МИНИСТЕРСТВО НАУКИ И ОБРАЗОВАНИЯ УКРАИНЫ

ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

И ЗАДАНИЯ К ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ ПО ДИСЦИПЛИНЕ "АППАРАТНЫЕ И ПРОГРАММНЫЕ СРЕДСТВА СИСТЕМ КОМПЬЮТЕРНОЙ ГРАФИКИ"

для студентов дневной формы обучения направления

6.050102 “Компьютерная инженерия”

У т в е р ж д е н о
на заседании методической комиссии
кафедры «Компьютерная инженерия»
Протокол N _1_ от 29.08.2013

Донецк, ДонНТУ, 2013

УДК 681.3

Методические указания и задания к выполнению лабораторных работ по дисциплине "Аппаратные и программные средства систем компьютерной графики", (для студентов специальности «Компьютерные системы и сети») / Сост.: Р.В.Мальчева - Донецк: ДонНТУ, 2013. – 34 с.

Приведены варианты заданий и порядок выполнения лабораторных работ по дисциплине.

Составители : Р.В. Мальчева, доцент

В В Е Д Е Н И Е

Целью лабораторных работ по курсу является формирование знаний общей постановки задачи синтеза трехмерного изображения и простейшего алгоритма для ее решения, а также навыков использования графического режима работы дисплея и разработки программного обеспечения, ориентированного на задачу синтеза некоторого объекта.

Выполнение настоящих лабораторных работ может рассматриваться в качестве начального этапа подготовки к самостоятельной разработке графических систем и устройств мультимедиа на базе специализированных аппаратурных и программных средств.

В результате выполнения лабораторных работ студенты должны получить навыки двумерного моделирования, а также сведения о простейшем типовом алгоритме синтеза трехмерного объекта, разработать и отладить соответствующее программное обеспечение и выполнить расчет двумерного изображения для визуализации простейшего многогранника. Разработку программного обеспечения рекомендуется проводить на языке C++ с использованием стандартной графической библиотеки, а также ассемблерных вставок для повышения качества выводимой информации.

На всех основных типах ПЭВМ принят растровый способ изображения графической информации - изображение представлено прямоугольной матрицей точек (пикселей), каждый из которых может иметь свой цвет, выбираемый из заданного набора цветов - палитры. Для реализации этого подхода в составе компьютера имеется видеоадаптер, который хранит в своей видеопамяти изображение и обеспечивает отображение содержимого видеопамяти на экране монитора с частотой 50 Гц.

№	Назва лабораторної роботи	Тривалість (ак.годин)
1	Вивчення операцій «застосування мірила» та «повернення» на прикладі візуалізації графіка функції.	4
2	Підготовка бази даних складної трьомірної сцени	4
3	Синтез двомірного зображення трьомірної сцени методом пріорітетов.	4
4	Синтез двомірного зображення трьомірної сцени методом трасування промінів.	4

Л А Б О Р А Т О Р Н А Я Р А Б О Т А 1

«Вивчення операцій «застосування мірила» та «повернення» на прикладі візуалізації графіка функції»

Цель: Целью данной работы является приобретение студентами навыков вывода результатов расчетов и моделирования в графическом режиме на различные устройства отображения, а также закрепление навыков, полученных ими при изучении дисциплин, связанных с численными методами и дискретной математикой.

Задание к лабораторной работе: Используя 3 последних цифры зачетной книжки (ЗК), из таблицы выбрать значения коэффициентов k_1, k_2, k_3 и выполнить вывод графика функции в диапазоне $-\frac{\pi}{3} \le x \le \frac{\pi}{3}$:

- в попиксельном режиме ;
- используя кусочно-линейную интерполяцию и функцию LINE с шагом $dx = \frac{\pi}{30}$.

Вид функции:

четный номер ЗК: $y = k_1 \cdot e^{k_2 \cdot x} \cdot \sin(k_3 \cdot x)$.

нечетный номер ЗК: $y = k_1 \cdot e^{k_2 \cdot x} \cdot \cos(k_3 \cdot x)$.

Таблица коэффициентов

Цифра	0	1	2	3	4	5	6	7	8	9
k	0,1	-0,1	0,2	-0,2	0,15	-0,15	0,25	-0,25	0,3	-0,3

Этапы выполнения работы:

1. Изучить материал по графическим режимам дисплея.
2. Выполнить анализ режимов, поддерживаемых дисплеями, находящимися в учебной лаборатории.
3. Выполнить тестирование графических режимов.
4. Подготовить эскиз графика функции в математической системе координат.
5. Подготовить формулы масштабирования и преобразования в пиксельный (целочисленный!!!) формат абсцисс и ординат функций, используя разрешение экрана Xmax,Ymax.
6. Разработать блок-схему алгоритма и текст программы.
7. Подготовить отчет по лабораторной работе.

Порядок демонстрации выполнения работы: Демонстрация работы выполняется во всех графических режимах, доступных на устройствах отображения без каких либо дополнительных изменений в тексте программы.

Содержание отчета: В отчете должны быть отражены результаты выполнения всех этапов работы, также заключение.

Пример и рекомендации по выполнению лабораторной работы

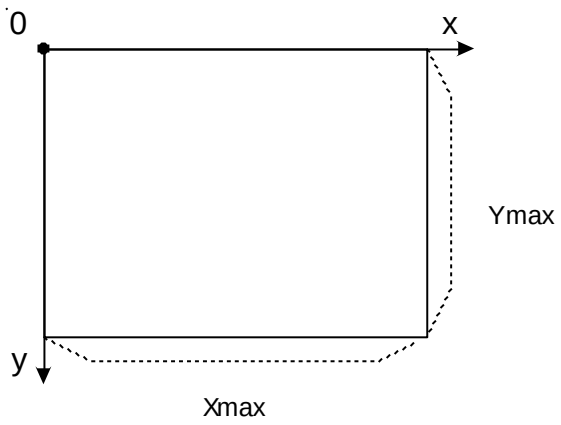
Приведенные процедуры интерполяции в качестве входных параметров используют координаты физических точек (пикселей) экрана. Если исходные параметры интерполируемого графика заданы в системе координат, отличных от экранной, необходимо сперва преобразовать их в координаты пикселей.

Как минимум нужно знать следующие параметры, необходимые для преобразования:

1. Размеры физического экрана вывода (viewport) (в пикселях).
2. Направление координатных осей экрана вывода.
3. Размеры экрана вывода в единицах, в которых задан график.
4. Направление его координатных осей.

Экраном вывода может служить либо физический экран (с разрешением Xmax x Ymax пикселей), либо отдельное окно (часть окна) размера Xmax x Ymax пикселей, куда производится вывод. Для экрана вывода обычно характерны следующие особенности:

- точка пересечения координатных осей находится в левом верхнем углу (координаты (0,0));
- ось X направлена слева направо (0…Xmax-1);
- ось Y направлена сверху вниз (0…Ymax-1).



На рисунке Xmax – максимальная ширина экрана вывода (в пикселях), Ymax – максимальная высота.

Если точка пересечения и направление координатных осей экрана вывода и плоскости графика совпадают, то для преобразования координат плоскости графика в координаты экрана вывода необходимо рассчитать коэффициенты kX и kY:

$$kX = \frac{X\ \max}{X\ \max G};$$

$$kY = \frac{Y\ \max}{Y\ \max G}$$

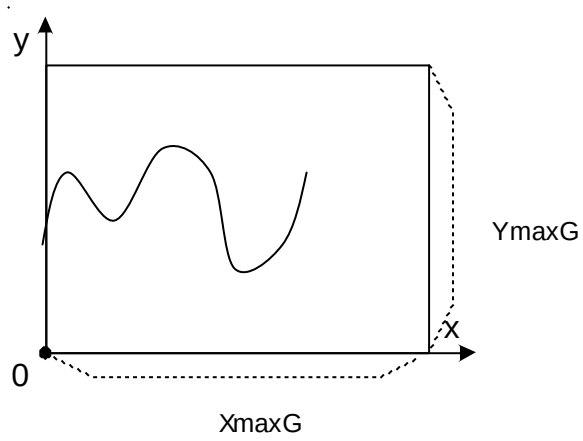
Здесь XmaxG – максимальная X-координата на плоскости графика, YmaxG – максимальная Y- координата. Зная данные коэффициенты, легко перевести координаты из системы координат графика в систему координат экрана:

Xvp = X*kX; Yvp = Y*kY,

где X, Y – координаты точки в системе координат графика,

Xvp, Yvp - координаты точки в системе координат экрана вывода (в пикселях).

Обычно график задан в декартовой системе координат. При этом ось Y направлена снизу вверх, ось X – слева направо.



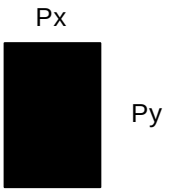
В этом случае дополнительно необходимо выполнить преобразования для Y-координаты. Формула для расчета Yvp будет выглядеть следующим образом:

Yvp = Ymax-Y*kY.

После выполнения данных преобразований точка (Xvp,Yvp) на экране не всегда будет однозначно соответствовать точке (X,Y) графика. Это зависит от выбранного масштаба и разрешения экрана. Например, если размеры экрана вывода численно вдвое меньше размеров в системе координат графика (типа 400x300 против 800x600), то точки с координатами

(500,500),(501,501),(500,501),(501,500) в системе координат графика на экране вывода будут представлены одной и той же точкой с координатами (250,250).

Другая проблема может заключаться в том, что физический пиксель на экране не является квадратным (отношение его сторон не равно 1).



В этом случае для наиболее верного соответствия экранного изображения физическому необходимо скорректировать размеры экрана вывода с учетом размеров пикселя:

XmaxC = Xmax*k1

YmaxC = Ymax*k2

Коэффициенты k1 и k2 рассчитываются исходя из размеров пикселя.

При k1 = 1 (изменяется высота экрана вывода) k2 = Px/Py, таким образом,

XmaxC = Xmax

YmaxC = Ymax * Px / Py.

