МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Новосибирский национальный исследовательский государственный университет»
(Новосибирский государственный университет, НГУ)

Физический факультет
Кафедра физико-технической информатики

УТВЕРЖДАЮ
Декан ФФ
академик РАН _____ А. Е. Бондарь
«07» 10 2020 г.

Рабочая программа дисциплины

МАШИННАЯ ГРАФИКА

направление подготовки: **03.03.02 Физика, Курс 4, семестр 8**
направленность (профиль): **Физическая информатика**

Форма обучения
Очная

Семестр	Общий объем	Виды учебных занятий (в часах)				Промежуточная аттестация (в часах)				
		Контактная работа обучающихся с преподавателем			Самостоятельная работа, не включая период сессии	Самостоятельная подготовка к промежуточной аттестации	Контактная работа обучающихся с преподавателем			
		Лекции	Практические занятия	Лабораторные занятия			Консультации	Зачет	Дифференцированный зачет	Экзамен
1	2	3	4	5	6	7	8	9	10	11
8	144	32		22	88				2	
Всего 144 часа / 4 зачётных единицы, из них: - контактная работа 56 часов - в интерактивных формах 22 часа										
Компетенции ПК-1, ПК-2										

Разработчик:
к.ф.-м.н., зав. каф. П. П. Кроковный

Заведующий кафедрой ФТИ ФФ НГУ
к.ф.-м.н. П. П. Кроковный

Ответственный за образовательную программу
д.ф.-м.н., проф. С. В. Цыбуля

Новосибирск, 2020

Содержание

Аннотация

к рабочей программе дисциплины
«Машинная графика»
Направление: **03.03.02 Физика**
Направленность (профиль): Физическая информатика

Программа курса **«Машинная графика»** составлена в соответствии с требованиями СУОС по направлению подготовки **03.03.02 Физика, направленность «Физическая информатика»,** а также задачами, стоящими перед Новосибирским государственным университетом по реализации Программы развития НГУ. Дисциплина реализуется на физическом факультете Федерального государственного автономного образовательного учреждения высшего профессионального образования Новосибирский национальный исследовательский государственный университет (НГУ) кафедрой физико-технической информатики. Дисциплина изучается студентами четвёртого курса физического факультета в качестве одной из дисциплин по выбору вариативной части образовательной программы.

Цели курса –практическое ознакомление студентов с алгоритмами и методами, применяемыми при создании графических изображений и средств человеко-машинного взаимодействия для решения различных прикладных задач на компьютере.

Дисциплина нацелена на формирование у выпускника профессиональных компетенций:

ПК-1 – способность использовать специализированные знания в области физики для освоения профильных физических дисциплин;

ПК-2 -способность проводить научные исследования в избранной области экспериментальных и (или) теоретических физических исследований с помощью современной приборной базы (в том числе сложного физического оборудования) и информационных технологий с учетом отечественного и зарубежного опыта.

В результате освоения дисциплины обучающийся должен:

- **Знать**: основные алгоритмы и методы машинной графики;
  основные растровые и векторные форматы представления изображений.
- **Уметь**: реализовывать в программном коде основные алгоритмы и методы машинной графики.
- **Владеть**: средой разработки Eclipse;
  навыками создания в ней графических Java-приложений.

Курс рассчитан на один семестр. Преподавание дисциплины предусматривает следующие формы организации учебного процесса: лекции, лабораторные занятия, самостоятельная работа студента и её контроль преподавателями с помощью заданий, дифференцированный зачёт.

Программой дисциплины предусмотрены следующие виды контроля:

Текущий контроль: выполнение практических заданий.

Промежуточная аттестация: дифференцированный зачёт.

Общая трудоемкость рабочей программы дисциплины составляет **144** академических часа / **4** зачетные единицы.

1. Перечень планируемых результатов обучения по дисциплине, соотнесённых с планируемыми результатами освоения образовательной программы.

Целью освоения дисциплины «Машинная графика» является практическое ознакомление студентов с алгоритмами и методами, применяемыми при создании графических изображений и средств человеко-машинного взаимодействия для решения различных прикладных задач на компьютере.

Для достижения поставленной цели ставятся следующие задачи:

1. Изучение теоретических основ двумерной и трёхмерной машинной графики, и наиболее важных алгоритмов;
2. Изучение практического применения этих алгоритмов в реальных приложениях на языке Java.

Профессиональная компетенция ПК-1 – способность использовать специализированные знания в области физики для освоения профильных физических дисциплин.

Профессиональная компетенция ПК-2 – способность проводить научные исследования в избранной области экспериментальных и (или) теоретических физических исследований с помощью современной приборной базы (в том числе сложного физического оборудования) и информационных технологий с учетом отечественного и зарубежного опыта.

При проведении научных исследований и решении практических задач в физике зачастую приходится работать с большими объёмами данных. Здесь исследователю в высшей степени помогают средства визуализации (в том числе интерактивные), позволяющие быстро понять общую картину. Не всегда удаётся воспользоваться готовыми приложениями, чтобы создать подходящую визуальную среду для управления каким-либо экспериментом или для обработки результатов. Нередко приходится разрабатывать специальные приложения. Данный курс даёт первоначальный опыт в разработке таких приложений.

В результате освоения дисциплины обучающийся должен:

- **Знать**:
  - основные алгоритмы и методы машинной графики (ПК 1.1);
  - основные растровые и векторные форматы представления изображений (ПК 2.1).
- **Уметь**:
  - реализовывать в программном коде основные алгоритмы и методы машинной графики (ПК-1.2).
- **Владеть**:
  - средой разработки Eclipse (ПК 1.3)
  - навыками создания в ней графических Java-приложений (ПК 2.3).

2. Место дисциплины в структуре образовательной программы.

Учебный курс «Машинная графика» относится к вариативной части программы дисциплин бакалавриата.

Для успешного освоения курса «Машинная графика» студенты должны обладать знаниями программирования.

3. Трудоёмкость дисциплины в зачётных единицах с указанием количества академических часов, выделенных на контактную работу обучающегося с преподавателем (по видам учебных занятий) и на самостоятельную работу.

Семестр	Общий объем	Виды учебных занятий (в часах)				Промежуточная аттестация (в часах)				
		Контактная работа обучающихся с преподавателем			Самостоятельная работа, не включая период сессии	Самостоятельная подготовка к промежуточной аттестации	Контактная работа обучающихся с преподавателем			
		Лекции	Практические занятия	Лабораторные занятия			Консультации	Зачет	Дифференцированный зачет	Экзамен
1	2	3	4	5	6	7	8	9	10	11
8	144	32		22	88				2	
Всего 144 часа / 4 зачётные единицы, из них: - контактная работа 56 часов - в интерактивных формах 22 часа										
Компетенции ПК-1, ПК-2										

Реализация дисциплины предусматривает практическую подготовку при проведении следующих видов занятий, предусматривающих участие обучающихся в выполнении отдельных элементов работ, связанных с будущей профессиональной деятельностью: лекции, лабораторные занятия, самостоятельная работа студента и её контроль преподавателями с помощью заданий, дифференцированный зачёт.

Программой дисциплины предусмотрены следующие виды контроля:

- текущий контроль успеваемости: выполнение практических заданий;
- промежуточная аттестация: дифференцированный зачёт.

