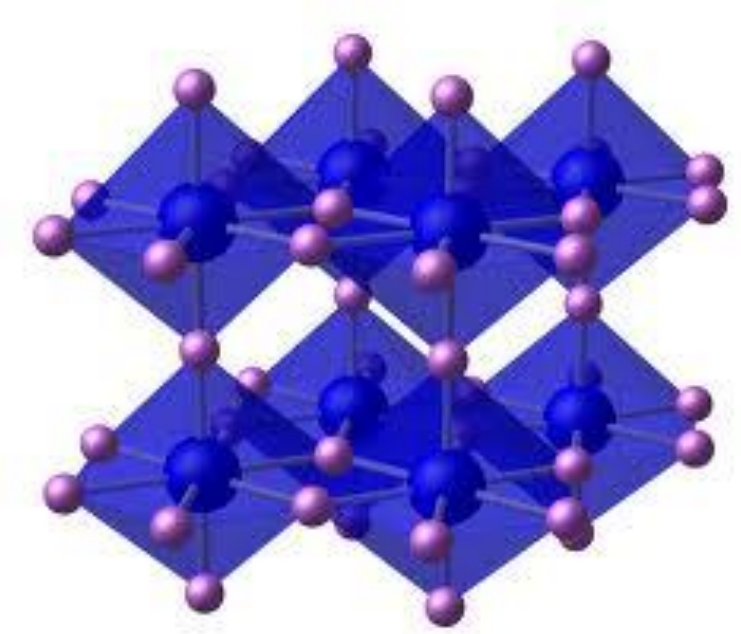
Н. Ю. Хабибулина, Д. И. Хабибулин, И.Н. Киселёв

КОМПЬЮТЕРНАЯ ГРАФИКА

Учебно-методическое пособие по выполнению лабораторных и самостоятельных работ для бакалавров технических направлений подготовки



Томск – 2024

Министерство науки и высшего образования РФ
Федеральное государственное бюджетное образовательное учреждение высшего образования

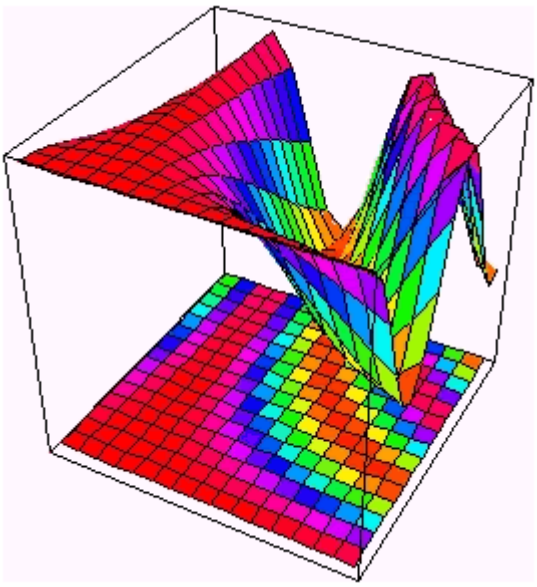
**ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)**

Кафедра компьютерных систем в управлении и проектировании (КСУП)

Н. Ю. Хабибулина, Д. И. Хабибулин, И.Н. Киселёв

КОМПЬЮТЕРНАЯ ГРАФИКА

Учебно-методическое пособие по выполнению лабораторных и самостоятельных работ для бакалавров технических направлений подготовки



Томск – 2024

Хабибулина Н. Ю., Хабибулин Д.И., Киселёв И.Н.

Компьютерная графика: учеб. методич. пособие по выполнению лабораторных и самостоятельных работ для студентов технических направлений подготовки / Н. Ю. Хабибулина, Д. И. Хабибулин, И.Н. Киселёв. – Томск : Томск. гос. ун-т систем упр. и радиоэлектроники, каф. КСУП, 2024. – 45 с.

В пособии изложены основные положения изучения дисциплины «Компьютерная графика»: цели и задачи дисциплины, содержание лекционного материала, методические рекомендации по выполнению, оформлению и защите лабораторных работ, задания к лабораторным работам, рекомендации к самостоятельной работе студентов. Предложен учебный материал для начального изучения среды программирования Visual Studio.

Пособие предназначено для бакалавров высших технических учебных заведений.

Содержание

Введение

Основной профессиональной образовательной программой обучения бакалавров технических направлений подготовки 15.03.04 «Автоматизация технологических процессов и производств», 27.03.03 «Системный анализ и управление», 27.03.04 «Управление в технических системах», 09.03.01 «Информатика и вычислительная техника» предусмотрено изучение дисциплины «Компьютерная графика». В соответствии с учебным планом содержание данной дисциплины состоит из лекционных занятий, лабораторных и самостоятельных работ. Описанию последних и посвящено настоящее учебно-методическое пособие.

В начале пособия приведена общая характеристика лекционного материала по рассматриваемой дисциплине.

Далее представлен курс лабораторных работ: приведены методические рекомендации по порядку выполнения лабораторных работ и задания к лабораторным работам, отображены требования к содержанию и оформлению отчета по работе. Кроме того, пособие содержит темы и методические указания по выполнению самостоятельной работы студентов, в том числе и требования к реферату.

Цели и задачи дисциплины

Основная цель изучения дисциплины «Компьютерная графика» – научить будущего специалиста применять методы отображения графической информации в двумерном и трехмерном пространстве, реализовывать алгоритмы компьютерной графики с помощью современных программных сред (в частности, интегрированной среды разработки Visual Studio), использовать программные и аппаратные средства компьютерной графики.

В соответствии с учебным планом данная дисциплина изучается студентами в течение одного семестра. В ходе изучения студенты должны ознакомиться с предоставленным курсом лекций, выполнить лабораторные работы, написать реферат по темам, изучаемым самостоятельно. В процессе

выполнения заданий необходимо проявить свое умение пользоваться дополнительными источниками информации и творческий подход к решению поставленной задачи.

Содержание дисциплины

Лекционный материал

1) Компьютерная графика; области применения компьютерной графики; тенденции построения современных графических систем: графическое ядро, приложения, инструментарий для написания приложений; основные функциональные возможности современных графических систем; организация диалога в графических системах; классификация и обзор современных графических систем; стандарты в области разработки графических систем;

2) Технические средства компьютерной графики: мониторы, графические адаптеры, плоттеры, принтеры, сканеры; графические процессоры, аппаратная реализация графических функций.

3) Растровая графика: алгоритмы генерации отрезка - цифровой дифференциальный анализатор (ЦДА обычный и несимметричный), алгоритм Брезенхема; алгоритм построения окружности Брезенхема; стиль линии; способы устранения ступенчатости изображения.

4) Растровая графика: четырех- и восьми связная области, гранично-определенная и внутренне-определенная область, простой и сложный контур, обход простого и сложного контура, закраска многоугольника, алгоритмы заливки области с затравкой (простой и построчный).

5) Растровая графика: алгоритмы отсечения (простой двумерный и Коэна-Сазерленда).

6) Векторная графика: системы координат; однородные координаты; 2D-преобразование (преобразование на плоскости – масштабирование, сдвиг, поворот вокруг начала координат, поворот вокруг произвольной точки): преобразование точек, преобразование прямых линий, преобразование параллельных линий, преобразование плоских фигур.

7) Векторная графика. Преобразование в трехмерном пространстве (3D-преобразование): виды геометрических моделей, их свойства, параметризация моделей; геометрические операции над моделями; матрицы преобразований (сдвиг, масштабирование, поворот вокруг оси).

8) Векторная графика: процедура перехода от мировых координат к экранным; виды проекций; матрицы перспективных и параллельных проекций.

9) Векторная графика: удаление невидимых линий и поверхностей - метод z – буфера, алгоритм Варнака, алгоритм, использующий нормаль поверхности, алгоритм Робертса.

10) Способы создания фотореалистических изображений: модели освещения, механизм диффузного и зеркального отражения света, модели закраски, прозрачность, тени, фактура, излучательность.

Лабораторные занятия

1) Лабораторная работа № 1 - Первый графический проект. Матричные операции. *Цель работы* – знакомство с технологией создания Windows Form - приложения, в частности создание WindowsForms-приложения в интегрированной среде разработки Visual Studio (далее по тексту возможно сокращенное название - VS), изучение основных элементов и принципов функционирования VS, создание первого графического и математического проекта.

2) Лабораторная работа № 2 - Алгоритмы растровой графики. Построение графических примитивов. Обход и заливка контура. Отсечение отрезков. *Цель работы* – изучение и реализация алгоритмов растровой графики: генерация отрезка алгоритмами Брезенхема и цифровым дифференциальным анализатором (обычным и несимметричным), генерация окружности алгоритмом Брезенхема, вывод линий различного стиля; обход сложного контура, заливка контура с затравкой, закраска многоугольника; алгоритмов отсечения отрезков.

3) Лабораторная работа № 3 - Преобразования на плоскости (2D-преобразования). *Цель работы* – изучение и реализация алгоритмов преобразования фигур на плоскости.

4) Лабораторная работа № 4 - Векторно-полигональная и аналитическая модель объекта. Преобразования в пространстве (3D-преобразования). Проекции. Удаление невидимых линий. *Цель работы* – изучение и реализация алгоритмов построения и преобразования трехмерного объекта с использованием векторно-полигональной модели; изучение и реализация алгоритмов построения проекции фигуры, удаления невидимых линий и граней.

Методические указания по выполнению, оформлению отчета и защите лабораторных работ приведены ниже.

Самостоятельная работа

Методические рекомендации по выполнению самостоятельной работы представлены в описании лабораторных работ в одноименных разделах.

Методические рекомендации по выполнению индивидуальной самостоятельной работы содержат темы и требования к реферату.