При k2 = 1 (изменяется ширина экрана вывода) k1 = Py/Px:

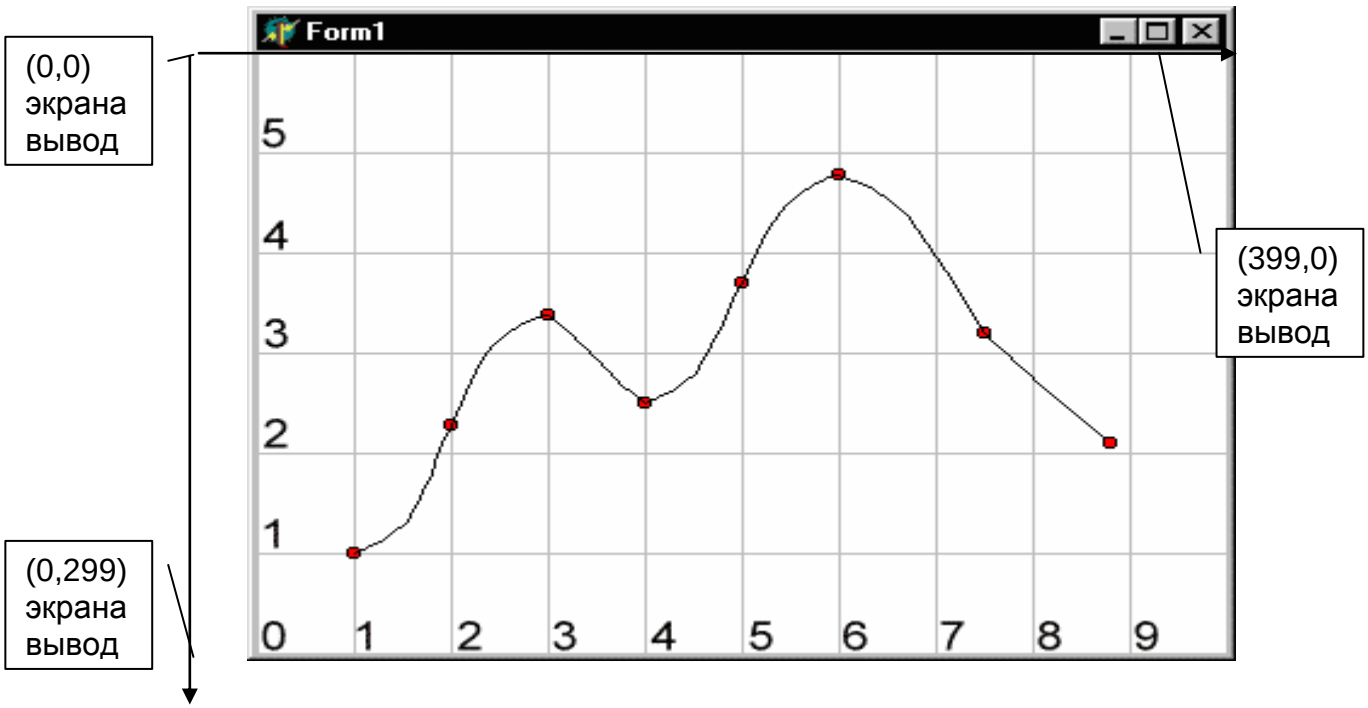
XmaxC = Xmax * Py / Px

YmaxC = Ymax

На рисунке приведен график функции, интерполированной целочисленным методом. Точки были заданы следующим массивом координат:

X	1.0	2.0	3.0	4.0	5.0	6.0	7.5	8.8
Y	1.0	2.3	3.4	2.5	3.7	4.8	3.2	2.1

При этом размеры физического экрана вывода (формы) были взяты 500 на 300 пикселей, размеры сетки координат графика – 10 на 6 единиц.

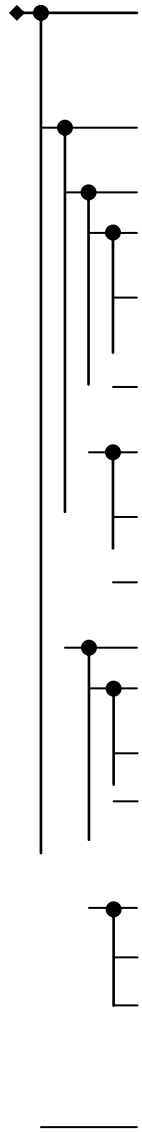


Коэффициенты kX и kY: kX = 500 / 10 = 50; kY = 300 / 6 = 50.

В следующей таблице приведено соответствие между точками в системе координат графика и системе координат экрана.

номер точки (i)	координаты (Xi,Yi)	(Xvpi, Yvpi) после масштабирования	(Xvpi, Yvpi) после переворота оси Y
1	1.0 1.0	50 50	50 250
2	2.0 2.3	100 115	100 185
3	3.0 3.4	150 170	150 130
4	4.0 2.5	200 125	200 175
5	5.0 3.7	250 185	250 115
6	6.0 4.8	300 240	300 60
7	7.5 3.2	375 160	375 140
8	8.8 2.1	440 105	440 195

Рассмотрим более детально интерполяцию части графика (5-6), заданную координатами в системе координат экрана (250,115)-(300,60) (после перевода и переворота оси Y) при заданной величине D = 8.



Ac (x,y)	A (x,y)	B (x,y)	Bc (x,y)	dx	dy	C (x,y)	комментарий
200,175	250,115	300,60	375,140	51	80	274,74	dx > 8, dy > 8, следовательно, разбиваем (5-6) на 2 части и вызываем 2 раза процедуру CurveInt
200,175	250,115	274,74	300,60	24	41	261,93	dx > 8, dy > 8 -> CurveInt
200,175	250,115	261,93	274,74	11	22	255,104	dx > 8, dy > 8 -> CurveInt
200,175	250,115	255,104	261,93	5	11	252,109	dx < 8, dy > 8 -> CurveInt
200,175	250,115	252,109	255,104	2	6	-	dx < 8, dy < 8, проводим отрезок (250,115) – (252,109)
250,115	252,109	255,104	261,93	3	5	-	dx < 8, dy < 8, проводим отрезок (252,109) – (255,104)
250,115	255,104	261,93	274,74	6	11	258,98	dx < 8, dy > 8 -> CurveInt
250,115	255,104	258,98	261,93	3	6	-	dx < 8, dy < 8, проводим отрезок (255,104) – (258,98)
255,104	258,98	261,93	274,74	3	5	-	dx < 8, dy < 8, проводим отрезок (258,98) - (261,93)
250,115	261,93	274,74	300,60	13	19	267,83	dx > 8, dy > 8 -> CurveInt
250,115	261,93	267,83	274,74	6	10	264,88	dx < 8, dy > 8 -> CurveInt
250,115	261,93	264,88	267,83	3	5	-	dx < 8, dy < 8, проводим отрезок (261,93) – (264,88)
261,93	264,88	267,83	274,74	3	5	-	dx < 8, dy < 8, проводим отрезок (264,88) – (267,83)
261,93	267,83	274,74	300,60	7	9	270,78	dx < 8, dy > 8 -> CurveInt
261,93	267,83	270,78	274,74	3	5	-	dx < 8, dy < 8, проводим отрезок (267,83) – (270,78)
267,83	270,78	274,74	300,60	4	4	-	dx < 8, dy < 8, проводим отрезок (270,78) – (274,74)
250,115	274,74	300,60	375,140	26	14	286,64	dx > 8, dy > 8 -> CurveInt
…	…	…	…	…	…	…	…

Содержание отчета:

1. Расчет графика функции.
2. Расчет рабочей области экрана, масштабов.
3. Формулы формирования координат.
4. Текст программы.
5. Файл с результатами.

Л А Б О Р А Т О Р Н А Я Р А Б О Т А 2
«Підготовка бази даних складної трьомірної сцени»

Стадия 1: ПОДГОТОВКА ОПИСАНИЯ ТРЕХМЕРНОГО ОБЪЕКТА»

В этой лабораторной работе в качестве объекта рассматривается выпуклый многогранник, т.е. геометрическое тело, аппроксимированное плоскими выпуклыми одноприоритетными гранями.

1. Требование к математической модели аппроксимации синтезируемой сцены

Повышение степени реализма выдвигает повышенные требования к моделям, применяемым для описания сцены. Среди необходимых составных частей, которые должны быть представлены в модели, можно назвать следующие:

***** основные элементы сцены и их взаимоотношения;

***** геометрия сцены, т.е. пространственное размещение, форма, размеры отдельных составляющих сцены;

***** топология сцены, т.е. информация о взаимном расположении и связности компонент сцены;

* специфические для рассматриваемого применения данные, например, электрические или механические характеристики; описательный текст.

В общем случае модель состоит из прикладной структуры данных и набора процедур прикладной программы для поддержки этой структуры.

2. Подготовка описания синтезируемой сцены

Построение и реализация математической модели визуальной обстановки в компьютерных генераторах изображений могут быть выполнены многими способами, основанными на различных методах и алгоритмах представления графических данных и генерации изображений.

Основное содержание этапа предварительной подготовки (препроцессирования) заключается в следующем. Создание и редактирование графической базы данных заданной сцены. В соответствии с выбранными графическими примитивами, способами описания и обработки объектов сцены на данном этапе определяются способы их представления в памяти и набор операций над ними. Создается и наполняется данными об отображаемой сцене графическая база данных. Редактирование данных графической базы необходимо только при изменении геометрических параметров объектов и/или энергетических свойств их поверхностей.

3 Основные модели описания геометрии синтезируемой сцены

Основные модели, описывающие геометрию объектов, можно классифицировать следующим образом

1) Поверхностные.
2) Модели сплошных тел: ячеечные, граничные, сплошных конструктивов.
3) Проволочные.
4) Точечные.

Поверхностные модели представляют объекты в виде тонких поверхностей, под которыми находится пустое пространство, не заполненное материалом объекта. Из поверхностей первого порядка можно составить описание объекта типа полигонального поля. Таким полем называют серию смежных многоугольников, не имеющих разрыва между собой. Многоугольник задается матрицей координат и нормальным вектором.

Из аналитической геометрии известно, что функция вида

$f2(x,y,z) = Ax^2 + By^2 + Cz^2 + 2Dxy + 2Eyz + 2Fzx + 2Gx + 2Hy + 2I z + K = 0$

в зависимости от набора коэф. A, B, ... , K может описывать поверхности сферы, эллипсоида, гиперболоида, конуса, параболоида, цилиндра и 2-х плоскостей.

Описание в виде ***сплошных тел*** подразумевает, что объекту или отдельному примитиву принадлежат все точки объекта - как наружные, так и внутренние.

Ячеечные методы; идеи очень просты. Весь моделируемый объект считается разбитым на большое число дискретных кубических ячеек . Моделирующая система должна просто записать информацию о принадлежности или непринадлежности каждого куба телу объекта. Структура данных представляется трехмерной матрицей, в которой каждый элемент соответствует пространственной ячейке. Недостаток метода : большой объем памяти , требуемый для записи объекта с высоким разрешением.

Граничное представление. В памяти хранятся поверхности , края поверхностей , указатели пересечений поверхностей. При этом структура данных строится одновременно с процессом синтеза. Модели описания границами допускают использование булевых операций и операций над множествами. Преимущества: большие возможности геометрического моделирования форм. Недостатки: модель логически неустойчива , т.е. возможно создание противоречивых конструкций.

Метод сплошных конструктивов представляет сложные объекты составленными из простых объемных примитивов: кубы, цилиндры, конусы, эллипсоиды. Структура данных такой модели идентична бинарному дереву (построенному снизу вверх). Узлы - операторы над примитивами ; листья - примитивы.

Описание типа ***проволочной модели*** заключается в представлении поверхности серией пересекающихся линий, принадлежащих поверхности объекта.

Точечное описание представляет поверхность объекта множеством отдельных точек. Основной особенностью такого описания служит отсутствие информации о поверхности между точками. Применяется в тех случаях, когда поверхность очень сложна, не обладает гладкостью а детальное представление многочисленных геометрических особенностей важно для практики. (Участки грунта планет, формы малых небесных тел - со спутников / микрообъекты - снятые при помощи микроскопов).

5. Пример файла описания параллелепипеда

Описание объекта представлено в файле s_obj.dat.

