

МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВГУ»)

УТВЕРЖДАЮ

Заведующий кафедрой
ПиИТ

наименование кафедры, отвечающей за реализацию дисциплины



проф. Махортов С.Д.,
подпись, расшифровка подписи
05.03.2025 г.

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ
Б1.В.ДВ.04.02 Избранные разделы 3D-графики

Код и наименование дисциплины в соответствии с учебным планом

1. Код и наименование направления подготовки/специальности:

09.04.02 Информационные системы и технологии

2. Профиль подготовки/специализация: Мобильные приложения и компьютерные игры

3. Квалификация выпускника: магистр

4. Форма обучения: очная

5. Кафедра, отвечающая за реализацию дисциплины: ПиИТ

6. Составители программы: Вахтин А.А., к.ф.-м.н., доц.

(ФИО, ученая степень, ученое звание)

7. Рекомендована: НМС ФКН протокол № 5 от 05.03.2025 г.

(наименование рекомендующей структуры, дата, номер протокола,

отметки о продлении вносятся вручную)

8. Учебный год: 2025/2026

Семестр(ы): 2

9. Цели и задачи учебной дисциплины

- сформировать понимание основных принципов обработки графической информации в игровых компьютерных системах;
- сформировать представление об основных технологиях в области компьютерной графики, специфичных для компьютерных игр;
- сформировать владение методами конструирования 3D-графических объектов для игровых приложений;
- выработать навыки использования графических библиотек;
- сформировать знание основных алгоритмов обработки графической информации;
- научить студентов профессионально проектировать игровые программные приложения с графикой;
- выработать практические навыки применения полученных знаний.

10. Место учебной дисциплины в структуре ООП: учебная дисциплина относится к части блока Б1, формируемой участниками образовательных отношений, дисциплина по выбору.

11. Планируемые результаты обучения по дисциплине/модулю (знания, умения, навыки), соотнесенные с планируемыми результатами освоения образовательной программы (компетенциями) и индикаторами их достижения:

Код	Название компетенции	Код(ы)	Индикатор(ы)	Планируемые результаты обучения
ПК-4	Способен проектировать архитектуру программного средства	ПК-4.1	Умеет определять состав компонентов программного средства	знать: Методы разработки графических 3D приложений в соответствии с принятыми стандартами и технологиями. Современные подходы в разработке приложений 3D-графики. Основные математические методы и алгоритмы 3D-графики. уметь: Обеспечивать соответствие проектирования и дизайна приложений 3D-графики принятым стандартам и технологиям. Разрабатывать программы 3D-графики с эффективным использованием ресурсов и дизайна ИС. владеть: Навыками разработки приложений 3D-графики. Самостоятельно проводить анализ и исследование и находить решение поставленной задачи. владеть: Основными алгоритмами и методами 3D-графики. владеть: Методами разработки графических 3D приложений в соответствии с принятыми

				стандартами и технологиями.
--	--	--	--	-----------------------------

12. Объем дисциплины в зачетных единицах/час — 5/180.

Форма промежуточной аттестации *зачет с оценкой*

13. Трудоемкость по видам учебной работы

Вид учебной работы		Трудоемкость		
		Всего	По семестрам	
			2 семестр	
Аудиторные занятия		56	56	
в том числе:	лекции	16	16	
	практические	0	0	
	лабораторные	32	32	
Самостоятельная работа		132	132	
в том числе: курсовая работа (проект)		0	0	
Форма промежуточной аттестации (экзамен – __ час.)		0	0	
Итого:		180	180	

13.1. Содержание дисциплины

п/п	Наименование раздела дисциплины	Содержание раздела дисциплины	Реализация раздела дисциплины с помощью онлайн-курса, ЭУМК
1. Лекции			
1.1	Основы компьютерной графики	Представление графических объектов в программе и на экране компьютера. Пиксель и вектор. Обзор технологий, применяемых для решения задач компьютерной графики.	
1.2	Математические методы, применяемые в компьютерной графике	Аффинные преобразования в пространстве, проекции, аппроксимация кривых и поверхностей.	
1.3	Алгоритмы визуализации трёхмерных тел	Удаление невидимых граней. Освещение, моделирование цвета. Метод трассировки лучей.	
1.4	Реализация методов 3D-моделирования	Аппроксимация поверхностей, заданных функциями. Метод шагающих кубов. Добавление или удаление вершин и граней многогранников, изменение координат вершин. Сечение 3D-объектов плоскостью. Алгоритмы композиций трёхмерных тел (объединение, пересечение, вычитание).	
1.5	Применение библиотеки OpenGL для изображения 3D-объектов	Трёхмерная графика в OpenGL. Трёхмерные преобразования. Цвет, освещение, свойства материала, текстура.	
2. Лабораторные занятия			
2.1	Основы компьютерной графики	Представление графических объектов в программе и на экране компьютера. Пиксель и вектор. Обзор технологий, применяемых для решения задач компьютерной графики.	
2.2	Математические методы, применяемые в	Аффинные преобразования в пространстве, проекции, аппроксимация кривых и поверхностей.	

