Федеральное государственное бюджетное образовательное учреждение
высшего образования
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ

На правах рукописи

Кочергин Максим Игоревич

МЕТОДИКА И АЛГОРИТМЫ ВИЗУАЛЬНОГО МОДЕЛИРОВАНИЯ НЕПРЕРЫВНЫХ И ДИСКРЕТНО-НЕПРЕРЫВНЫХ ФИЗИКО-ТЕХНИЧЕСКИХ ЗАДАЧ МЕТОДОМ КОМПОНЕНТНЫХ ЦЕПЕЙ

05.13.18 – Математическое моделирование, численные методы и комплексы программ

Диссертация
на соискание учёной степени
кандидата технических наук

Научный руководитель
доктор технических наук, профессор
Дмитриев Вячеслав Михайлович

Томск 2019

Оглавление

Список сокращений 5
Введение 6
Глава 1. Компьютерное моделирование физико-технических задач 15
1.1 Физико-технические задачи как объект моделирования 15
1.1.1 Физические, технические и физико-технические задачи 15
1.1.2 Классификация физико-технических задач 20
1.1.3 Математическое описание класса решаемых задач 22
1.2 Инструментальные средства моделирования физико-технических задач 24
1.2.1 Обзор систем компьютерного моделирования физико-технических задач 24
1.2.2 Примеры компьютерных моделей 29
1.3 Метод многоуровневых компонентных цепей для моделирования технических систем 36
1.3.1 Назначение, основные понятия метода многоуровневых компонентных цепей 36
1.3.2 Многоуровневый подход к представлению компьютерных моделей 38
1.3.3 Язык моделирования многоуровневых компонентых цепей 40
Выводы по Главе 1 45
Глава 2. Развитие языка многоуровневых компонентных цепей для моделирования физико-технических задач 46
2.1 Алгоритм моделирования физико-технических задач 46
2.1.1 Декомпозиция задачи и синтез структуры многоуровневой компьютерной модели 49
2.1.2 Построение многоуровневой компьютерной модели 54
2.2 Моделирование дискретного поведения объектов в физико-технических задачах 57
2.2.1 Традиционные подходы к моделированию дискретно-непрерывных систем 57
2.2.2 Интерпретация диаграмм состояний в язык метода многоуровневых компонентных цепей 63

2.2.3 Компенсация амплитудно-временной погрешности 71
2.3 Моделирование непрерывного поведения объектов в физико-технических задачах 85
2.3.1 Отображение геометрических свойств объектов в многоуровневой компьютерной модели 85
2.3.2 Отображение физических свойств в компьютерных моделях 93
2.3.3 Обработка табличных результатов моделирования 100
Выводы по главе 2 108
Глава 3. Комплекс программ для компьютерного моделирования физико-технических задач 110
3.1 Библиотека моделей компонентов для моделирования физико-технических задач 111
3.2 Программный модуль обучения компьютерному моделированию физико-технических задач на основе анализа словесного портрета 114
3.2.1 Назначение программного модуля 114
3.2.2 Структура и функции программного модуля обучения моделированию 115
3.2.3 Указания к применению программного модуля обучения моделированию 133
3.3 Информационная система управления виртуальной лабораторией моделирования физико-технических задач 136
3.3.1 Назначение и структура программного модуля 138
3.3.2 Описание интерфейса программного модуля 140
3.4 Сравнение с аналогами 142
Выводы по Главе 3 144
Глава 4. Моделирование физико-технических задач методом компонентных цепей 146
4.1 Непрерывная модель полёта тела в атмосфере Земли 146
4.2 Моделирование усилий на полированном штоке штангового глубинного насоса 153

4.3 Моделирование отскоков упругого тела от поверхности со сложным профилем 160
4.4 Аппроксимация результатов моделирования 162
Выводы по Главе 4 166
Заключение 167
Список литературы 169
Приложение А. Примеры моделей задач 186
Приложение Б. Диаграммы классов комплекса программ 217
Приложение В. Акты внедрения результатов исследования 238
Приложение Г. Свидетельства о регистрации программ для ЭВМ 241

СПИСОК СОКРАЩЕНИЙ

MFC	Microsoft Foundation Classes
UML	Unified Modeling Language
ВИП	виртуальные инструменты и приборы
ГА	гибридные автоматы
ГС	гибридная система
ИМАП	интерактивная математико-алгоритмическая панель
ИМП	интерактивная математическая панель
ИСУЛ	информационная система управления лабораторией
ИЭ	информационные элементы
КЦ	компонентная цепь
МАК	моделирование алгоритмических конструкций
МКМ	многоуровневая компьютерная модель
МКЦ	метод компонентных цепей
ММКЦ	метод многоуровневых компонентных цепей
МНК	метод наименьших квадратов
МО	модель отношений
ОМ	объектная модель
ПВП	панели виртуальных приборов
ПО	предметная область
САО	среднее абсолютное значение отклонения
СДО	система дистанционного обучения
СКО	среднеквадратичное отклонение
СЛАУ	система линейных алгебраических уравнений
СМ	среда моделирования
ССИД	слабо структурированные исходные данные
ТЕ	текстовые единицы
ТЗ	техническая задача
УИМ	учебно-иллюстративный модуль
ФЗ	физическая задача
ФТЗ	физико-технические задачи
ХСИД	хорошо структурированные исходные данные
ХТС	химико-технологические системы
ШГН	штанговый глубинный насос

ВВЕДЕНИЕ

Широкое распространение в различных областях физики и техники для исследовательских, образовательных целей или в рамках задачи функционального проектирования находят физико-технические задачи (ФТЗ). ФТЗ характеризуются отсутствием априорной информации о структуре модели: так, например, при моделировании технических систем заранее известна и в достаточной мере формально представлена информация о структуре будущей модели некоторого искусственного устройства, в то время как структура модели физической системы формируется в ходе процессов осмысления, формализации и непосредственно моделирования и при этом может изменяться (ввиду процедур упрощения или усложнения модели, учёта новых факторов и пр.). ФТЗ представляет собой постановку некоторой проблемы о техническом (искусственном) объекте, решаемую с помощью различных знаний из специализированных разделов физики и имеющую определенное практическое или исследовательское значение. В данной работе рассматриваются ФТЗ о сложных динамических объектах. Под сложностью понимается 1) сложность поведения, 2) сложность структуры, 3) переменный, зависящий от времени и режимов, состав моделируемых объектов, 4) взаимодействие объектов различной природы (например, техническое устройство с дискретным поведением и физический объект с непрерывным поведением) [Сениченков, 2005].

Исследованию сложных динамических систем в целом посвящены работы Ю.Б. Колесова, Ю.Б. Сениченко, Ю.В. Шорникова, А.В. Бессонова, P. Mosterman, F. Cellier, C.W. Gear. Моделированию физических задач с применением метода компонентных цепей в аспекте образовательной деятельности посвящены работы А.Ю. Филиппова, О.Н. Шаровой. В работах Е.А. Арайса, В.М. Дмитриева, А.В. Шутенкова, Т.Н. Зайченко заложены и развиты основы метода компонентных цепей (МКЦ), которые получили развитие в работе Т.В. Ганджи в виде метода многоуровневых компонентных цепей (ММКЦ), направленного на моделирование сложных технических

устройств и систем и предполагающего декомпозицию компьютерной модели системы любой природы на 3 уровня: объектный, логический (алгоритмический) и визуальный. Эта концепция декомпозиции отвечает изменчивой структуре ФТЗ и соответствует разделению поведения физических систем на непрерывное (физическое), дискретное (логическое или алгоритмическое) и визуальное. Однако ММКЦ на текущий момент (как и среда моделирования (СМ) МАРС – комплекс программ, реализующий данный метод) не имеет инструментов для эффективного моделирования ФТЗ, методики формализации таких задач и алгоритма построения их моделей.

Необходимо отметить, что в практике моделирования ФТЗ в качестве инструмента, как правило, применяются математические пакеты, языки программирования, табличные процессоры, специализированные САПР, не отвечающие в полной мере специфике решения ФТЗ.

В связи с вышеизложенным **актуальным** является развитие и адаптация ММКЦ к моделированию технических систем для моделирования ФТЗ (обладающих как непрерывным, так и дискретно-непрерывным поведением и включающих геометрическую составляющую), формирование методики их многоуровневого моделирования, а также разработка программных средств для компьютерного моделирования ФТЗ и обучения основам их компьютерного моделирования.

Цель работы – разработка методики, алгоритма и программных средств моделирования и для исследования непрерывных и дискретно-непрерывных ФТЗ, а также реализация на его основе комплекса программ для моделирования и обучения моделированию физико-технических задач.

Для достижения поставленной цели были решены следующие задачи:

1. Описание и классификация ФТЗ и средств их моделирования.

2. Разработка алгоритма многоуровневого компьютерного моделирования ФТЗ, включающего этапы формализации моделируемой задачи и её многоуровневой декомпозиции на объектный, логический (алгоритмический) и визуальный уровни.

3. Разработка средств моделирования для отображения физических свойств объектов, геометрических свойств их межобъектных связей в ФТЗ, а также инструментов для моделирования дискретно-непрерывного (гибридного) поведения объектов с реализацией механизма компенсации накапливаемой амплитудно-временной погрешности.

4. Разработка на базе поисковых методов оптимизации метода численной аппроксимации для приближения табличных результатов моделирования ФТЗ функцией произвольного вида.

5. Разработка комплекса программ для многоуровневого моделирования ФТЗ на базе СМ МАРС, включающего библиотеку моделей компонентов для моделирования ФТЗ, программный модуль для самостоятельного обучения моделированию ФТЗ и информационную системы управления виртуальной лабораторией моделирования ФТЗ для сопровождения аудиторных занятий по моделированию.

Объектом исследования являются дискретно-непрерывное поведение объектов, их физические свойства и геометрические свойства их связей в ФТЗ.

Предметом исследования являются модели дискретно-непрерывного поведения объектов, их физических свойств и геометрических свойств их связей в ФТЗ и алгоритмы их исследования.

Научная новизна работы:

1. В области математического моделирования. Предложен алгоритм моделирования ФТЗ, отличительной особенностью которого является возможность автоматизации перевода естественноязыкового представления модели в формальное представление ММКЦ. Разработаны единицы языка многоуровневых компонентных цепей для моделирования физических свойств, гибридного поведения (с применением диаграмм состояний) объектов и геометрических свойств их межобъектных связей в ФТЗ, отличающиеся от существующих более высоким уровнем абстракции. Предлагаемые диаграммы состояний отличаются от других средств моделирования гибридного поведения тем, что представляют собой самостоятельный элемент имитационной модели

объекта, отдельный от его аналитической модели, а также реализуют механизм компенсации накапливаемой амплитудно-временной погрешности.

2. В области численных методов. Предложен численный метод аппроксимации результатов моделирования ФТЗ, базирующийся на решении задачи аппроксимации комбинацией поисковых методов и отличающийся от существующих тем, что позволяет находить коэффициенты приближающей функции произвольного вида (не требуя при этом аналитических преобразований), обеспечивающие глобальный, а не только локальный минимум суммы квадратов отклонения аппроксимирующей функции от аппроксимируемой. Разработан численный алгоритм компенсации амплитудно-временной погрешности, отличающийся от существующих тем, что заключается в решении задачи обратной интерполяции и обеспечивает неитерационный расчёт точки перехода дискретного состояния.

