Ниже приведен возможный вариант функции для рисования линии между точками  идляцветомcolor для работы с использованием библиотеки классов MFC.

6. **Стиль линии. Перо**

Для описания различных по виду изображений на основе линий используют термин**стиль линий** или**перо**[2]. Терминперо иногда делает более понятной суть алгоритма вывода линий для некоторых стилей – в особенности для толстых линий. Например, если для тонкой непрерывной линии перо соответствует одному пикселу, то для толстых линий перо можно представить себе как фигуру или отрезок линии, который скользит вдоль оси линии, оставляя за собой след (рис.4.4).

**4.3.1. Алгоритмы вывода толстой линии**

Взяв за основу любой алгоритм вывода обычных тонких линий (например, алгоритм Брезенхэма), запишем его в следующем обобщенном виде:

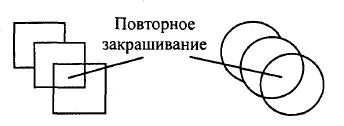
**Вывод пиксела (х, у)**

Можно представить себе такой алгоритм, как цикл, в котором определяются координаты каждого пиксела. Этот алгоритм можно модифицировать для вывода толстой линии следующим образом:

**Вывод фигуры (или линии) пера с центром (х, у)**

Вместо вывода отдельного пиксела стоит вывод фигуры или линии, соответствующей перу – прямоугольник, круг, отрезок прямой.

Такой подход к разработке алгоритмов толстых линий имеет преимущества и недостатки. Преимущество – можно прямо использовать эффективные алгоритмы для вычисления координат точек линии оси, например, алгоритмы Брезенхэма. Недостаток – неэффективность для некоторых форм пера. Для перьев, которые соответствуют фигурам с заполнением, количество тактов работы алгоритма пропорционально квадрату толщины линии. При этом большинство пикселов многократно закрашивается в одних и тех же точках (рис. 4.5).



Такие алгоритмы более эффективны для перьев в виде отрезков линий. В этом случае каждый пиксел рисуется только один раз. Но здесь важным является наклон изображаемой линии. Ширина пера зависит от наклона (рис. 4.6).

Очевидно,что горизонтальное перо не может рисовать толстую горизонтальную линию.

Для вывода толстых линий с помощью пера в качестве отрезка линии чаще всего используются отрезки горизонтальной или вертикальной линий, реже – диагональные отрезки под углом 45 градусов. Целесообразность такого способа определяется большой скоростью вывода горизонтальных и вертикальных отрезков прямой. Для того, чтобы достигнуть минимального количества тактов вывода, толстые линии, которые по наклону ближе к вертикальным, рисуют горизонтальным пером, а пологие линии – вертикальным пером.

**4.3.2. Алгоритмы вывода пунктирной линии**

Алгоритм для рисования тонкой пунктирной линии можно получить из алгоритма вывода тонкой непрерывной линиизаменой процедуры вывода пиксела более сложной конструкцией:

**Вывод пиксела (х, у)**

**Проверка значения счетчика С :**

**Если С удовлетворяет некоторым условиям,**

**то вывод пиксела (х, у)**

**Значение С увеличивается на единицу**

При выводе полилиний, которые состоят из отрезков прямых, необходимо предотвратить обнуление значения счетчика в начале каждого отрезка и обеспечить продолжение непрерывного приращения вдоль всей сложной линии. Иначе будут нестыковки пунктира. Использование переменной-счетчика затруднено при генерации пунктирных линий в алгоритмах, которые используют симметрию, например, при выводе круга или эллипса. В этом случае будут нестыковки пунктира на границах октантов или квадрантов.

7. **Стиль заполнения. Кисть. Текстура**

При выводе фигур могут использоваться различные стили заполнения. Для обозначения стилей заполнения, отличных от сплошного стиля, используют такие понятия, как**кисть**и **текстура**. Их можно считать синонимами, однако понятие текстуры обычно используется применительно к трехмерным объектам, а кисть – для изображения двумерных объектов. Текстура – это стиль заполнения, закрашивание, которое имитирует сложную рельефную объемную поверхность, выполненную из какого-то материала.