Содержимое файла:

```
//Габаритный радиус объекта      :  240
//Количество граней объекта      :  6
 //Описание грани 1              :
 //Количество вершин грани       :  4
 //Код цвета грани               :  3
 //Координата вершины 1          :  100.0   50.0 -200.0
 //Координата вершины 2          :  100.0   50.0  200.0
 //Координата вершины 3          :  100.0  -50.0  200.0
 //Координата вершины 4          :  100.0  -50.0 -200.0
 //Нормаль грани  1              :   1.0    0.0    0.0
 //Описание грани 2              :
 //Количество вершин грани       :  4
 //Код цвета грани               :  4
 //Координата вершины 1          :  100.0   50.0 -200.0
```

```
//Координата вершины 2          : -100.0   50.0 -200.0
//Координата вершины 3          : -100.0   50.0  200.0
//Координата вершины 4          :  100.0   50.0  200.0
//Нормаль грани  2              :    0.0    1.0    0.0
//Описание грани 3              :
//Количество вершин грани       :   4
//Код цвета грани               :   5
//Координата вершины 1          :  100.0   50.0 -200.0
//Координата вершины 2          :  100.0  -50.0 -200.0
//Координата вершины 3          : -100.0  -50.0 -200.0
//Координата вершины 4          : -100.0   50.0 -200.0
//Нормаль грани  3              :    0.0    0.0   -1.0
//Описание грани 4              :
//Количество вершин грани       :    4
//Код цвета грани               :    6
//Координата вершины 1          :  100.0  -50.0  200.0
//Координата вершины 2          : -100.0  -50.0  200.0
//Координата вершины 3          : -100.0  -50.0 -200.0
//Координата вершины 4          :  100.0  -50.0 -200.0
//Нормаль грани  4              :    0.0   -1.0    0.0
//Описание грани 5              :
//Количество вершин грани       :     4
//Код цвета грани               :     7
//Координата вершины 1          : -100.0   50.0  200.0
//Координата вершины 2          : -100.0  -50.0  200.0
//Координата вершины 3          :  100.0  -50.0  200.0
//Координата вершины 4          :  100.0   50.0  200.0
//Нормаль грани  5              :    0.0    0.0    1.0
//Описание грани 6              :
//Количество вершин грани       :     4
//Код цвета грани               :     1
//Координата вершины 1          : -100.0  -50.0 -200.0
//Координата вершины 2          : -100.0  -50.0  200.0
//Координата вершины 3          : -100.0   50.0  200.0
//Координата вершины 4          : -100.0   50.0 -200.0
//Нормаль грани  6              :   -1.0    0.0    0.0
```

Задание по лабораторной работе

1. По вариантам табл.4.1 подготовить описание объекта в собственной системе координат. Описание должно включать:

- габаритный радиус объекта,
- количество граней объекта,
- описание граней (количество вершин грани; координаты вершин или список вершин, если координаты вершин заданы отдельным массивом; параметры нормального вектора).

Порядок задания вершин - по часовой стрелке с видимой стороны.

Таблица 4.1 - Тип объекта (по последней цифре зачетной книжки)

0	прямая пирамида с основанием из 3 вершин
1	прямая пирамида с основанием из 4 вершин
2	прямая пирамида с основанием из 5 вершин
3	Усеченная прямая пирамида с основанием из 3 вершин
4	Усеченная прямая пирамида с основанием из 4 вершин
5	Усеченная прямая пирамида с основанием из 5 вершин
6	Наклонная пирамида с основанием из 4 вершин
7	Наклонная пирамида с основанием из 5 вершин
8	Усеченная наклонная пирамида с основанием из 4 вершин
9	Усеченная наклонная пирамида с основанием из 5 вершин

Содержание отчета по лабораторной работе

1. Задание.
2. Рисунок объекта синтеза в локальной системе координат с указанием нумерации вершин, граней, нормалей.
3. Содержание базы данных описания объекта визуализации с пояснением и/или расчетом нормалей каждой грани.

Л А Б О Р А Т О Р Н А Я Р А Б О Т А 3
«СИНТЕЗ ИЗОБРАЖЕНИЯ СЦЕНЫ МЕТОДОМ ПРИОРИТЕТОВ»

1. Теоретические основы

1.1 Алгоритм удаления скрытых поверхностей с использованием списка приоритетов

Сущность данного алгоритма в том, что бы все элементы сцены выводились в соответствии с некоторым списком, в котором записан порядок отображения объектов предварительно отсортированный по глубине или приоритету. Если такой список окончателен, то никакие два элемента не будут взаимно перекрывать друг друга. Тогда можно записать все элементы в буфер кадра поочередно, начиная с элемента, наиболее удаленного от положения наблюдателя. Более близкие к наблюдателю элементы будут затирать информацию о более далеких элементах сцены. Поэтому задача об удалении невидимых поверхностей решается тривиально.

Во многих случаях моделирования в реального времени, например при имитации посадки самолета, сцена статична, а меняется только точка наблюдателя. По предложению Шумейкера начали вычислять в автономном режиме некоторые более общие приоритетные характеристики, такие, как список приоритетов для моделирования упомянутых выше статических сцен.

В алгоритме Шумейкера в сцене допускаются только выпуклые многоугольники. Такие многоугольники группируются в кластеры.

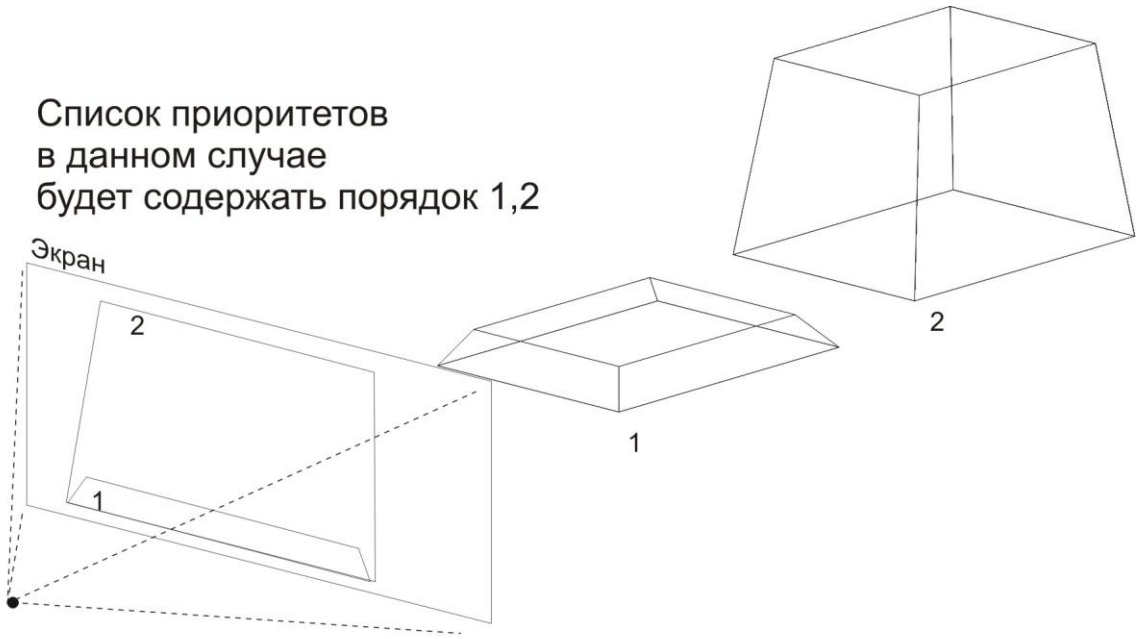


Рисунок 1.1 Алгоитм художника в действии

1.2 Алгоритм трехмерного отсечения

Отсечение – процесс удаления некоторой части базы данных. Играет важную роль в задачах машинной графики.

Алгоритмы отсечения бывают дву- или трехмерные и применяются как к регулярным, так и не регулярным областям и объемам.. Эти алгоритмы можно реализовать аппаратно или программно. Алгоритмы отсечения, реализованные программно, зачастую оказываются недостаточно быстродействующими для приложений, ориентированных на процессы, протекающие в реальном времени. Поэтому как трех-, так и двумерные алгоритмы отсечения реализуются аппаратными или микропрограммными средствами. В подобных реализациях обычно ограничиваются дву- или трехмерными отсекателями типовых форм.

Двумя наиболее распространенными отсекателями являются: прямоугольный параллелепипед , т.е. полый брусок, используемый при аксонометрической проекции, а также усеченная пирамида (пирамида видимости), которая используется при центральном проецировании.

У каждой из таких форм есть шесть граней: левая, правая, верхняя, нижняя, дальняя ближняя. Для идентификации положения отрезка относительно отсекателя удобно использовать, как и в 2D-отсечении, коды концевых точек Коэна-Сазерленда. Только в трехмерном случае используется 6-битовый код. Самый правый бит кода считается первым. В биты кода заносятся единицы:

- в первый, если конец отрезка находится левее объема;

-во второй, если конец отрезка правее;

-в третий, если конец отрезка ниже;

-в четвертый, если конец отрезка выше;

-в пятый, если конец отрезка ближе;

-в шестой, если конец отрезка дальше;

В противном случае в соответствующие биты заносится ноль.

После получения кода легко можно определить необходимость отсечения. Если коды обоих концов отрезка равны нулю, то отрезок является полностью видимым. Если же

побитовое логическое произведения кодов конечных точек отрезка не равны нулю, то он полностью невидим. А если это произведение равно нулю, то данный отрезок может оказаться как частично так и полностью невидимым.

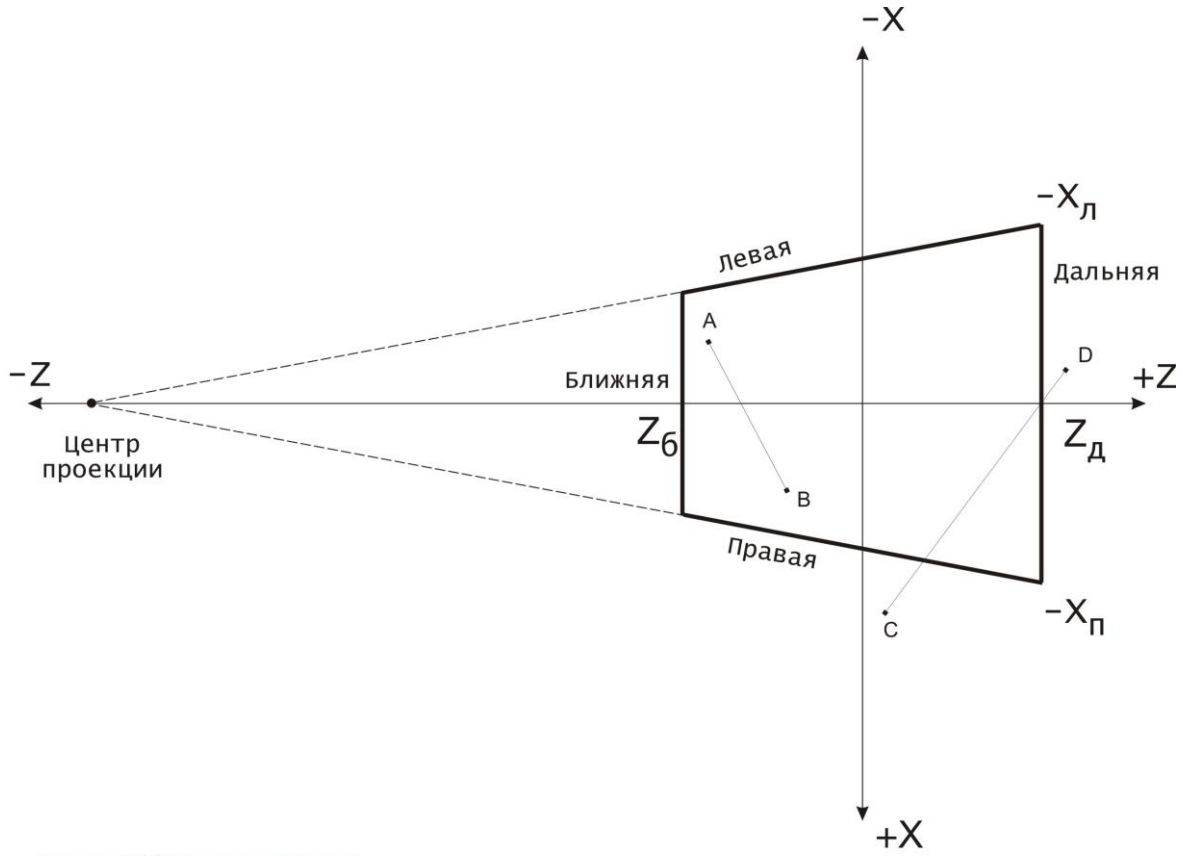


Рисунок 1.2 Усеченная пирамида

Как видно с рисунка 1.2 отрезок AB является полностью видимым, а отрезок CD является частично видимым (коды его точек будут 000010 и 100000 соответственно).