3. В области комплексов программ. Предложен алгоритм функционирования диаграмм состояний, отличающийся тем, что применим для математических моделей, представленных не только в аналитическом виде, но и схемотехническом и структурном. Впервые разработан алгоритм формализации текстовых условий ФТЗ, позволяющий автоматизировать процедуру перевода естественноязыкового описания задачи в компонентное описание в соответствии с формализмом ММКЦ.

Теоретическая значимость работы определяется вкладом в развитие общей теории моделирования, а именно развитием метода многоуровневых компонентных цепей. Произведена интерпретация диаграмм состояний в язык моделирования алгоритмических конструкций ММКЦ, что в отличие от других инструментальных средств позволяет строить модели гибридных систем, представленные как в блочно-символьном (аналитическом) виде, так и в компонентном виде. Впервые на основе решения задачи обратной интерполяции предложен алгоритм компенсации накапливаемой амплитудно-временной погрешности, возникающей при смене состояний дискретно-непрерывной системы. Впервые предложен численный метод аппроксимации, основанный на

комбинации поисковых методов оптимизации, позволяющий находить глобальный минимум отклонения аппроксимируемой функции от аппроксимирующей за счёт исследования ряда локальных минимумов.

Практическая значимость.

1. Разработанные библиотека моделей компонентов для моделирования ФТЗ и компьютерная модель штангового глубинного насоса, построенная с её использованием, были использованы в АО «Энергонефтемаш» для определения оптимальных эксплуатационных характеристик нефтедобывающей установки и построения алгоритма работы системы управления штанговым глубинным насосом.

2. Результаты работы (комплекс программ, алгоритмы моделирования ФТЗ, численный алгоритм аппроксимации) использованы в учебном процессе ФГБОУ ВО «Томский государственный университет систем управления и радиоэлектроники» при преподавании дисциплин «Компьютерное моделирование физических задач», «Основы компьютерного моделирования физико-технических задач», «Моделирование систем», а также при организации работы учебной лаборатории «Элементы и устройства робототехнических систем».

Результаты работы применялись в следующих НИР:

1. «Исследование и разработка интеллектуальной системы управления штанговым глубинным насосом для поддержания оптимального динамического уровня жидкости в нефтяной скважине» (соглашение № 14.574.21.0157 от 26 сентября 2017 г.) – по федеральной целевой программы «Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2014 – 2020 гг.».

2. «Разработка программных средств автоматической параметризации компьютерных моделей эколого-экономических систем предприятий нефтегазовой промышленности», грант РФФИ №16-37-00027, 2016–2017 г.

Обоснованность и достоверность результатов научного исследования обеспечивается глубоким анализом работ в данной тематике, корректностью

использованных методов и подтверждается сравнением качества и точности численного моделирования с применением разработанного комплекса программ с результатами моделирования в других инструментальных средах (*Simulink, Rand Model Designer*, *Mathcad*, *Matlab*) и успешным решением практических задач.

Методы исследования. В работе применялись методы теории систем, компьютерного моделирования, численные методы оптимизации, методы теории конечных автоматов, теории гибридных автоматов. При разработке комплекса программ использовались методы объектно-ориентированного программирования и проектирования программного обеспечения.

На защиту выносятся следующие положения:

1. Разработанные алгоритм многоуровневого моделирования ФТЗ и методика многоаспектного анализа ФТЗ детерминируют их приведение к формализму ММКЦ для дальнейшего построения моделей в СМ МАРС, а также позволяют автоматизировать процедуру перевода словесного описания ФТЗ в формальное. Разработанные единицы языка ММКЦ позволяют строить модели ФТЗ из блоков высокого уровня абстракции, снижая при этом размерность аналитической модели, решаемой вычислительным ядром СМ МАРС, вследствие предварительного решения части уравнений модели имитационным ядром. (*Область исследований: 1. Разработка новых математических методов моделирования объектов и явлений*).

2. Интерпретированные в язык ММКЦ диаграммы состояний позволяют моделировать ФТЗ, математические модели которых представляются не только в символьном виде, но и в схемотехническом и структурном, обеспечивая при этом компенсацию амплитудно-временной погрешности моделирования, возникающей и накапливающейся при смене дискретных состояний гибридного поведения объекта. (*Область исследований: 2. Развитие качественных и приближенных аналитических методов исследования математических моделей*).

3. Разработанный численный метод аппроксимации на базе поисковых

методов оптимизации позволяет аппроксимировать табличные результаты моделирования ФТЗ приближающими функциями произвольного вида, обеспечивая выбор глобального минимума отклонения значений приближающей функции от аппроксимируемой из ряда локальных минимумов. (*Область исследований: 4. Реализация эффективных численных методов и алгоритмов в виде комплексов проблемно-ориентированных программ для проведения вычислительного эксперимента*).

4. Разработанный комплекс программ состоит из 3 блоков, нацеленных как непосредственно на моделирование ФТЗ, так и на обучение их моделированию. *Библиотека компонентов* позволяет строить наглядные и адекватные визуальные модели непрерывных и дискретно-непрерывных ФТЗ из компонентов высокого уровня абстракции. *Программный модуль обучения* формирует (за счёт использования методов автоматического анализа текста) пользовательские справочные инструкции о процедуре формализации и декомпозиции словесного портрета введённой в него ФТЗ при внеаудиторной работе учащихся, направленной на развитие навыков анализа и моделирования задач. *Информационная система управления лабораторией моделирования* сопровождает аудиторные занятия студентов и организует их взаимодействие с электронными курсами и средой моделирования. (*Область исследований: 8. Разработка систем компьютерного и имитационного моделирования*).

Апробация результатов. Основные положения данного исследования докладывались и обсуждались на 18 следующих международных и всероссийских конференциях: Международная научно-практическая конференция «Электронные средства и системы управления» (г. Томск, 2015, 2016, 2017, 2018), Международная конференция студентов и молодых учёных «Перспективы развития фундаментальных наук» (г. Томск, 2016, 2017, 2018, 2019), Международная научно-техническая конференция студентов, аспирантов и молодых учёных «Научная сессия ТУСУР» (г. Томск, 2016, 2017, 2018, 2019), Открытая выставка научных достижений молодых учёных ТУСУРа «РОСТ.up» (г. Томск, 2016), Международная научно-методическая конференция

«Современное образование: развитие технологий и содержания высшего профессионального образования как условие повышения качества подготовки выпускников» (г. Томск, 2017), 55-я Международная Научная Студенческая Конференция (г. Новосибирск, 2017), XIV Международная научно-техническая конференции «Новые информационные технологии и системы» (г. Пенза, 2017), 17-ая Международная молодежная научно-практическая конференция «Моделирование, фундаментальные исследования, теория, методы и средства» (г. Новочеркасск, 2017), Международная научно-методическая конференция «Современное образование: повышение профессиональной компетентности преподавателей вуза – гарантия обеспечения качества образования» (г. Томск, 2018).

Публикации по теме диссертации. Основные результаты исследования изложены в 32 публикациях, из них 3 статьи в ведущих рецензируемых научных журналах, рекомендованных ВАК. Получено 2 свидетельства о регистрации программ для ЭВМ.

Личный вклад автора. Постановка цели и задач исследования осуществлялась совместно с научным руководителем. Основные результаты диссертационной работы получены автором лично. Совместно с научным руководителем и Ганджой Т.В. осуществлялась разработка информационной системы управления лабораторией [Кочергин, Дмитриев, Ганджа, 2019].

Структура работы. Диссертация состоит из введения, четырёх глав, заключения, списка используемой литературы из 140 наименований и 4 приложений. Общий объём основной части составляет 185 страниц и включает 101 рисунок и 12 таблиц.

В Главе 1 ФТЗ рассматривается как объект моделирования, проводится обзор инструментальных сред для моделирования ФТЗ, закладываются теоретико-методологические основы исследования.

В Главе 2 предлагается разработанный алгоритм моделирования ФТЗ методом МКЦ, описываются разработанные средства для моделирования физических, геометрических свойств объектов и их дискретно-непрерывного

поведения в ФТЗ.

В Главе 3 описывается комплекс разработанных программ для моделирования ФТЗ и обучения их компьютерному моделированию, исследуются возможности автоматизации формализации текстовых условий ФТЗ – их перевода из естественноязыкового представления в информационную модель согласно формализма ММКЦ.

В Главе 4 приводятся несколько примеров моделей ФТЗ, построенных в СМ МАРС с применением разработанного комплекса программных инструментов, остальные примеры моделей вынесены в приложение А.

В приложении Б приводится описание разработанного комплекса программ в нотациях языка UML.

В приложении В представлены копии актов внедрения результатов исследования.

В приложении Г приведены свидетельства о регистрации программ для ЭВМ.

ГЛАВА 1. КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ ФИЗИКО-ТЕХНИЧЕСКИХ ЗАДАЧ

1.1 ФИЗИКО-ТЕХНИЧЕСКИЕ ЗАДАЧИ КАК ОБЪЕКТ МОДЕЛИРОВАНИЯ

1.1.1 ФИЗИЧЕСКИЕ, ТЕХНИЧЕСКИЕ И ФИЗИКО-ТЕХНИЧЕСКИЕ ЗАДАЧИ

Введём следующие базовые определения для определения понятийного аппарата исследования.

Задача – это проблемная ситуация с явно заданной целью [Филиппов, 2007]. Задача состоит из условия (исходного предмета задачи – той информации, которая известна о проблемной ситуации) и требования (указаний к требуемому результату). В более узком смысле иногда под задачей понимают саму цель рассматриваемой проблемной ситуации.

Физическая задача (ФЗ) – постановка некоторой проблемы из предметной области «Физика», решаемая с помощью математических действий, логических умозаключений или эксперимента. Физические задачи как правило не имеют прикладной значимости и представляют интерес только для образовательных целей. Примером физической задачи может служить формулировка задачи физики из любого раздела соответствующего задачника, например, такая задача из раздела «Импульс»: «Человек массой 60 кг бежит со скоростью 7,2 км/ч, догоняет тележку массой 80 кг, движущуюся со скоростью 1,8 км/ч, и вскакивает на нее. Найти, с какой скоростью будет двигаться тележка». Классифицироваться ФЗ могут, например, по разделам физики (механика, молекулярная физика и т.д.) или по характеру поведения объектов относительно переменной «время» (статические и динамические задачи). Более подробная классификация будет представлена ниже.

Физико-технические задачи (ФТЗ) – постановка некоторой проблемы о техническом (искусственном) объекте, решаемая с помощью различных знаний из специализированных разделов физики и имеющая определенное практическое или исследовательское значение. Исследователи [Клишкова, 2011] отмечают, что

такие задачи представляют интерес в том числе и в образовании: умения решать физико-технические проблемы занимает важное место при освоении принципов действия современных технических устройств. Понятие ФТЗ может быть связано с понятием физико-технической проблемы – проблемы «достижения технических результатов, решение которых требует получения и применения фундаментальных знаний» [Клишкова, 2011].

В качестве примера ФТЗ приведём следующие задачи: а) задачу внешней баллистики о полёте снаряда в атмосфере Земли, которая может рассматриваться на различных уровнях абстракции (в случае построения простейшей модели полёта тела – идеального полёта, – эта задача может быть отнесена к классу ФЗ, а не ФТЗ); б) задача о поле земной волны (распространении электромагнитного поля), рассматриваемая в работе [Балханов, 2007]; задача моделирования усилий на полированном штоке штангового глубинного насоса, которая рассматривается в параграфе 4.2 данной работы.