Для описания алгоритмов заполнения фигур с определенным стилем можно использовать тот же способ, что и для описания алгоритмов рисования линий. Описание всех разновидностей подобных алгоритмов можно дать с помощью обобщенной схемы:

**Вывод пиксела заполнения цвета С с координатами (х, у)**

Например, в алгоритме вывода полигонов пикселы заполнения рисуются в теле цикла горизонталей, а все другие операции предназначены для подсчета координат этих пикселов. Сплошное заполнение означает, что цвет (С) всех пикселов одинаков, то есть C=const. Для получения определенного узора необходимо изменять цвет пикселов заполнения. Преобразуем алгоритм заполнения следующим образом:



**Вывод пиксела заполнения цветом**

Функция  будет определять стиль заполнения. Аргументами функции цвета являются координаты текущего пиксела заполнения. Однако в отдельных случаях эти аргументы не нужны. Например, если цвет С вычислять как случайное значение в определенных границах: , то можно создать иллюзию шершавой матовой поверхности (рис.4.7, а).

|  |  |
| --- | --- |
|  |  |
| а) | б) |
| Рис. 4.7 | |

Другой стиль заполнения – штриховой (рис. 4.7, б). Для него функцию цвета также можно записать в аналитической форме:

,

где– период, – толщина штрихов,– цвет штрихов, цвет фона.

Подобную функцию можно записать и для других типов штриховки. Аналитическая форма описания стиля заполнение позволяет достаточно просто изменять размеры штрихов при изменениях масштаба показа, например, для обеспечения режима WYSIWYG.

**8. *Геометрические основы компьютерной графики***. Системы координат (декартова, полярная, сферическая) и векторы. Скалярное и векторное произведение векторов

Рассмотрим различные способы представления векторов на плоскости и в пространстве. Пусть  некоторый вектор, заданный в декартовой системе координат  на плоскости (рис.7.1, *а*) и декартовой системе координат  в пространстве (рис. 7.1, *б*).

В прямоугольной системе координат  направление осей задается тройкой перпендикулярных единичных векторов , , .

Система координат называется **правой,** если при повороте от вектора  к вектору  на  направление вектора совпадает с поступательным движением винта с правой резьбой (рис. 7.2). Начальная точка векторов обозначается буквой *О*.Ниже представлены различные формы записи векторов на плоскости и в пространстве.

 (7.1)

 (7.2)

Другая форма записи векторов

(7.3

 (7.4)

Модуль (длина) вектора

 (7.5)

 (7.6)

Пусть заданы два вектора и в форме (7.4)

и,

Тогда

 (7.7)

 (7.8)

При этом справедливы равенства

 (7.9)

**Скалярное произведение векторов**

Скалярное произведение векторов  иопределяется выражением

, (7.10)

где- угол между векторами и 

В координатной форме

 (7.11) или

, (7.12

если вектора и  представлены в форме (7.4).

Из (7.10) получаем

 (7.13)

Если , то угол между векторами  и  является острым , при– вектора и  ортогональны, а если , то угол между векторами является тупым .

Отметим некоторые свойства скалярного произведения.

Пусть,,–некоторые числа. Тогда справедливы равенства:

 (7.14)

**Векторное произведение векторов**

Векторное произведение векторов  и определяется выражением

, (7.15

где  – угол между векторами  и . Направление вектора  перпендикулярно векторам  и и таково, что вектора , ,  в приведенной последовательности образуют правостороннюю тройку (рис. 7.2).

Отметим некоторые свойства векторного произведения.

Длина вектора равна

 (7.16)

Если, где  скаляр, то

, (7.17)

так как  (см. ниже).

Пусть  – некоторая константа, тогда справедливы соотношения

 (7.18)

Рассмотрим еще одну форму записи векторного произведения



. (7.19

**9. Уравнение прямой на плоскости**

Пусть в декартовой системе координат на плоскости  заданы две точк

и 

Тогда уравнение прямой, проходящей через эти точки имеет вид:

 (7.20)

Полагая в (7.20)

,

получаем параметрическое уравнение прямой на плоскости, проходящей через точки  и .