На рис.1.3, 1.4 приведены блок-схемы.

2 . Последовательность выполнения работы

Целью данной лабораторной работе является разработка программной системы синтеза динамического трехмерного объекта. Последовательность выполнения работы следующая:

- аппроксимация объекта 8 - 10 выпуклыми многогранниками;
- подготовка файлов исходного описания трехмерного объекта, аппроксимированного выпуклыми многогранниками;
- разработка программы синтеза объекта заданным методом;
- разработка интерактивной системы управления перемещением объекта (или точки наблюдения) для тестирования объекта.

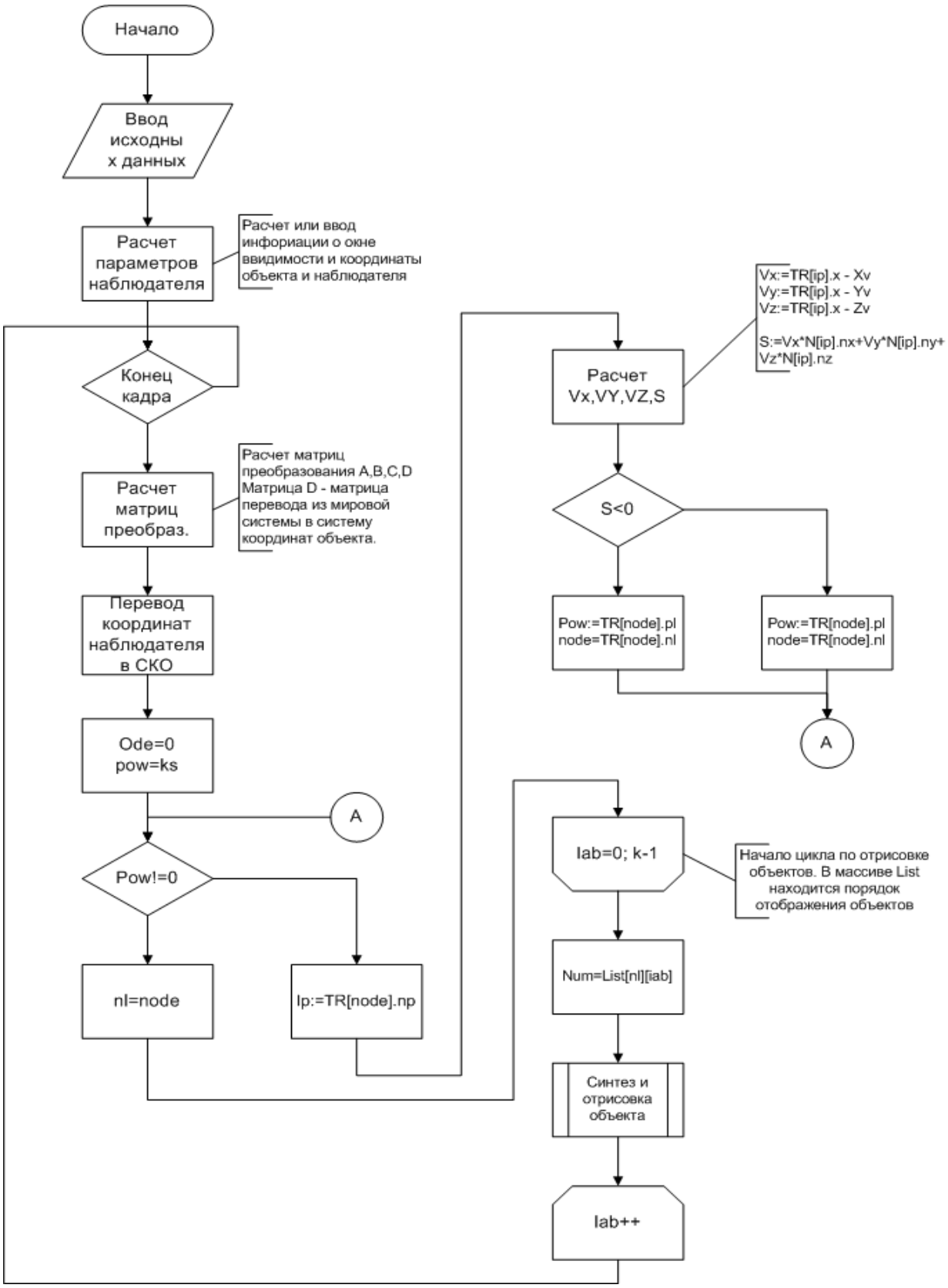


Рисунок 1.3 Блок схема реализации метода приоритетов

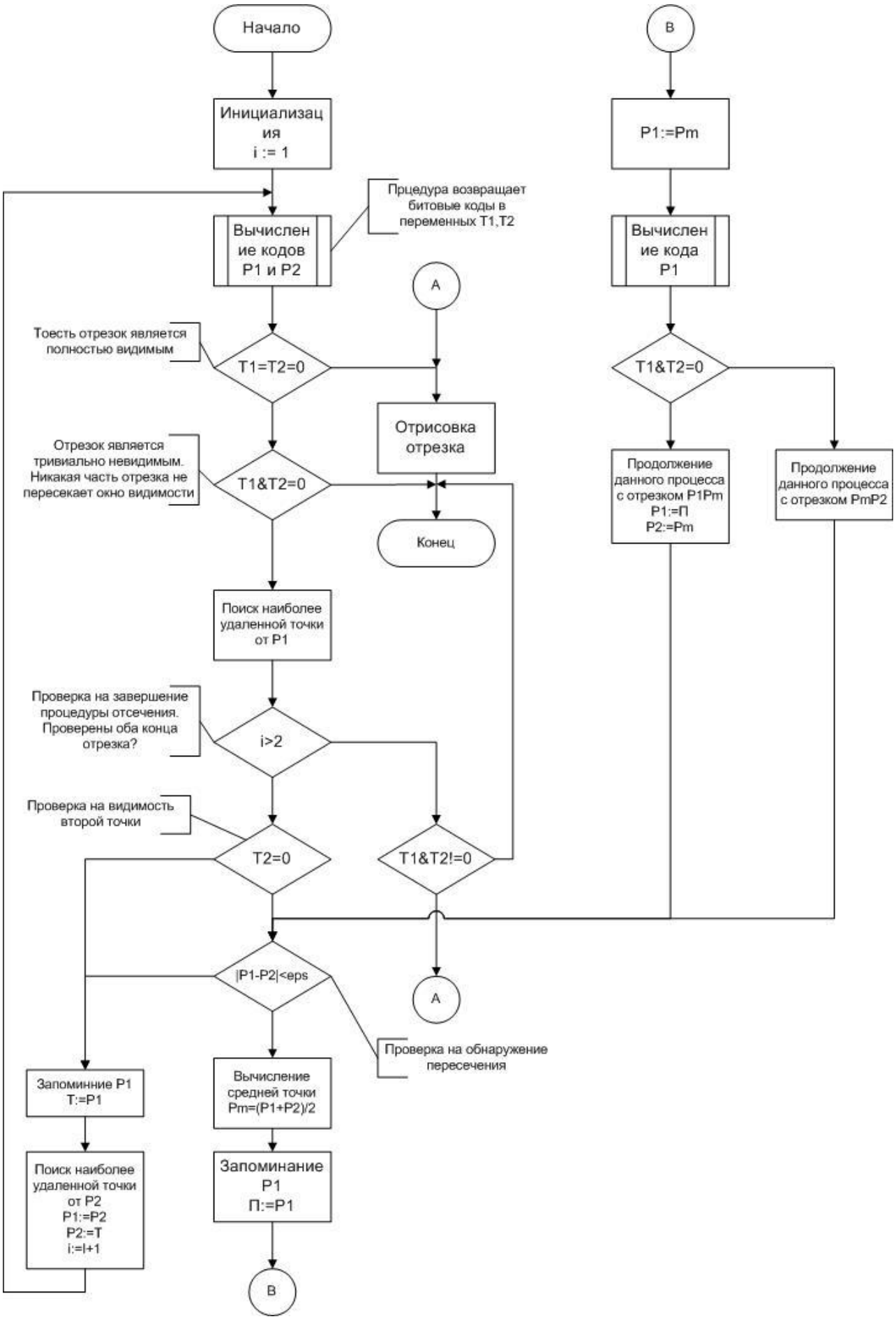


Рисунок 1.4 Блок схема алгоритма отсечения по границе видимого объема

2.1. Аппроксимация объекта

Под объектом визуализации понимается тело в трехмерном пространстве, заданное в своей системе координат и имеющее единое функциональное назначение. Объект визуализации (составной объект) подразделяется на примитивные объекты, представляющие собой выпуклые многогранники и имеющие единое геометрическое представление и функциональное назначение. Топология объекта определяет взаимное расположение составляющих его примитивных объектов.

Для описания выпуклого многогранника применяется полигональная аппроксимация, т.е. поверхность объекта разделяется на конечное число граней. Грань представляется ограничивающими ее ребрами и вершинами. Геометрия объекта задается координатами его вершин и ориентацией граней в пространстве, а его топология определяет связи между многогранниками в пространстве.

- координаты вершин граней 4;
- нормали граней.

Грань может быть описана, например, следующим образом:

```
struct
  {
    unsigned    int num_edge, /* Номер грани           */
                priznak_edge, /* Признак грани         */
                num_points,   /* Количество вершин     */
                points[16];   /* Список номеров вершин */
        struct
             {
               long int x, y, z;   /* Трехмерные
                                      координаты вершины*/
               unsigned char color;/* Цвет вершины      */
             }pnt[16];       /* Массив структур вершин */
  }edge;                     /* Описание одной грани */
```

Так как реальные объекты, в общем случае, ограничены довольно сложными геометрическими поверхностями, в качестве первого этапа описания примитивного объекта выполняется аппроксимация его плоскими гранями. Все вершины и грани полученного выпуклого многогранника нумеруются (для каждого примитивного объекта используется своя нумерация), при этом количество граней и количество вершин в каждой грани ограничено. Аппроксимация примитивного объекта выполняется проектировщиком вручную на основании имеющихся чертежей.