Отметим, что исследователи не дают чёткого разделения физических и физико-технических задач, так, например, в работах [Низамов, 1980; Мусабеков, 2017; Аладьев, Богдявичюс, Хунт, 1999] вводится такой класс задач как физические задачи с техническим содержанием, которые обладают как признаками ФЗ, так и ФТЗ.

Обозначим отличие ФТЗ от технических задач. Техническая задача (ТЗ) – *инженерная* задача о техническом объекте (искусственном устройстве), решаемая с помощью знаний из различных наук (например, химии и биологии, а не только физики). Процесс решения любой технической задачи включает 4 основных этапа: 1) постановка задачи, 2) поиск вариантов решения, 3) анализ вариантов решения, 4) оценка вариантов и выбор решения [Боголюбова, Скроботова, Федоров, 2007]. Каждую техническую задачу упрощенно можно представить себе состоящей из трех основных компонентов: начального состояния, конечного результата и процесса превращения первого в последний. Эти компоненты могут быть известными, заданными или неизвестными, искомыми [Буш, 1976]. Если один или два компонента задачи даны или заданы

через какое-либо отношение между ними, а оставшиеся компоненты неизвестны, то задача является ***изобретательской***, решаемой в условиях дефицита исходной информации. Приведём следующий пример технической задачи: «Ведущий вал вращается со скоростью от 400 до 4000 об/мин. Ведомый вал должен постоянно иметь 400 об/мин. Как это осуществить?» [Голдовский, Вайнерман, 1990].

Ввиду того, что определение ТЗ даётся через термин «инженерная задача», поясним и его. Инженерная задача – это задача преобразования или перехода объекта из исходного состояния в требуемое конечное при наличии объективных ограничений: технических, технологических, энергетических, информационных, по материальным ресурсам и т.д. [Бояршинова, Фишер, 2006]. Инженерной задача может считаться только тогда, когда существует несколько альтернативных путей ее решения и инженеру нужно выбрать из этих путей наиболее предпочтительный, удовлетворяющий сформулированным условиям и ограничениям.

Таким образом, суммируя вышеописанное, можно выделить следующие отличительные особенности ФТЗ: 1) в отличие от ТЗ она не представляется в виде трёх компонентов (начальное состояние, процесс превращения и конечный результат) – её поведение определяется структурой поведения исследуемого объекта и не известна заранее, а формируется в ходе построения её представления (модели) и зависит от уровня детализации и степени абстракции исследователя; 2) ФТЗ в отличие от ТЗ является не инженерной задачей, а, в первую очередь, исследовательской; 3) ФТЗ, в отличие от ФЗ, имеет некоторое прикладное или исследовательское значение, так как являются задачами из области техники; 4) ФТЗ, в отличие от ФЗ, представляет собой более сложные задачи по характеру поведения объектов и структуре их связей. Приведём также и особенности общие для рассматриваемых классов: 1) областью знаний (предметной областью) для ФЗ и ФТЗ является физика; 2) объектной областью для ФТЗ и ТЗ является техника.

Особенности рассматриваемых физико-технических задач

В данной работе рассматриваются ФТЗ, содержащие сложные динамические объекты, для которых определено понятие *состояния* как совокупности некоторых величин в данный момент времени и задан *закон «эволюции»*, который описывает изменение начального состояния с течением времени. Законы эволюции для динамических объектов могут задаваться различными способами (с помощью дифференциальных уравнений, теории марковских цепей и пр.) однако в данной работе рассматриваются законы, выраженные с помощью систем алгебро-дифференциальных уравнений.

«Сложность» динамической системы определим следующим образом:

- система многокомпонентна, её состав не является статичным и может изменяться во времени;
- компоненты системы имеют различную физическую природу;
- связи между компонентами могут носить как физический, так и информационный характер;
- структура системы в общем случае система иерархичная и многоуровневая;
- поведение компонентов системы может носить как непрерывный, дискретный так и непрерывно-дискретный (гибридный) характер [Колесов, 2012].

«Гибридное поведение» представляет собой **сложное поведение**. Оно исследовалось в работах [Колесов, 2003, Сениченков, 2005] с точки зрения распространения теории конечных автоматов на сложное поведение систем. Гибридная система – «это математическая модель, предназначенная для описания реальных или конструируемых объектов, меняющих свое непрерывное поведение при наступлении определенных событий. Гибридные системы также могут называть событийно-управляемыми динамическими системами, реактивными и непрерывно-дискретными системами» [Сениченков, 2005]. Также употребляются наименование «дискретно-событийные системы».

В качестве гибридных систем могут представляться системы из различных областей:

- механические системы (движение и столкновение объектов),
- электрические цепи (зарядка конденсаторов при смене положения ключа в цепи),
- химические системы (химические реакции, управляемые переключением клапанов, насосов),
- вычислительные системы (взаимодействие цифровых и аналоговых устройств)

Для моделирования гибридной системы может использоваться гибридный автомат – ориентированный граф, вершины которого соответствуют системам алгебро-дифференциальных уравнений, а ветви – переходам. Гибридные автоматы используются для моделирования дискретно-непрерывного поведения в среде моделирования Rand Model Designer (ранее MvStudium).

Можно выделить **три основных фактора**, обусловливающих появление гибридного поведения [Колесов, Сениченков, 2012].

1) *совместное функционированием непрерывных и дискретных объектов;*

2) *мгновенные качественными изменения* в непрерывном объекте (естественные или искусственные – связанные с представлением исследователя о системе);

3) *изменение состава системы*.

Помимо поведенческой сложности для ФТЗ характерна также **структурная сложность** – искусственные объекты, рассматриваемые в таких задачах, могут иметь жёсткие или упругие внешние связи с другими объектами при рассмотрении их механического движения (например, изучение движения двух сцепленных вагонов).

Таким образом 2 основными характеристиками ФТЗ, рассматриваемыми в данной работе являются **поведенческая** и **структурная** сложность.

Очевидно, что ФТЗ представляют собой промежуточное звено между физическими задачами (зачастую учебными) и техническими задачами – иначе

говоря инженерными задачами, связанными с преобразованием технической системы и изменением её конфигурации.

Сформулируем особенности ФТЗ, которые необходимо учитывать при выборе среды для создания их компьютерных моделей:

1) характер поведения объектов в задаче может быть как непрерывный (физическое поведение), так и дискретно-непрерывный (например, мгновенные качественные изменения в модели непрерывного поведения);

2) геометрические свойства объектов в ФТЗ могут варьироваться (например, при смене типа связи между телами), поэтому должны моделироваться отдельным от основной системы уравнений блоком;

3) степень детализации задачи может изменяться в ходе моделирования (используется подход итерационного моделирования);

4) предметной областью таких задач, является физика, что определяет набор типовых моделей (элементов), которые могут представляться готовыми блоками;

5) при декомпозиции ФТЗ явно выделяются следующие уровни: 1. непрерывное поведение, 2. дискретное поведение, 3. управляемое поведение (внешнее воздействие на модель со стороны исследователя).

Далее перейдём к рассмотрению классификации ФЗ и ФТЗ.

1.1.2 КЛАССИФИКАЦИЯ ФИЗИКО-ТЕХНИЧЕСКИХ ЗАДАЧ

В научной и учебной литературе отсутствуют последовательные классификации ФТЗ, несмотря на то, что этот термин находит в них употребление. В работе [Филиппов, 2007] рассматриваются критерии классификации учебных **ФЗ,** которые можно взять в качестве опоры для формулирования критериев для классификации ФТЗ: по характеру и методу исследования; по характеру требований; по содержанию; по способу предъявления; по полноте содержания; по способу решения; по целевому назначению.

В качестве критериев классификации **ТЗ** могут выступить различные, такие как реальность, объективность, уровень сложности решения, информация, функциональность, определённость, типология (хорошо определенные и плохо определенные задачи), характер творческого вклада (задачи поиска нового решения, задачи поиска нового применения известного решения и задачи на «заполнение пробелов») [Буш, 1976].

Отметим следующие фрагментарные классификации ФТЗ. В работе [Шиян, 2000] рассматриваются два типа ФТЗ: «исследовательские задачи, состоящие в установлении механизмов физических процессов, определяющие функциональные свойства материалов и приборных структур электроники», и «конструкторские задачи, состоящие в определении возможностей создания электронных компонентов с заданными свойствами» [Клишкова, 2011] – т.е. задачи функционального проектирования.

В работе [Имашев, 2007] предлагается разделение ФТЗ на а) творческие и репродуктивные в зависимости от их содержания, характера и цели, б) качественные и вычислительные. Также автор выделяет задачи с производственно-техническим содержанием и политехническим содержанием – задачи отражающие общие принципы действия и устройства различного рода установок и машин, таких разнообразных отраслей: промышленного производства, сельского хозяйства, связи, транспорта и пр. Однако в рассматриваемой работе акцентируется внимание только на образовательной стороне решения таких задач.

Обобщая вышеприведённые классификации, приведём следующую классификацию (табл. 1.1).

Таблица 1.1 – Предлагаемая классификация ФТЗ

Классификационные признаки	Классы ФТЗ
1 По изменчивости свойств	а) статические, б) динамические
2 По характеру динамического поведения	а) непрерывные, б) дискретные, в) дискретно-непрерывные (гибридные).

3 По характеру непрерывного поведения	а) алгебраические, б) алгебро-дифференциальные, в) интегральные и др.
4 По характеру гибридного поведения	а) совместное функционирование непрерывных и дискретных объектов; б) мгновенные качественные изменения в непрерывном объекте; в) изменение состава системы; г) не характеризующиеся гибридным поведением;
5 По степени детализации модели (по степени упрощения)	а) с учётом геометрических свойств объектов и без их учёта; б) с учётом сил трения, сопротивления среды; в) с учётом тепловых процессов при механическом взаимодействии и т.д.
6 По полноте известных данных	а) с необходимыми данными; б) с достаточными данными; в) с избыточными данными
7 Цели решения задачи	а) исследовательские задачи, б) задачи функционального проектирования, в) учебные задачи
8 Область физики	а) гидравлика, б) баллистика, в) механика и пр.
9 По содержанию (соотношению физики и техники)	а) физические задачи с техническим содержанием; б) технические задачи с физическим содержанием;
10 По исходным данным	а) задачи с хорошо структурированными исходными данными (известны все её элементы и связи между ними), б) задачи со слабо структурированными исходными данными (структура задачи скрыта или задана неявно)

1.1.3 МАТЕМАТИЧЕСКОЕ ОПИСАНИЕ КЛАССА РЕШАЕМЫХ ЗАДАЧ

В данной работе рассматриваются ФТЗ с непрерывным и дискретно-непрерывным поведением. Такой подкласс задач может быть описан как $T = (P(a, x), G(a, x), S(a, x))$, где $P(a, x)$ – детерминированное непрерывное (физическое) поведение, описываемое системой алгебро-дифференциальных уравнений (x – переменные, a – параметры), определяющее непрерывное состояние системы; $G(a, x)$ – геометрические свойства объекта и его межобъектных связей, описываемые системой алгебраические уравнений; $S(a, x)$ – дискретное поведение, представленное графом переходов (определяющее

дискретное состояние системы), вершины которого соответствуют некоторой подсистеме из $P(a,x)$, а ветви – переходы.

Непрерывная часть модели ФТЗ представляется системой алгебро-дифференциальных уравнений, а дискретная часть – набором дополнительных уравнений, рассчитываемых в момент возникновения некоторого дискретного события. Проиллюстрируем на рис. 1.1 соотнесение дискретного S и непрерывного $S(x,t)$ состояний системы.