	компьютерной графике		
2.3	Алгоритмы визуализации трёхмерных тел	Удаление невидимых граней. Освещение, моделирование цвета. Метод трассировки лучей.	
2.4	Реализация методов 3D-моделирования	Аппроксимация поверхностей, заданных функциями. Метод шагающих кубов. Добавление или удаление вершин и граней многогранников, изменение координат вершин. Сечение 3D-объектов плоскостью. Алгоритмы композиций трёхмерных тел (объединение, пересечение, вычитание).	
2.5	Применение библиотеки OpenGL для изображения 3D-объектов	Трёхмерная графика в OpenGL. Трёхмерные преобразования. Цвет, освещение, свойства материала, текстура.	

13.2. Темы (разделы) дисциплины и виды занятий

№ п/п	Наименование темы (раздела) дисциплины	Виды занятий (количество часов)				
		Лекции	Практические	Лабораторные	Самостоятельная работа	Всего
1.	Основы компьютерной графики	2	0	0	22	24
2.	Математические методы, применяемые в компьютерной графике	2	0	8	26	36
3.	Алгоритмы визуализации трёхмерных тел	4	0	8	28	40
4.	Реализация методов 3D-моделирования	4	0	8	28	40
5.	Применение библиотеки OpenGL для изображения 3D-объектов	4	0	8	28	40
	Итого:	16	0	32	132	180

14. Методические указания для обучающихся по освоению дисциплины

Студентам рекомендуется самостоятельно изучать рекомендованную литературу с целью анализа и поиска решений поставленных лабораторных задач.

15. Перечень основной и дополнительной литературы, ресурсов интернет, необходимых для освоения дисциплины

а) основная литература:

№ п/п	Источник
1.	Боресков А.В. Программирование компьютерной графики / А.В. Боресков. – М.: ДМК Пресс, 2019. – 372 с.
2.	Евченко, А. И. OpenGL и DirectX. Программирование графики / А.И. Евченко. — СПб : Питер, 2006. — 349 с.

б) дополнительная литература:

№ п/п	Источник
1.	Тюкачёв Н.А. С#. Программирование 2D и 3D векторной графики: учебное пособие / Н.А. Тюкачёв, В.Г. Хлебостроев. – 4-е изд., стер. – СПб.: Лань, 2020. – 320 с.
2.	Шикин, Е. В. Компьютерная графика. Полигональные модели : [учебное пособие] / Е. В. Шикин, А. В. Боресков. — М. : ДИАЛОГ-МИФИ, 2005. — 461 с.
3.	Жарков В. А. Компьютерная графика, мультимедиа и игры на Visual C# 2005 / В.А. Жарков. — М. : Жарков Пресс, 2005. — 811 с.

в) информационные электронно-образовательные ресурсы (официальные ресурсы интернет):

№ п/п	Ресурс
-------	--------

1.	https://lib.vsu.ru/?p=4&t=8
2.	Боресков А.В. Программирование компьютерной графики / А.В. Боресков. – М.: ДМК Пресс, 2019. – 372 с. (https://e.lanbook.com/book/131728)
3.	Тюкачёв Н.А. С#. Программирование 2D и 3D векторной графики: учебное пособие / Н.А. Тюкачёв, В.Г. Хлебостроев. – 4-е изд., стер. – СПб.: Лань, 2020. – 320 с. (https://e.lanbook.com/book/126152)

16. Перечень учебно-методического обеспечения для самостоятельной работы

№ п/п	Источник
1	Боресков А.В. Программирование компьютерной графики / А.В. Боресков. – М.: ДМК Пресс, 2019. – 372 с.
2	Евченко, А. И. OpenGL и DirectX. Программирование графики / А.И. Евченко. — СПб : Питер, 2006. — 349 с.

17. Образовательные технологии, используемые при реализации учебной дисциплины, включая дистанционные образовательные технологии (ДОТ), электронное обучение (ЭО), смешанное обучение):

На лабораторных и практических занятиях допускается использовать студентами любой язык программирования и любую среду программирования.