Методические требования по организации выполнения и защите лабораторных работ

Выполнение лабораторной работы происходит в **малых группах.**

Каждая лабораторная работа рассчитана на 2 занятия по 2 пары.

1. Разбейтесь на группы по 3-4 человека
2. Выберите «руководителя» и «технического писателя». Руководитель – координирует работу группы и собирает созданные индивидуально программные модули в единый групповой проект. Технический писатель – оформляет отчет.
3. Каждый участник группы (включая руководителя и технического писателя) выполняют задание лабораторной работы самостоятельно, т.е. создают собственные проекты и описывают полученные результаты (п.3. отчета).
4. Затем весь полученный материал передается (копируется) руководителю (созданные программные модули и сопутствующие файлы) и техническому писателю (файлы *.doc, содержащие описание результатов работы каждого участника группы). Они производят соединение всего материала в единый проект. В п.3 отчета необходимо прописать ФИО ответственного исполнителя каждой части проекта.
5. В конце второго занятия группе необходимо защитить свой проект.
6. На следующей лабораторной работе участники группы меняются ролями.

Методические указания по выполнению лабораторных работ. Требования к оформлению отчета

По задумке автора, результатом выполнения курса лабораторных работ по дисциплине «Компьютерная графика» должен быть программный продукт, объединяющий в себе выполненные задания всех лабораторных работ. Поэтому перед началом выполнения заданий необходимо создать проект, основная экранная форма которого содержит четыре (по числу лабораторных работ) кнопок. Каждая кнопка имеет название «Лабораторная работа № __», при нажатии на кнопку выполняется программный модуль соответствующей лабораторной работы. Заголовок формы должен содержать название дисциплины, фамилию, имя, отчество и группу студентов.

Для защиты лабораторной работы необходимо представить созданный программный продукт (исходный код и загрузочный файл) и отчет.

Содержание отчета по лабораторной работе:

1) титульный лист;
2) цель работы и постановка задачи;
3) анализ задачи (краткая теория по теме лабораторной работы: математические модели решения поставленной задачи, описание используемых алгоритмов);
4) описание структуры и алгоритма программы;
5) описание основных функций;
6) руководство пользователя;
7) ответы на контрольные вопросы;
8) заключение.

Образец титульного листа и оформления отчета по лабораторной работе представлены в приложении А.

Лабораторная работа № 1

Первый графический проект. Матричные операции

Цель работы – знакомство с технологией создания Windows-приложения, в частности создание приложения типа WindowsForms Application в среде Visual Studio (далее по тексту - VS), изучение основных элементов и принципов функционирования VS, а также создание первого графического и математического проекта.

Порядок выполнения и задания лабораторной работы № 1

1 Знакомство с интегрированной средой разработки Visual Studio

1.1) Первое приложение в VS

Изучите учебный материал, посвященный разработке приложений с графическим интерфейсом (WindowsForms). Для этого перейдите по одной из представленных ссылок:

1) **https://learn.microsoft.com/ru-ru/visualstudio/ide/create-csharp-winform-visual-studio?view=vs-2022**
2) https://metanit.com/sharp/windowsforms/1.1.php (разделы 1.1 и 1.2, остальные главы необходимо изучить самостоятельно).

Задание 1.1 Реализуйте первое WindowsForms-приложение – форма с кнопкой, при нажатии на которую на форме появляется надпись «Hello World!»

1.2) Добавление форм. Взаимодействие между формами

Изучите учебный материал, представленный по ссылке:

https://metanit.com/sharp/windowsforms/2.3.php

Задание 1.2 Реализуйте WindowsForms-приложение, осуществляющее вызов из главной формы второстепенной формы по описанной в примере схеме.

2 Разработка первого математического приложения

Изучите учебный материал, посвященный разработке Windows Form приложения для работы с матрицами. Для этого перейдите по ссылке:

http://www.bestprog.net/ru/2016/04/29/011-c-%D0%BF%D1%80%D0%B8%D0%BC%D0%B5%D1%80-%D1%81%D0%BE%D0%B7%D0%B4%D0%B0%D0%BD%D0%B8%D1%8F-%D0%B4%D0%B2%D1%83%D0%BC%D0%B5%D1%80%D0%BD%D0%BE%D0%B9-%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86%D1%8B/

Задание 2. Выполните пример создания на форме и умножения двумерных матриц.

3 Расширение функций математического приложения

Задание 3. Модифицируйте созданный программный модуль, реализовав в нем операции сложения, вычитания и умножения матриц (размерность матриц вводит пользователь)

Задания для самостоятельного выполнения к лабораторной работе № 1

– ***Общие требования к программному продукту и защите проекта***

1. В программе необходимо предусмотреть следующее:

– размер матрицы и вектора вводится с клавиатуры (максимальный размер – 4x4);

– элементами матрицы являются вещественные числа;

– ввод массивов с клавиатуры или заполнение их случайными числами;

– вывод исходных данных и результатов на экран;

– проверка ошибок ввода и возможности выполнения операции (например, сложение матриц выполняется для матриц одинакового размера).

2. При оформлении отчета по индивидуальной части лабораторной работы придумайте по два тестовых вопроса по изученным и реализованным алгоритмам. В каждом отчете должны быть представлены вопросы разного типа (закрытые вопросы с выбором одного правильного ответа, закрытые

вопросы с выбором нескольких правильных ответов, открытый вопрос с вводом ответа в виде числа или строки, вопрос на установление соответствия, вопрос-эссе, вопрос на установление правильной последовательности и т.д.).

– ***Задание для руководителя***

1. Создайте проект, основная экранная форма которого содержит четыре (по числу лабораторных работ) кнопок. Каждая кнопка имеет название «Лабораторная работа № __», при нажатии на кнопку выполняется программный модуль соответствующей лабораторной работы. Заголовок формы должен содержать название дисциплины. На форме разместите фамилию, имя, отчество и группу студентов, участников малой группы.

2. Для обработки события нажатия кнопки «Лабораторная работа № 1» создайте проект, реализующий выполнение задания 3 лабораторной работы.

3. Реализуйте один из нижеперечисленных вариантов заданий (номер варианта необходимо получить у преподавателя).

4. Подготовьтесь к выполнению лабораторной работы № 2. Для этого ознакомьтесь с теоретическим материалом по теме лабораторной работы № 2.

– ***Задание для технического писателя***

1. Реализуйте один из нижеперечисленных вариантов заданий (номер варианта необходимо получить у преподавателя).

2. Напишите отчет к лабораторной работе. Требования к отчету см.п. «Методические указания по выполнению и защите лабораторных работ. Требования к оформлению отчета».

3. Подготовьтесь к выполнению лабораторной работы № 2. Для этого ознакомьтесь с теоретическим материалом по теме лабораторной работы № 2.

– ***Задание для остальных участников группы***

1. Реализуйте два из нижеперечисленных вариантов заданий (номера вариантов необходимо получить у преподавателя).

2. Подготовьтесь к выполнению лабораторной работы № 2. Для этого ознакомьтесь с теоретическим материалом по теме лабораторной работы № 2.

Варианты индивидуальных заданий на самостоятельную работу к лабораторной работе № 1

1) Умножение матрицы на константу.
2) Модуль вектора.
3) Скалярное произведение векторов.
4) Векторное произведение векторов.
5) Транспонирование матрицы.
6) Умножение матрицы на вектор.
7) Умножение вектора на константу
8) Определитель матрицы*.
9) Обратная матрица*.

Контрольные вопросы к лабораторной работе № 1

1. Что такое VS?
2. Как запустить VS?
3. Что такое «кнопка быстрого доступа»?
4. Что такое «Обозреватель решений (Solution Explorer)»?
5. Что такое «Окно свойств (Properties)»?
6. Что такое «Панель инструментов (Toolbox)»?
7. Что такое «дизайнер форм»? Чем отличается дизайнер форм от формы?
8. Что такое «свойство»?
9. Что такое «событие»?
10. Как получить доступ к свойствам, расположенным на странице «События» (Events)?
11. Для чего нужно окно редактора кода?

12. Что такое «проект»? Перечислите способы создания нового проекта.
13. Как сохранить проект? Перечислите все возможные способы.
14. Как переключиться между формой и модулем?
15. Какие основные файлы входят в проект VS?
16. С какими компонентами VS Вы познакомились в данной лабораторной работе?

Лабораторная работа № 2

Алгоритмы растровой графики. Построение графических примитивов. Обход контура. Заливка замкнутой области. Отсечение отрезков

Цель работы – изучение и реализация алгоритмов растровой графики: генерация отрезка алгоритмами Брезенхема и цифровым дифференциальным анализатором (обычным и несимметричным), генерация окружности алгоритмом Брезенхема, вывод линий различного стиля; обход сложного контура, заливка контура с затравкой, закраска многоугольника; алгоритмы отсечения отрезков.

Краткая теория и порядок выполнения лабораторной работы

Для выполнения лабораторной работы ознакомьтесь с лекционным материалом по рассматриваемой теме.

Продолжим изучать приемы программирования в среде Visual Studio на примере создания проекта вывода отрезка обычным алгоритмом цифрового дифференциального анализатора (ЦДА).

1) Пример 1. *Алгоритм генерации отрезка «Обычный ЦДА»*

С помощью ЦДА решается дифференциальное уравнение отрезка, имеющее вид:

$$\frac{dY}{dX} = \frac{Py}{Px}$$

где $Py = Y_k - Y_n$ приращение координат отрезка по оси Y,

$Px = X_k - X_n$ - приращение координат отрезка по оси X,

X_n, Y_n – координаты начальной точки отрезка,

X_k, Y_k – координаты конечной точки отрезка.