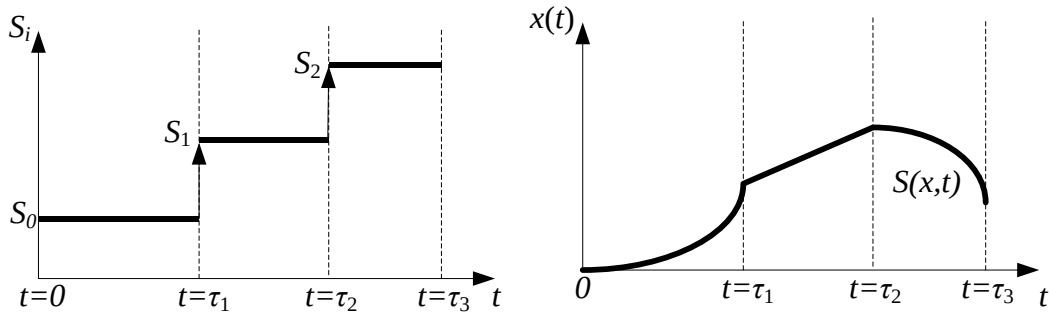


Рисунок 1.1 – Соотнесение дискретного и непрерывного состояний

Дискретное состояние представляет собой счётное количество систем уравнений, которые поочерёдно описывают поведение системы на определённых участках непрерывного времени (на интервалах τ_i). Непрерывное состояние в момент времени t_i представляет собой точку в пространстве состояний динамической системы, при этом дискретное состояние S обусловливает градиент $\nabla S(x,t)$.

Наличие дискретно-непрерывного поведения соответствует сложности **поведения** объекта. **Структурной** сложности соответствует наличие геометрических свойств, описываемых $G(a, x)$, под которыми понимается в первую очередь межобъектные связи: движение тела по некоторому нелинейному профилю (поверхности), наличие жесткой/упругой механической связи между объектами и пр.

1.2 ИНСТРУМЕНТАЛЬНЫЕ СРЕДСТВА МОДЕЛИРОВАНИЯ ФИЗИКО-ТЕХНИЧЕСКИХ ЗАДАЧ

1.2.1 ОБЗОР СИСТЕМ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ ФИЗИКО-ТЕХНИЧЕСКИХ ЗАДАЧ

В целом способы представления непрерывных моделей в инструментальных средах можно разделить на:

1) схемотехнические (блочные) модели, состоящие из блоков низкого уровня (интеграторы, дифференциатор, передаточная функция, сумматоры, демпфер, инерционное звено и пр.) с направленными связями (подобно аналогового моделирования), характерные для *Simulink, VisSim*, СМ МАРС.

2) структурные (физические) модели, состоящие из блоков более высокого уровня абстракции, структурно соответствующие моделируемому объекту (труба, бак, насос и пр.). Подобный подход реализован, например, в *Modelica, Rand Model Designer*, СМ МАРС, надстройках *SimMechanics, SimPowerSystems* среды *Simulink.*

3) аналитические модели, которые задаются в символьном (аналитически) или (реже) в блочно-символьном виде (например, в СМ МАРС с применением интерактивных математических панелей (ИМП)).

Отмечаются следующие недостатки существующих моделирующих системах [Колесов, 2012]: 1) отсутствие средств эффективного моделирования дискретно-событийных (гибридных), 2) отсутствие удобного языка планирования и проведения вычислительного эксперимента, 3) отсутствие интерфейсов между различными средами моделирования (отсутствие возможности обмена моделями между пакетами), 4) отсутствие стандартизированных компонентов в разных пакетах.

На основании приведённых в пункте 1.1.1 особенностей ФТЗ, сформулируем *критерии* для оценивания систем компьютерного моделирования: 1) наличие инструментов для моделирования *непрерывного* и *дискретно-непрерывного поведения* объектов; 2) наличие инструментов для

отображения *геометрических* свойств объектов и межобъектных связей; 3) возможность постепенной детализации модели; 5) наличие готовых (параметрируемых) блоков (элементов), содержащих модели из предметной области; 6) возможность декомпозиции поведения объектов в задаче на несколько уровней.

Все инструменты, используемые для компьютерного моделирования ФТЗ, можно условно разделить на следующие группы: 1) системы компьютерной математики, 2) среды моделирования, 3) узкоспециализированные пакеты, в том числе САПР, 4) языки программирования высокого уровня (а также программы, требующие реализации в них численных методов) [Кочергин, 2017].

Системы компьютерной математики. Сюда относятся такие прикладные пакеты для осуществления математических расчётов как *Mathcad* [Охорзин, 2006], *Mathematica* (*Wolfram*), *Maxima*. Они реализуют общий подход к созданию аналитических моделей, но отличаются некоторыми особенностями, например, интерактивные документы *Mathcad* легче и удобнее позволяют решать несложные задачи, *Mathematica* имеет большое количество различных расширений (нейронные сети, обработка текста и языка), в том числе и для визуализации данных. Главное достоинство таких систем – прозрачность математических расчетов и простота их записи. Однако необходимо отметить и большее количество их недостатков: а) необходимость математического описания модели «вручную»; б) связи между этапами моделирования просматриваются неявно; в) отсутствие типовых блоков-моделей; г) отсутствие функциональной схемы модели (диаграммы); д) отсутствие специальных элементов для ввода/вывода; е) отсутствие виртуальных приборов; ж) отсутствие возможность быстрой модификации модели в случае изменения условий задачи.

Таким образом, системы компьютерной математики представляются удобным и эффективным средством для проведения расчётов и создания интерактивных документов (в частности, *Mathcad* и *Wolfram Mathematica*), а также позволяют создавать математические модели *непрерывных* систем, записываемые на символьном языке близком к математическому как в явной, так

и в неявной формах. Моделирование же дискретно-непрерывных систем требует привлечения элементов программирования, что значительно усложняет процедуру создания модели и фактически приближает её по сложности к разработке модели на языке программирования. Также, ввиду того, что такие пакеты являются универсальными, в них отсутствуют специфические элементы, необходимые для создания моделей ФТЗ (например, отображающие геометрические свойства объектов).

Системы компьютерного моделирования. К этому типу относятся такие «универсальные» системы, как *LabView*, *Simulink*, *VisSim*, СМ МАРС, Rand Model Designer, ИСМА. Инструменты данной категории наиболее приемлемы для компьютерного моделирования физических процессов, т.к. реализуют объектно-ориентированный подход к моделированию.

Simulink – графическая среда имитационного моделирования, входящая в состав пакета *Matlab*. Разработана в 1984 г, текущая версия – 9.3 (входит в Matlab версии 2019a). Среда содержит готовые расширяемые библиотеки блоков для различных областей физики и техники (гидравлика и др.), алгоритмические и структурные блоки; полностью интегрирована с *Matlab* и имеет доступ к различным расширениям (нейронные сети, нечёткая логика, 2-D и 3-D графика, анимация и т.д.) и языку программирования. Имеет возможность сокрытия фрагмента модели внутри создаваемого функционального блока с входами и выходами («вложенные структуры»). Модели представляются на одном слое, что негативно сказывается на наглядности моделей со сложными системами управления. Для моделирования гибридного поведения используются диаграммы состояний *stateflow* (аналог диаграммы конечного автомата в языке *UML*).

VisSim – визуальный язык моделирования динамических систем. Разработан в 1989 г, последняя выпущенная версия – 9 (2015). Содержит арифметические блоки, математические функции, передаточные функции, блоки интерактивного вывода и численного интегрирования. Основные области моделирования: электрические, тепловые, механические, гидравлические

процессы, эконометрика и др. Имеет расширения для нейронных сетей, клиента передачи данных с сервером и др. Также имеет возможности для интеграции с *Mathcad, Matlab*. Имеет наглядные средства анимации изображений. Готовые блоки для моделирования физических процессов отсутствуют. Модели составляются из блоков математических и функциональных операций и отображаются на одном слое. Для представления модели используются «вложенные структуры» аналогичные *Matlab*. Начиная с 2016 входит в пакет программ solidThinking's Model Based Development Suite.

LabView – среда разработки с графическим языком программирования. Разработана в 1986 году, текущая версия – 2019. Имеет широкий спектр алгоритмических и структурных блоков, а также арифметических и функциональных. Специальные блоки для моделирования физических процессов отсутствуют. Символьные модели записываются в явном виде в специальных блоках, соединяемых с источниками данных и средствами визуализации. В среде имеется возможность создания новых компонентов с моделью внутри, а также пополнения ими библиотеки моделей. Есть дополнительные библиотеки для удалённой передачи данных, математических методов обработки данных, доступа к базам данных, визуализации данных (2-*D* и 3-*D*). LabView осуществляет декомпозицию модели на 2 уровня: лицевая панель (интерфейс) и блочная диаграмма, которая содержит как модель объекта, так и модель его системы управления.

СМ МАРС – мультифизическая среда многоуровневого моделирования. Имеет различные структурные, алгоритмические, арифметические компоненты, компоненты для доступа к базам данных и пр. В среде присутствуют специальные компоненты для моделирования физических процессов, а также возможность пополнения библиотеки компонентов. СМ МАРС базируется на принципах метода многоуровневых компонентных цепей (ММКЦ), что предполагает разделение модели объекта на визуальный, логический (алгоритмический) и объектный уровни, что позволяет отделять модель объекта от модели его системы управления.

Rand Model Designer (промышленная версия академической СМ *MvStudium*) представляет собой среду визуального моделирования для создания однокомпонентных, многокомпонентных и гибридных моделей сложных динамических систем. Среда опирается на теорию гибридных автоматов (ГА) при построении моделей дискретно-непрерывных систем. Аналитические модели (в виде алгебро-дифференциальных уравнений) задаются символьно «внутри» гибридного автомата и жёстко к нему привязаны. Компонентные модели могут содержать внутри (инкапсулировать по аналогии с *Simulink*) себя ГА, однако ГА не могут включать в себя многокомпонентные модели, что ограничивает набор решаемых задач.

Системы автоматизированного проектирования (например, *SolidWorks*, в частности её модуль *Flow Simulation*), позволяющие создавать трёхмерные параметрические модели различных конструкций для целей функционального проектирования сложных технических объектов. Такие системы, например, позволяют моделировать течение жидкостей и газов по типовым физическим моделям с наглядной визуализацией. Главным недостатком таких систем является отсутствие средств имитационного моделирования и, как следствие, невозможность моделирования систем управления проектируемым объектом (что объясняется их другой целевой ориентацией). Однако возможно встраивать созданные в таких системах трёхмерные параметрические «микромодели» в «макромодели», создаваемых в средах компьютерного моделирования (например, [Жуков, Горбунов, Лычагин, 2017]). Инструментальные средства данного типа в работе не рассматриваются, так как являются узкоспециализированными и позволяют решать задачи только одного класса.

Далее проиллюстрируем модельное представление ФТЗ в различных средах моделирования.

1.2.2 ПРИМЕРЫ КОМПЬЮТЕРНЫХ МОДЕЛЕЙ

Продемонстрируем возможности современных СМ для моделирования ФТЗ, а также их недостатки на примере моделей непрерывных, дискретно-непрерывных систем и задач с геометрической составляющей.

Моделирование непрерывных систем

Рассмотрим модели непрерывных систем, построенных в *Simulink, VisSim,* СМ МАРС на примере движения падающего тела [Simulation of bouncing …] (рис. 1.2). Данная задача может рассматриваться как в качестве непрерывной, так и дискретно-непрерывной (с точки зрения последовательности сменяющих друг друга моделей движения тела в отскоке).