Общая трудоемкость рабочей программы дисциплины составляет 4 зачетных единицы.

- занятия лекционного типа – 32 часа;
- лабораторные занятия – 22 часа;
- самостоятельная работа обучающегося в течение семестра, не включая период сессии – 88 часов;
- промежуточная аттестация (дифференцированный зачёт) – 2 часа.

Объём контактной работы обучающегося с преподавателем (занятия лекционного типа, лабораторные занятия) составляет 56 часов.

Работа с обучающимися в интерактивных формах составляет 22 часа (лабораторные занятия).

4. Содержание дисциплины, структурированное по темам (разделам) с указанием отведённого на них количества академических часов и видов учебных занятий.

Общая трудоемкость дисциплины составляет 4 зачётные единицы, 144 академических часа.

№ п/п	Раздел дисциплины	Неделя семестра	Виды учебной работы, включая самостоятельную работу студентов и трудоёмкость (в часах)					Консультации перед экзаменом (часов)	Промежуточная аттестация (в часах)
			Всего	Аудиторные часы		Сам. работа во время занятий (не включая период сессии	Сам. работа во время промежуточной аттестации		
				Лекции	Лабораторные занятия				
1	2	3	4	5	6	7	8	9	10
1.	Цели и задачи курса. Понятие компьютерной графики, историческая сводка. Области применения: генерация, обработка и анализ изображений. Основные задачи: методы графического представления данных, алгоритмы, языки, графические программные средства. Стандарты в компьютерной графике. Понятие дисплейного файла.	1	2	2					
2.	Изображения: растровые и векторные. Буфер кадра: индексный и полноцветный. Двойная буферизация. Пиксельные области: внутренне и гранично-	2	2	2					

	определённые. 4- и 8-связность. Заполнение пиксельных областей: span-алгоритм. Заполнение шаблоном.								
3.	Брезенхэма. Растеризация со сглаживанием, алгоритм У Сяолиня. Многоугольники: определения, ориентация, характеристическая функция. Выпуклость. Площадь многоугольника.	3	2	2					.
4.	Обрезание (клиппирование) многоугольников: алгоритмы Сазерленда-Ходжмана и Вейлера Азертона. Растеризация многоугольников. Дизеринг: алгоритм упорядоченного дизеринга, алгоритм Флойда-Стейнберга.	4	2	2					
5.	Яркость и интенсивность. Восприятие интенсивности света. Гамма-коррекция. Попиксельные операции: негатив, чёрно-белое, яркость, контраст. Композиция изображений. Альфа-канал. Алиасинг и антиалиасинг. Муары. Субпиксельный рендеринг. Временной	5-6	4	4					

	алиасинг в анимации. Разрешение изображения: пространственное, временное, по интенсивности. Регуляризация изображений. Свёрточные фильтры: сглаживание, резкость, тиснение. Выделение контуров, оператор Собеля, оператор Робертса. Медианный фильтр. Акварелизация.								
6.	Визуализация в научных вычислениях. Цели визуализации. Визуализация функции одной переменной: график функции, гистограмма. Методы совместной визуализации нескольких величин. Визуализация функций двух переменных: понижение размерности, цветовая карта, изолинии, векторные поля, линии тока. Алгоритм марширующих квадратов. Визуализация функций многих переменных: объёмные изображения, изоповерхности, повышение	7	2	2					

	информативности изображения, анимация, интерактивность. Алгоритм марширующих кубов. Визуализация объёмных плотностей. Лица Чернова.								
7.	Элементы вычислительной геометрии на плоскости и в пространстве. Точка, вектор, отрезок, прямая, плоскость, нормаль. Основные формулы. Симплекс. Барицентрические координаты. Разбиение единицы. Триангуляция.	8	2	2					
8.	Преобразования. Понятия модельных, видовых, анимационных преобразований. Векторная алгебра, базисы. Линейные и аффинные преобразования. Сдвиг, масштабирование и поворот на плоскости. Матричные записи для линейных. Задача поворота относительно точки (невозможно представить одной матрицей из-за сдвига). Примеры языка PostScript.	9	2	2					

	Однородные координаты и 2Д преобразования. Соглашения о записи вектора: строка / столбец. Запись и применение преобразований. Левосторонняя и правосторонняя СК. Положительное вращение вокруг осей. Однородные координаты и 3Д преобразования. Поворот: углы Эйлера. Преобразование твердого тела. Поворот вокруг произвольной оси.								
9.	Видовое преобразование: перспективное и параллельное проецирование. Пирамида видимости (POV), видимый объем. Матрицы проецирования, вывод. Преобразование видимого объема к полукубу. Алгоритм Z-буфера. Конвейер преобразования координат: модельные СК – мировая СК – СК камеры – полукуб – клиппирование – плоскость изображения – порт вывода.	10-11	2	2					
10.	Конструирование кривых с локальной модификацией. Метод Эрмита, метод Безье. Геометрические	12	2	2					

	свойства отрезка кривой Безье. B-сплайны. Геометрические свойства B-сплайновой кривой. Сопряжение участков кривых (Эрмит, Безье). Геометрическая и параметрическая непрерывности.								
11.	Параметрические поверхности. Параметрические линии на поверхностях и касательные к ним. Нормаль к поверхности. Уравнение пути фрезы. Конструирование участков поверхностей. Билинейная функция. Линейчатая поверхность, построенная на опорных кривых. Участок поверхности по методу Кунса. Участок поверхности по методу Эрмита. Участок поверхности по методу Безье. Геометрические свойства. B сплайны. Сшивка двух участков поверхности. Суперквадрики. Поверхности вращения. Другие моделирующие преобразования: скручивание, экструзия и т.п. Морфинг поверхностей.	13	2	2					

12.	Пересечение луча с поверхностями: сфера, бокс, плоскость, многоугольник, квадрика. Нормаль, отражённый и преломлённый луч.	14	2	2					
13.	Задачи трёхмерной графики. Психофизиология зрения. Отражение света от материалов: двулучевая функция отражательной способности. Зеркальное и диффузное (Ламбертово) отражение. Способы визуализации: лучевая трассировка, трассировка Монте-Карло, радиосити.	15	2	2					
14.	Локальная модель освещённости. Модель полного отражения по Фонгу. Сканирующие алгоритмы: монотонный, алгоритм Гуро, алгоритм Фонга. Генерация полиго-нальных сеток. Усредне-ние нормалей. Алгоритм лучевой трассировки.	16	2	2					
15.	Полигональный рендеринг, особенности. Пространственны е структуры данных. Ограничивающие объёмы. Сетки.	16	2	2					

	Вложенные сетки. Квадродеревья, октодеревья. Визуализация теней.								
16.	Повторение языка Java. Среда разработки Eclipse, создание простейшего графического приложения	1	10		2	8			
17.	Задача 1. Редактор ломаных. Требуется реализовать графическое приложение на Java, позволяющее в интерактивном режиме создавать векторное изображение из набора ломаных с возможностью сохранения в файл и загрузки.	2-3	14		2	12			
18.	Задача 2. Заливка. Требуется расширить приложение из первой задачи, добавив возможность заливать 4-связные и 8-связные области span-алгоритмом с использованием шаблона.	4-5	14		2	12			
19.	Задача 3. Визуализация двумерной функции. Требуется реализовать приложение на Java, отображающее двумерную функцию различными способами: изолинии, цветотоновая	6-8	18		4	14			