В *обычном ЦДА* тем или иным образом определяется количество узлов N, используемых для аппроксимации отрезка. Затем за N циклов вычисляются координаты очередных узлов:

$$X_0 = X_n;\ X_{i+1} = X_i + \frac{P_x}{N}$$

$$Y_0 = Y_n;\ Y_{i+1} = Y_i + \frac{P_y}{N}$$

Получаемые значения X_i, Y_i преобразуются в целочисленные значения координаты очередного пикселя либо округлением, либо отбрасыванием дробной части. Данный пиксель выводится на экран.

Координаты концов отрезка будем задавать в интерактивном режиме: пользователь нажимает правую кнопку мыши и, не отпуская ее, протягивает отрезок до конечной точки, отпускает правую кнопку – на экране получаем отрезок.

1. Создайте новый проект – *Rastr_algoritm.*

2. С помощью дизайнера форм создайте следующий интерфейс основной формы программы (рис.2.1):

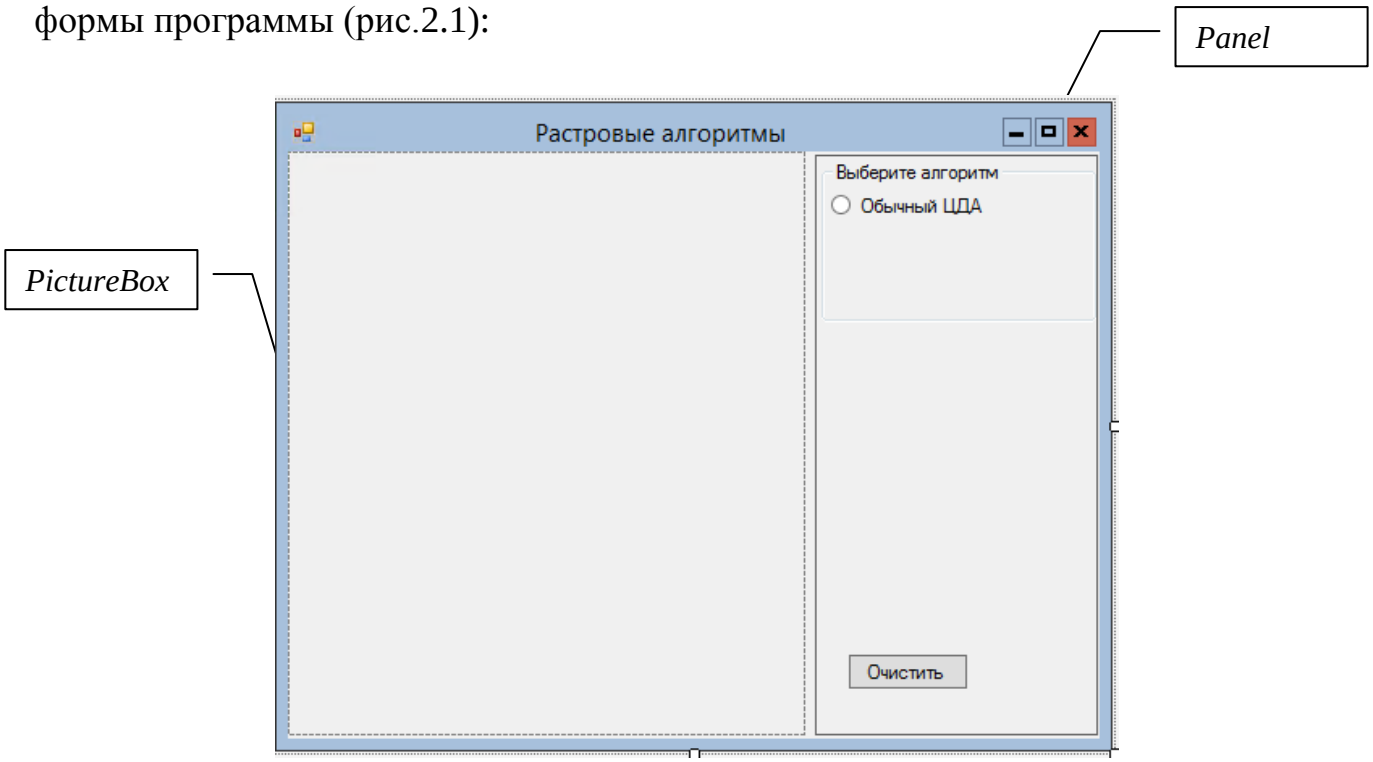


Рисунок 2.1 – Интерфейс программы *Linii*

– в свойстве Text компонента Form1 наберите «Растровые алгоритмы»;

– выберите компонент Panel (ветвь панели компонентов Контейнеры) и разместите его на правой стороне формы; установите высоту панели максимально возможную по форме; свойство BorderStyle установите в FixedSingle;

– на панели (Panel) разместите компонент GroupBox (ветвь Контейнеры); его свойство Text замените на «Выберите алгоритм»; в нем разместите один компонент RadioButton. Его свойство Text замените на «Обычный ЦДА»;

– добавьте на панель кнопку (Button) с названием «Очистить»;

– выберите компонент PictureBox (в ветви Стандартные элементы управления) и разместите его на большую часть формы.

Рекомендация: *для быстрого поиска элемента на панели элементов можно воспользоваться строкой поиска данной панели - достаточно начать набирать название искомого элемента в строке.*

3. Напишем программный код попиксельного вывода отрезка обычным алгоритмом ЦДА. Координаты начальной точки отрезка фиксируются при нажатии пользователем правой кнопки мыши на компоненте PictureBox. Для этого на странице Events (События) инспектора объектов сделайте двойной щелчок напротив события MouseDown. Перейдите в редактор кода и наберите следующие строки:

```
private void pictureBox1_MouseDown(object sender, MouseEventArgs e)
        {
            if (radioButton1.Checked == true)
            {
                xn = e.X;
                yn = e.Y;
            }

       else MessageBox.Show ("Вы не выбрали алгоритм вывода фигуры!");

 }
```

Пояснения:

– если в GroupBox не выбран ни один переключатель, то на экран выводится сообщение «Вы не выбрали алгоритм вывода фигуры!».

– Для проверки выбора RadioButton1 используется свойство Checked, принимающее значение True, если выбран данный переключатель, и False – в противном случае.
– Параметры e.X, e.Y – это координаты пикселя, в который попал курсор при нажатии правой кнопки мыши (по умолчанию метод - обработчик события нажатия правой кнопки мыши – возвращает структуру *e* с двумя полями X и Y - координаты пикселя). В нашем случае это координаты начальной точки отрезка.
– Кроме этого, не забудьте описать глобальные переменные xn, yn. Для этого в программный код вставьте строчку с описанием переменных в начале описания класса Form1

```
public partial class Form1 : Form
    {
        public int xn, yn, xk, yk;

        public Form1()
        {
            InitializeComponent();
        }
```

4. При отпускании правой кнопки мыши фиксируем конечную точку отрезка, рассчитываем в цикле координаты очередного пикселя и в результате выполнения цикла выводим отрезок на экран красным цветом. Для фиксирования отпускания правой кнопки мыши используется событие MouseUp компонента PictureBox. На странице Events (События) инспектора объектов сделайте двойной щелчок напротив данного события и в редакторе кода напишите:

```
private void pictureBox1_MouseUp(object sender, MouseEventArgs e)
    {
        //numberNodes – переменная, задающая количество узлов для
        //аппроксимации отрезка
        //xOutput – x-координата очередного узла
        //yOutput – y-координата очередного узла
        int index, numberNodes;
        double xOutput, yOutput, dx, dy;

        //Объявляем объект "myPen", задающий цвет и толщину пера
        Pen myPen = new Pen(Color.Red, 1);
```

```
        //Объявляем объект "g" класса Graphics и предоставляем
        //ему возможность рисования на pictureBox1:
        Graphics g = Graphics.FromHwnd(pictureBox1.Handle);

        //реализация обычного алгоритма ЦДА
        xk = e.X;
        yk = e.Y;
        dx = xk - xn;
        dy = yk - yn;
        numberNodes = 200;
        xOutput = xn;
        yOutput = yn;
        for (index = 1; index <= numberNodes; index++)
        {
            //Рисуем прямоугольник
            g.DrawRectangle(myPen, (int)xOutput, (int)yOutput, 2, 2);

            //Рисуем закрашенный прямоугольник:
            //Объявляем объект "redBrush", задающий цвет кисти
            // SolidBrush redBrush = new SolidBrush(Color.Red);

            //Рисуем закрашенный прямоугольник
            // g.FillRectangle(redBrush, (int)xOutput, (int)yOutput, 2, 2);

            xOutput = xOutput + dx / numberNodes;
            yOutput = yOutput + dy / numberNodes;
        }

    }
```

Пояснения:

- для рисования используем объект Pen (карандаш) с параметрами – цвет – красный и толщина – 1) и создаем объект g класса Graphics, который имеет много встроенных методов для вывода фигур;
- реализуем обычный алгоритм ЦДА с numberNodes = 200;
- в данном примере вывод отрезка в цикле можно производить двумя способами: первый (не закомментирован) - отрезок рисуется прямоугольниками размером 2X2 пикселя; второй – (закомментировано в примере) – закрашенными прямоугольниками. Попробуйте рисование обоими способами – опишите разницу преподавателю;
- `(int)`xOutput - преобразование переменно *xOutput* типа *double* в переменную типа *int*.