18. Материально-техническое обеспечение дисциплины:

Мультимедийная лекционная аудитория, компьютерный класс, программное обеспечение: среда программирования Visual Studio.

19. Оценочные средства для проведения текущей и промежуточной аттестаций

Порядок оценки освоения обучающимися учебного материала определяется содержанием следующих разделов дисциплины:

№ п/п	Наименование раздела дисциплины (модуля)	Компетенция(и)	Индикатор(ы) достижения компетенции	Оценочные средства
1.	Основы компьютерной графики	ПК-4	ПК-4.1	Задания лабораторных работ
2.	Математические методы, применяемые в компьютерной графике	ПК-4	ПК-4.1	Задания лабораторных работ
3.	Алгоритмы визуализации трёхмерных тел	ПК-4	ПК-4.1	Задания лабораторных работ
4.	Реализация методов 3D-моделирования	ПК-4	ПК-4.1	Задания лабораторных работ
5	Применение библиотеки OpenGL для изображения 3D-объектов	ПК-4	ПК-4.1	Задания лабораторных работ
Промежуточная аттестация форма контроля – зачет с оценкой				По результатам лабораторных работ

20 Типовые оценочные средства и методические материалы, определяющие процедуры оценивания

20.1 Текущий контроль успеваемости

Контроль успеваемости по дисциплине осуществляется с помощью контроля и оценки следующих лабораторных заданий:

20.1.1 Перечень лабораторных заданий

Лабораторная работа № 1.

1. Написать программу для построения многоугольников со скруглениями радиуса R . Программа должна позволять создавать любое количество фигур и управлять ими при помощи мыши (в т.ч. при создании указывать точки, после создания - перемещать точки)

2. Написать процедуру пересечения двух произвольных треугольников, заданных на плоскости. Программа должна позволять задавать параметры двух треугольников. Было бы замечательно, если бы треугольники можно было изменять с помощью мыши. Основной метод должен принимать два треугольника в качестве аргумента и возвращать упорядоченный набор точек для построения контура пересечения.

3. Написать программу построения графика функции, заданной на плоскости. Предусмотреть корректное отображение разрывов.

4. Написать программу динамического изменения графика функции на плоскости. (динамическое изменение задается ещё одной функцией от (t)).

5. Написать процедуру постепенного превращения одной кривой, заданной набором точек, в другую кривую, заданную другим набором точек. Пользователь задаёт две кривые с помощью мыши через опорные точки и время, за которое одна кривая должна превратиться в другую. После нажатия кнопки запуска происходит соответствующая анимация (программа должна каждый кадр вычислять "текущую" линию на основе двух граничных и текущего времени). Алгоритм рисования кривой реализовать самостоятельно. Например, кривую Безье или др. сплайны.

6. Создать приложение, демонстрирующее выполнение аффинных преобразований на плоскости, на примере фигуры -- ромб и (*) на примере эллипса (В качестве усложнённого варианта). Предполагается разработка гибкой архитектуры, где каждое преобразование будет наследоваться от какого-то базового интерфейса с объявленной функцией трансформации точки. Далее потребуется сделать инструмент, который бы прозрачно для основной программы, рисующей фигуру, применял эти преобразования к точкам перед их фактическим выводом на экран. Для этого потребуется создать интерфейс `IAffine`, в котором будет объявлен метод преобразования точки. После этого потребуется реализовать несколько классов для каждого преобразования. Далее можно сделать класс, который будет также наследоваться от `IAffine` и хранить в себе другие преобразования (паттерн компоновщик). Ромб стоит унаследовать от какой-то `IFigure`. Далее где-то написать класс/метод, который бы умел рисовать список фигур с применением указанного Аффинного преобразования. Таким образом, можно получить приложение, которому всё равно, какие фигуры рисуются и с применением каких преобразований (в т.ч. не будет важен порядок). В пользовательском интерфейсе должна быть возможность по настройке порядка и параметров применяемых преобразований.

7. Нарисовать обычную стрелку. Предусмотреть возможность управления параметрами стрелки (исходная и конечная точки, длина и ширина самой стрелки) с помощью мыши.

8. Прямоугольная спираль. Предусмотреть возможность изменения параметров спирали с помощью мыши.

9. Свечные диаграммы. Разработать приложение для вывода набора данных с помощью диаграммы. (*) Учесть изменение внешнего вида при изменении масштаба. Т.е. адаптировать график под текущий масштаб и область просмотра.