	карта, режим интерполяции, дизеринг.								
20.	Задача 4. Графические фильтры. Требуется реализовать приложение, отображающее растровое изображение из файла и позволяющее изменить его следующими способами: перевод в чёрно-белое, изменение яркости, контраста, гамма-коррекция, негатив, размытие, повышение резкости, фильтр тиснения, выделение контуров, акварельный фильтр.	9-11	18		4	14			
21.	Задача 5. Трёхмерная проволочная модель. Требуется создать приложение, отображающее в перспективной проекции набор трёхмерных тел вращения в виде проволочной модели, используя матричные преобразования для формирования сцены. Образующие тел вращения пользователь задаёт при помощи Безье-сплайна или B-сплайна в интерактивном	12-14	18		4	14			

	редакторе. Сцену можно вращать перед камерой с помощью мыши.								
22.	Задача 6. Полигональная визуализация. Требуется расширить приложение из предыдущей задачи так, чтобы тела вращения отображались сканирующим алгоритмом Гуро с использованием модели полного отражения по Фонгу для нескольких источников света.	15-16	18		4	14			
23.	Дифференцированный зачёт	17	2						2
Всего			**144**	**32**	**22**	**88**			**2**

Программа и основное содержание лекций (32 часа)

1. Цели и задачи курса. Понятие компьютерной графики, историческая сводка. Области применения: генерация, обработка и анализ изображений. Основные задачи: методы графического представления данных, алгоритмы, языки, графические программные средства. Стандарты в компьютерной графике. Понятие дисплейного файла (2 часа).
2. Изображения: растровые и векторные. Буфер кадра: индексный и полноцветный. Двойная буферизация. Пиксельные области: внутренне и гранично-определённые. 4- и 8-связность. Заполнение пиксельных областей: span-алгоритм. Заполнение шаблоном. (2 часа)
3. Брезенхэма. Растеризация со сглаживанием, алгоритм У Сяолиня. Многоугольники: определения, ориентация, характеристическая функция. Выпуклость. Площадь многоугольника. (2 часа)
4. Обрезание (клиппирование) многоугольников: алгоритмы Сазерленда-Ходжмана и Вейлера Азертона. Растеризация многоугольников. Дизеринг: алгоритм упорядоченного дизеринга, алгоритм Флойда-Стейнберга. (2 часа)
5. Яркость и интенсивность. Восприятие интенсивности света. Гамма-коррекция. Попиксельные операции: негатив, чёрно-белое, яркость, контраст. Композиция изображений. Альфа-канал. Алиасинг и антиалиасинг. Муары. Субпиксельный рендеринг. Временной алиасинг в анимации. Разрешение изображения: пространственное, временное, по интенсивности. Регуляризация изображений. Свёрточные фильтры: сглаживание, резкость, тиснение. Выделение контуров, оператор Собеля, оператор Робертса. Медианный фильтр. Акварелизация. (4 часа)
6. Визуализация в научных вычислениях. Цели визуализации. Визуализация функции одной переменной: график функции, гистограмма.
   Методы совместной визуализации нескольких величин. Визуализация функций двух переменных: понижение размерности, цветовая карта, изолинии, векторные поля, линии тока. Алгоритм марширующих квадратов. Визуализация функций многих переменных:

объёмные изображения, изоповерхности, повышение информативности изображения, анимация, интерактивность. Алгоритм маршрирующих кубов. Визуализация объёмных плотностей. Лица Чернова. (2 часа)

7. Элементы вычислительной геометрии на плоскости и в пространстве. Точка, вектор, отрезок, прямая, плоскость, нормаль. Основные формулы. Симплекс. Барицентрические координаты. Разбиение единицы. Триангуляция. (2 часа)
8. Преобразования. Понятия модельных, видовых, анимационных преобразований. Векторная алгебра, базисы. Линейные и аффинные преобразования. Сдвиг, масштабирование и поворот на плоскости. Матричные записи для линейных. Задача поворота относительно точки (невозможно представить одной матрицей из-за сдвига). Примеры языка PostScript. Однородные координаты.
   и 2Д преобразования. Соглашения о записи вектора: строка / столбец. Запись и применение преобразований. Левосторонняя и правосторонняя СК. Положительное вращение вокруг осей. Однородные координаты и 3Д преобразования. Поворот: углы Эйлера. Преобразование твердого тела. Поворот вокруг произвольной оси. (2 часа)
9. Видовое преобразование: перспективное и параллельное проецирование. Пирамида видимости (POV), видимый объем. Матрицы проецирования, вывод. Преобразование видимого объема к полукубу. Алгоритм Z-буфера. Конвейер преобразования координат: модельные СК – мировая СК – СК камеры – полукуб – клиппирование – плоскость изображения – порт вывода. (2 часа)
10. Конструирование кривых с локальной модификацией. Метод Эрмита, метод Безье. Геометрические свойства отрезка кривой Безье. B-сплайны. Геометрические свойства B-сплайновой кривой. Сопряжение участков кривых (Эрмит, Безье). Геометрическая и параметрическая непрерывности. (2 часа)
11. Параметрические поверхности. Параметрические линии на поверхностях и касательные к ним. Нормаль к поверхности.
    Уравнение пути фрезы. Конструирование участков поверхностей. Билинейная функция. Линейчатая поверхность, построенная на опорных кривых. Участок поверхности по методу Кунса. Участок поверхности по методу Эрмита. Участок поверхности по методу Безье. Геометрические свойства. B сплайны. Сшивка двух участков поверхности. Суперквадрики. Поверхности вращения. Другие моделирующие преобразования: скручивание, экструзия и т.п. Морфинг поверхностей. (2 часа)
12. Пересечение луча с поверхностями: сфера, бокс, плоскость, многоугольник, квадрика. Нормаль, отражённый и преломлённый луч. (2 часа)
13. Задачи трёхмерной графики. Психофизиология зрения. Отражение света от материалов: двулучевая функция отражательной способности. Зеркальное и диффузное (Ламбертово) отражение. Способы визуализации: лучевая трассировка, трассировка Монте-Карло, радиосити. (2 часа)
14. Локальная модель освещённости. Модель полного отражения по Фонгу. Сканирующие алгоритмы: монотонный, алгоритм Гуро, алгоритм Фонга. Генерация полиго-нальных сеток. Усредне-ние нормалей. Алгоритм лучевой трассировки. (2 часа)
15. Полигональный рендеринг, особенности. Пространственные структуры данных. Ограничивающие объёмы. Сетки. Вложенные сетки. Квадродеревья, октодеревья. Визуализация теней. (2 часа)

Программа лабораторных занятий (22 часа)

1. Повторение языка Java. Среда разработки Eclipse, создание простейшего графического приложения (2 часа).
2. Задача 1. Редактор ломаных. Требуется реализовать графическое приложение на Java, позволяющее в интерактивном режиме создавать векторное изображение из набора ломаных с возможностью сохранения в файл и загрузки (2 часа).