– ***Не забудьте описать глобальные целочисленные переменные xk, yk, xn, yn.***

5. Сохраните и запустите программу. Нарисуйте несколько линий. Экранная форма программы может выглядеть так (рис.2.2):

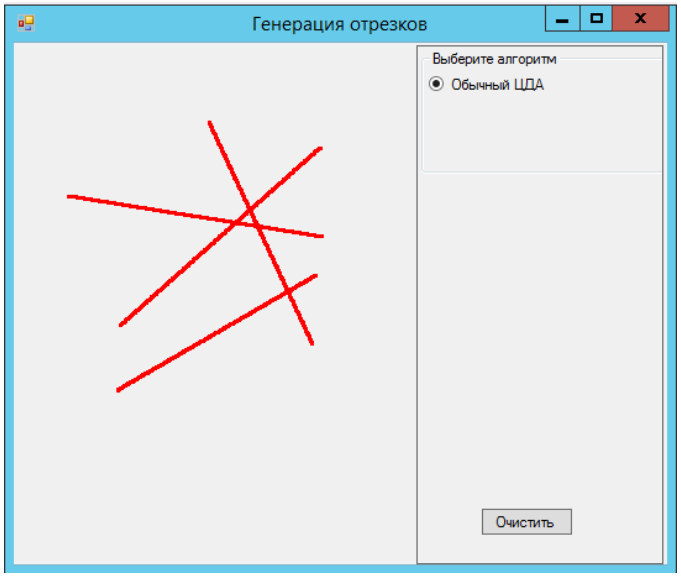


Рисунок 2.2 – Результаты работы программы

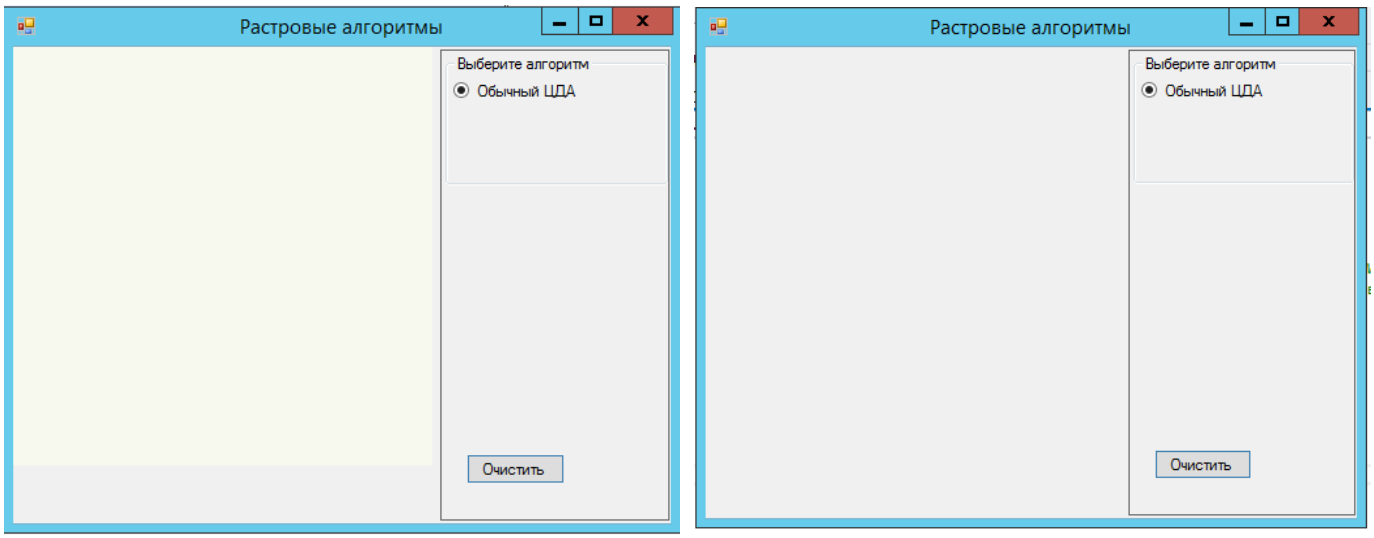
6. Пропишем процедуру очистки поля рисования. Дважды щелкните на кнопке «Очистить» и перейдите в редактор кода. Если набрать следующие строки:

```
private void button1_Click(object sender, EventArgs e)
        {
            Bitmap myBitmap = new Bitmap(pictureBox1.Height, pictureBox1.Width);

            //Задаем цвет пикселя по схеме RGB (от 0 до 255 для каждого цвета)
            Color newPixelColor = Color.FromArgb(247, 249, 239);

            for (int x = 0; x < myBitmap.Width; x++)
            {
                for (int y = 0; y < myBitmap.Height; y++)
                {
                    myBitmap.SetPixel(x, y, newPixelColor);
                }
            }

            pictureBox1.Image = myBitmap;

          //  pictureBox1.Image = null;
        }
```

то все пиксели компонента PictureBox1 перекрасятся в светлый цвет (рис.2.3, а). Если же прописать:

```
private void button1_Click(object sender, EventArgs e)
        {
          pictureBox1.Image = null;
        }
```

то использовали метод обновления компонента PictureBox, результат будет аналогичным рис. 2.3, б.



а) б)

Рисунок 2.3

2) *Пример 2.* *Рекурсивный алгоритм заливки замкнутой 4-х связной гранично-определенной области с затравкой*

Рекурсивный алгоритм заливки 4-х связной гранично-определенной области:

– определяется, является ли пиксель граничным или уже закрашенным,
– если нет, то пиксель перекрашивается (выполняется функция ЗАЛИВКА), затем проверяются и если надо перекрашиваются 4 соседних пикселя.

Функцию ЗАЛИВКА определим следующим образом.

Функция ЗАЛИВКА (x, y)

```
{
Если цвет пикселя (x, y) не равен цвету границы и цвету заполнения, то
{   Установить для пикселя (x, y) цвет заполнения;
        ЗАЛИВКА (x+ 1, y);
        ЗАЛИВКА (x-1, y);
        ЗАЛИВКА (x, y+1);
        ЗАЛИВКА (x, y-1);
        }
}
```

В качестве закрашиваемой области возьмем фигурку, полученную пересечением отрезков (координаты концов отрезка вводятся в программе). Отрезки будут выводиться красным цветом, а область будет заливаться синим цветом.

1. Модифицируйте созданный ранее проект *Rastr_algoritm*. С помощью дизайнера форм создайте следующий интерфейс основной формы программы (рис.2.4):

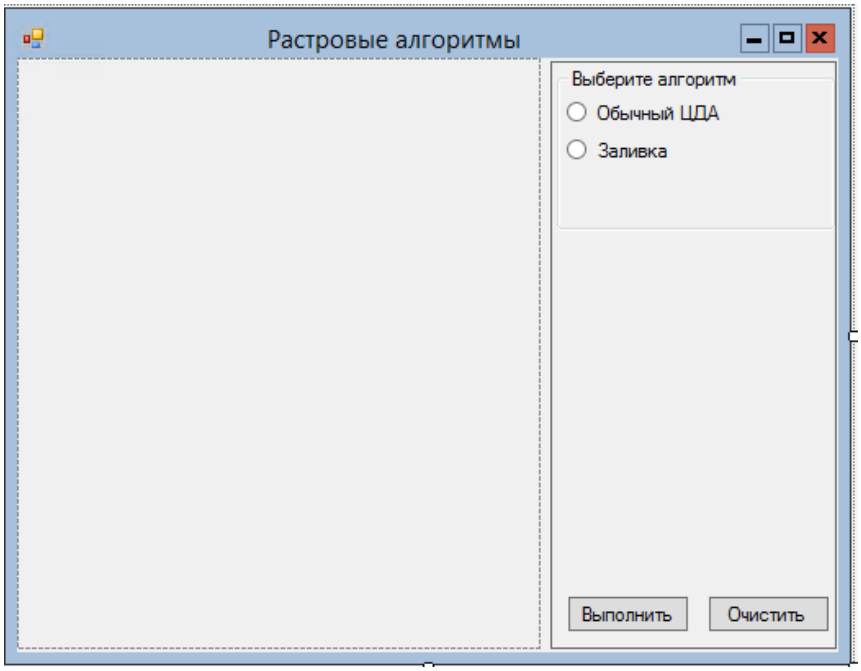


Рисунок 2.4

– добавьте в компоненте GroupBox компонент RadioButton, назначив соответственно свойству Text значение «Заливка»;

– добавьте кнопку (Button) с названием «Выполнить».

2. Для вывода отрезков используем программный код метода вывода отрезка обычным алгоритмом ЦДА, созданный в примере. Оформим его в виде отдельной функции *CDA* с входными параметрами: *xStart, yStart, xEnd, yEnd* – соответственно координаты начального и конечного концов отрезка.

```
// вывод отрезка
        private void CDA(int xStart, int yStart, int xEnd, int yEnd)
        {
            int index, numberNodes;
            double xOutput, yOutput, dx, dy;

            xn = xStart;
            yn = yStart;
            xk = xEnd;
            yk = yEnd;

            dx = xk - xn;
            dy = yk - yn;
            numberNodes = 200;
            xOutput = xn;
            yOutput = yn;

            for (index = 1; index <= numberNodes; index++)
            {
                myBitmap.SetPixel((int)xOutput, (int)yOutput,
currentBorderColor);
                xOutput = xOutput + dx / numberNodes;
                yOutput = yOutput + dy / numberNodes;
            }
          }
```

Пояснения:

- не забудьте объявить глобальные переменные

```
public partial class Form1 : Form
    {
        public int xn, yn, xk, yk; // концы отрезка
        Bitmap myBitmap; // объект Bitmap для вывода отрезка
        Color currentBorderColor; // текущий цвет отрезка и
   текущий цвет заливки
```

- переменная *currentBorderColor* будет определять цвет линии. Ее значение будем получать из диалогового окна выбора цвета.

