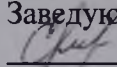


МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ «ВИТЕБСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ ИМЕНИ П.М. МАШЕРОВА»

Факультет математики и информационных технологий

Кафедра прикладного и системного программирования

Допущена к защите
«3» февраля 2020 г.
Заведующий кафедрой
 С.А.Ермоченко

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

**МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ ФИЗИЧЕСКИХ ЯВЛЕНИЙ В
ВИРТУАЛЬНОЙ СРЕДЕ**

Специальность 1-40 80 04 Математическое моделирование, численные методы и комплексы программ

Ковальчинский Максим Евгеньевич,
магистрант

Научный руководитель:
Корчевская Елена Алексеевна,
доцент

Витебск, 2020

**Учреждение образования «Витебский государственный университет
имени П.М. Машерова»**

Задание по подготовке магистерской диссертации студента
Ковальчинскому Максиму Евгеньевичу

Тема работы «Математическое моделирование физических явлений в виртуальной среде»

утверждена «1» сентября 2018 г.

протокол № 7 заседания Совета факультета математики и информационных технологий

1. Срок сдачи студентом законченной работы

2. Исходные данные для работы

Магистерская диссертация выполняется с учетом возможности использования результатов разработки во внутренних проектах ООО «Фабрика игр».

3. Перечень вопросов, подлежащих разработке в магистерской диссертации:

- Исследование и анализ современных подходов моделирования физических явлений;

- Создание прототипа библиотеки физических эффектов, соответствующей требованиям гибкости и масштабируемости;

4. Перечень графического материала

Демонстрационные изображения пользовательского интерфейса, листинги исходного кода.

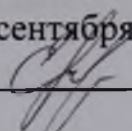
5. Консультанты по работе

Троицкий В.Н. – директор ООО «Фабрика игр»

Дата выдачи задания «1» сентября 2018 г.

Кафедра коррекционной работы

Утверждаю «1» сентября 2018 г.

Зав. кафедрой  С.А. Ермоченко

Научный руководитель _____ Е.А. Корчевская

Задание принял к исполнению М.Е.Ковальчинский

Реферат

Магистерская диссертация 33 с., 15 источников.

ТВЕРДОЕ ТЕЛО, МОДЕЛИРОВАНИЕ, ФИЗИЧЕСКОЕ ЯВЛЕНИЕ, ФИЗИЧЕСКИЕ СИМУЛЯЦИИ, ФИЗИЧЕСКИЕ БИБЛИОТЕКИ.

Цель исследования – разработка и исследование математической модели, описывающей физическое твердое тело.

Объект исследования: подходы к реализации математических моделей, описывающей физическое твердое тело, критерии применимости данных подходов.

Предмет исследования: математическое моделирование физических явлений.

Методы исследования: описательно-аналитический, сравнительно-сопоставительный.

Практическая значимость работы: задача математических моделей физических явлений является типичной проблемой, с которой сталкиваются конструкторы, инженеры, а также разработчики моделей виртуальной реальности. Задачей исследования является анализ существующих подходов к построению физической модели, а также поиск новых, или модификация существующих подходов.

Содержание

Введение	4
1. Теоретическая база исследования.....	9
1.1 Подходы к созданию физических движков.....	9
1.2 Пример реализации правдоподобной баллистики в играх	12
2. Процесс исследования и разработки	14
2.2. Принцип Д'Аламбера.....	14
2.3. Генераторы силы.....	15
2.4. Разрешение коллизий	17
Заключение	20
Список использованных источников	21
Приложение А	23
Приложение Б.....	25
Приложение В	26
Приложение Г.....	27
Приложение Д.....	28
Приложение Е	29
Приложение Ж	30
Приложение З.....	31
Приложение И.....	32