3. Задача 2. Заливка. Требуется расширить приложение из первой задачи, добавив возможность заливать 4-связные и 8-связные области span-алгоритмом с использованием шаблона (2 часа).
4. Задача 3. Визуализация двумерной функции. Требуется реализовать приложение на Java, отображающее двумерную функцию различными способами: изолинии, цветотоновая карта, режим интерполяции, дизеринг. (4 часа)
5. Задача 4. Графические фильтры. Требуется реализовать приложение, отображающее растровое изображение из файла и позволяющее изменить его следующими способами: перевод в чёрно-белое, изменение яркости, контраста, гамма-коррекция, негатив, размытие, повышение резкости, фильтр тиснения, выделение контуров, акварельный фильтр. (4 часа)
6. Задача 5. Трёхмерная проволочная модель. Требуется создать приложение, отображающее в перспективной проекции набор трёхмерных тел вращения в виде проволочной модели, используя матричные преобразования для формирования сцены. Образующие тел вращения пользователь задаёт при помощи Безье-сплайна или B-сплайна в интерактивном редакторе. Сцену можно вращать перед камерой с помощью мыши. (4 часа)
7. Задача 6. Полигональная визуализация. Требуется расширить приложение из предыдущей задачи так, чтобы тела вращения отображались сканирующим алгоритмом Гуро с использованием модели полного отражения по Фонгу для нескольких источников света. (4 часа)

Самостоятельная работа студентов (88 часов)

Перечень занятий на СРС	Объем, час
Подготовка практических заданий	88

5. Перечень учебной литературы.

5.1. Основная литература

1. Грузман И. С., Киричук В. С., Косых В. П., Перетягин Г. И., Спектор А. А. Цифровая обработка изображений в информационных системах. — Новосибирск, 2000.

5.2. Дополнительная литература

2. Дебелов В. Объектно-ориентированная система машинной графики для Windows (C++ и Microsoft DirectX), Новосибирск, 1999.

6. Перечень учебно-методических материалов по самостоятельной работе обучающихся.

Самостоятельная работа студентов поддерживается следующими учебными пособиями:

3. Грузман И. С., Киричук В. С., Косых В. П., Перетягин Г. И., Спектор А. А. Цифровая обработка изображений в информационных системах. — Новосибирск, 2000.

7. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины.

Для освоения дисциплины используются следующие ресурсы:
- электронная информационно-образовательная среда НГУ (ЭИОС);
- образовательные интернет-порталы;
- информационно-телекоммуникационная сеть Интернет.

Интернет-ресурсы:

1. Дебелов В. А., Валеев Т. Ф. Машинная графика. Электронный лекционный курс. НГУ, 2014.
2. Голованов Н. Н. Геометрическое моделирование. — М.: Академия, 2011. — 272 с.
3. Дегтярев В. М. Компьютерная геометрия и графика. — М.: Академия, 2011. — 192 с.
4. Ньюмен П., Спрулл Р., "Основы интерактивной машинной графики", Москва, Мир, 1976.
5. Порев В. Н. Компьютерная графика. — СПб.: BHV, 2002.
6. Роджерс Д., Адамс Дж. Математические основы машинной графики. Пер. с англ. — М.: Мир, 2001. — 604 с.
7. Шикин Е. В., Боресков А. В. Компьютерная графика. Полигональные модели. — М.: ДИАЛОГ-МИФИ, 2000. — 464 с.
8. Tufte, E. R. Envisioning Information. — Graphics Press, 1990. — 126 pp.
9. Java 7 JDK documentation http://docs.oracle.com/javase/7/docs/
10. Eclipse documentation http://www.eclipse.org/documentation/
11. Дёмин А. Ю., Кудинов А. В. Курс компьютерной графики Томского Политехнического Университета http://compgraph.tpu.ru/
12. Курс лекций по «Компьютерной графике» Новосибирского Государственного Технического Университета. http://ermak.cs.nstu.ru/kg_rivs/graf.htm

7.1 Современные профессиональные базы данных

Не используются.

7.2. Информационные справочные системы

Не используются.

8. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине.

Для обеспечения реализации дисциплины используется стандартный комплект программного обеспечения (ПО), включающий регулярно обновляемое лицензионное ПО Windows и MS Office.

Использование специализированного программного обеспечения для изучения дисциплины не требуется.

9. Материально-техническая база, необходимая для осуществления образовательного процесса по дисциплине.

Для реализации дисциплины используются специальные помещения:

1. Учебные аудитории для проведения занятий лекционного типа, практических занятий, курсового проектирования (выполнения курсовых работ), групповых и индивидуальных консультаций, текущего контроля, промежуточной и итоговой аттестации.

2. Помещения для самостоятельной работы обучающихся.

Учебные аудитории укомплектованы специализированной мебелью и техническими средствами обучения, служащими для представления учебной информации большой аудитории.

Помещения для самостоятельной работы обучающихся оснащены компьютерной техникой с возможностью подключения к сети "Интернет" и обеспечением доступа в электронную информационно-образовательную среду НГУ.

Материально-техническое обеспечение образовательного процесса по дисциплине для обучающихся из числа лиц с ограниченными возможностями здоровья осуществляется согласно «Порядку организации и осуществления образовательной деятельности по образовательным программам для инвалидов и лиц с ограниченными возможностями здоровья в Новосибирском государственном университете».

10. Оценочные средства для проведения текущего контроля и промежуточной аттестации по дисциплине.

Порядок проведения текущего контроля и промежуточной аттестации по дисциплине

Текущий контроль

Текущий контроль успеваемости осуществляется в виде выполнения практических заданий.

Промежуточная аттестация

Освоение компетенций оценивается согласно шкале оценки уровня сформированности компетенции. Положительная оценка по дисциплине выставляется в том случае, если заявленные компетенции ПК-1 и ПК-2 сформированы не ниже порогового уровня в части, относящейся к формированию способности использовать специализированные знания в области машинной графики в профессиональной деятельности.

Окончательная оценка работы студента в течение семестра происходит на дифференцированном зачёте. Он проводится в конце семестра в устной форме. Вопросы подбираются таким образом, чтобы проверить уровень сформированности компетенций ПК-1 и ПК-2.

Вывод об уровне сформированности компетенций принимается преподавателем. Положительная оценка ставится, когда все компетенции освоены не ниже порогового уровня. Оценки «отлично», «хорошо», «удовлетворительно» означают успешное прохождение промежуточной аттестации.

Описание критериев и шкал оценивания индикаторов достижения результатов обучения по дисциплине «Машинная графика».