3. Создадим метод-обработчик выбора цвета линии. Добавьте еще одну кнопку на форму, назовите ее «Цвет линии». Добавьте элемент ColorDialog (рис.2.5)

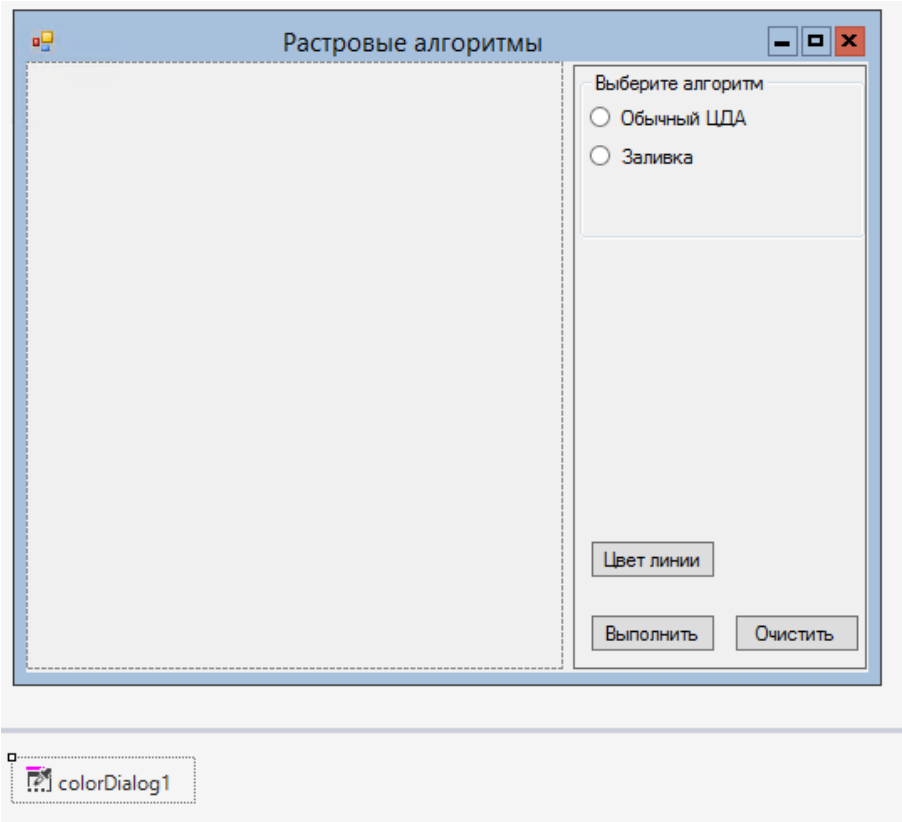


Рисунок 2.5

Перейдите в код обработчика нажатия кнопки «Цвет линии» (например, дважды кликните не кнопке «Цвет линии»). Введите программный код в метод-обработчик нажатия на кнопку. Должен получиться следующий код:

```
private void button2_Click(object sender, EventArgs e)
    {
        DialogResult dialogResult = colorDialog1.ShowDialog();
        if (dialogResult == DialogResult.OK && radioButton1.Checked)
        {
            currentBorderColor = colorDialog1.Color;
        }
    }
```

4. Создадим обработчик нажатия на кнопку «Выполнить». При нажатии на данную кнопку в случае выбора «Обычный ЦДА» на экране появляется прямоугольник и треугольник, нарисованные выбранным цветом. Для их вывода на экран вызовем несколько раз функцию CDA с разными

координатами концов отрезка. Если же на форме выбрано «Заливка» - вызываем метод заливки с координатами затравки (160,40) – точка лежит внутри треугольника:

```
private void button1_Click(object sender, EventArgs e)
{
    //отключаем кнопки
    button1.Enabled = false;
    button2.Enabled = false;

    //создаем новый экземпляр Bitmap размером с элемент
    //pictureBox1 (myBitmap)
    //на нем выводим попиксельно отрезок

    myBitmap = new Bitmap(pictureBox1.Width, pictureBox1.Height);
    using (Graphics g = Graphics.FromHwnd(pictureBox1.Handle))
      {
        if (radioButton1.Checked == true)
        {
            //рисуем прямоугольник
            CDA(10, 10, 10, 110);
            CDA(10, 10, 110, 10);
            CDA(10, 110, 110, 110);
            CDA(110, 10, 110, 110);

            //рисуем треугольник
            CDA(150, 10, 150, 200);
            CDA(250, 50, 150, 200);
            CDA(150, 10, 250, 150);
        }
        else
        {
            if (radioButton2.Checked == true)
            {
                //получаем растр созданного рисунка в mybitmap
                myBitmap = pictureBox1.Image as Bitmap;

                // задаем координаты затравки
                xn = 160;
                yn = 40;

          // вызываем рекурсивную процедуру заливки с затравкой
                     FloodFill(xn, yn);
                }
        }
        //передаем полученный растр mybitmap в элемент pictureBox
        pictureBox1.Image = myBitmap;
```

```
                // обновляем pictureBox и активируем кнопки
                pictureBox1.Refresh();
                button1.Enabled = true;
                button2.Enabled = true;
            }
        }
```

Пояснения: для вывода рисунка на элемент PictureBox создаем растр (Bitmap), на нем можно работать отдельно с каждым пикселем.

5. Создадим рекурсивную процедуру заливки (заливаем зеленым цветом):

```
        // Заливка с затравкой (рекурсивная)
        private void FloodFill(int x1, int y1)
        {
            // получаем цвет текущего пикселя с координатами x1, y1
            Color oldPixelColor = myBitmap.GetPixel(x1, y1);

           // сравнение цветов происходит в формате RGB
           // для этого используем метод ToArgb объекта Color

            if ((oldPixelColor.ToArgb() != currentBorderColor.ToArgb())
&& (oldPixelColor.ToArgb() != Color.Green.ToArgb()))
            {
               //перекрашиваем пиксель
                myBitmap.SetPixel(x1, y1, Color.Green);

                //вызываем метод для 4-х соседних пикселей
                FloodFill(x1 + 1, y1);
                FloodFill(x1 - 1, y1);
                FloodFill(x1, y1 + 1);
                FloodFill(x1, y1 - 1);

            }
            else
            {
                //выходим из метода
                 return;
            }

         }
```

6. Сохраните и запустите программу. Выберите «Обычный ЦДА», нажмите кнопку «Цвет линии» и выберите цвет линии. Нажмите кнопку «Выполнить». Экранная форма программы может выглядеть так (рис.2.6).

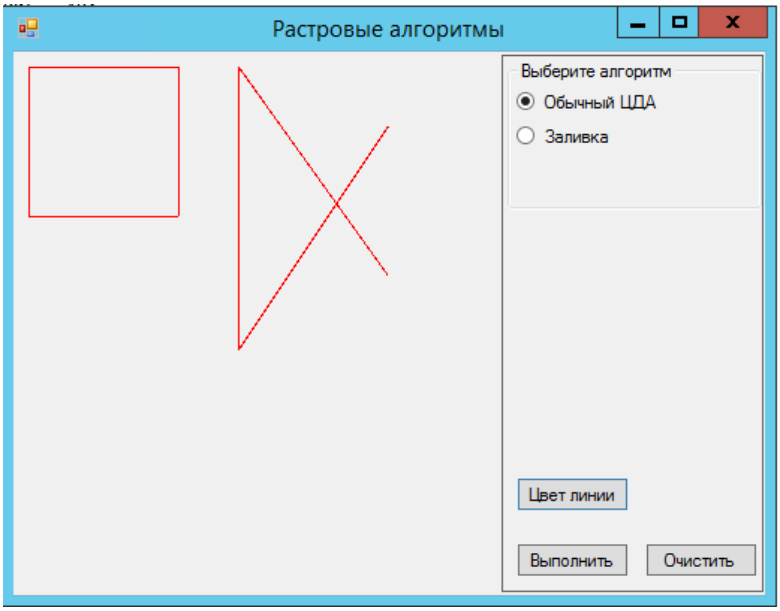


Рисунок 2.6 – Результат работы алгоритма вывода отрезка

7. Выберите «Заливка» и нажмите кнопку «Выполнить». Экранная форма программы может выглядеть так (рис.2.7).

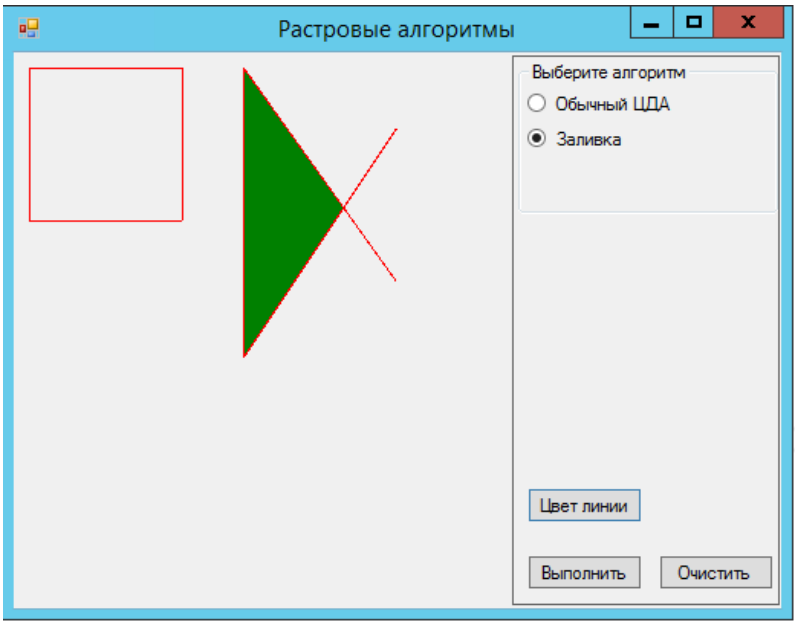


Рисунок 2.7 – Результат работы алгоритма заливки с затравкой

Задания к лабораторной работе № 2

– ***Задание 1***

Реализуйте все примеры из лабораторной работы 2.