Введение

Физика - это огромная дисциплина, академическая физика имеет сотни областей. Каждый из них описывает какой-то аспект физического мира, будь то как работает свет или протекают ядерные реакции внутри звезды. Некоторые части физики могут быть полезны в играх. Например, можно использовать оптику, чтобы моделировать движение и отражение света, и использовать его для создания реалистичной графики. Так работает трассировка лучей и, хотя она все еще очень медленная, она использовалась в нескольких тайтлах. Другие разделы физики имеют более слабую связь с миром игр: невозможно использовать симуляцию ядерной физики в игре, если только ядерные реакции не были главной целью игрового процесса. Но это не то, что мы имеем в виду, когда мы говорим об игровой физике. Говоря о физике в игре, мы действительно имеем в виду классическую механику: законы, управляющие движением крупных объектов под действием силы тяжести и других сил. В академической физике эти законы были в значительной степени заменены новыми теориями: теорией относительности и квантовой механикой. В играх они используются, чтобы придать объектам ощущение целостности, массы, инерции, упругости и плавучести. Физика игры существует почти с тех пор, как были написаны первые игры. Впервые это было видно по движению частиц: искрам, огню, баллистике пуль, дыму и взрывам. Физическое моделирование также использовалось для создания авиационных симуляторов в течение почти трех десятилетий. Затем появилась автомобильная физика с постоянно растущей сложностью моделей шин, подвески и двигателя. Когда вычислительная мощность стала выросла, мы увидели ящики, которые можно было перемещать или укладывать в штабеля, стены, которые можно было разрушить и разбить на составляющие их блоки. Это и есть физика твердого тела, которая быстро расширяется и стала включать более мягкие объекты, такие как одежда, флаги и веревки.

Хотя физика присутствует в играх уже более тридцати лет, в последние годы произошли определенные изменения в способах реализации физики. Первоначально каждый эффект был запрограммирован сам по себе, создавая игру с только физикой, необходимой для определенной игры. Если в игре нужны стрелы для движения по траекториям, тогда уравнение траектории может быть задано в игре. Это было бы бесполезно для моделирования чего-либо, кроме траектории стрел, но свою функцию выполняет идеально.

Это подходит для простых симуляций, когда объем кода невелик, а область действия физики весьма ограничена. Реализация базовая системы частиц может занимать в сотни строк кода. Когда сложность возрастает, может быть трудно сразу получить правдоподобный физический эффект. Например, в оригинальной игре Half-Life вы можете перемещать ящики по кругу, но физический код не совсем верен, а способ перемещения ящиков выглядит странно. Сложность создания реалистичной физики в сочетании с необходимостью почти одинаковых эффектов в игре за игрой, побуждала разработчиков искать общие решения, которые можно было бы использовать повторно. Технология повторного использования должна быть достаточно гибкой: в баллистическом симуляторе, который будет работать только со стрелами, поведение стрел может быть жестко запрограммировано. Если тот же код должен справляться и с пулями, тогда программное обеспечение должно абстрагироваться от конкретных снарядов и моделировать общую физику, которая у них всех общая. Это то, что мы называем «физическим движком»: общий фрагмент кода, который знает физику в целом, но не завязан на специфику сценария каждой игры. Здесь есть очевидная проблема. Если у нас был специальный код для симуляции стрелы, то нам больше ничего не нужно для симуляции. Если у нас есть общий физический движок для моделирования любого снаряда, и мы хотим смоделировать стрелу, нам также нужно сообщить движку характеристики того, что мы моделируем. Нам нужны физические свойства стрел, пуль или ящиков и так далее. И это очень важно. Физический

движок отлично справляется с математическими вычислениями, но он не знает, что нужно моделировать. В дополнение к движку нам также нужны данные, специфичные для игры, которые представляют игровой уровень. [1]

Есть два убедительных преимущества использования физического движка в играх. Во-первых, это экономия времени. Если предполагается использовать физические эффекты более чем в одной игре, то усилия по созданию физического движка окупаются, когда его можно просто импортировать в каждый новый проект. Легкая физическая система общего назначения не должна быть сложной для программирования. Для большинства необходимых игровых эффектов будет настроено несколько тысяч строк кода.

Вторая причина - качество. Было бы разумно включать в игру все больше и больше физических эффектов. Вы можете реализовать каждый из них так, как вам нужно: создать симулятор ткани для накидок и флагов, симулятор воды для плавучих ящиков и отдельный движок частиц. Каждый из них может работать отлично, но комбинировать эти эффекты будет сложно. Когда персонаж с плащом окажется в воде, как поведет себя его одежда? Если плащ продолжает развеиваться на ветру даже под водой, иллюзия будет испорчена. Физический движок дает вам возможность взаимодействовать с эффектами правдоподобно. Ящики в Half-Life составили основу только одной или двух головоломок в игре. Когда дело дошло до Half-Life 2, физика ящиков была заменена полным физическим движком. Это дает много новых возможностей. Куски разбитого ящика плавают на поверхности воды; объекты могут быть сложены, использованы как подвижные щиты и так далее. Нелегко создать физический движок, чтобы справиться с водой, ветром и одеждой, но это гораздо проще, чем пытаться взять три отдельных фрагмента кода и заставить их работать вместе во всех ситуациях.