Построение дерева топологии объекта и формирование приоритетных списков

Построение дерева топологии объекта необходимо для синтеза объета приоритетным методом. Идея метода высказана Шумейкером, который предложил заранее формировать приоритетные списки неподвижных выпуклых объектов сцены, разбивая пространство сцены на субпространства - кластеры, которые линейно независимы. Идея здесь заключается в том, что для каждого такого субпространства заранее верно определимы приоритеты объектов, определяющие порядок их вывода на экран дисплея для правильного удаления невидимых с точки зрения наблюдателя поверхностей в процессе синтеза изображения. Алгоритмы статической пространственной сортировки позволяют за счет односторонней видимости граней статически переставить их так, чтобы при любом положении наблюдателя порядок обработки граней соответствовал порядку их потенциального закрывания друг другом. Порядок следования от ближних к дальним относительно положения наблюдателя называют прямым приоритетным порядком. В этом случае грани, попавшие на обработку первыми, не могут перекрыться последующими гранями. При обратном приоритетном порядке (от дальних к ближним) каждая

последующая грань потенциально закрывает предыдущую. Приоритетные системы отображения удаляют невидимые поверхности либо путем последовательного наложения (при обратном приоритетном порядке), либо за счет пометки уже занятых пикселей (при прямом приоритетном порядке). Грани, образующие выпуклый объект, одноприоритетны. Наличие такой информации значительно уменьшает объем вычислений на этапе синтеза изображения.

Дерево топологии строится на основе разбиения пространства разделяющими плоскостями на кластеры, в каждом из которых порядок отображения многогранников определяется однозначно. В узлах дерева 4задаются нормали разделяющих плоскостей, листья дерева содержат приоритетные списки отображения многогранников. Дл нессиметричного дерева можно использовать следующее описание (нормаль задана начальной и конечной точкой):

```
struct
  {
    int N1,ST1,N2,ST2;
    signed long X1,Y1,Z1;
    signed long X2,Y2,Z2;
  } VERT[NVMAX];
```

«СИНТЕЗ МНОГОГРАННИКА»

2.1 Постановка задачи синтеза изображения

Задача синтеза изображений сводится к моделированию процессов образования у наблюдателя зрительных образов некоторой реальной ситуации и представляет собой генерацию последовательности кадров изображения с заданной частотой смены кадров. Каждый кадр изображения формируется в результате выполнения алгоритма визуализации, исходными данными для которого служат база данных модели отображаемой сцены и параметры, определяющие положение в пространстве наблюдателя и направление его взгляда, а также положение отображаемой сцены в каждый момент времени выполнения процесса моделирования.

2.2 Системы координат

Положение сцены в стационарном пространстве (Xm,Ym,Zm) в дискретные моменты времени определяется значением параметров (Xo,Yo,Zo) начала системы координат сцены и углами (Po,Qo,Go) ориентации ее в пространстве (Xm,Ym,Zm). Положение глаза наблюдателя совпадает с началом координатной системы наблюдателя (Xн, Yн,Zн). Направление взгляда наблюдателя совпадает с осью Xн и определяется углами Эйлера Pн,Qн,Gн, которые задают ориентацию системы (Xн,Yн,Zн) в пространстве (Xm,Ym,Zm). Перпендикулярно направлению взгляда на расстоянии Do от точки наблюдения расположено прямоугольное окно плоскости обзора размером Ao,Bo.

Наблюдатель видит изображение перспективной проекции части сцены, попадающей в поле зрения, т.е. в пределы окна на плоскости обзора. Задача синтеза изображения требует, чтобы зрительные образы в реальных и модельных условиях были адекватны. Степень адекватности образа, который оператор видит на экране графического устройства вывода, реальной обстановке характеризует качество системы синтеза изображений.

Для синтеза изображения трехмерного объекта необходимо подготовить базу данных описания объекта (в соответствии с принятым способом аппроксимации и требованиями алгоритма синтеза) и загрузить ее, при возможности, в оперативную память.

В общем случае, процесс синтеза – это бесконечный цикл смены кадров изображения. Смена кадров должна выполняться с частотой не менее 20 кадров в секунду (требования непрерывности «картинки» на экране). Реально, исходными данными для процесса синтеза являются результаты динамики движения объекта и наблюдателя. Для простейших примеров для задания и изменения векторов положения можно использовать изменение параметров при помощи управляющих клавиш или в цикле по некоторому закону.

3. Последовательность синтеза одного кадра

3.1. Ввод положений систем координат и расчет матриц преобразования.

В каждом кадре как наблюдатель, так и объекты сцены могут изменять свое расположение. Для отработки этих изменений в начале синтеза каждого кадра вводятся параметры (Xo,Yo,Zo,Po,Qo,Go,Xн,Yн,Zн,Pн,Qн,Gн), определяющих взаимное расположение центра отображаемой сцены и наблюдателя в мировой системе координат. По введенным значениям рассчитываются матрицы преобразований из одной системы в другую:

- A - преобразование из системы координат отображаемой сцены в мировую систему координат.
- B - преобразование из мировой системы координат в систему координат наблюдателя.
- D - матрица, обратная A по формулам:
- C - преобразование из системы координат отображаемой сцены в систему координат наблюдателя.

Аналитические выражения для вычисления значений коэффициентов матриц получены путем последовательного применения операций преобразований 3-D координат и приведены в приложении к м.у. по индивидуальной работе.

3.2 Задача отсечения относительно видимого объема

Необходимо отметить, что решение задачи удаления невидимых поверхностей обычно начинается на этапе формирования дисплейного списка кадра, когда из базы выбираются объекты и/или их элементы,
потенциально видимые через экран в текущем положении наблюдателя.

Прежде чем выполнять преобразование координат объекта в систему координат наблюдателя (процесс, требующий больших затрат времени), целесообразно выполнить проверку: виден ли вообще (хотя бы частично) данный объект при текущем взаимном расположении объекта и наблюдателя. Объект может находиться за наблюдателем (направление взгляда противоположно расположению объекта) или находиться перед наблюдателем, но настолько высоко (низко, слева или справа), что не будет виден через окно наблюдения, т.е. его изображение не попадет на плоскость окна (экрана). Наличие такой информации может существенно сократить временные затраты при синтезе текущего кадра изображения.

При определении потенциальной видимости объекта задача решается в системе координат наблюдателя, поэтому в качестве первого шага алгоритма на данном этапе необходимо выполнить перевод координат центров объектов из мировой системы координат в систему координат наблюдателя, т.е. умножить их на матрицу B. Затем последовательно произвести анализ видимости каждого объекта. При определении видимости достаточно представлять объекты их габаритными шарами, что значительно упрощает анализ и в то же время позволяет отбрасывать заведомо невидимые в данном кадре объекты.

Для каждого объекта входными данными на этом шаге являются:

- размеры a*b экрана, расположенного перпендикулярно оси наблюдения (совпадает с осью Ox системы координат наблюдателя) ;
- расстояние от экрана до наблюдателя, d;
- координаты центра объекта, Xi,Yi,Zi;

- радиус габаритного шара объекта, Ri.

Исходя из размеров экрана, область ограничения видимости на расстоянии d равна:

Y = (-a/2 ; a/2);

Z = (-b/2 ; b/2);

Соответственно, размеры окна, ограничивающие видимость наблюдателя в плоскости, проведенной через центр объекта перпендикулярно оси наблюдения, будут определяться по следующим соотношениям :

a' = X * a/d; b' = X * b/d;

Очевидно, что объект находится вне поля видимости, если:

- Объект расположен полностью за экраном, то есть значение координаты X центра экрана меньше разности d и R:

X < d - R

- Центр объекта находится вне границ a' и b', т.е.

(Y < -R - b'/2) или (Y > b'/2 + R);

(Z < -R - a'/2) или (Z > a'/2 + R);

3.3. Анализ видимости грани

На этом этапе в начале выполняется анализ направляющего вектора каждой грани объекта, при этом нормаль грани преобразуется в систему координат наблюдателя

NXi'=NXi*C11+NYi*C12+NZi*C13

NYi'=NXi*C21+NXi*C22+NZi*C23

NZi'=NXi*C31+NYi*C32+NZi*C33

Затем координаты одной вершины грани объекта преобразуются в систему координат наблюдателя в соответствии со следующими соотношениями:

X1'=X1*C11+Y1*C12+Z1*C13+C14

Y1'=X1*C21+X1*C22+Z1*C23+C24

Z1'=X1*C31+Y1*C32+Z1*C33+C34

и рассчитывается скалярное произведение вектора наблюдения на нормаль грани

S=X1'*(NXi')+(Y1')*(NYi')+ (Z1')*(NZi')

3.4. Задача преобразования координат объекта в систему координат наблюдателя

На следующем этапе координаты каждой лицевой грани объекта преобразуются в систему координат наблюдателя в соответствии со следующими соотношениями:

Xi'=Xi*C11+Yi*C12+Zi*C13+C14

Yi'=Xi*C21+Xi*C22+Zi*C23+C24

Zi'=Xi*C31+Yi*C32+Zi*C33+C34

3.5.Задача отсечения по плоскости окна

Затем проверяется положение каждой грани относительно оконной плоскости. Если грань вся лежит перед окном, то она отбрасывается, если вся за окном, то передается для дальнейшей обработки. В случае, если грань пересекает плоскость, которой принадлежит окно, рассчитываются значения координат точек пересечения грани с плоскостью окна в соответствии со следующими выражениями:

Xпп=Dэ

Yпп=(Dэ-Xпэ)*(Yзэ-Yпэ)/(Xзэ-Xпэ)+Yпэ

Zпп=(Dэ-Xпэ)*(Zзэ-Zпэ)/(Xзэ-Xпэ)+Zпэ,

где индексы переменных имеют следующий смысл:

пп -точка пересечения,

пэ -точка перед экраном,

зэ- точка за экраном.

Dэ- расстояние от точки наблюдения до экранной плоскости.

3.6. Проецирование на плоскость окна и перевод в экранные координаты

Проецирование трехмерных координат на оконную плоскость производится по формулам:

Xiэ=Kx*Zi/Xi

Yiэ=Ky*Yi/Xi

Этап масштабирования координат – перевод в экранные координаты (пиксели) с учетом линейных размеров экрана, которые в общем случае могут не совпадать с размерами окна, обычно, совмещается с этапом проецирования, при этом масштабные коэффициенты учитываются при расчете коэффициентов проецирования Кх и Ку.

3.7. Отсечение по границам экрана

Затем следует этап отсечения граней по границам экрана. Для получения координат точек пересечения грани с границами экрана выполняются проверки по всем четырем границам. При использовании формата с плавающей точкой для отсечения по границе "+Xэ" применяются следующие соотношения:

Xiэ=Xэ

Yiэ=(Xэ-Xiпр)*(Yiпс-Yiпр)/(Xiпс-Xiпр)+Yiпр

Для остальных границ соотношения аналогичны.