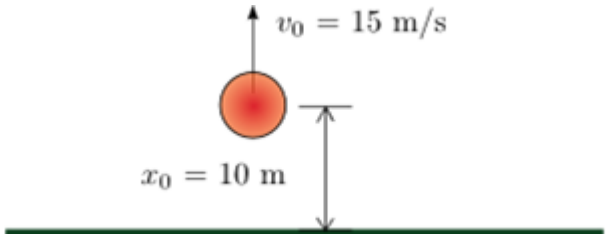


Рисунок 1.2 – Иллюстрация к непрерывной задаче

Simulink. Представим пример модели, прыгающего мяча, собранной в Simulink с помощью компонентов интеграторов (рис. 1.3), отвечающей следующей математической модели: $\frac{dv}{dt} = -g, \frac{dx}{dt} = v$, где g – ускорение свободного падения, $x(t)$ – координата мяча, а $v(t)$ – скорость мяча.

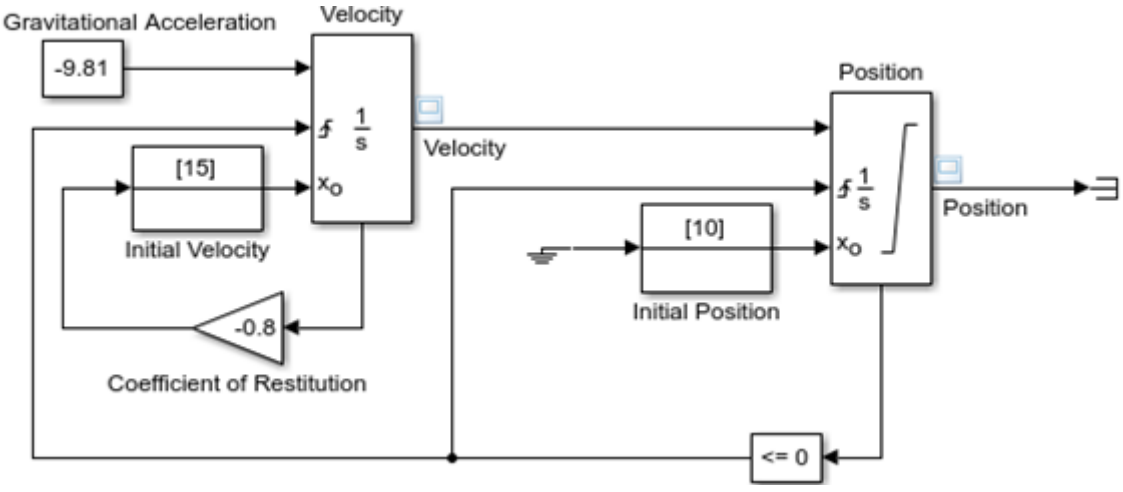


Рисунок 1.3 – Модель движения мяча в Simulink

На рис. 1 блок *Velocity* – интегратор скорости, представляющий собой первое уравнение математической модели, блок *Position* – интегратор координаты. Блок «<=0», расположенные ниже правого интегратора, задаёт ограничение на координату мяча (мяч не может опуститься ниже $x = 0$). Результаты моделирования представлены на рис. 1.4 в виде графиков зависимостей $x(t)$ и $v(t)$ соответственно.

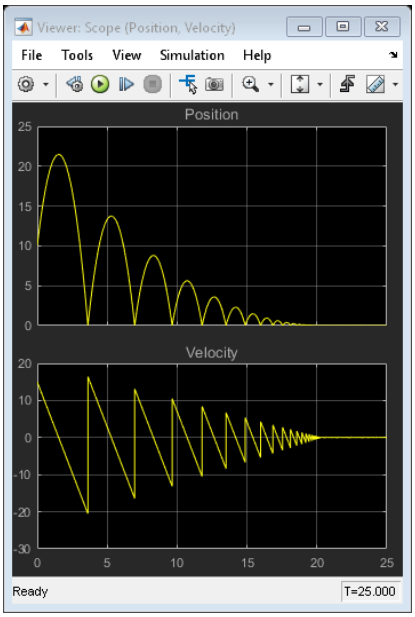


Рисунок 1.4 – Результаты моделирования в Simulink

На графике можно наблюдать затухающие отскоки тела и пилообразное изменение значения скорости.

VisSim. Рассмотрим аналогичную модель прыгающего мяча в среде VisSim [Дьяконов, 2004] (рис. 1.5).

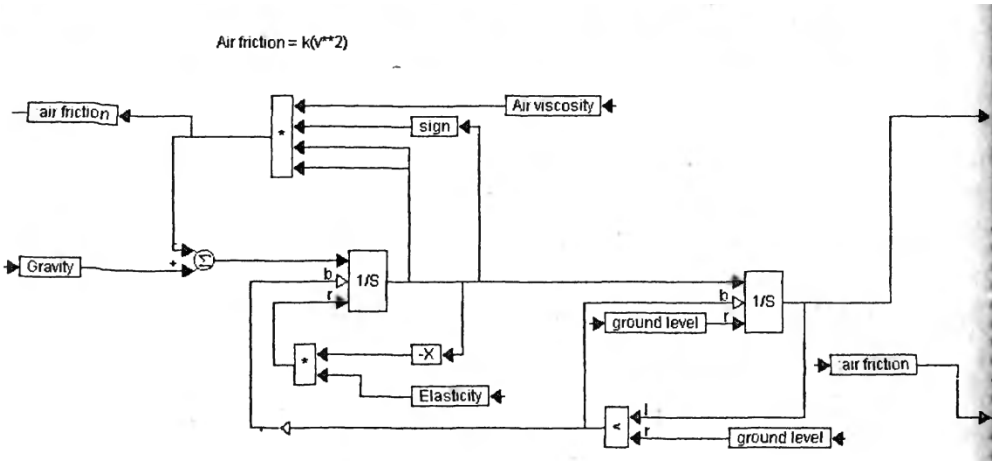


Рисунок 1.5 – Диаграмма непрерывной модели в *VisSim*

Данное решение аналогично решению в *Simulink* представленному ранее и отличается только графическим языком СМ. Результаты моделирования представлены на рис. 1.6.

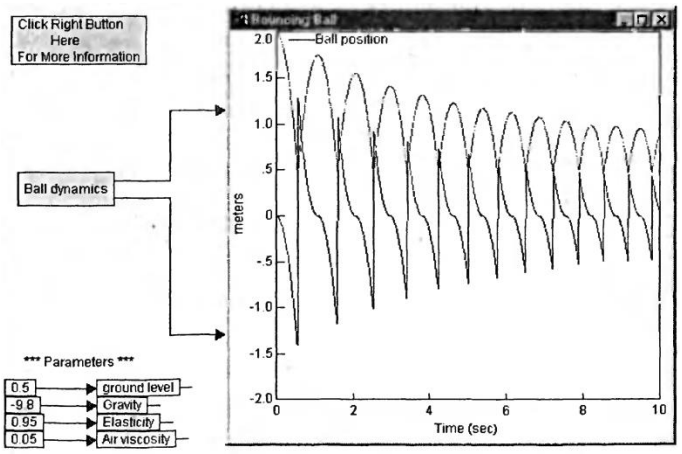


Рисунок 1.6 – Результаты моделирования непрерывной задачи

Моделирование дискретно-непрерывных систем

Рассмотрим предыдущую задачу о прыгающем мяче в качестве дискретно-непрерывной, считая падение и подъём мяча в качестве *состояний*, а удар о землю в качестве *события* (которое приводит к мгновенному изменению значения скорости), и представим её модельное представления в соответствующих средах.

Rand Model Designer. Среда использует ГА для отображения дискретно-непрерывных моделей. На рис. 1.7 представлена модель дискретного поведения, состоящего из двух (не считая начального – в виде закрашенного круга) состояний: «Движение» и «Отскок». Сторожевые условия, выполнение которых инициирует переход в другое состояние, записываются в пунктирных выносках.

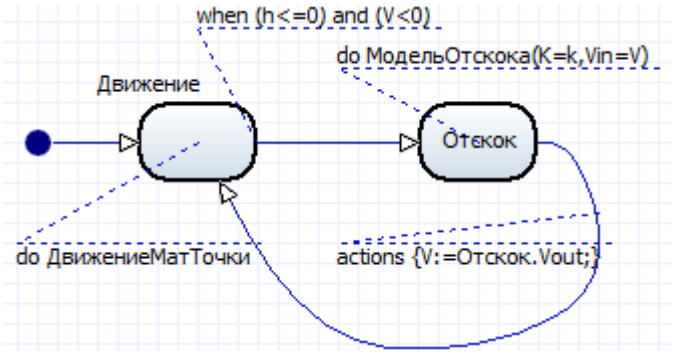
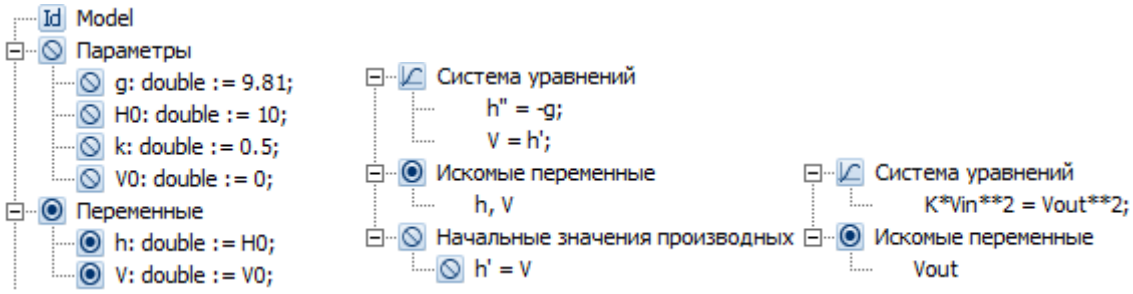


Рисунок 1.7 – Гибридная модель в Rand Model Designer

Каждое состояние содержит в себе систему алгебро-дифференциальных уравнений (рис. 1.8), которая характеризует непрерывное поведение системы.



а б в

Рисунок 1.8 – Аналитическая модель в Rand Model Designer

а – параметры и переменные модели, б – аналитическая модель ГА «Движение», в – аналитическая модель ГА «Отскок»

Результаты моделирования могут быть представлены графически (рис. 1.9).

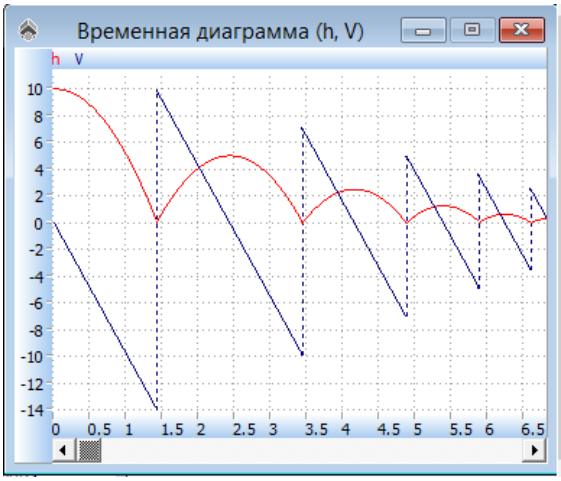


Рисунок 1.9 – Результаты моделирования в Rand Model Designer

Simulink. Для моделирования дискретно-непрерывных систем в *Simulink* используются диаграммы *stateflow*, являющиеся развитием конечных автоматов языка *UML*, берущих начало от карт состояний Харела [Harel, 1987]. *Stateflow* представляют собой отдельные блоки (подсистемы) на общей схеме модели. Отметим, что блоки данного типа (как и многие другие в *Simulink*) содержат вложенные структуры – схемы из последовательности связанных блоков, начало и конец которой соответствуют входу и выходу «родительского» блока. Из тела (содержимого) блока *stateflow* могут вызываться только функции, написанные на

языке программирования *Matlab*, а взаимодействовать с другими блоками схемы они могут **только** через свои входы и выходы.