– ***Задание 2***

Добавьте в созданный проект возможность вывода толстой линии. В качестве пера используйте квадрат, полученный закрашиванием 8 соседних пикселей.

– ***Задание 3***

Добавьте в проект возможность выбора цвета заливки.

– ***Задание 4***

Модифицируйте программный модуль так, чтобы можно было заливать замкнутый контур, построенный и выбранный интерактивно (т.е. рисуем отрезки интерактивно на форме, получаем с помощью их пересечений некоторый замкнутый контур, потом кликаем внутрь данного контура, внутренность контура заливается).

Задания для самостоятельного выполнения к лабораторной работе № 2

– ***Общие требования к программному продукту и защите проекта***

1. Для ввода исходных данных (концов отрезка, центра и радиуса окружности) можно использовать и интерактивный ввод (приведенный в примере 1), и диалоговый ввод через поля ввода (компоненты TextBox).

2. При реализации алгоритма обхода контура: программный модуль должен позволять визуализировать контур, состоящий из отрезков, окружностей, многоугольников (алгоритм построения отдельной фигуры может быть реализован другими участниками группы). Затем контур должен медленно (по мере его обхода) закрашиваться другим цветом. Предусмотрите настройку цвета обхода контура и заливки.

3. При реализации алгоритма отсечения: на входе программный модуль получает координаты отсекающего окна и концов отрезков (предусмотрите интерактивное добавление пользователем отрезков). Затем выводятся отсекающее окно и отрезки на экран. После нажатия соответствующей кнопки, отсеченные части отрезков, расположенные вне окна, должны быть закрашены другим цветом.

4. При оформлении отчета по индивидуальной части лабораторной

работы придумайте по два тестовых вопроса по изученным и реализованным алгоритмам. В каждом отчете должны быть представлены вопросы разного типа (закрытые вопросы с выбором одного правильного ответа, закрытые вопросы с выбором нескольких правильных ответов, открытый вопрос с вводом ответа в виде числа или строки, вопрос на установление соответствия, вопрос-эссе, вопрос на установление правильной последовательности и т.д.).

– ***Задание для руководителя***

1. Откройте проект, созданный в лабораторной работе №1. Для кнопки «Лабораторная работа №2» создайте обработчик (модуль), демонстрирующий все задания участников группы.

2. Реализуйте настройку цвета и стиля линии – толстая (предусмотрите выбор толщины и вида пера), тонкая, сплошная, пунктирная линия (предусмотрите настройку шага пунктира). *Примечание*: для ввода исходных данных (концов отрезка, толщины линии, длины штриха линии) можно использовать диалоговый ввод через компоненты TextBox.

3. Реализуйте индивидуальное задание (номер варианта необходимо получить у преподавателя).

4. Подготовьтесь к выполнению лабораторной работы № 3. Для этого ознакомьтесь с теоретическим материалом по теме лабораторной работы № 3.

– ***Задание для технического писателя***

1. Реализуйте индивидуальное задание (номер варианта необходимо получить у преподавателя).

2. Напишите отчет к лабораторной работе. Требования к отчету см.п. «Методические указания по выполнению и защите лабораторных работ. Требования к оформлению отчета» (не забудьте про теоретическое описание реализованных алгоритмов)

3. В отчете нарисуйте детальную блок-схему алгоритма обхода сложного замкнутого контура.

4. Подготовьтесь к выполнению лабораторной работы № 3. Для этого ознакомьтесь с теоретическим материалом по теме лабораторной работы № 3.

– ***Задание для остальных участников группы***

1. Реализуйте индивидуальное задание (номера вариантов необходимо получить у преподавателя).

2. Подготовьтесь к выполнению лабораторной работы № 3. Для этого ознакомьтесь с теоретическим материалом по теме лабораторной работы № 3.

Варианты индивидуальных заданий на самостоятельную работу к лабораторной работе № 2

Вариант 1

Реализуйте генерацию окружности по алгоритму Брезенхема.

Вариант 2

Реализуйте простой итеративный алгоритм заливки с затравкой.

Вариант 3

Реализуйте генерацию отрезка по алгоритму Брезенхема.

Вариант 4

Реализуйте построчный алгоритм заливки с затравкой.

Вариант 5

Реализуйте генерацию отрезка по несимметричному алгоритму ЦДА.

Вариант 6

Реализуйте алгоритм закраски многоугольника.

Вариант 7

Реализуйте алгоритм обхода сложного контура.

Вариант 8

Реализуйте двумерный алгоритм отсечения Коэна-Сазерленда.

Вариант 9

Реализуйте простой алгоритм двумерного отсечения.

Контрольные вопросы к лабораторной работе № 2

1. Чем отличаются обычный и несимметричный алгоритмы ЦДА?
2. Кратко опишите основную идею алгоритма Брезенхема генерации отрезка.
3. Кратко опишите основную идею алгоритма Брезенхема генерации окружности.
4. Что такое «растр»?
5. Что такое «растровая компьютерная графика»?
6. Что такое «пиксель»?
7. Как получить отображение толстой линии?
8. Как получить отображение пунктирной линии?
9. Что означают термины «гранично-определенная область» и «внутренне-определенная область»?
10. Объясните понятия «4-х связная» и «8-ми связная» область.
11. Чем отличаются простой и сложный контуры?
12. Чем отличаются алгоритмы обхода простого и сложного контура.
13. Что такое «затравка»?
14. Чем отличаются простой алгоритм заливки с затравкой и построчный алгоритм заливки с затравкой.
15. Чем отличаются алгоритмы заливки с затравкой и алгоритм закраски многоугольника.
16. Что означают термины «полностью видимые», «полностью невидимые», «подозрительные» отрезки?
17. Кратко опишите основную идею отсечения отрезков алгоритмом Коэна-Сазерленда.
18. В чем основная идея простого алгоритма двумерного отсечения.

Методические рекомендации по выполнению индивидуальной самостоятельной работы

Для более глубокого и самостоятельного изучения студентам предлагаются следующие темы для реферата:

1) История компьютерной графики, основные даты и события (например, 50-е годы: от текстовых изображений к графической консоли; 60-е годы: от "Альбома" к анимации; 70-е годы: эпоха алгоритмов; 80-е годы: компьютерная графика в кино; 90-е годы; время стандартов, Интернета и компьютерных игр; 21 век, перспективы компьютерной графики; или предложить свои этапы развития компьютерной графики).
2) Выдающиеся личности в компьютерной графике (П. Безье, А. Сазерленд, Стив Рассел, Джон Уорнок, Джим Кларк, Генри Гуро, Мартин Ньюелл, Ву Тонг Фонг, Бенуа Мандельброт, Джеймс Блинн, Эд Катмалл, Лорен Карпентер, Алвай Рей Смит, и др.).
3) Области применения компьютерной графики (научная графика, деловая графика. конструкторская графика, иллюстративная графика, художественная и рекламная графика, компьютерная анимация, мультимедиа).
4) Стандарты и языки компьютерной графики (CGI, IGES, Direct3D, DirectX, VRML,OpenGL, ActionScrip)
5) Классификация компьютерной графики (двухмерная, трехмерная, растровая, векторная, фрактальная, воксельная и т.д.)
6) Форматы хранения графической информации.
7) Представление цветов в компьютере.
8) Технические средства компьютерной графики: мониторы.
9) Технические средства компьютерной графики: графические адаптеры.
10) Технические средства компьютерной графики: плоттеры.
11) Технические средства компьютерной графики: принтеры.
12) Технические средства компьютерной графики: сканеры.

13) Технические средства компьютерной графики: световое перо, джойстик, трекбол, тачпад, трекпойнт, дигитайзеры.
14) Технические средства компьютерной графики: графические планшеты.
15) Графические процессоры.
16) Программные средства компьютерной графики.
17) Виртуальная и дополненная реальность.
18) Графика и игры. Технологии, применяющиеся в «игрострое». История и современность.
19) Графические технологии будущего.
20) Цифровое искусство.
21) ГИС как разновидность интерактивной компьютерной графики

В результате самостоятельного изучения каждый студент должен написать реферат на одну из предложенных тем. Студент может предложить собственную тему реферата, соответствующую дисциплине «Компьютерная графика».

Требования к структуре, оформлению и защите реферата

Структура реферата

- титульный лист (приложение Б)
- введение
- основная часть
- заключение
- список использованных источников
- приложения.

Введение должно содержать краткое описание рассматриваемой темы, цель работы и рассматриваемые вопросы. Объем введения – 1-2 стр. Основная часть может быть разбита на несколько пунктов. Заключение содержит основные выводы по проделанной работе (объем – 1 стр.). Список использованных источников содержит перечень учебников, журналов, электронных источников и источников Интернет, используемых при написании

реферата. Необходимо использовать информацию не старше 5 лет. На каждый источник по тексту реферата обязательно должны быт ссылки.

Стиль написания реферата - строгий технический. Поэтому по тексту нельзя использовать рекламные слоганы, слэнги и личные местоимения. Повествование должно вестись от третьего неопределенного лица (например, «рассмотрены...», «предложены...», «предложена классификация...» и т.п.).

Оформление реферата

Оформление реферата должно быть выполнено в соответствии с «Образовательный стандарт вуза ОС ТУСУР 01-2021. Работы студенческие по направлениям подготовки и специальностям технического профиля. Общие требования и правила оформления» (https://regulations.tusur.ru/documents/70). Наиболее важные требования:

- формат листа – А4;
- поля: верхнее, нижнее – 2 см; правое – 1,5 см; левое – 3см;
- межстрочный интервал – полуторный;
- красная строка – 1,25 см;
- шрифт – Times New Roman 12-14 пт;
- все рисунки и таблицы должны иметь номер и название;
- в конце заголовков точка не ставиться;
- нумерация страниц – по центру внизу страницы (первая страница – титульный лист, на нем номер не проставляется).