Критерии оценивания результатов обучения	Планируемые результаты обучения (показатели достижения заданного уровня освоения компетенций)	Уровень освоения компетенции			
		Не сформирован (0 баллов)	Пороговый уровень (3 балла)	Базовый уровень (4 балла)	Продвинутый уровень (5 баллов)

1	2	3	4	5	6
Полнота знаний	ПК 1.1 ПК 2.1	Уровень знаний ниже минимальных требований. Имеют место грубые ошибки.	Минимально допустимый уровень знаний. Допускается значительное количество негрубых ошибок.	Уровень знаний соответствует программе подготовки по темам/разделам дисциплины. Допускается несколько негрубых/ несущественных ошибок. Не отвечает на дополнительные вопросы.	Уровень знаний соответствует программе подготовки по темам/разделам дисциплины. Свободно и аргументированно отвечает на дополнительные вопросы.
Наличие умений	ПК 1.2	Отсутствие минимальных умений. Не умеет решать стандартные задачи. Имеют место грубые ошибки.	Продемонстрированы частично основные умения. Решены типовые задачи. Допущены негрубые ошибки.	Продемонстрированы все основные умения. Решены все основные задания с негрубыми ошибками или с недочетами.	Продемонстрированы все основные умения. Решены все основные задания в полном объеме без недочетов и ошибок.
Наличие навыков (владение опытом)	ПК 1.3 ПК 2.3	Отсутствие владения материалом по темам/разделам дисциплины. Нет навыков в решении стандартных задач. Наличие грубых ошибок.	Имеется минимальный набор навыков при решении стандартных задач с некоторыми недочетами.	Имеется базовый набор навыков при решении стандартных задач с некоторыми недочетами.	Имеется базовый набор навыков при решении стандартных задач без ошибок и недочетов. Продемонстрированы знания по решению нестандартных задач.

Типовые контрольные задания и материалы, необходимые для оценки результатов обучения

Примеры практических заданий

Задача 1. Редактор ломаных

Требуется реализовать редактор полилиний с возможностью сохранения и загрузки. Полилиния представляет собой незамкнутый многоугольник или конечную последовательность отрезков, для которых начало следующего совпадает с концом предыдущего.

Программа должна позволять:

- Создать новый документ — удалить все созданные полилинии.
- Интерактивно нарисовать сегмент полилинии в режиме резиновой линии.
- Завершить сегмент и начать новый. Уже созданные сегменты менять нельзя.
- Начать новую полилинию. После этого вернуться к редактированию предыдущей нельзя.
- По кнопке удалять последнюю нарисованную полилинию (либо ту, которая рисуется в данный момент).

- Сохранить результат в текстовый файл с расширением. le строго определённого формата (см. ниже). Если пользователь не ввёл расширение, оно подписывается само.

- Загрузить полилинии из. le-файла, отобразить их и позволить добавлять новые к загруженным (если перед загрузкой что-то было нарисовано, это должно быть удалено).

- Произвольно менять размер окна. Если окно уменьшили так, что часть рисунка исчезла из виду, она не должна теряться, когда окно увеличат снова, и должна сохраняться в файл.

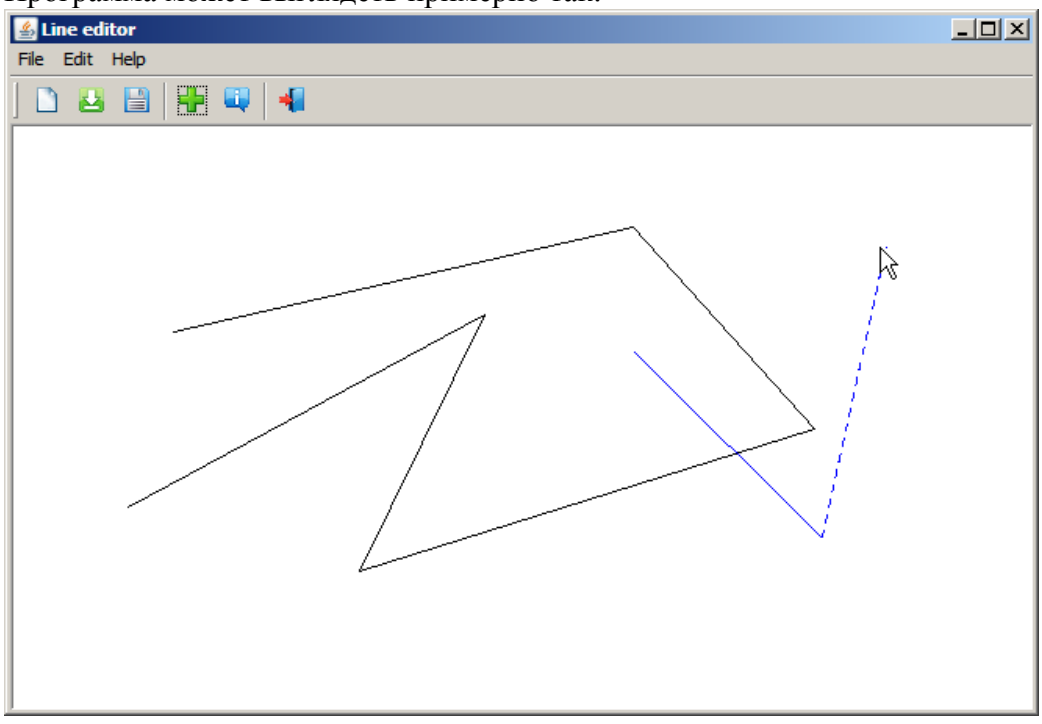
- Все возможности должны быть представлены на тулбаре (новый документ, загрузка, сохранение, окно About — обязательно). Все кнопки тулбара должны иметь всплывающие подсказки с описанием, что делает та или иная кнопка.

- Если программа поддерживает загрузку, то в комплекте с программой должен идти как минимум один пример файла, который корректно загружается и отображается.

- В файле About должны быть описаны все клавиши, кнопки мыши, которые приводят к определённым действиям. К примеру, «Для проведения отрезка следует прижать левую кнопку мыши и вести мышь с прижатой кнопкой». Или «Чтобы начать новую фигуру, нажмите правую кнопку мыши в любом месте окна».

- (MVC) Программа должна разделять модель и представление (view) на уровне классов. Код должен легко модифицироваться, чтобы подключить новое представление к существующей модели, причём представления должны обновляться синхронно. Рекомендуется для класса модели реализовать интерфейс Observable. По желанию можно контроллер реализовать в отдельном классе, но не запрещается объединить его с представлением.

Программа может выглядеть примерно так:



Кнопка About должна выглядеть так, как показано на рисунке —  . Это icons\About.gif в архиве icons.zip на сайте курса. Остальные кнопки — по желанию, можете взять другие картинки или нарисовать свои.

Формат. le-файла

Текстовый файл. При чтении символы пробела и табуляции считать равнозначными, а пробелы и табуляции в конце строк игнорировать. При записи использовать только пробелы и не допускать лишних пробелов в конце строк.

Первая строка — одно число, количество полилиний.

Далее последовательность записей, каждая запись описывает одну полилинию. Запись выглядит следующим образом:

Polyline — одно слово, написанное именно так

<x><y> — последовательность строк, описывающих координаты узлов полилинии. <x>, <y> — целые неотрицательные числа, координаты очередного узла в экранных пикселях от верхнего левого угла.

Пустая строка завершает запись. В конце файла (для последней записи) пустая строка может отсутствовать.

Пример корректного файла:

```
3
Polyline
95 49
87 273
176 275

Polyline
354 68
255 66
228 276
358 276

Polyline
242 160
314 165
```

Задача 2. Заливка.

Построить Java-приложение, реализующее рисование ломаной линии алгоритмом Брезенхэма, а также заливку 4- и 8-связных областей Span-алгоритмом. Приложение должно позволять рисовать на полотне размером 1920×1200 пикселей и включать режим прокрутки для окон меньшего размера (можно использовать ScrollPane).