10. Пересечение или объединение прямоугольников. Разработать приложения для вычисления фигуры, являющейся результатом пересечения/(*)объединения нескольких прямоугольников, стороны которых параллельны осям координат. Должна быть возможность редактировать прямоугольники с помощью мыши. Метод, решающий задачу должен принимать в качестве аргумента список экземпляров класса Прямоугольник и возвращать список из замкнутых линий (каждая линия -- набор точек). В случае пересечения, конечно же, результатом будет один прямоугольник. В случае объединения -- могут получиться несколько замкнутых линий.

Лабораторная работа № 2.

1. Написать процедуру пересечения двух произвольных треугольников, заданных в трехмерном пространстве.

Пользователь задаёт координаты вершин треугольников в трёхмерном пространстве с помощью полей ввода. Для решения этой задачи необходимо разработать класс Triangle, два экземпляра которого передавать в функцию, решающую поставленную задачу и возвращающую список трёхмерных точек, являющихся вершинами контура пересечения. Если треугольники находятся в одной плоскости, то задача сводится к двумерной.

2. Написать программу построения тени от платонова тела, освещенного точечным источником.

Пользователь выбирает, какое именно платоново тело использовать, местоположение точечного источника света (просто точка в пространстве) и плоскость, на которой выводить тень (параметры A, B, C и D из канонического уравнения). Платоновы тела должны быть реализованы в виде отдельных классов, унаследованных от интерфейса IModel. Возможно, было бы удобно использовать один из порождающих паттернов проектирования для создания выбранного платонового тела.

3. Построить на экране изображения платоновых тел: тетраэдра, куба, октаэдра, икосаэдра, додекаэдра.

Реализовать в виде нескольких классов, унаследованных от IModel. Вершины должны генерироваться автоматически на основе заданного размера тела (радиуса описанной сферы), положения и (*)ориентации в пространстве. (*)Хотелось бы увидеть алгоритм генерации НЕ использующий кучу непонятных коэффициентов.

4. Написать программу, вращающую трёхмерное тело (на примере икосаэдра) вокруг заданной оси. Предусмотреть возможность выбора оси вращения (точка, принадлежащая оси и вектор-направление) и угловой скорости.

Конечно же, основной алгоритм вращения должен оперировать IModel'ями. В отличии от задачи 3 вращаться должна сама модель (оси координат остаются на экране неподвижными).

5. Написать программу освещения 3D-поверхности двумя источниками света.

Пользователь задаёт координаты точек положения источников света, а также их цвета с помощью соответствующего пользовательского интерфейса. Освещать стоит IModel. В качестве примеров поверхностей предлагается использовать куб, цилиндр и сферу. В рамках данной задачи потребуются учитывать некоторые параметры материала, из которого сделана поверхность. В простом случае - цвет материала. В более сложном - отражение и поглощение света.

6. Написать программу построения цилиндрических и конусных поверхностей.

Пользователь должен иметь возможность задавать основные параметры таких объектов: положение в пространстве, сдвиг (плоскость оснований не всегда перпендикулярна оси), внешние радиусы оснований (в случае нулевого значения одного из радиусов получается конус) и внутренние.

7. Написать программу изображения трехмерных тел с двухточечным проецированием.

Пользователь должен указывать координаты точек схода. В качестве модели стоит использовать IModel, а фактически продемонстрировать на примере куба, сферы и м.б. каких-либо других фигур.

Проецирование не должно привязываться к камере. Т.е. мы изменяем объект, а потом можем посмотреть на него с разных сторон.

8. Написать программу построения пересечения платоновых тел произвольной плоскостью.

Пользователь выбирает платоново тело, указывает параметры плоскости (коэффициенты A, B, C, D из канонического уравнения плоскости). В метод, решающий задачу, передаётся IModel и экземпляр класса Plane(плоскость), а в результате возвращается массив из получившихся частей - отдельных объектов. Каждый из которых можно вывести независимо друг от друга. Кроме того, было бы замечательно увидеть сам срез (плоский многоугольник).

(*) Можно развить данную задачу до пересечения любых трёхмерных тел плоскостью.

9. Написать программу построения ленты Мёбиуса.

Пользователь должен задавать основные её параметры, а после построения иметь возможность рассмотреть со всех сторон.

(*) Было бы интересно увидеть анимацию движения вдоль плоскости от первого лица.

10. Написать программу построения бутылки Клейна.

Пользователь должен задавать основные параметры, а после построения иметь возможность рассмотреть со всех сторон. Кроме того, было бы полезно иметь возможность указывать прозрачность стенок.