, (7.21)

Пусть.

Тогда (7.21) можно записать в виде

, (7.22)

или в более компактной форме

. (7.23)

Из (7.23) следует, что при , а при  получаем, что .

Таким образом, приточка  принадлежит отрезку .

Следовательно, при, уравнение (7.23) можно считать уравнением отрезка .

Рассмотрим задачу о взаимном расположении двух отрезков на плоскости.

***Условие задачи***. На плоскости заданы два отрезка прямой:и

, ,

, .

***Определить :*** пересекаются или нет отрезки и .

Запишем уравнения прямыхи , определяемых отрезкамии  соответственно в параметрической форме

 (7.24)

 (7.25)

Предположим, что прямые пересекаются, и - точка их пересечения.

В этой точке справедливо равенство

.

или

.

Отсюда





или

. (7.26)

Определитель системы (7.26) раве

 (7.27)

Если , то система уравнений (7.26) решения не имеет:отрезки параллельны (или накладываются).

Если , то система уравнений (7.26) имеет единственное решение  и .

При этом если и , то отрезки пересекаются, иначе пересекаются их продолжения, или один отрезок пересекается с продолжением другого.

Подставляя найденные значения  и в (7.24) или в (7.25), можно определить координаты точки пересечения отрезков .

 (7.28)

или

. (7.29)

**10.Уравнение прямойв пространстве**

Пусть теперьдве точки заданы в пространстве

и (7.30)

Тогда уравнение прямой, проходящей через эти точки, будет иметь вид:

. (7.31)

И в параметрической форме

 (7.32)

Уравнения (7.32), как и ранее, можно представить в компактной форме

. (7.33)

Рассмотрим задачу о пересечении плоскости отрезком прямой.

**Условие задачи.** В декартовой системе координат заданаплоскость , определяемая уравнением

 (7.34)

и отрезок прямой , где точки  и  определяются по (30).

**Определить:** пересекает отрезок  плоскость (рис. 7.3) или нет (рис. 7.4).

|  |  |
| --- | --- |
|  |  |
|  |  |

Для решения поставленной задачи определим значение параметра , при котором прямая (7.32) и плоскость (7.34) имеют общую точку.

Подставляя выражения для координат (7.32) в уравнение плоскости (7.34) и вводя обозначения ,, , получим

 (7.35)

Из (7.35) получаем

 (7.36)

для .

Тогда,если , то отрезок  пересекает плоскость . В противном случае плоскость  пересекает продолжение рассматриваемого отрезка.

Координаты точки пересечения определяются путем подстановки значения в (7.32)

 , (7.37)

или в компактной форме

. (7.38)

Рассмотрим случай, когда

. (7.39)

Известно, что в уравнении плоскости коэффициенты , и  являются координатами вектора , нормального к этой плоскости, т. е.. Рассмотрим вектор , построенный на отрезке . Тогда условие (7.39) означает, что

. (7.40)

Равенство нулю скалярного произведения (7.40) означает, что вектор перпендикулярен вектору нормали и, следовательно, параллелен плоскости . Если при этом также справедливо равенство , то вектор (отрезок ) лежит в плоскости .

Рассмотрим один частный случай рассмотренной задачи, когда плоскость (7.34) параллельна плоскости  и определяется уравнением  или

. (7.41)

В этом случае ,,  и . Подставляя эти значения в (7.36), получаем

 (7.42)

Подставив найденное значение в (37), находим

 (7.43)

Рассмотренная задача возникает при построении линий уровня

11 ***Геометрические основы компьютерной графики***. Вычисление координат вектора нормали к поверхности.

В данном разделе рассматривается задача вычисления вектора нормали к поверхности, которая возникает во многих задачах компьютерной графики.

Сначала вычислим нормаль к плоскости, которая определяется расположенными на ней тремя точками (рис.7.5).