Модель

Модель состоит из набора ломаных и заливок. Заливка описывается положением затравки, цветом (0 — тёмный, 1 — светлый, конкретные цвета на усмотрение автора) и связностью (4 или 8). Ломаная описывается набором координат вершин.

В клиентской области окна показывается набор линий примерно, как в задаче LE, но с прокруткой (можно использовать свой код из LE).

Работа с файлами

Проект должен уметь сохранять модель в файле и читать ее из файла. Расширение файла — *.spn. Все строки, начинающиеся с //, а также пустые строки должны игнорироваться! Отдельные числа в строке разделяются любым числом пробелов и табуляций. Все значения целочисленные.

В файле присутствуют записи о ломаных и о заливках вперемешку.

Строки с описанием ломаных

```
POLYLINE // Ключевое слово: дальше идёт ломаная
kp wp // число вершин ломаной, толщина линии
x y // координаты первой вершины
x y // координаты очередной вершины
x y // координаты очередной вершины
x y // координаты очередной вершины
…
x y // координаты последней (kp-й) вершины
```

Строки с описанием заливок

```
FILL // Ключевое слово: дальше идёт описание заливки в одну строку
xs ys col pc // координаты затравки, цвет (0 или 1), связность (4
или 8)
```

Что должно быть из средств управления.

Чтение из файла и запись в файл. По новому документу – чистая модель: 0 ломаных и 0 заливок.

Возможность рисовать линии разной толщины (хотя бы 1 пиксель и 3 пикселя) в режиме резиновой линии. Линии толщиной 1 пиксель обязательно рисовать с использованием собственной реализации алгоритма Брезенхэма. Линии толщиной 3 можно рисовать с использованием библиотечной функции.

Возможность выбрать режим заливки 4-связный или 8-связный. Возможность выбрать цвет заливки (тёмный или светлый). Возможность осуществить заливку. Заливку реализовать вручную span-алгоритмом.

В модель сохраняется набор ломаных и заливок в том порядке, в котором они были сделаны пользователем. При сохранении в файл порядок действий сохраняется. При загрузке все ломаные и заливки должны быть «проиграны» в том же порядке, чтобы получилась такая же картинка!

При прокрутке ScrollPane, минимизации окна и восстановлении изображение и информация о заливке не теряется (для этого может пригодиться BufferedImage). Можно залить разные фрагменты картинки разными цветами или, например, залить область, затем поделить её ломаной на две области и одну из них залить поверх другим цветом.

Задача 3. Визуализация двумерной функции.

Построить приложение, рисующее "портрет" однозначной функции 2-х переменных в виде цветной карты и карты изолиний.

Требования к программе.

1. Читать параметры задачи из файла (должен быть приготовлен свой файл данных).

2. Изменять параметры задачи в диалоге (JDialog): k, m, a, b, c, d (обязательно), остальные (цвета) по желанию. В диалоге обязательно должны быть проставлены tab-stops. Диалог вызывается по кнопке на тулбаре.

3. Отображать функцию в одном из трёх видов: цветовая карта, интерполяция, интерполяция с дизерингом Флойда-Стайнберга. Переключать виды кнопками на тулбаре. Перерисовывать легенду для каждого вида. Легенда — это как бы отдельная функция, равномерно возрастающая по вертикали, плюс подписи; будет красиво, если вы отобразите легенду с помощью того же кода, что и основную картинку, просто подсунув в этот код другую функцию.

4. Отображать поверх функции изолинии для уровней z1…zn; возможность включить и выключить изолинии кнопкой на тулбаре

5. Отображать поверх функции сетку k*m; возможность включить и выключить кнопкой на тулбаре; возможность сочетать с изолиниями

6. Добавить интерактивное построение изолиний по клику мыши: определяете по координатам пикселя, куда кликнули, точку в области определения функции f, берёте её значение и строите изолинию. Можно проводить изолинию через курсор, пока кнопка мыши прижата (тогда при движении мыши изолиния будет двигаться вместе с ней).

7. При движении мыши над полем значения **x**, **y** и **f(x,y)**, соответствующее точке экрана,выводится в определенное место, напр., в правом нижнем углу, под легендой.

8. Легенда состоит не только из палитры используемых цветов, но и цифровых значений уровней.

9. В описании точно указать, где в программе задается функция. Это должен быть отдельный метод.

10. Все действия осуществляются по кнопкам: режим отображения функции, вкл/выкл карты из изолиний, вкл/выкл сетки. Сетка строится либо при помощи линий (толщины 1), либо в виде узловых точек (отдельным цветом). Возможны варианты личных решений.

11. Поле изолиний и легенда должны вписываться в клиентскую область окна, т.е. необходимо производить расчет в программе.

12. Отсутствие разрывов в изолиниях.

13. Наличие анализа случаев 4-х точек пересечения изолинии с клеткой.

Формат файла:

```
k m      // число значений сетки по X и по Y
n        // число уровней
r0 g0 b0 // цвета легенды
r1 g1 b1
...
rn gn bn
ris gis bis // цвет построения изолиний
```

Задача 4. Графические фильтры.

Построить приложение, в котором показать знание следующих вопросов:

- Перевод цветного изображения в черно-белое
- Пиксельные фильтры
- Матричные фильтры
- Другие фильтры.

Общая характеристика программы:

1. Клиентская область окна делится на 2 зоны:

SRC **DST**

Зоны SRC и DST совершенно одинаковых размеров (пусть N×M пикселей), как правило, неквадратные. Между зонами и границами окна должен быть бордюр не менее 5 пикселей.

2. Имеется некоторое цветное изображение в файле .bmp/.png/.gif/.jpg. Размер его может быть больше размера зоны. В этом случае выводится на экран блок N×M пикселей из центра картины. Если изображение меньше размера зоны, оно должно выводиться в центре зоны, а остальная часть зоны должна закрашиваться синим цветом. Не фиксировать размеры зон. Они должны зависеть от размера клиентской области и меняться при растягивании окна. Все операции должны проводить для полного изображения, даже если оно не влезает в экран. После этого я могу растянуть окно и увидеть недостающие фрагменты.
3. Имеются кнопки для копирования изображения с зоны SRC в зону DST и наоборот. Можно просимулировать drag'n'drop мышью.
4. Все последующие операции берут в качестве исходного изображения зону SRC и помещают результат в зону DST. Какие операции:
5. Перевод цветного в черно-белое (0.299·R+0.587·G+0.114·B).
6. Выделение контуров. Вводится порог (JSlider). При движении слайдера картинка должна автоматически обновляться
7. Фильтры-свёртки: сглаживание (blur) и резкость (sharpen).
8. Идентичное преобразование
9. Негатив
10. Тиснение
11. Акварелизация
12. Гамма-коррекция. Уровень вводится через JSlider; тоже можно в JOptionPane; должна быть возможность вводить дробные величины как минимум от 0.1 до 10.0 с шагом 0.1. При движении слайдера картинка должна автоматически обновляться.
13. Произвольное матричное преобразование для матрицы 3×3. Матрицу задаёт пользователь в диалоге. В отдельном поле задаётся нормировочный коэффициент (на него всё делится в результате). В отдельном поле задаётся сдвиг (константа, которая прибавляется к итоговому значению цвета). Должна быть возможность автоматически вычислить нормировочный коэффициент (сумма всех элементов матрицы). Если вы сделали всё правильно, то сглаживание, резкость и тиснение будут лишь частными случаями этого фильтра (но для них всё равно должны быть предусмотрены отдельные кнопки).