При использовании целочисленного значения целесообразнее использовать метод половинного деления.

Разработка программы синтеза объекта заданным методом приоритетов

Для синтеза метолом приоритетов необходимо добавить внешний цикл по многогранникам, а, следовательно, выполнить выборку приоритетного списка многогранников для каждой точки наблюдения. Анализ кластера, в котором находится наблюдатель выполняется путем проверки знака скалярного произведения вектора наблюдения на нормаль разделяющей грани. Анализ выполняется в системе координат объекта, поэтому точка наблюдения, заданная в мировой системе координат, преобразуется в систему координат объекта.

Фрагмент программы для выбора списка многогранников:

```
struct
     {
     int N1,ST1,N2,ST2;
     signed long X1,Y1,Z1;
     signed long X2,Y2,Z2;
     } VERT[NVMAX];
struct
     {
     char namef[20];
     } FLIST[NSMAX];

int main()
{
  ...

  / *чтение файла  данных*/
cikl='Y';
while(cikl=='Y')
{
//  Ввод векторов положения точки наблюдения и объекта

       matr_a=&a[0][0]; matr_b=&b[0][0];
       matr_d=&d[0][0]; matr_c=&c[0][0];
  matr_fun(pso,teo,gao,pss,tes,gas,        //вычисление
           matr_a,matr_b,matr_d,matr_c);   //матриц
```

```
  fun_smn(x_o,y_o,z_o,x_s,y_s,z_s,matr_b,matr_c);//смещения
// перевод точки наблюдения в систему объекта
  x_s1=mul_x_y(x_s,d[0][0])+mul_x_y(y_s,d[0][1])
                 +mul_x_y(z_s,d[0][2])-x_o;
  y_s1=mul_x_y(x_s,d[1][0])+mul_x_y(y_s,d[1][1])
                 +mul_x_y(z_s,d[1][2])-y_o;
  z_s1=mul_x_y(x_s,d[2][0])+mul_x_y(y_s,d[2][1])
                 +mul_x_y(z_s,d[2][2])-z_o;
// спуск по дереву - выборка номера списка
    ivs=1;
    sts=1;
  while(sts!=0)
        {
         iv=ivs-1;
         XX1=VERT[iv].X1;
         YY1=VERT[iv].Y1;
         ZZ1=VERT[iv].Z1;
         XX2=VERT[iv].X2;
         YY2=VERT[iv].Y2;
         ZZ2=VERT[iv].Z2;
         S=(XX1-x_s1)*(XX2-XX1)+(YY1-y_s1)*(YY2-YY1)+(ZZ1-z_s1)*(ZZ2-ZZ1);
         if(S<0)
                { sts=VERT[iv].ST1; ivs=VERT[iv].N1;                }
                else
                { sts=VERT[iv].ST2; ivs=VERT[iv].N2;                 }
         }
          ils=ivs;
          il=ils-1;
// выборка списка номер il длиной DLS
  for(k=0;k<DLS+1;k++) //основной цикл
  {
  ...
  // синтез многогранника
  ...
  } //конец основного цикла
  ...
} // коней программы
```

Варианты задания:

Сцена состоит из 8-ми выпуклых многогранников различного размера. Тип многогранника тот же, что и в 4-й работе. Четные варианты используют симметричное дерево, нечетные – несимметричное.

ПРИМЕР.

3. Исходные данные сцены

Сцена, в данном случае, имеет три составляющие: объект, наблюдатель и объем видимости (окно). Каждый из объектов имеет свои параметры, как, то координаты, угол вращения и др заданные в мировой или своих системах координат.

Рассмотрим подробнее объект сцены:

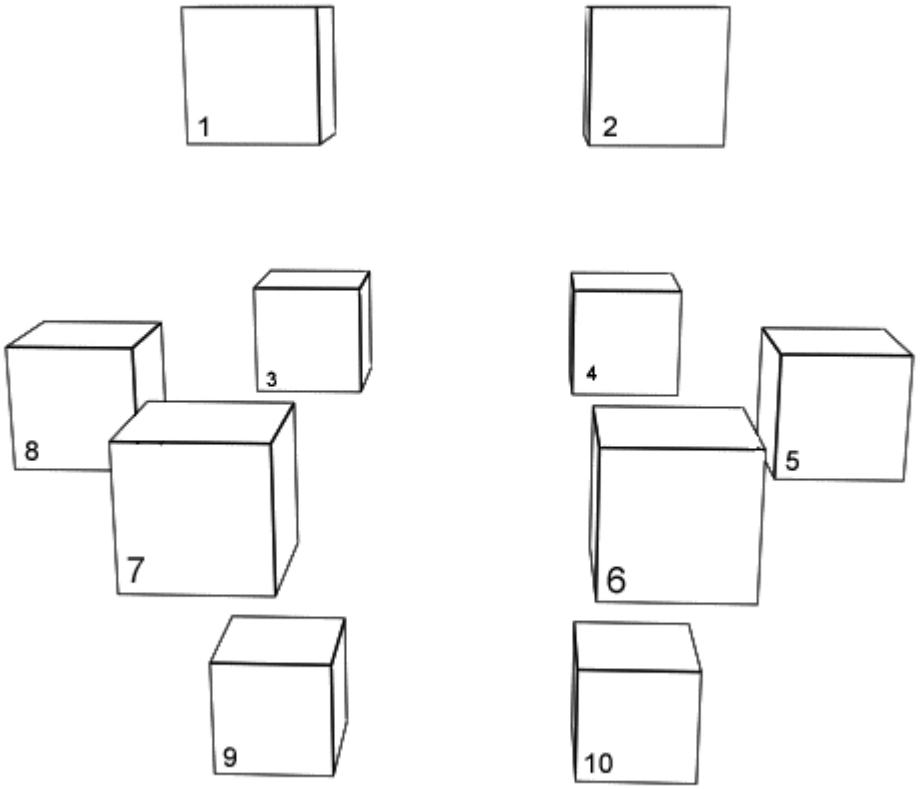


Рисунок 2.1 Сцена из 10-ти объектов

Объект состоит из десяти кубов, каждый в свою очередь состоит из 8 граней и 12 вершин. Как уже говорилось выше для данного метода необходимо составить список приоритетов, а для этого в свою очередь надо разбить сцену разделяющими плоскостями так, чтоб для каждого элемента сцены была выделен свой ограничивающий объем. Для данного случая разделяющие плоскости будут распределены следующим образом :

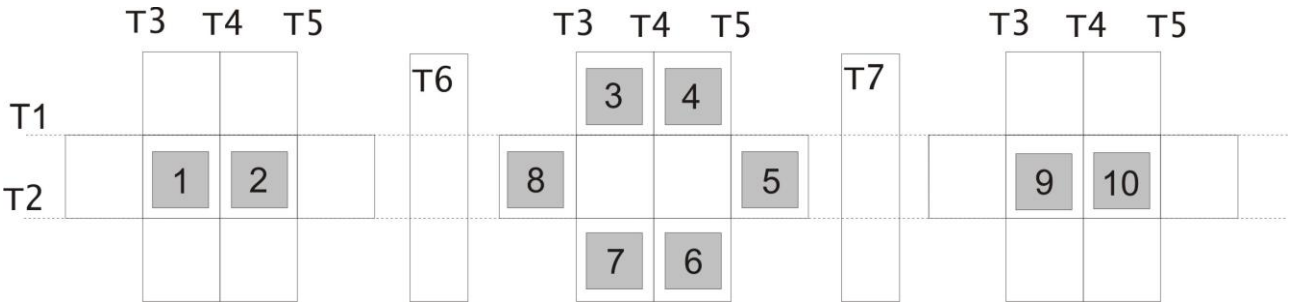


Рисунок 2.2 Слева на право: плоскость выше T6, плоскость ниже T6, но выше T7, плоскость ниже T7

В связи с чем дерево будет иметь вид, показанный на рис.2.3.

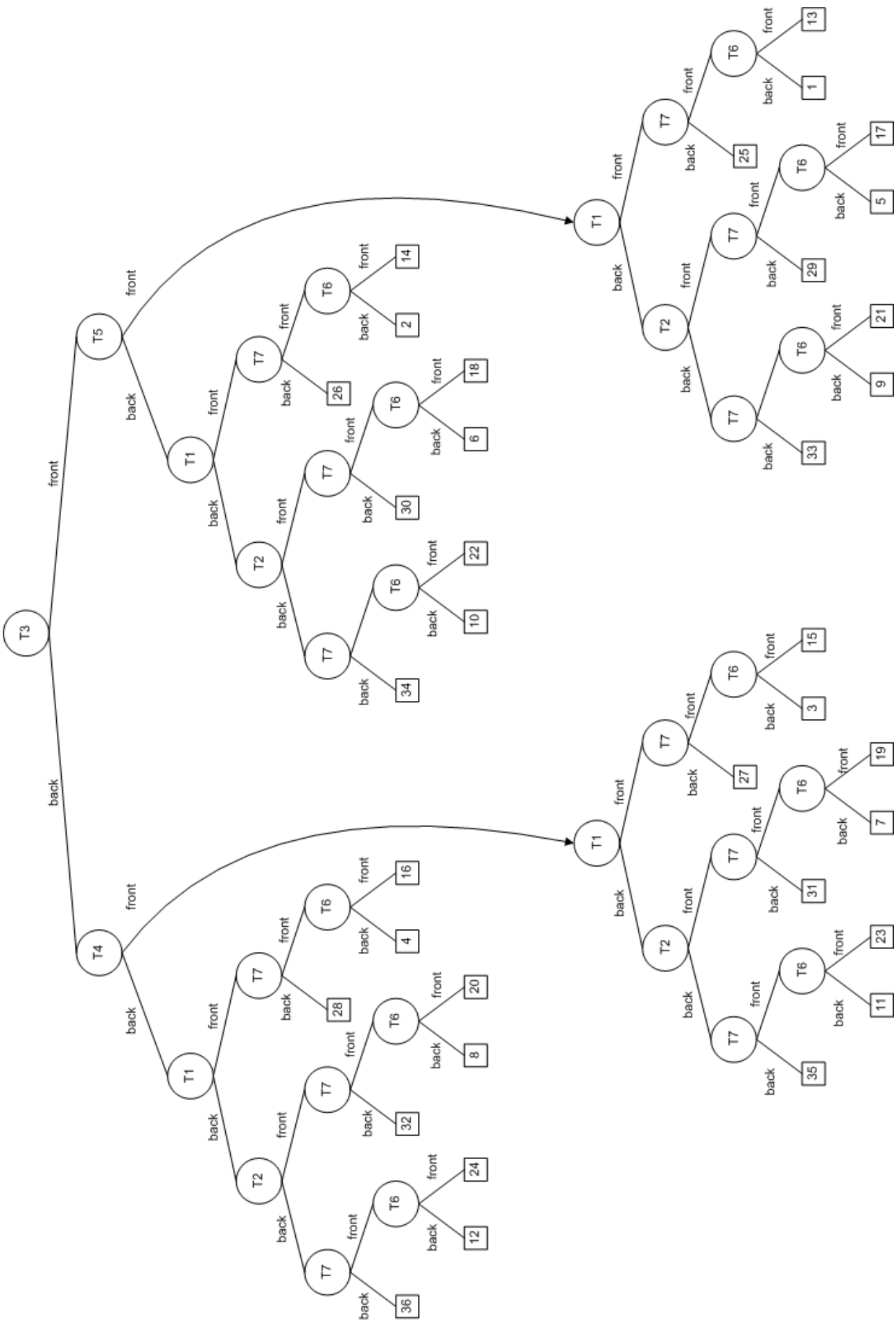
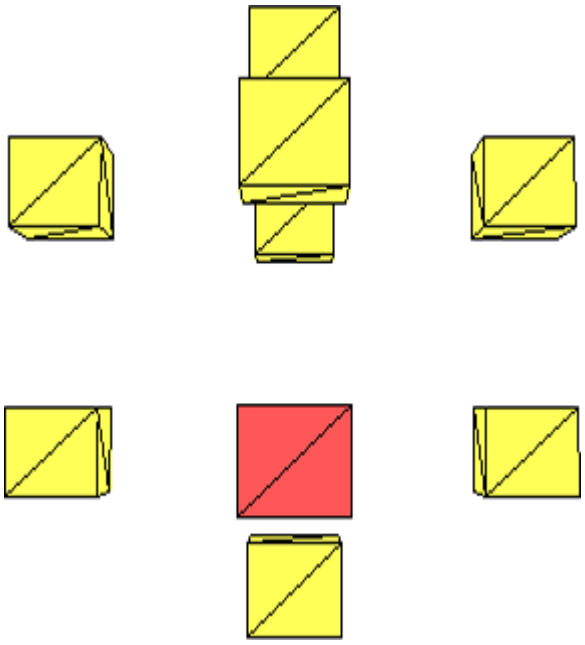