11. "Вокселизовать" трёхмерный объект и/или поверхность. Пользователь должен задавать размер кубиков.

На входе имеется IModel, а на выходе - объект, который состоит как бы из кубиков. Было бы замечательно иметь 2 реализации. Первая возвращает список кубиков (в т.ч. и внутренние), а вторая - единый большой объект, у которого поверхность имеет соответствующую форму.

12. Написать программу для бинарных операций над трехмерными телами.

В идеале, на вход подуются две IModel, а в результате получается результирующая модель, являющаяся объединением, пересечением или же вычитанием одной из другой.

Лабораторная работа № 3

1. Написать программу построения цилиндра вдоль заданной замкнутой линии. Примерами таких объектов могут выступать тор (цилиндр вдоль окружности) и Трилистник(узел).

(*) Интересным дополнением может служить построение заданного профиля вдоль незамкнутой линии (можно не ограничиваться прямыми отрезками).

2. Написать программу построения многоугольника вдоль заданной линии (в т.ч. замкнутой) с поворотом вокруг этой линии на заданный угол. Получается закрученная призма, (*)не обязательно построенная вдоль прямой.

В простом случае пользователь выбирает линию из какого-либо набора функций (для начала стоит сделать просто отрезок), а также задаёт параметры многоугольника (количество углов и размер, задаваемый через радиус описанной окружности).

3. Написать программу построения сферы с лучиками/иглами.

Желательно сделать в максимально общем виде, чтобы можно было задавать количество, размер и форму лучиков (не только конусы, но и цилиндры или призмы).

4. Написать программу построения модели болта и шурупа.

Пользователь должен задавать основные размеры, в т.ч. характеристики резьбы.

5. Написать программу построения трёхмерной решётки (из трубок/цилиндров).

Пользователь задаёт все параметры решётки: шаги по каждой из осей, отображаемую область, размер трубок. В простейшем случае трубки можно представить в виде обычных линий.

6. Написать программу построения зубчатого колеса со спицами.

Пользователь задаёт параметры зубчатого колеса: толщина, количество зубьев, все необходимые радиусы (их будет три).

7. Написать программу, позволяющую скруглить две и (*)три пересекающиеся плоские грани.

8. Написать программу построения спирали (пружины). Пользователь должен иметь возможность задавать основные параметры (положение в пространстве и размер пружины, шаг и направление витков, диаметр проволоки, из которой сделана пружина).

(*) Усложнённая версия включает в себя возможность сделать пружину с изменяемым шагом витков (можно реализовать, позволив пользователю указывать функцию, по которой будет меняться шаг или же интерполируя по ключевым точкам)

(*) Ещё один вариант усложнённой версии - возможность задавать изменяемый радиус (опять же с помощью указания функции).

9. Поверхности второго порядка. Создать инструмент для рисования поверхностей второго порядка. Проект должен позволять выбирать в трехмерном пространстве параллелепипед просмотра, перемещать и поворачивать его, изменять масштаб. Поверхность второго порядка может быть выбрана из полного списка таких поверхностей с возможностью поворота, смещения и растягивания по любой оси, или поверхность может быть задана уравнением второго порядка.

10. Линии уровня. Создать инструмент для рисования линий уровня. Этот проект должен позволять вводить функцию двух переменных либо в аналитическом виде, либо в табличном - значениями в узловых точках. Если функция вводится в аналитическом виде, то вам понадобится интерпретатор, на худой конец можно менять исходный текст и перетранслировать его. В случае табличного задания функции придётся применять подходящую двумерную интерполяцию.

Должна быть предусмотрена возможность изменять прямоугольник проектирования и число линий уровня. Рекомендуется кроме линий применить подкраску: определённому промежутку изменения значений функции поставить в соответствие некоторый цвет и этим цветом закрасить область между линиями уровня, так, как это делают на топографических картах (чем ниже - тем зеленее, чем выше - тем коричневее).

Описание технологии проведения

Оценка знаний, умений и навыков, характеризующая этапы формирования компетенций в рамках изучения дисциплины осуществляется в ходе текущей и промежуточной аттестаций.

Текущая аттестация проводится в соответствии с Положением о текущей аттестации обучающихся по программам высшего образования Воронежского государственного университета. Текущая аттестация проводится на основе оценки выполнения практических задач.

20.2 Промежуточная аттестация

Промежуточная аттестация по дисциплине осуществляется по итогам результатов лабораторных работ.

Промежуточная аттестация проводится в соответствии с Положением о промежуточной аттестации обучающихся по программам высшего образования. При оценивании используются качественные шкалы оценок.