14. Запись в файл .png изображения из зоны DST. Сохранять полное изображение, а не только то, что влезло в экран.
15. По чтению нового файла программа должна приводиться в исходное состояние.
16. Кнопки New быть не должно. По умолчанию программа должна открывать ваш пример (если не найдёт файл, можно или принудительно требовать выбрать изображение или просто ругаться и выходить).
17. По желанию (на 5 баллов не требуется) можно реализовать гауссово размытие. Придумайте, как сделать, чтобы работало не очень медленно.

Обязательно реализовать вручную все фильтры. Следите, чтобы не было переполнений цветов во всяких фильтрах. Если у вас при увеличении резкости компонент цвета стал больше 255 (меньше 0), и вы это не обработали, значит фильтр не работает.

Задача 5. Трёхмерная проволочная модель.

Построить приложение, рисующее набор объектов в виде проволочной модели. Удаление невидимых линий необязательно.

Цель: усвоить практическое применение преобразований в однородных координатах, преобразование модели в мировую систему координат (размещение на сцене), затем в систему координат камеры, проецирующие преобразования (перспективное!!!), клиппирование по полукубу.

Конкретно. Построить приложение, рисующее "портрет" поверхностей, заданных параметрически.

Итак, поверхность задана как

$$\mathbf{r = r(u,v) = (x(u,v), y(u,v), z(u,v),}$$

$$\mathbf{(u,v) \in D = [a,b]*[c,d].}$$

Шаг 1. Построить проволочную модель поверхности в модельном пространстве – XYZ. Для этого строим сетку на **D**: **n*m*клеток** (а не узлов) – **u0 = a, …, un = b; v0 = c, …, vm = d.** Далее строим пространственные ребра – 2 семейства отрезков:

А) для **i = 1..n, j = 0..m**,

$$\mathbf{[r(u(i-1), v(j)) – r(u(i), v(j))],}$$

Б) для **i = 0..n, j = 1..m**,

$$\mathbf{[r(u(i), v(j-1)) – r(u(i), v(j))].}$$

Другими словами, мы построили два семейства линий для поверхности, которые соответствуют параметрическим линиям **u = const, v = const.**

Модель строится как поверхность вращения **PRot.** Образующую задаёт пользователь в специальном диалоге (или режиме работы программы). Пользователем задаются контрольные точки, по которым строится B-сплайн. Можно добавлять, удалять и двигать точки.

На сцене может присутствовать до 10 моделей, заданных пользователем. Должна быть возможность добавить новую фигуру (открывается диалог для задания образующей) и удалить последнюю добавленную. По желанию — возможность редактировать любую из 10 фигур. После добавления или удаления фигуры пересчитывается габаритный бокс, и сцена приводится к начальному положению (см. шаг 3).

Для всех тел вращения параметры a, b, c, d одинаковые.

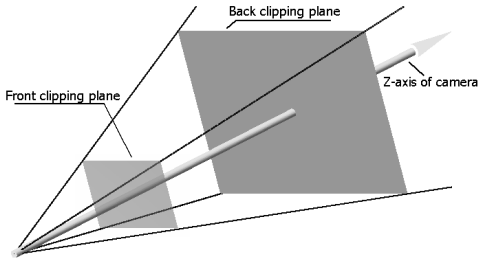
Сразу после запуска программы какая-то фигура должна визуализироваться (предопределите набор точек в программе).

Шаг 2. Выбор **n** и **m** может оказаться таким, что некоторые особенности поведения поверхности окажутся упущенными. Для более точного приближения проволочной модели к поверхности зададим сечения более подробно, т.е. выберем еще несколько дополнительных точек на каждом ребре – разобьем его на **k** более мелких отрезков. Можно, конечно, взять более мелкую сетку (большие значения (**n**и**m**), но тогда может получиться совсем неразборчивый чертеж. Таким образом, мы увеличиваем число пространственных отрезков, представляющих нашу проволочную (wireframe) геометрическую модель поверхности, в **k**раз. На приведенном рисунке **k** = 5. В таком случае изображение выглядит более гладким.

Шаг 3. Мы построилив модельных координатах нашу сцену в виде набора отрезков – множество отрезков **MO**. Подсчитываем габаритный объем – бокс. Сдвигаем и масштабируем так, чтобы габаритный бокс в мировой системе координат совпал с боксом **[-1,1]x[-1,1]x[0,1].** Дополнительно выполняем 3 поворота на углы Эйлера **ex, ey, ez**. Здесь у нас появляется промежуточная матрица однородного преобразования **M1.**

Шаг 4. Задаем положение камеры, это следующие 3 точки в XYZ:

- **(xc,yc,zc)** – точка положения верхушки камеры – точка (-10, 0, 0)
- **(xv,yv,zv)** – точка, задающая направление камеры (viewdirection) – точка (10, 0, 0)
- **(xu,yu,zu)** – точка, по которой определяется "направление вверх" камеры (upvector) – точка (0, 1, 0)



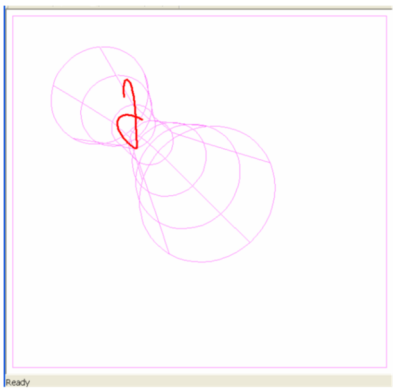
Камера

Далее зададим параметры пирамиды видимости (POV – pyramidofview) и объема визуализации (viewingfrustum):

- **zn** – расстояние до ближней клиппирующей плоскости
- **zf** – расстояние до дальней клиппирующей плоскости
- **sw, sh**– размеры грани объема визуализации на ближней плоскости

На клиентской области окна приложения выбираем прямоугольник **P**с углами **(u0,v0)** и **(u1,v1)** в экранных координатах (пикселях), как обычно.

Внимание: под изображение отводится не вся область окна, а должен оставаться небольшой бордюр. Его обвести. По нему будет вестись клиппирование.



Шаг 5. Для перевода всех отрезков из мировой системы координат в систему координат камеры служит матрица **МК** (построить). Пересчитываем координаты всех отрезков множества **МО** в систему координат камеры, используя полное преобразование, которое определяется матрицей **МП** равной **МК*М1**. Это отрезки, представляющие проволочную модель поверхности.

Применяем полное преобразование проецирования – матрица **Mpr**, т.е. переводим точки из модельной СК в точки видимого объема – точки полукуба **[-1,+1]*[-1,+1]*[0,+1]**.

Шаг 6. Клиппируем все отрезки границей полукуба и изображаем те их части, которые попали внутрь его.