Объем реферата – 20-25 стр.

Реферат не может быть засчитан при наличии хотя бы одного из ниже перечисленных недостатков:

- если полностью или в значительной части работа выполнена несамостоятельно, т.е. путем механического переписывания учебников, специальной или другой литературы, копирования источников Интернет;

– если выявлены существенные ошибки, свидетельствующие о том, что содержание тем не раскрыто;

– если работа отличается узконаправленным замкнутым подходом к решаемым проблемам без применения комплексного анализа, позволяющего студенту проявить широкий объем знаний;

– если работа написана небрежно, неразборчиво, с несоблюдением правил оформления.

Защита реферата

До 01 мая необходимо сдать реферат на проверку преподавателю.

На последнем лекционном занятии проводится защита реферата в форме лекции-презентации (точное время проведения назначает преподаватель заранее - чаще всего защита проводится за неделю до окончания семестра). На данном занятии студент выступает с докладом продолжительностью 5-7 минут. Доклад должен содержать краткое описание рассмотренной в реферате теме. Рекомендуется в процессе доклада использовать демонстрационный материал в виде мультимедийной презентации (возможно включение роликов). После доклада предоставляется возможность присутствующим задать докладчику вопросы. Результатом защиты являются рейтинговые баллы.

Приложение А

Образец титульного листа и отчета по лабораторной работе

Министерство науки и высшего образования РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Томский государственный университет систем управления и радиоэлектроники
(ТУСУР)»

Кафедра компьютерных систем в управлении и проектировании (КСУП)

ПРЕОБРАЗОВАНИЯ В ПРОСТРАНСТВЕ. ПРОЕКЦИИ. УДАЛЕНИЕ НЕВИДИМЫХ ЛИНИЙ

Отчет по лабораторной работе № 4
по дисциплине «Компьютерная графика»

Вариант –

Студенты гр.581
_____ И. И. Иванов
_____ И. И. Петров

«___» _____ 20__ г.

Проверил
канд. техн. наук,
доцент каф.КСУП
_____ Н. Ю. Хабибулина
«___» _____ 20__ г.

20__ г.

1 Цель работы и постановка задачи

Цель работы – изучение и реализация алгоритмов построения проекции фигуры, удаления невидимых линий и граней.

2 Задания к лабораторной работе № 4

1. Откройте проект, созданный в предыдущих лабораторных работах. Для кнопки «Лабораторная работа №4» создайте соответствующий обработчик (модуль), удовлетворяющий нижеследующим требованиям.

2. Напишите программный модуль для отображения преобразований многогранника в пространстве.

Первым действием отобразите многогранник на экране – высота многогранника параллельна вертикальной оси.

Далее реализуйте:

- построение оси вращения (при этом координаты точки, через которую проходит ось вращения, задаются пользователем интерактивно);
- поворот многогранника так, чтобы его высота совпала с заданной осью вращения;
- вращение относительно высоты многогранника, совпадающей с заданной осью вращения;
- перемещение многогранника (по каждой координатной оси, по двум или трем осям одновременно);
- изменение размера многогранника (по каждой оси, по двум или трем координатным осям одновременно);
- настройку цвета и толщины линий;
- изменение направления и скорости вращения / перемещения.

3. Нарисуйте систему координат и ось вращения.

4. Координаты точек – мировые.

Максимальное и минимальное значение мировых координат изобразите на экране. Осуществите автоматический подбор размеров изображения.

5. При выполнении этого задания при удалении невидимых линий программа должна выдавать контурное изображение (невидимые линии рисовать штрих-пунктиром).

6. Отображаемую фигуру, ось вращения, вид проекции и алгоритм удаления невидимых линий выберите в соответствии с вариантом (таблица 3):

№	Фигура	Проекция	Ось вращения	Алгоритм удаления невидимых линий
	Тетраэдр	Изометрия	Прямая, заданная произвольными направляющими косинусами	алгоритм, использующий нормаль грани

3 Анализ задачи

Для выполнения поставленной задачи необходимо реализовать следующие алгоритмы:

1) переход от мировых координат к экранным координатам;
2) трехмерные сдвиг и масштабирование;
3) трёхмерное вращение вокруг произвольной оси;
4) изометрическое проецирование трёхмерного изображения на плоскость;
5) удаление невидимых линий (прорисовка пунктиром).

1. Алгоритм перехода от мировых координат к экранным координатам.

Для получения 3D-преобразований воспользуемся векторно-полигональной моделью фигуры. В соответствии с ней любая фигура в трехмерном пространстве может быть представлена матрицей координат своих вершин. Затем, используя данную матрицу, проводят вектора, которые соединяют соответствующие вершины. В результате получают каркасное изображение фигуры. Далее, используя алгоритмы удаления невидимых линий, получают изображение фигуры с отсечением невидимых ребер.

Изначально любая трехмерная фигура задается мировыми однородными координатами. Для ее отображения на экране необходимо получить экранные координаты. Для получения экранных координат используют следующую цепочку преобразований:

мировые координаты (x, y, z) -> видовые координаты $(x_в, y_в, z_в)$ *->* $(x_{пр}, y_{пр}, z_{пр})$ *-> экранные координаты* $(x_Э, y_Э, z_Э)$.

2. Преобразования сдвига и масштабирования в однородных координатах относительно центра координат имеют одинаковую форму произведения вектора исходных координат вершины фигуры на матрицу преобразования.

Преобразование сдвига.

Преобразование сдвига выглядит следующим образом:

$$v \cdot T = (x\ y\ z\ h) \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ l & m & n & 1 \end{pmatrix},$$

где v – вектор трехмерных однородных координат вершины фигуры, h – произвольный множитель не равный нулю (в нашем случае 1), T – матрица сдвига, l, m, n – величины смещения по осям OX, OY, OZ соответственно.

Преобразование масштабирование.

Преобразование сдвига выглядит следующим образом:

$$v \cdot T = (x\ y\ z\ h) \cdot \begin{pmatrix} a & 0 & 0 & 0 \\ 0 & e & 0 & 0 \\ 0 & 0 & j & 0 \\ 0 & 0 & 0 & s \end{pmatrix},$$

где v – вектор трехмерных однородных координат вершины фигуры, , T – матрица масштабирования: a, e, j – величины масштабирования по осям OX, OY, OZ соответственно; s – полное изменение масштаба (в данном случае a, e, j должны быть равны 1).

3. Трёхмерное вращение вокруг произвольной оси.

Преобразование поворота вокруг произвольной оси, заданной направляющими косинусами и проходящей через заданную точку, выглядит следующим образом:

$$v \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -l & -m & -n & 1 \end{pmatrix} \cdot R \cdot \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ l & m & n & 1 \end{pmatrix},$$

где v – вектор трехмерных однородных координат вершины многогранника,

l, m, n – заданные координаты точки, через которую проходит ось вращения,

R – матрица поворота вокруг оси, заданной направляющими косинусами:

$$R := \begin{bmatrix} (n_1)^2 + (1 - n_1)^2 \cos(\theta) & n_1 \cdot n_2 (1 - \cos(\theta)) + n_3 \sin(\theta) & n_1 \cdot n_3 (1 - \cos(\theta)) & 0 \\ n_1 \cdot n_2 \cdot (1 - \cos(\theta)) - n_3 \cdot \sin(\theta) & (n_2)^2 + (1 - n_2)^2 \cdot \cos(\theta) & n_2 \cdot n_3 (1 - \cos(\theta)) + n_1 \cdot \sin(\theta) & 0 \\ n_1 \cdot n_3 \cdot (1 - \cos(\theta)) + n_2 \cdot \sin(\theta) & n_2 \cdot n_3 \cdot (1 - \cos(\theta)) - n_1 \cdot \sin(\theta) & (n_3)^2 + (1 - n_3)^2 \cdot \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

где n_1, n_2, n_3 – направляющие косинусы относительно осей OX, OY и OZ соответственно, Θ – угол поворота.

Положительным направлением считается направление против часовой стрелки, если смотреть вдоль оси вращения к началу координат.

4. Проецирование изображения.

В данной лабораторной работе применяется изометрическая проекция. Матрица проецирования имеет вид:

$$pro := \begin{pmatrix} -\cos\left(\frac{\pi}{6}\right) & \sin\left(\frac{\pi}{6}\right) & 0 & 0 \\ \cos\left(\frac{\pi}{6}\right) & \sin\left(\frac{\pi}{6}\right) & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

Проецирование осуществляется умножением вектора трехмерных однородных координат на матрицу проецирования. Оси системы координат расположены под углом 120 градусов относительно друг друга.

5. Для построения более реалистичного изображения трёхмерных сцен необходимо удалять невидимые части объектов (рёбра и грани).

Существует 2 основных подхода к решению данной задачи.

Первый подход заключается в непосредственном сравнении объектов друг с другом для выяснения того, какие части объектов являются видимыми. В данном случае работа ведётся в пространстве объектов.

Второй подход заключается в определении для каждого пикселя экрана ближайшего к нему объекта (вдоль направления проецирования). При этом работа ведётся в пространстве экранных координат.

Рассмотрим один из алгоритмов, реализующий первый подход.

Отсечение нелицевых граней.

Пусть для каждой грани некоторой фигуры задан единичный вектор внешней нормали. Если вектор нормали грани составляет с направлением проецирования (направлением взгляда на объект) тупой угол, то такая грань не может быть видна и называется нелицевой. В случае, когда данный угол является острым, грань видна и называется лицевой.

В случае, когда трёхмерная сцена представляет собой один выпуклый многогранник, удаление нелицевых граней полностью решает задачу удаления невидимых граней.