Это не значит, что физический движок - это панацея. Есть причины, по которым использование полного физического движка нецелесообразно. Наиболее распространенной причиной является скорость. Физический движок

общего назначения довольно сильно нагружает процессор. Поскольку он должен быть гибким, он не может делать предположений о типах объектов, которые он моделирует. Когда вы работаете с очень простой игровой средой, это обобщение означает потерю вычислительной мощности. Это не столько проблема современных консолей или ПК, но на портативных устройствах, таких как телефоны и КПК, это может быть фатально. Можно создать игру, используя полный физический движок на ПК, но та же игра на мобильном телефоне будет работать быстрее с некоторой специализированной физикой. Необходимость предоставления движку данных также может быть серьезной проблемой. Еще одна причина, по которой следует избегать физических движков - это условия разработки. Если команда разработчиков небольшая, то для разработки полного физического решения может потребоваться время на улучшение других аспектов игры: например, графики или игрового процесса. С другой стороны, даже любительские игры должны конкурировать с коммерческими играми за внимание, а физика высшего качества является обязательным требованием для игры любого качества. [2]

Этим и обусловлена полезность и актуальность исследований методов моделирования математических моделей, описывающей физическое твердое тело, создания программного обеспечения, упрощающего разработку симуляций.

Цель написания магистерской диссертации состоит в разработке и исследовании математической модели, описывающей физическое твердое тело.

Достижение поставленной цели предполагает постановку и решение следующих задач:

- анализ существующих подходов к построению физической модели
- выделить общие черты и отличия этих методов;
- формулирование требований к построению и модификации физической модели;
- произвести реализацию модифицированного подхода построения

математической модели пружин, плавучих тел и соударений

- проанализировать полученный результат на предмет соответствия выдвинутым требованиям

Список использованных источников

- 1) I. Millington. *Game Physics Engine Development*, Morgan Kaufmann publications, 2007.
- 2) C. Ericson. *Real-Time Collision Detection*, Morgan Kaufmann publications, 2004.
- 3) D. h. Eberly. *Game Physics*, Morgan Kaufmann publications, 2010.
- 4) Э. Фримен. *Паттерны Проектирования*, O`Reilly, 2011.
- 5) Typescript Documentation. [Электронный ресурс]. – Режим доступа: <http://www.typescriptlang.org>– Дата доступа: 20.12.2019.
- 6) PixiJS Examples. [Электронный ресурс]. – Режим доступа: <http://www.pixijs.io/#/demos-basic> – Дата доступа: 5.12.2019.
- 7) Линейная алгебра для разработчиков игр. [Электронный ресурс]. – Режим доступа: [http:// habr.com/ru/post/131931](http://habr.com/ru/post/131931)– Дата доступа: 15.12.2019.
- 8) Физические объекты и явления. [Электронный ресурс]. – Режим доступа: [http:// wikipedia.org/wiki/Категория:Физические_эффекты_и_явления](http://wikipedia.org/wiki/Категория:Физические_эффекты_и_явления) – Дата доступа: 18.12.2019.
- 9) Основы кинематики. [Электронный ресурс]. – Режим доступа: http://k-a-t.ru/tech_mex/22-dinamika_4 – Дата доступа: 18.12.2019.
- 10) Руководство по Typescript. [Электронный ресурс]. – Режим доступа: <http://matanit.com/web/typescript> – Дата доступа: 18.12.2019.
- 11) Физический движок. [Электронный ресурс]. – Режим доступа: [http:// wikipedia.org/wiki/ Физический_движок](http://wikipedia.org/wiki/Физический_движок)– Дата доступа: 3.12.2019.
- 12) Физический движок изнутри. [Электронный ресурс]. – Режим доступа: [http:// tproger.ru/ translations/whats-in-a-projectile-physics-engine-part-1](http://tproger.ru/translations/whats-in-a-projectile-physics-engine-part-1) – Дата доступа: 13.01.2019.
- 13) Learning Pixi. [Электронный ресурс]. – Режим доступа: <https://github.com/kittykatattack/learningPixi> – Дата доступа: 23.12.2019.

14) Generator. [Электронный ресурс]. – Режим доступа: [https://en.wikipedia.org/wiki/Generator_\(computer_programming\)](https://en.wikipedia.org/wiki/Generator_(computer_programming)) – Дата доступа: 18.12.2019.

15) Façade Pattern. [Электронный ресурс]. – Режим доступа: <https://refactoring.guru/design-patterns/facade> – Дата доступа: 10.02.2019.