На рис. 1.10 представлена модель данной задачи, состоящая из двух блоков: *Bouncing Ball*, содержащим внутри себя конечный автомат, и *Scope*, отображающего результаты моделирования пользователю.

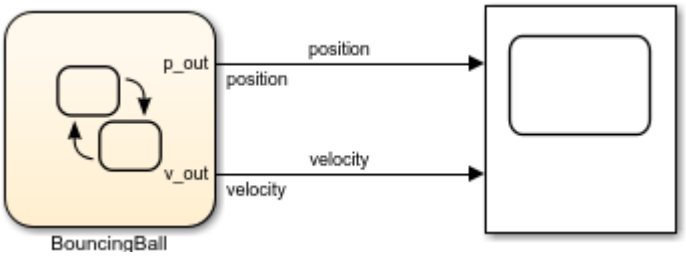


Рисунок 1.10 – Модель в Simulink

Содержимое блока *Bouncing Ball* представлено на рис. 1.11, где в нотациях *stateflow* (также схожими с нотациями UML) представлена диаграмма дискретного поведения.

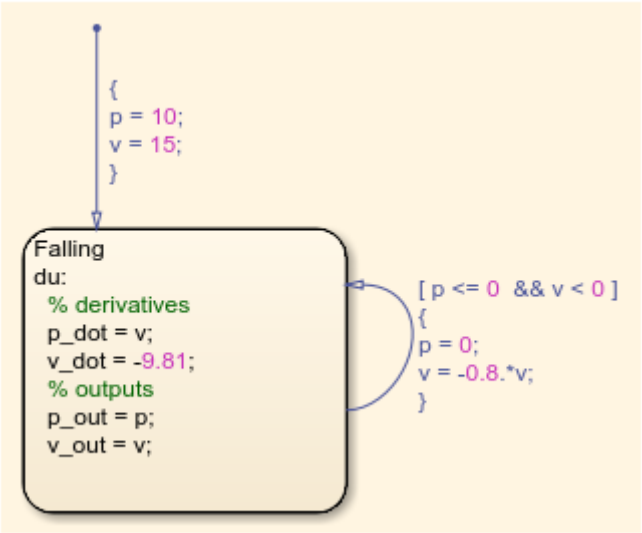


Рисунок 1.11 – Содержимое блока stateflow

Модель непрерывного поведения записана в символьной форме внутри блока *du* (сигнализирующего о повторяемых действиях), где системные (предопределённые) переменные *p_dot* и *v_dot* соответствуют приращению координаты и скорости мяча в единицу времени. Заканчивается исполнение блока при переходе в следующее состояние (по стрелке), которое произойдёт при

выполнении сторожевых условий, записанных справа в квадратных скобках (в фигурных скобках записана операция, выполняемая при переходе).

Результаты выполнения такое модели представлены на рис. 1.12.

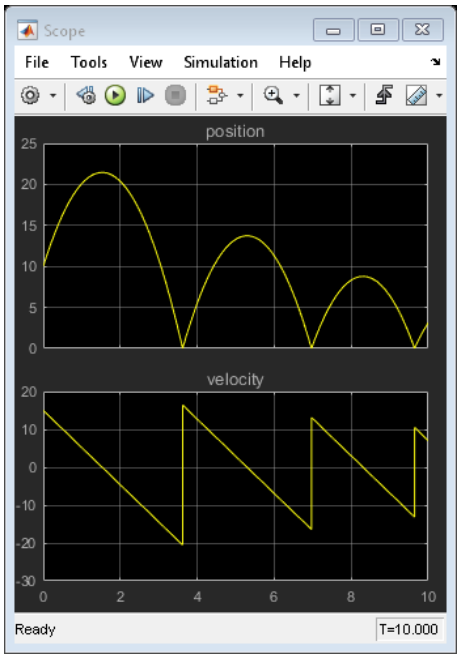


Рисунок 1.12 – Результаты моделирования

Моделирование геометрических свойств объектов

Проиллюстрируем способы отображения геометрических свойств в СМ *Simulink*. Остальные среды (за исключением *LabView*) не имеют готовых компонентов для отображения геометрических свойств объектов в задаче.

Рассмотрим задачу о скатывании тела по наклонной плоскости (рис. 1.13).

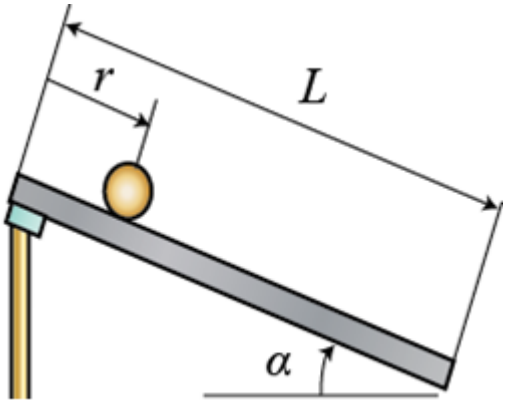


Рисунок 1.13 – Иллюстрация к физико-геометрической задаче

Simulink. В *Simulink* осуществляется сочленение механической части модели и модели управления в единую схему с помощью блоков типа *JointSensor* и *JointActuator* [Тихонов, Тишков, 2010]. Механическая часть представляет

собой цепи из моделей твёрдых тел, массово-инерционные характеристики устанавливаются в параметрах блоков.

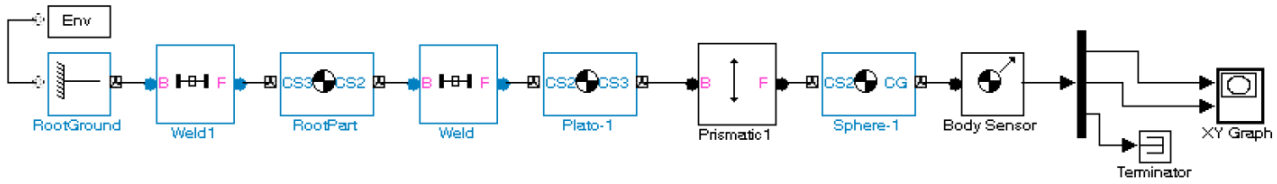


Рисунок 1.14 – Модель «Шар на плоскости» в *Simulink*

Результаты моделирования (рис. 1.15) иллюстрируют изменение положения центра масс тела.

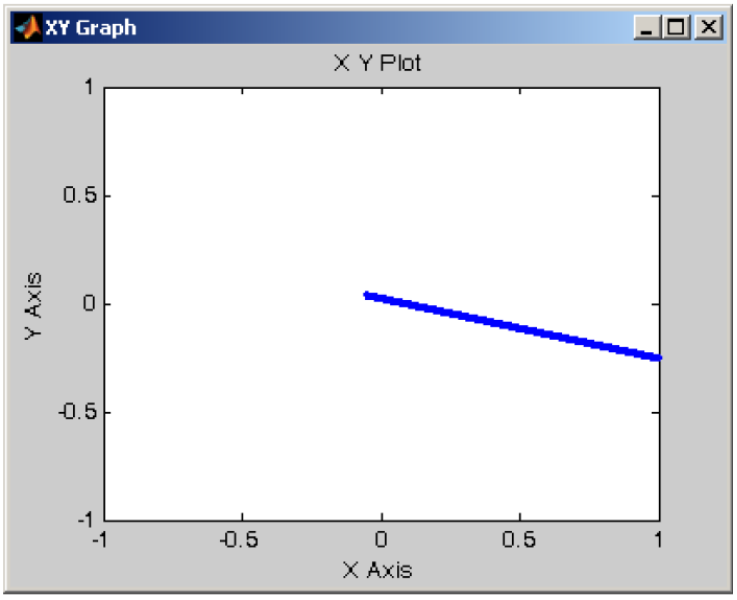


Рисунок 1.15 – График изменения положения центра масс тела

***

Наиболее приемлемыми типом программ для компьютерного моделирования ФТЗ нам представляются системы компьютерного моделирования, такие как СМ МАРС, *Simulink* и др.

Опишем системы компьютерного моделирования согласно представленным ранее требованиям к инструментам моделирования ФТЗ (табл. Таблица 1.2).

Таблица 1.2 – Характеристика пригодности СМ для моделирования ФТЗ

	Simulink, VisSim, SimInTech	LabView	Rand Model Designer	СМ МАРС
1) наличие инструментов для моделирования непрерывного и дискретно-непрерывного поведения объектов	диаграммы stateflow	–	ГА	компоненты «логические операторы»
2) наличие инструментов для отображения геометрических свойств	–	–	–	–
3) возможность постепенной детализации	+	+	+	+
4) наличие готовых блоков (элементов), содержащих модели из предметной области	+	–	–	+
5) возможность декомпозиции поведения объектов в задаче на несколько уровней	–	частично	–	+

В качестве среды наиболее приемлемой для моделирования ФТЗ будем рассматривать СМ МАРС, базирующуюся на методе многоуровневых компонентных цепей, т.к. она имеет расширяемую библиотеку компонентов и, что наиболее важно, предполагает декомпозицию задачи на 3 уровня: 1) схемно-объектный (непрерывное поведение), 2) алгоритмический (дискретное поведение), 3) визуальный (измерение и управление моделью).

1.3 МЕТОД МНОГОУРОВНЕВЫХ КОМПОНЕНТНЫХ ЦЕПЕЙ ДЛЯ МОДЕЛИРОВАНИЯ ТЕХНИЧЕСКИХ СИСТЕМ

1.3.1 НАЗНАЧЕНИЕ, ОСНОВНЫЕ ПОНЯТИЯ МЕТОДА МНОГОУРОВНЕВЫХ КОМПОНЕНТНЫХ ЦЕПЕЙ

ММКЦ относится к классу универсальных методов компьютерного моделирования [Ганджа, 2017] и предназначен для моделирования сложных объектов и систем, обладающих информационными и энергетическими потоками в связях; позволяет создавать как имитационные (в виде алгоритмических конструкций), так и аналитические (в виде систем алгебро-

дифференциальных уравнений) модели. ММКЦ позволяет строить многоуровневые компьютерные модели сложных, физически неоднородных технических (технологических) объектов с информационными и энергетическими связями.

Базовыми понятиями ММКЦ являются *компонент* и *компонентная цепь* (КЦ). **Компонент** – формализованное отображение некоторого элемента или функционального блока моделируемой системы, описываемого отдельной математической (математико-алгоритмической) моделью. Уравнения (выражения) модели компонента составляются относительно его первичных (входных и выходных) параметров и переменных.

Компонентная цепь (рис. 1.16) является основной компьютерной моделью исследуемой системы в формате ММКЦ, в общем случае представляет собой произвольную совокупность связанных друг с другом *компонентов.* Формально КЦ представляет собой совокупность трех взаимосвязанных множеств $C = K, B, N$, где K – множество компонентов цепи; B – множество ветвей цепи (связей); N – множество узлов цепи [Дмитриев, Ганджа, 2004].