В общем случае описанная проверка не решает задачу полностью, но позволяет значительно сократить количество рассматриваемых граней.

Алгоритм удаления нелицевых плоскостей, используемый в данной работе.

1. Сформировать многоугольники граней, исходя из списка вершин тела.

2. Вычислить нормальный вектор для каждой полигональной грани тела.

3. Определить нелицевые плоскости:

Определить косинус угла между нормалью к грани и вектором направления наблюдателя.

Если этот косинус угла меньше 0, то плоскость невидима.

Вывести весь многоугольник, лежащий в этой плоскости пунктиром.

4. Вывести остальные грани.

4 Описание структуры и алгоритма программы

Разработанная программа имеет структуру, изображенную на рисунке 4.1.

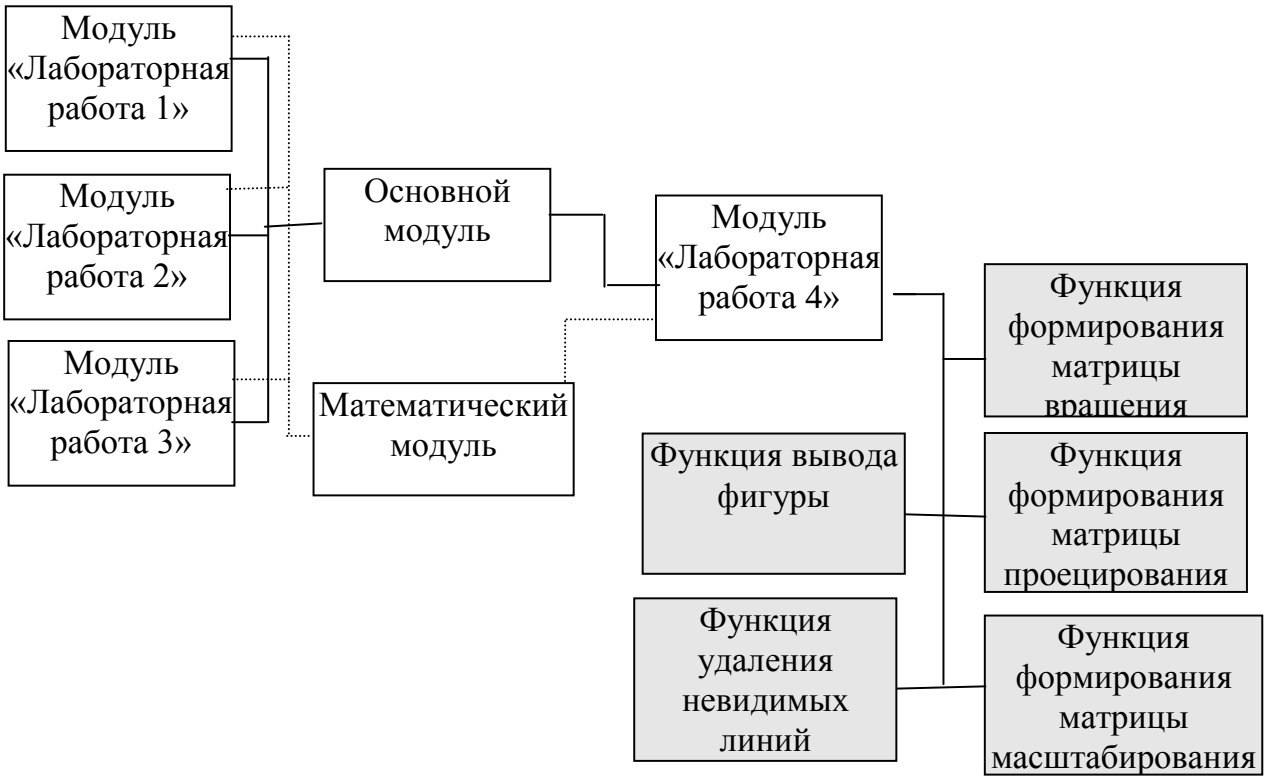


Рисунок 4.1 – Структура приложения

Разработанный в данной лабораторной работе модуль имеет имя lab4.cs. Его форма lab4.cs.

Основной алгоритм программного модуля:

1. Формируем матрицу мировых координат фигуры

2. Формируем матрицы преобразования.

3. Организуем цикл поворота фигуры:

3.1 очистка экрана;

3.2 умножаем матрицу фигуры на матрицу смещения;

3.3 умножаем матрицу фигуры на матрицу поворота;

3.4 умножаем матрицу фигуры на матрицу обратного смещения;

3.5 умножаем матрицу фигуры на матрицу проецирования;

3.6 определяем видимость ребер и выводим фигуру красным цветом с новыми координатами;

3.7 увеличиваем угол поворота;

3.8 переходим на шаг 3.

4.1 Описание основных функций

Модуль содержит несколько функций:

. . .

(описание функций – название, входные, выходные параметры, ее назначение и подробное пошаговое описание основных процедур)

void FormCreate(Sender: TObject); - процедура для создания формы.

void Draw(A:TMatrix; f:boolean); – в зависимости от значения f выводит грань пунктирной или сплошной линией.

void ImageClear; – очистка экрана.

Tmatrix fVectorMul(A:TMatrix); – функция возвращает нормаль к грани.

double ScalarMul(Vector:TMatrix); – функция определяет косинус угла между нормалью и направлением проекцирования.

void DrawXY(A:TMatrix); – выводит оси на экран.

Пошаговое описание основной процедуры Draw

Шаг 1. Выделяем память под массивы граней фигуры, нормалей к этим граням, матрицу переноса, матрицу поворота фигуры вокруг оси, матрицу изометрической проекции;

Шаг 2.Задаем начальные значения для всех этих матриц;

Шаг 3.Начальный угол поворота равен 0;

Шаг 4.Цикл пока не нажата кнопка «Стоп»;

Шаг 5. Повернуть 1-ю грань фигуры на угол Phi

Шаг 6. Определить нормаль к этой грани

Шаг 7. Определить угол между нормалью и вектором проецирования

Шаг 8. Если угол тупой, грань не лицевая f1=false, иначе лицевая f1= true

Шаг 9. Повернуть 2-ю грань фигуры на угол Phi

Шаг 10. Определить нормаль к этой грани

Шаг 11. Определить угол между нормалью и вектором проецирования

Шаг 12. Если угол тупой, грань не лицевая f2=false, иначе лицевая, f2= true

Шаг 13. Повернуть 3-ю грань фигуры на угол Phi

Шаг 14. Определить нормаль к этой грани

Шаг 15. Определить угол между нормалью и вектором проецирования

Шаг 16. Если угол тупой, грань не лицевая f3=false, иначе лицевая f3=true

Шаг 17. Повернуть 4-ю грань фигуры на угол Phi

Шаг 18. Определить нормаль к этой грани

Шаг 19. Определить угол между нормалью и вектором проецирования

Шаг 20. Если угол тупой, грань не лицевая f4=false, иначе лицевая f4=true

Шаг 21. Если 1-ая грань не лицевая – нарисовать пунктиром, иначе ничего не выводить

Шаг 22. Если 2-ая грань не лицевая – нарисовать пунктиром, иначе ничего не выводить

Шаг 23. Если 3-ая грань не лицевая – нарисовать пунктиром, иначе ничего не выводить

Шаг 24. Если 4-ая грань не лицевая – нарисовать пунктиром, иначе ничего не выводить

Шаг 25. Если 1-ая грань лицевая – нарисовать сплошной линией, иначе ничего не выводить

Шаг 26. Если 2-ая грань лицевая – нарисовать сплошной линией, иначе ничего не выводить

Шаг 27. Если 3-ая грань лицевая – нарисовать сплошной линией, иначе ничего не выводить

Шаг 28. Если 4-ая грань лицевая – нарисовать сплошной линией, иначе ничего не выводить

Шаг 29. Увеличить угол вращения

Шаг 30. Обновить матрицу вращения

Шаг 31. Кнопка «Стоп» нажата? Да выход, Иначе переход на Шаг 5.

5 Руководство пользователя

Исполняемый файл программы – lab4.exe. После запуска программы появляется основная форма (рисунок 2).

. . .

Для запуска лабораторной работы 4 нажмите кнопку «Лабораторная работа4». Появиться форма (рисунок 3).

. . .

Для поворота фигуры нажмите кнопку «Поворот». Для изменения скорости вращения передвиньте ползунок на полосе прокрутки (рисунок 4).

. . .

Для выхода из программы закройте форму.

6 Ответы на контрольные вопросы

…….

7 Тестовые вопросы

…….

Заключение

В ходе проделанной работы изучили алгоритмы получения 3D-преобразований фигуры, а именно: алгоритм формирования векторно-полигональной модели фигуры, алгоритм перехода от мировых координат к экранным, алгоритмы вращения, смещения, масштабирования в пространстве, алгоритм проецирования, алгоритм удаления невидимых линий.

В результате создан программный продукт, реализующий все изученные алгоритмы и предоставляющий следующие возможности:

- отображение трехмерной фигуры на экране;
- задание оси вращения;
- вращение фигуры вокруг заданной оси;
- перемещение и масштабирование фигуры по каждой координатной оси, по двум или трем осям одновременно;
- удаление невидимых ребер фигуры;
- настройку скорости и направления вращения / перемещения;
- настройку цвета и стиля линий.

Приложение Б

Пример оформления титульного листа реферата

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)»

Кафедра компьютерных систем в управлении и проектировании (КСУП)

Реферат
по дисциплине «Компьютерная графика»
на тему: «Графические процессоры»

Выполнил:
студент гр. _____
_____ И. И. Иванов
«___» _____ 20 __ г.

Проверил:
канд.техн. наук,
доцент каф. КСУП,
_____ Н. Ю. Хабибулина
«___» _____ 20 __ г.

20__ г.