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Рис. 7.5 |  | Рис. 7.6 |

Пусть точки , и  лежат в одной плоскости.

Образуем векторы

и .

Тогда вектор нормали  к плоскости можно вычислить как векторное произведение векторов  и

 (7.44)

Выражение (44) используется для вычисления координат вектора нормали в случае описания поверхности векторно-полигональной моделью (см. ниже).

Рассмотрим теперь произвольную поверхность, заданную в явном виде функцией

. (7.45)

При таком задании поверхности вектор нормали будет имеет вид

, (7.46

где ,,  – координаты вектора нормали.

Единичный вектор нормали будет определяться выражением

, (7.47)

где – модуль вектора нормали, , ,  – координаты вектора единичной нормали.

Пусть теперь поверхность задана неявно уравнением

. (7.48)

При неявном определении поверхности вектор нормали примет вид []

, (7.49)

где ,,  – координаты вектора нормали,

 (7.50)

– градиент функции .

Единичный же вектор нормали в этом случае

, (7.51)

где

 – (7.52)

модуль градиента функции , , ,  – координаты вектора единичной нормали.

При выборе вектора нормали необходимо следить за тем, чтобы получить **нормаль нужного направления**, т. е. правильно выбрать **нужную сторону поверхности**. Если для поверхности указано, что нормаль к ней составляет с осью  угол больший, чем , то вместо (7.46) в качестве вектора нормали необходимо взять вектор

, (7.53)

Аналогично для поверхности . В этом случае вместо (7.49) для вычисления вектора нормали следует воспользоваться выражением

. (7.54)

Соответственно выражение для вектора единичной нормали будет иметь вид

.(7.55)

Пусть теперь поверхность задана параметрически:

 (7.56)

Тогда вектор нормали  к поверхности имеет вид:

, (7.57)

где

,,  (7.58)

12. При отображении пространственных объектов на экране или на листе бумаги с помощью принтера необходимо знать координаты объектов. Мы рассмотрим две системы координат. Первая **–мировая система координат(МСК)**, которые описывают истинное положение объектов в пространстве с заданной точностью. Другая **– оконные координаты** или**система координат устройства изображения**, в котором осуществляется вывод изображения объектов в заданной проекции.

Мировые координаты объектов являются трехмерными. Положение объекта может быть описано, например, в прямоугольной или сферической системе координат. Где располагается центр системы координат и каковы единицы измерения вдоль каждой оси, не очень важно. Важно то, что для отображения должны быть известны какие-то числовые значения координат отображаемых объектов.

Первоначально рассмотрим работу с плоскими изображениями, для описания которых используются двухмерные мировые координаты. Положим, что изображение объекта сформировано в некоторой плоскости пространства. Рассмотрим задачу получения формул для пересчета мировых координат объекта в оконные.

Пусть в мировом пространстве задана плоскость . Свяжем с этой плоскостью систему координат и выделим на ней прямоугольную область (рис. 5.1).

Рассмотрим оконную систему координат  с началом в левом верхнем углу окна и выделим в ней некоторую прямоугольную область (рис. 5.2).

Пусть  – некоторая точка в мировой системе координат, а  ее образ в оконной системе координат.

Из приведенных рисунков можно получить пропорции

(5.1)













0









P





















Рис. 5.2

(5.2)

Введем обозначения:

|  |  |
| --- | --- |
|  | – ширина области отображения в оконных координатах, |
|  | – ширина области отображения в мировых координатах, |
|  | – высота области отображения в оконных координатах, |
|  | – высота области отображения в мировых координатах. |

Тогда выражение (5.2) можно переписать в виде

(5.3)

или

, (5.4)

где ,.

Представим соотношения (5.4) в матричном виде

(5.5)

или

(5.6)

где

, , .

Перепишем пару рассматриваемых равенств в виде, удобном для дальнейшего использования.

(5.7)

Представим (5.7) также и в матричном виде

 , (5.8)

или

,(5.9)

где

, , 