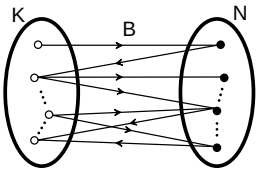


Рисунок 1.16 – Обобщённый вид КЦ

КЦ C может быть представлена совокупностью подцепей: $C = C_1 \cup C_2 \cup ... \cup C_n$, где $C_1, C_2, ..., C_n$ – также КЦ.

Множество компонентов K в общем случае имеет три подмножества:

1) K_W – компоненты источники энергии или сигналов;

2) K_P – компоненты-преобразователи энергии или сигналов;

3) K_Z – компоненты-измерители энергии или сигналов [Дмитриев, Ганджа, 2004].

Каждый компонент *K* цепи *C* имеет произвольное число связей, каждой из которых соответствует пара топологических координат $S_j = (N_j, B_j)$, – переменные связей, где V_{Nj}– потенциальная (например, скорость, ток, сила или давление), а V_{Bj} – потоковая (например, сила, ток, поток) переменные цепи. Для переменных, ведущих в один и тот же узел КЦ, выполняются законы: для ***потенциальных*** переменных – *закон равенства*, а для ***потоковых*** – *закон равенства нулю их суммы*.

1.3.2 МНОГОУРОВНЕВЫЙ ПОДХОД К ПРЕДСТАВЛЕНИЮ КОМПЬЮТЕРНЫХ МОДЕЛЕЙ

Формализм ММКЦ предполагает декомпозицию любой модели на 3 уровня (каждый из которых соответствует определенному слою в редакторе СМ МАРС):

1. **Схемно-объектный уровень**, на котором средствами аналитического моделирования описывается непрерывное (физическое) поведение объекта.

2. **Алгоритмический уровень**, на котором средствами имитационного моделирования описывается дискретное поведение объекта – алгоритм его функционирования в задаче.

3. **Визуальный уровень**, на котором отображается внешнее представление объекта или видимое проявление любого его поведения (т.е. регистрируемое и отображаемое, например, на интерфейсе прибора), и задаётся управляющее воздействие со стороны исследователя.

На рис. 1.17 представлена общая структура многоуровневого (многослойного) представления компьютерной модели в рамках формализма ММКЦ в СМ МАРС, где *схемно-объектному уровню* многоуровневой компьютерной модели (МКМ) соответствует *объектный* слой, *алгоритмическому – логический*, *визуальному – визуальный*.

Рисунок 1.17 – Структура многоуровневой модели

Далее опишем назначение слоёв редактора СМ МАРС, находящихся во взаимно-однозначном соответствии с уровнями декомпозиции МКМ в ММКЦ.

Визуальный слой (*V*-слой) представляет собой пользовательский интерфейс для взаимодействия с моделью. На нём осуществляется ввод входных данных модели, вывод пользователю выходных данных и визуализации результатов моделирования в виде графиков, диаграмм и пр. [Кочергин, 2017] Основная цель этого слоя – интерактивное управление моделью.

Объектный слой (*C*-слой) содержит математическую (аналитическую) модель системы в виде КЦ, состоящей из схемно-технических и схемно-аналитических блоков, характеризующей их непрерывное (физическое) поведение [Дмитриев, Ганджа, 2016].

Логический слой (*L*-слой) осуществляет передачу данных между C- и V-слоями, а также содержит алгоритмическую КЦ, определяющую алгоритм дискретного поведения моделируемого объекта или сценарий проведения вычислительного (виртуального) эксперимента на модели [Кочергин, 2017]. Также на этом слое содержатся сценарии исследовании модели и обработки результатов моделирования, сценарии взаимодействия модели с источниками данных (например, базами данных).

Таким образом реализуется следующая схема работы модели: данные с V-слоя слоя подаются на L-слой, оттуда – на C-слой. После этого на C-слое производится расчёт модели на текущей итерации, результаты которого передаются обратно на L-слой, а оттуда на V-слой и отображаются пользователю на графиках или компонентах-визуализаторах. Далее процедура повторяется.

1.3.3 ЯЗЫК МОДЕЛИРОВАНИЯ МНОГОУРОВНЕВЫХ КОМПОНЕНТЫХ ЦЕПЕЙ

В работе [Ганджа, 2017] описывается язык многоуровневых компонентных цепей, используемый в СМ МАРС на разных слоях многослойного редактора. Данный язык состоит из трёх своих подмножеств: на *C*-слое используется *язык моделирования химико-технологических систем* (язык ХТС), на *L*-слое – *язык моделирования алгоритмических конструкций* (язык МАК), на *V*-слое – *язык виртуальных инструментов и приборов* (язык ВИП). Все языки являются графическими, что позволяет реализовать подход визуального моделирования – представления моделей в виде схем, состоящих из связанных блоков. Аналогичные языки используются в таких средах как *LabView* и *Simulink*. Дадим краткую характеристику каждому из языков СМ МАРС.

Язык ХТС предназначен для представления на С-слое МКМ таких КЦ, которые соответствует моделям процессов преобразования мультифизических энергетических и многокомпонентных вещественных потоков, протекающим в управляемых ХТС [Ганджа, 2017].

Компоненты языка ХТС делятся на: 1. Компоненты-источники: а) *источники энергии* (источник потенциальной переменной: *источник потенциала* (скорости, давления, температуры), *источник разности потенциальных переменных* (насосы, источники напряжения) и *источник потоковой переменной*); б) и*сточники вещественного потока* (подающие смесь веществ). 2) Компоненты-преобразователи: *преобразователи энергии* и *преобразователи вещества*. 3) Компоненты-измерители: *измерители характеристик энергии* и *измерители характеристик вещества*.

Язык МАК предназначен для построения сценариев сложных технических управляемых систем, а также сценариев функционирования устройств управления. Эти КЦ относятся к L-слою и представляют собой алгоритмические КЦ. Язык МАК расширяет язык ХТС, позволяя формировать и анализировать математико-алгоритмические конструкции.

Язык ВИП включают в себя визуальные компоненты и предназначен для построения панелей виртуальных приборов (ПВП), которые используются для автоматизации вычислительных (виртуальных) и натурных (реально-виртуальных) экспериментов.

На рис. 1.18 представлена схема соответствия структуры МКМ в СМ МАРС структуре многоуровневого представления моделей в ММКЦ.

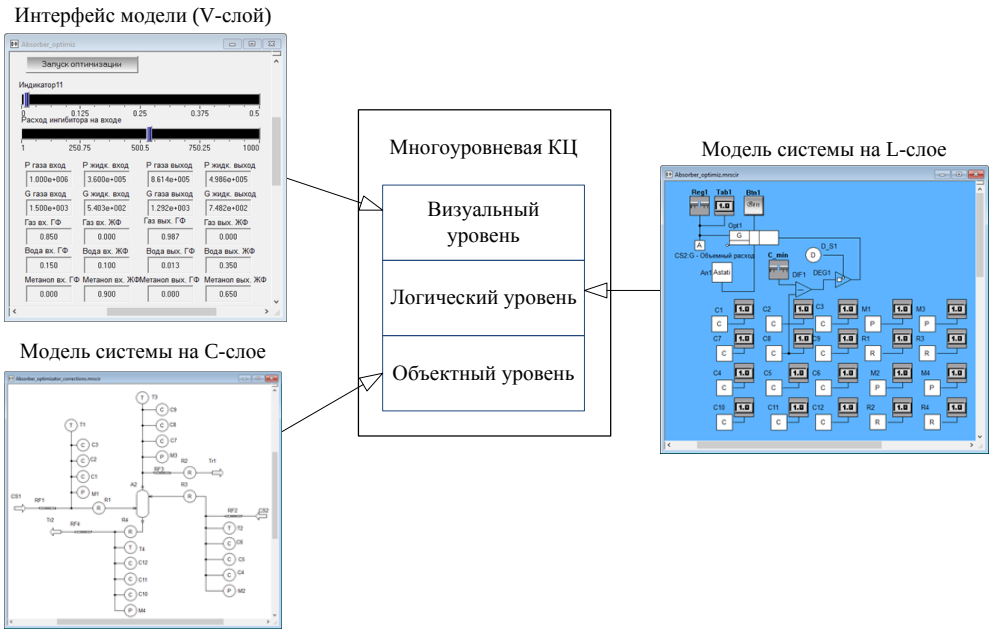


Рисунок 1.18 – Многоуровневое представление модели ХТС

ФТЗ как объект моделирования методом МКЦ

На текущий момент язык ММКЦ не удовлетворяет в полной мере требованиям к инструментальным средам для моделирования ФТЗ, изложенным в пункте 1.2.1 по ряду причин: 1) отсутствуют инструменты для моделирования непрерывного и дискретно-непрерывного поведения объектов – язык МАК содержит только компоненты низкого уровня абстракции (логические операторы, алгоритмические компоненты), что затрудняет построение сложных систем управления объектами в ФТЗ (рассмотрим ниже); 2) геометрические свойства могут задаваться только в блочно-символьном виде, так как специализированные компоненты отсутствуют; 3) готовые блоки (элементы), содержащие модели из предметной области представлены не в достаточном объёме. Однако необходимо отметить возможность декомпозиции поведения

объектов в задаче на несколько уровней, что является одним из ключевых особенностей, позволяющих использовать этот язык в качестве базового (но с модификацией) для моделирования ФТЗ.

Язык МАК, с помощью которого осуществляется построение модели в СМ МАРС на *L*-слое, использует общеалгоритмический подход для построения моделей поведения объектов. Это означает что этот язык располагает базовыми компонентами, соответствующими арифметическим операциям, циклам, ветвлениям и другим математико-алгоритмическим конструкциям, следовательно, моделирование сложного поведения будет требовать составления большой и трудночитаемой КЦ, что не всегда является приемлемым. В связи с этим актуальным является интерпретация в язык МАК более сложных структур, описывающих дискретное поведение объектов и систем различной природы. Сочетание использования таких структур на L-слое (для описания дискретного поведения исследуемых объектов) и специальных компонентов на С-слое (для описания непрерывного поведения) позволит эффективнее и удобнее строить модели систем и объектов с дискретно-непрерывным поведением.

Рассмотрим моделирование задачи о движении материальной точки по трём участкам пути с различным ускорением, которую можно описать следующей формулировкой: «Материальная точка совершает равноускоренное движение по трём участкам своего пути: 1) с ускорением a_1 на первом участке длиной s_1, 2) a_2 – на втором (длиной s_2), 3) a_3 – на оставшемся (длиной s_3)». Непрерывное поведение объекта (материальной точки) в этой задаче описывается уравнением $s(t) = v_0t+at^2/2$ (его также можно описать дифференциальным уравнением второго порядка или двумя дифференциальными уравнениями первого порядка: $dx/dt = v$, $dv/dt = a$). Дискретное поведение объекта в данной задаче характеризуется сменой численного значения параметра «текущее ускорение» в связи с наступлением определённого события (в данном случае – достижения объектом конца текущего участка) и может быть описано следующим выражением (1.1).

$$a=\begin{cases} a_1, & if \quad s(t)<s_1 \\ a_2, & if \quad s_1 \le s(t) \le s_2 \\ a_3, & if \quad s(t)>s_1+s_2 \end{cases}, \qquad (1.1)$$

Таким образом дискретное поведение объекта в задаче можно разделить на 3 элементарных дискретных промежутка времени, в которых он характеризуется соответствующим значением вектора состояния. На рис. 1.19 представлен вид алгоритмической КЦ на L-слое, характеризующей дискретное поведение автомобиля, описанное выражением (1.1).