Рисунок 2.3 Дерево приоритетов

Тогда списки получаются следующими

<list var="10:9:2:1:5:4:6:3:7:8"/> <!-- 1 -->
<list var="10:9:2:1:6:7:5:8:4:3"/> <!-- 2 -->
<list var="9:10:1:2:7:8:6:3:5:4"/> <!-- 3 -->
<list var="9:10:1:2:8:3:7:4:6:5"/> <!-- 4 -->
<list var="10:9:2:1:5:4:6:3:7:8"/> <!-- 5 -->
<list var="10:9:2:1:5:4:6:3:7:8"/> <!-- 6 -->
<list var="9:10:1:2:8:3:7:4:6:5"/> <!-- 7 -->
<list var="9:10:1:2:8:3:7:4:6:5"/> <!-- 8 -->
<list var="10:9:2:1:5:4:6:3:7:8"/> <!-- 9 -->
<list var="10:9:2:1:4:3:5:8:6:7"/> <!-- 10 -->
<list var="9:10:1:2:3:4:8:5:7:6"/> <!-- 11 -->
<list var="9:10:1:2:8:3:7:4:6:5"/> <!-- 12 -->
<list var="10:9:5:4:6:3:7:8:2:1"/> <!-- 13 -->
<list var="10:9:6:7:5:8:4:3:2:1"/> <!-- 14 -->
<list var="9:10:7:8:6:3:5:4:1:2"/> <!-- 15 -->
<list var="9:10:8:3:7:4:6:5:1:2"/> <!-- 16 -->
<list var="10:9:5:4:6:3:7:8:2:1"/> <!-- 17 -->
<list var="10:9:5:4:6:3:7:8:2:1"/> <!-- 18 -->
<list var="9:10:8:3:7:4:6:5:1:2"/> <!-- 19 -->
<list var="9:10:8:3:7:4:6:5:1:2"/> <!-- 20 -->
<list var="10:9:5:4:6:3:7:8:2:1"/> <!-- 21 -->
<list var="10:9:4:3:5:8:6:7:2:1"/> <!-- 22 -->
<list var="9:10:3:4:8:5:7:6:9:1"/> <!-- 23 -->
<list var="9:10:8:3:7:4:6:5:1:2"/> <!-- 24 -->
<list var="2:1:5:4:6:3:7:8:10:9"/> <!-- 25 -->
<list var="2:1:6:7:5:8:4:3:10:9"/> <!-- 26 -->
<list var="1:2:7:8:6:3:5:4:9:10"/> <!-- 27 -->
<list var="1:2:8:3:7:4:6:5:9:10"/> <!-- 28 -->
<list var="2:1:5:4:6:3:7:8:10:9"/> <!-- 29 -->
<list var="2:1:5:4:6:3:7:8:10:9"/> <!-- 30 -->
<list var="1:2:8:3:7:4:6:5:9:10"/> <!-- 31 -->
<list var="1:2:8:3:7:4:6:5:9:10"/> <!-- 32 -->
<list var="2:1:5:4:6:3:7:8:10:9"/> <!-- 33 -->
<list var="2:1:4:3:5:8:6:7:10:9"/> <!-- 34 -->
<list var="1:2:3:4:8:5:7:6:9:10"/> <!-- 35 -->
<list var="1:2:8:3:7:4:6:5:9:10"/> <!-- 36 -->

4. Результаты работы программы



Рассмотрим данный пример. Квадрат, выделенный красным цветом, является первым, то есть наблюдатель находится в зоне 18. Тогда список последовательности отображения будет - 10:9:5:4:6:3:7:8:2:1, что и видно на рисунке.

Л А Б О Р А Т О Р Н А Я Р А Б О Т А 4
«СИНТЕЗ ИЗОБРАЖЕНИЯ СЦЕНЫ МЕТОДОМ ТРАССИРОВКИ ЛУЧЕЙ»

Цель работы: ознакомиться с методом трассировки лучей на примере синтеза небольшой сцены.

Теория алгоритма обратной трассировки лучей

Главная идея, лежащая в основе этого метода, заключается в том, что наблюдатель видит любой объект посредством испускаемого неким источником света, который падает на этот объект и затем каким-то путем доходит до наблюдателя. Свет может достичь наблюдателя, отразившись от поверхности, преломившись или пройдя через нее.

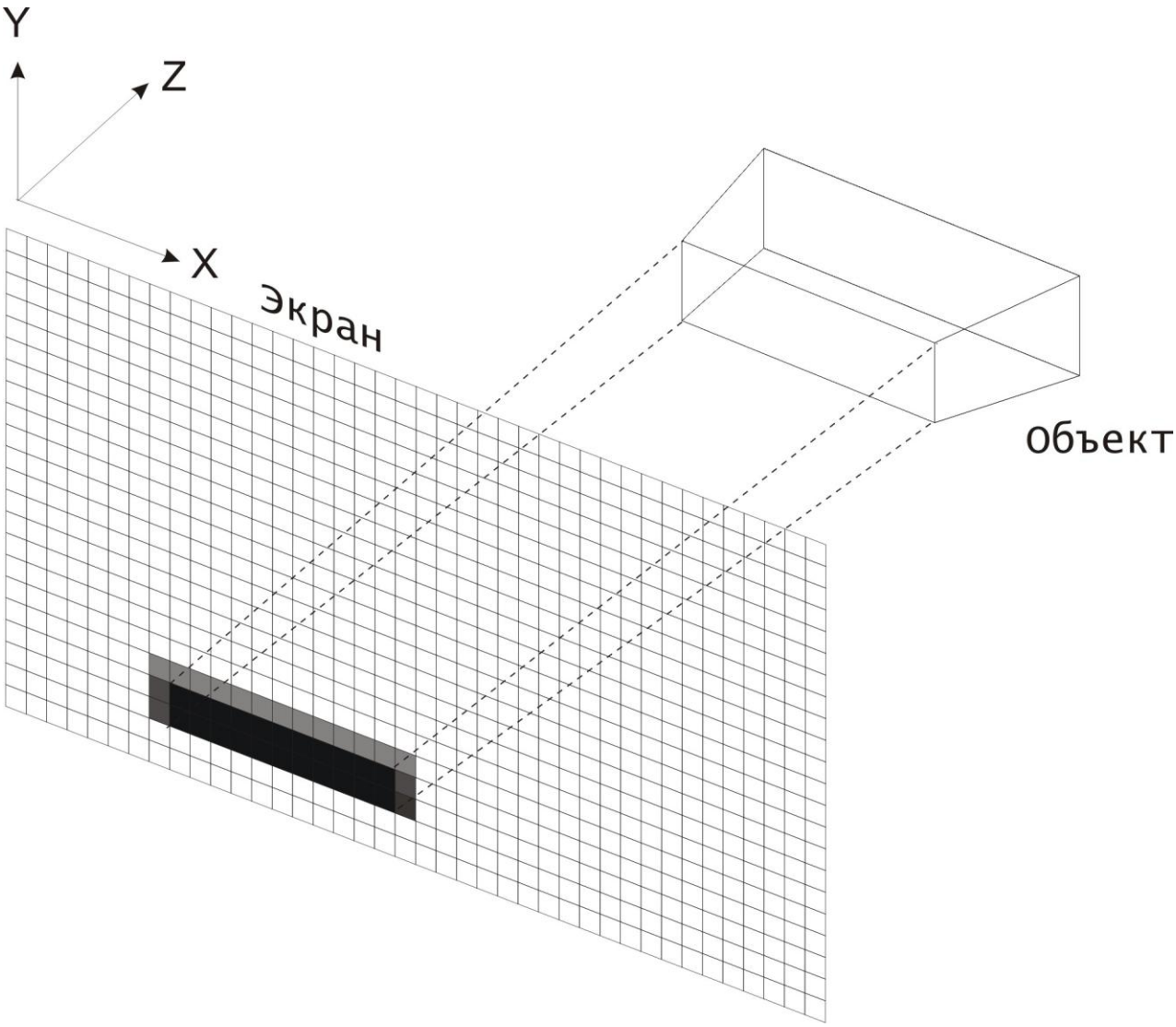


Рисунок 1.1 Трассировка луча.

Наиболее важным элементом алгоритма определения видимых поверхностей путем трассировки лучей, является процедура определения пересечений. В состав сцены можно включать любые объекты, для которых можно создать процедуру построения пересечений. Объекты сцены могут состоять из набора многоугольников (наиболее часто используются трех и четырехугольники), многогранников или тел.

Вычислительная стоимость определения пересечений произвольной пространственной прямой(луча) с одним с одним выделенным объектом может оказаться высокой. Что бы избавится от ненужного поиска пересечений, производится проверка пересечения луча с объемной оболочкой рассматриваемого объекта. И если луч не пересекает оболочки, то не нужно больше искать пересечение этого объекта с лучом. В качестве оболочки можно использовать параллелепипед или сферу.

Наиболее точно (в общем случае) объект охватывает оболочка в виде параллелепипеда, но определить попадание луча в сферу проще и это требует меньше математических вычислений. Тест со сферической оболочкой сводится к определению расстояния от точки до трехмерной прямой, т.е. луча. Будем использовать параметрическое представление прямой проходящей через точки *P1(x1,y1,z1)* и *P2(x2,y2,z2)*.

$$P(t) = P1 + (P2-P1)t$$

с компонентами:

$$x = x1 + (x2-x1)t = x1 + at$$
$$y = y1 + (y2-y1)t = y1 + bt$$
$$z = z1 + (z2-z1)t = z1 + ct$$

Тогда минимальное расстояние d от этой прямой до точки P0(x0,y0,z0) равно:

$$d2 = (x - x0)^2 + (y - y0)^2 + (z - z0)^2,$$

где параметр t, определяющий ближайшую точку P(t) равен:

$$t = - (a(x1-x0) + b(y1-y0) + c(z1-z0)) / (a^2 + b^2 + c^2)$$

Если d2 > R2, где R – радиус сферической оболочки, то луч не может пересечься с объектом.