Реализация

1. **Самое главное: должна применяться только матричная арифметика.** Все преобразования рассчитывать, как матрицы, формировать результирующую матрицу, а затем ее применять к точкам. Использовать однородные координаты. Нарушение этого условия – сразу 2 балла.
2. Должна быть возможность изменять значения **n, m, k, a, b, c, d.** Эти значения задаются изначально в файле данных. **Файл с данными должен быть.** Углы поворота ex, ey, ez меняются с помощью перемещения мыши с прижатой клавишей (придумайте, как это лучше сделать).
3. Параметры n, m, k, a, b, c, d, zn, zf, sw, sh должны также задаваться в диалоге
4. Значения, задающие положение и параметры камеры: **(xc,yc,zc), (xv,yv,zv), (xu,yu,zu) – заданы жестко**
5. Просто wireframe без удаления невидимых линий.
6. Управление сценой перед камерой с помощью мыши.

Формат файла

```
// В первой строке через пробел или табуляцию:
n m k a b c d zn zf sw sh
K // количество тел вращения
// Дальше описание отдельных тел
N // Количество опорных точек B-сплайна для образующей текущего тела
X1 Y1 // координаты первой точки
X2 Y2 // координаты второй точки
…
XN YN // координаты N-й точки
// потом второе тело и т. д.
```

Задача 6. Полигональная визуализация.

Построить приложение, рисующее набор трёхмерных объектов с использованием модели локальной освещённости Фонга и затенением Гуро. Удаление невидимых линий обязательно.

Цель: усвоить практическое применение модели освещённости и понять устройство простых трёхмерных движков.

Сцена и фигуры задаются так же, как в задаче WF — до 10 тел вращения с образующими в виде B-сплайнов, которые можно считывать из файла и редактировать в программе. Дополнительно на сцене присутствуют:

- Рассеянный свет

До трёх цветных точечных источников света, которые характеризуются координатами и цветом (r,g,b). Расставьте их по умолчанию как-нибудь разумно на сцене.
Тела вращения обладают материалом, который характеризуется:

- Три коэффициента (r,g,b) рассеянного и диффузного цвета
- Три коэффициента (r,g,b) зеркального отражения
- Показатель зеркальности по Фонгу.

То есть семь чисел (эмиссия отсутствует). Для лицевой и изнаночной стороны материалы независимые. Значения по умолчанию могут быть выбраны на усмотрение автора, но должны заметно отличаться для лицевой и изнаночной сторон (например, красный и синий цвета, либо тёмный и светлый).

Значение **k** при визуализации не нужно! При чтении из файла его пропускайте.

Основной ход алгоритма:

- Каждое тело вращения содержит **n** × **m** узловых вершин. Вычисляем их координаты и нормаль. Нормаль лежит в плоскости образующей, две другие координаты определите, используя формулу для B-сплайна (касательную получить через производную, из неё легко получить нормаль).
- Каждую вершину преобразуем к мировым координатам и освещаем, используя значения материала и источники света. Можно сразу тут выяснить, какая сторона направлена к камере и использовать соответствующий материал.
- Каждую вершину преобразуем к полукубу матричными преобразованиями, не забывая координату Z. Теперь у вас есть массив преобразованных и освещённых вершин.
- Из каждых четырёх вершин, формирующих ячейку сетки, делаем два треугольника (разбиваем по какой-нибудь диагонали).
- Для каждого треугольника определяем Z-координату (можно по-простому, усредняя Z-координаты всех вершин треугольника).
- Сортируем треугольники по глубине
- Отрисовываем их в экранных координатах, используя затенение Гуро. Двигаемся от дальних к ближним. Может пригодиться алгоритм заливки многоугольника из 4-й лекции.
- Рисуем всё в бэкбуфер. По окончании отрисовки копируем на экран.

В идеале (на 5 баллов) приложение должно позволять вращать сцену мышкой (как в задаче WF). Если сцена перерисовывается дольше 5 секунд (для n=m=20, 3 тела) и вы не можете оптимизировать перерисовку лучше, делайте вращение в режиме каркасной модели (в точности как в WF) и отдельную кнопку render для отрисовки отрендеренной сцены.

Как и в WF, в диалоге должна быть возможность менять n, m, a, b, c, d, zn, zf, sw, sh. Также должна быть возможность задавать показатель зеркальности и цвета материалов, цвета фона, рассеянного света и источников света (координаты источников необязательно – достаточно читать их из файла). Цвета хорошо бы задавать специальным контролом (см. javax.swing.JColorChooser)

Главное — делайте сперва алгоритм, а потом уже остальное! Отсутствие поддержки файлов и диалога при идеальном рендерере — это 4 балла, но при серьёзных недоработках рендерера наличие диалога и чтения файлов не спасает.

Формат файла

// В первой строке через пробел или табуляцию:

rbackgbackbback // цвет фона, все компоненты от 0 до 255

rambgambbamb // цвет рассеянного света

rl1 gl1 bl1 xl1 yl1 zl1 // цвет и положение первого источника света

rl2 gl2 bl2 xl2 yl2 zl2 // цвет и положение второго источника света

rl3 gl3 bl3 xl3 yl3 zl3 // цвет и положение третьего источника света. Чтобы выключить источник, просто задайте ему чёрный цвет. В программе вы можете проверить, если rlx=glx=blx=0, то источник не обсчитывать

rfdgfdbfdrfsgfsbfsnf // лицевой (front) материал — цвет рассеянный/диффузный (diffuse) и зеркальный (specular), показатель зеркальности

rbdgbdbbdrbsgbsbbsns // изнаночный (back) материал. Далее файл выглядит в точности как для WF

n m k a b c d zn zf sw sh

K // количество тел вращения

// Дальше описание отдельных тел

N // Количество опорных точек B-сплайна для образующей текущего тела

X1 Y1 // координаты первой точки

X2 Y2 // координаты второй точки

…

XNYN // координаты N-й точки

// потом второе тело и т. д.

Примерные вопросы на дифференцированный зачёт

1. Алгоритм дизеринга Флойда-Стайнберга.
2. Алгоритм Сазерленда-Ходжмана
3. Алгоритм Вейлера-Азертона
4. Алгоритм Брезенхэма для растеризации окружностей.
5. Алгоритм марширующих квадратов.
6. Преобразование видимого объёма к полукубу, матрица проецирования.

7. Оператор вращения вокруг оси в трёхмерном пространстве.
8. Сканирующий алгоритм Гуро.
9. Модель полного отражения по Фонгу.
10. Основные пространственные структуры данных.
11. Пересечение луча со сферой.
12. Формула преломлённого луча.
13. Методы выделения контуров.
14. Графические фильтры на основе свёртки.
15. Span-алгоритм заливки для 4- и 8-связных пиксельных областей.

Оценочные материалы по промежуточной аттестации, предназначенные для проверки соответствия уровня подготовки по дисциплине требованиям СУОС, хранятся на кафедре-разработчике РПД в печатном и электронном виде.

Лист актуализации рабочей программы
по дисциплине «Машинная графика»
по направлению подготовки 03.03.02 Физика
Профиль «Физическая информатика»

№	Характеристика внесенных изменений (с указанием пунктов документа)	Дата и № протокола Учёного совета ФФ НГУ	Подпись ответственного