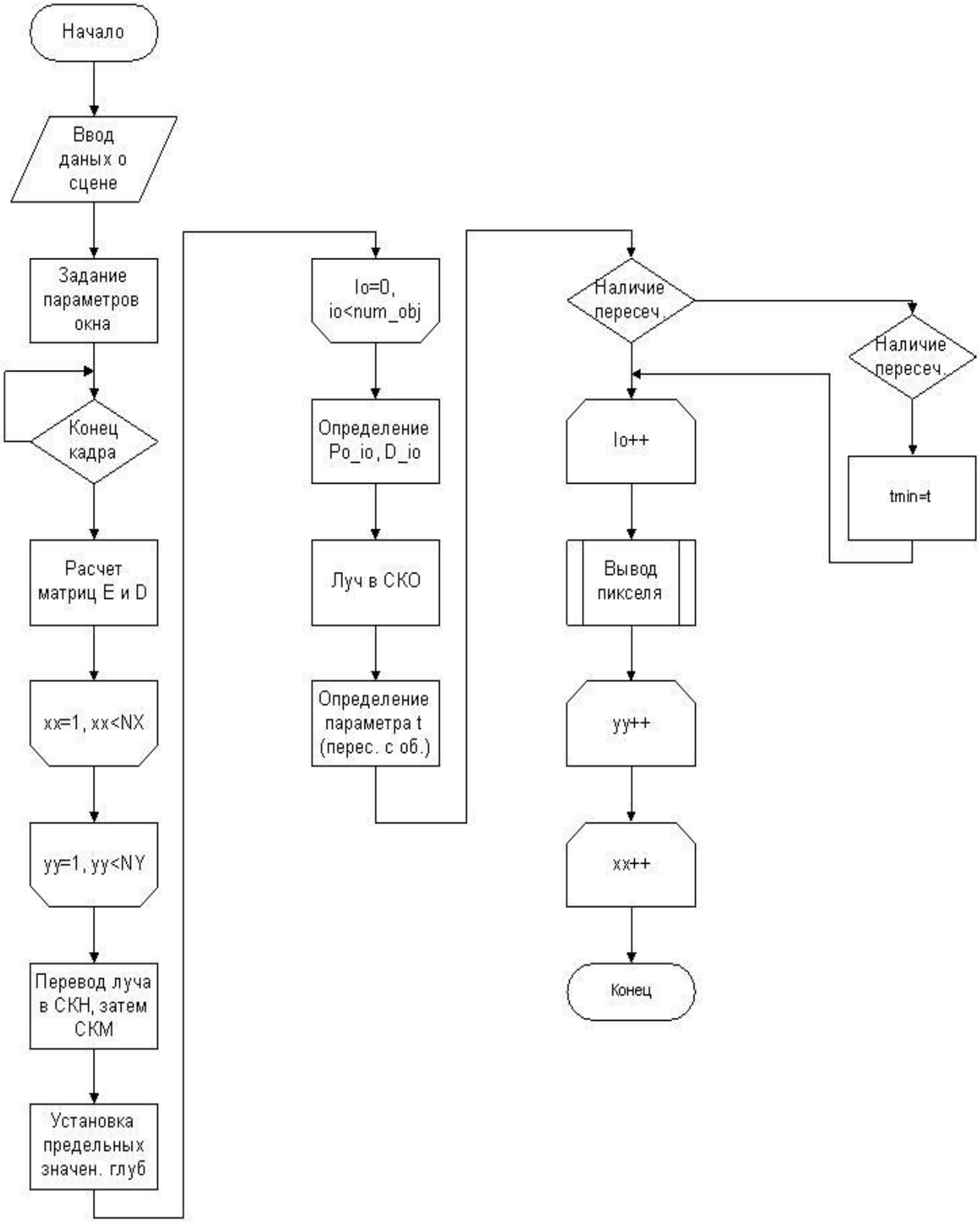


Рис. Блок – схема алгоритма

Результаты работы программы

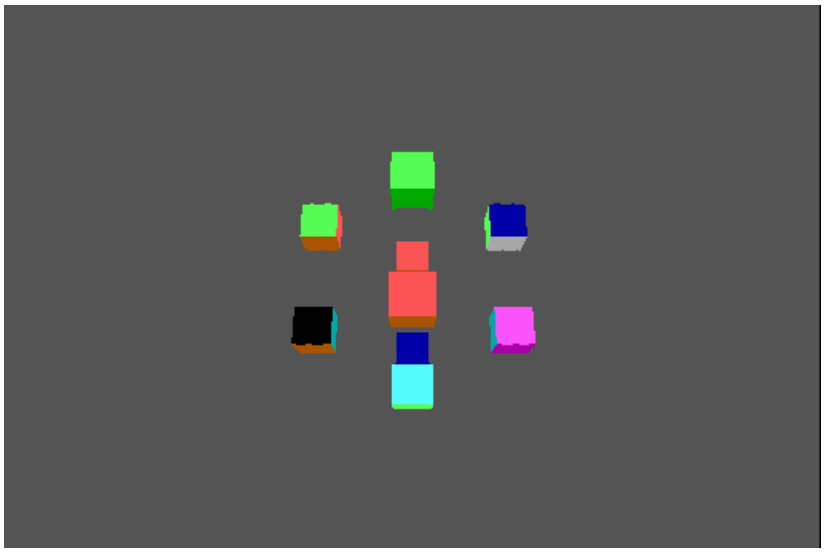


Рисунок 3.1 Пример №1 (ps=0, te=70, ga=0)

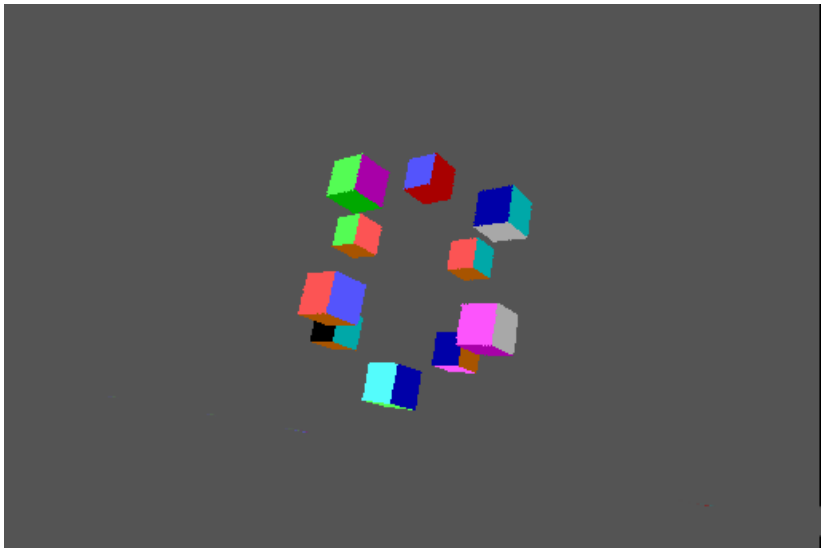


Рисунок 3.2 Пример №2 (ps=0, te=70, ga=30)

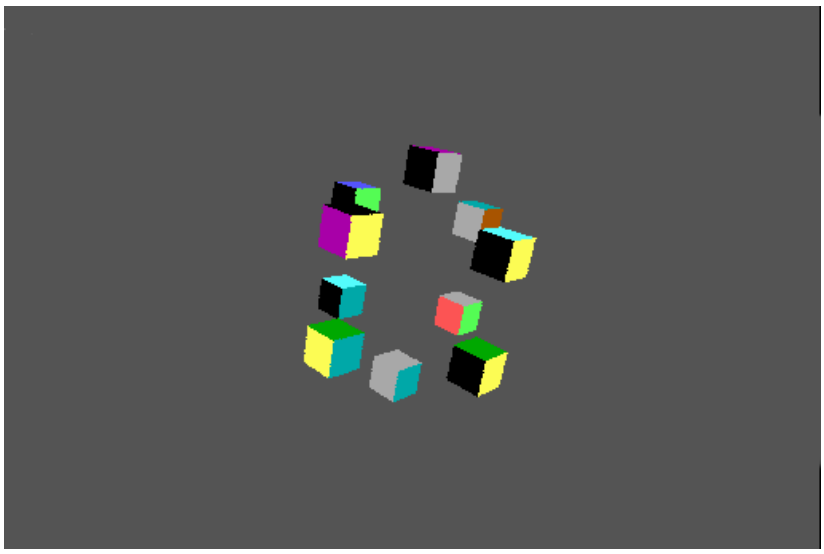


Рисунок 3.3 Пример №3 (ps=150, te=70, ga=70)

Варианты задания:

Сцена состоит из 8-ми выпуклых многогранников различного размера. Тип многогранника тот же, что и в 5-й работе.

СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

1. Ньюмен У., Спрул Р. Основы интерактивной машинной графики. Пер. с англ. /Под ред. В.А.Львова. -М.: Мир, 1976.2. Гилой В. Интерактивная машинная графика: Структуры данных, алгоритмы, языки. Пер. с англ. /Под ред. Ю.М.Баяковского, М,: Мир, 1981.

3. Роджерс Д., Адамс Дж. Математические основы машинной графики. Пер. с англ. /Под ред. Ю.И.Топчеева.- М.: Машиностроение, 1980.

4. Дж. Фоли, А. вэн Дэм. Основы интерактивной машинной графики. В 2-х кн. Кн.1. Пер. с англ. М.: Мир.- 1985.- 367с.

5. Дж. Фоли, А. вэн Дэм. Основы интерактивной машинной графики. В 2-х кн. Кн.2. Пер. с англ. М.: Мир.- 1985.- 368с.

6. Роджерс Д. - Алгоритмические основы машинной графики : Пер. с англ. - М. : Мир, 1989, 512 с.

7. Препарата Ф., Шеймос М. Вычислительная геометрия: Введение: Пер. с англ.- М.: Мир, 1989, -478 с.

8. Ж. Эгрон. Синтез изображений. Базовые алгоритмы. М.: Радио и связь. - 1993. - 216с.

9. Е.В.Шикин, А.В.Боресков, А.А.Зайцев. Начала компьютерной графики. М.: ДИАЛОГ-МИФИ. - 1993.- 138с.

10. В.П.Иванов. Трехмерная компьютерная графика. М.: Радио и связь, 1995.

11. А.Аммерал. Интерактивная трехмерная машинная графика. М.: «СолСистем». 1992.-315с.

12. Foley J., Dam A. Computer Graphics. Principles and practice. - 2nd ed. In C. - AWPC, 1997. - 1175p.

Учебное издание

Методические указания и задания
к лабораторным работам по курсу
"Аппаратные и программные средства систем компьютерной графики"
(для студентов направления 6.050102
«Компьютерная инженерия»)

Составитель: Мальчева Раиса Викторовна