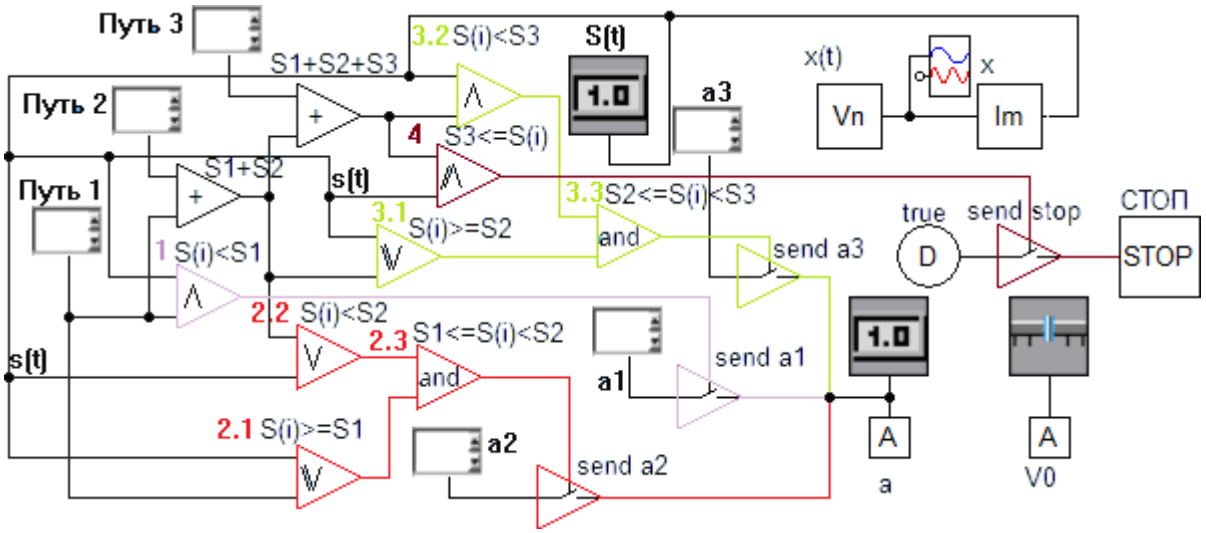


Рисунок 1.19 – Алгоритмическая КЦ задачи средствами языка МАК

На схеме используются компоненты-накопители «send a1», «send a2», «send a3», которые передают на выход (справа) значение со входа (слева) при поступлении управляющего воздействия «истина» (сверху). Так при выполнении условия $s(t)<s_1$ компонент «s(t)<s1» (помечен цифрой «1» на рис. 1.19) на каждой итерации работы модели будет подавать значение «истина» на накопитель «send a1», который будет передавать значение a_1 на *C*-слой, содержащий КЦ, описывающую непрерывное поведение автомобиля. При одновременном выполнении условий 2.1 и 2.2 (помечены на рис. 1.19) на *C*-слой будет поступать значение a_2, которое сменится значением a_3 при выполнении условий 3.1, 3.2. При выполнении условия 4 работа модели будет остановлена.

Представленный пример алгоритмической КЦ, составленной средствами языка МАК, позволяет сделать вывод о необходимости интеграции новых

компонентов, позволяющих более наглядно и компактно изображать дискретное поведение объекта.

Дискретное поведение в данной задаче может быть компактно представлено в виде конечного автомата, например, в нотациях языка UML – в виде диаграмм состояний (*statechart* в UML 1.0 или *state machine diagram* в UML 2.0). Представим на рис. 1.20 диаграмму состояний материальной точки в представленной выше задачи в нотациях языка UML.

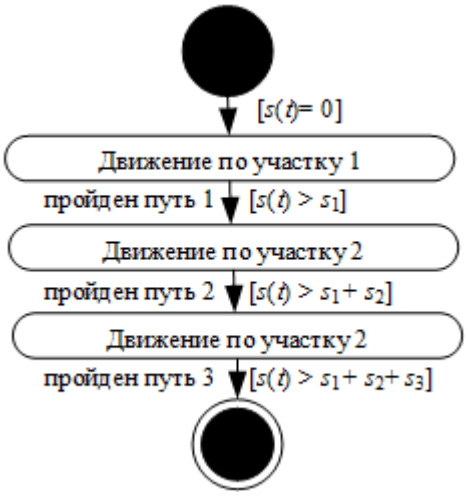


Рисунок 1.20 – Диаграмма состояний материальной точки

Очевидно, что модель дискретного поведения на рис. 1.19 и диаграмма состояний на рис. 1.20 не сопоставимы, что обусловливает актуальность развития ММКЦ (в частности разработку единиц языка более высокого уровня абстракции), направленное на создание средств адекватного и эффективного моделирования ФТЗ.

ВЫВОДЫ ПО ГЛАВЕ 1

1. ФТЗ представляют собой задачи о сложном динамическом объекте и обладают рядом особенностей, предъявляющих требования к инструментальным средствам моделирования.

2. В практике моделирования ФТЗ в качестве инструмента, как правило, применяются математические пакеты, языки программирования, табличные процессоры, специализированные САПР, не отвечающие специфике решения ФТЗ.

3. Формализм метода многоуровневых компонентных цепей, заложенный в СМ МАРС, отвечает специфике ФТЗ, однако методика многоуровневого компьютерного моделирования на данный момент отсутствует, как и специализированные компоненты СМ МАРС, необходимые для отражения физических, геометрических и поведенческих свойств объектов в ФТЗ.

ГЛАВА 2. РАЗВИТИЕ ЯЗЫКА МНОГОУРОВНЕВЫХ КОМПОНЕНТНЫХ ЦЕПЕЙ ДЛЯ МОДЕЛИРОВАНИЯ ФИЗИКО-ТЕХНИЧЕСКИХ ЗАДАЧ

Представим алгоритм многоуровневого моделирования ФТЗ и опишем его основные этапы, включая формализацию условий, декомпозицию задачи, построение информационной и компьютерной модели.

2.1 АЛГОРИТМ МОДЕЛИРОВАНИЯ ФИЗИКО-ТЕХНИЧЕСКИХ ЗАДАЧ

Взяв за основу алгоритм моделирования ФЗ, предложенный в [Филиппов, 2007] представим **алгоритм многоуровневого компьютерного моделирования ФТЗ,** учитывающий их специфику и формализм ММКЦ:

1 Изучение исследуемого объекта и анализ данных задачи

1.1 Анализ поставленной задачи с опорой на знания предметной области: объективизация и параметризация задачи; изучение известных данных о моделируемых объектах.

1.2 Определение целей моделирования

1.3 Оценка полноты и точности имеющихся данных для достижения поставленной цели. Доопределение неизвестных данных и выбор степени детализации задачи.

2 Декомпозиция и формализация задачи. Построение информационной модели задачи

2.1 Декомпозиция условий задачи в соответствии с методикой многоаспектного анализа

2.2 Формирование диаграммы состояний задачи (выделение фаз поведения)

2.3 Математическое описание непрерывной модели физического поведения в каждом дискретном состоянии

2.4 Дополнение модели описанием пространственно-геометрической составляющей задачи

2.5 Формирование сценария исследования модели

2.6 Оценка полноты полученной информационной модели и определения характера связей в задаче (детерминированные, вероятностные, нечёткое управление)

3 Построение компьютерной модели

3.1 Определение компонентного базиса будущей модели

3.2 Формирование дискретной структуры задачи из компонентов языка моделирования

3.3 Формирование КЦ на С-слое для описания непрерывного поведения в задаче и пространственно-геометрической задачи.

3.4 Оценка корректности модели

3.5 Формирование КЦ на L-слое для реализации сценария исследования модели, обработки и визуализации результатов

3.6 Тестирование сценария исследования

4 Проведение исследования и анализ результатов моделирования

Графическая иллюстрация алгоритма представлена ниже (Рисунок 2.1).

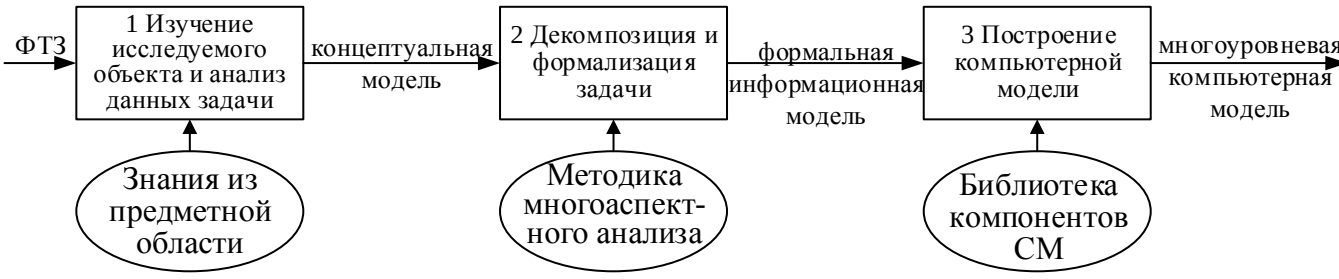


Рисунок 2.1 – Алгоритм моделирования ФТЗ

Проиллюстрируем этап 1 алгоритма моделирования задачи на примере следующей физической задачи: «Рассмотрим задачу движения упругого мяча, брошенного под углом к горизонту. Траекторию движения мяча можно представить в виде последовательно убывающих перевернутых парабол (рис. 2.2). Пусть мяч при своем приземлении падает на некоторую поверхность. Мяч, упав на поверхность, испытывает действие упругой силы, выталкивающей

материальную тело (мяч). Возникает упругий удар и мяч, отскочив от поверхности, продолжает движение» [Маликов, 2005].

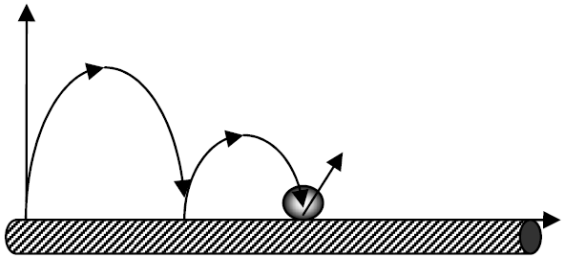


Рисунок 2.2 – Траектория движения мяча

Так как исходная задача не является физико-технической, усложним её за счёт введения профиля поверхности движения, задаваемого табличной функцией (см. Рисунок 2.3). Вид профиля обусловлен следующими причинами: а) задаётся таблично, б) содержит неоднородные участки, в) позволяет продемонстрировать отскок мяча как от горизонтальной, так и наклонной и вертикальной поверхностей. Также определим следующую цель решения задачи – установить координаты центра мяча в момент, когда величина проекции скорости мяча на ось OY составит менее 5% от начальной, что условно будем считать отсутствием отскока. Теперь данная задача представляет собой физическую задачу с техническим содержанием.

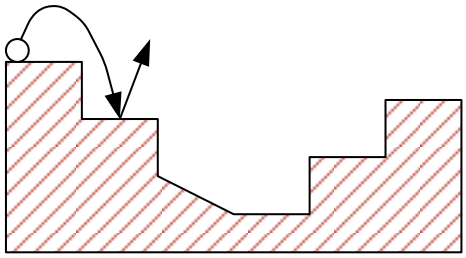


Рисунок 2.3 – Профиль поверхности табличной функции

Проиллюстрируем представленный выше алгоритм многоуровневого моделирования ФТЗ на данном примере.

Этап 1. Изучение исследуемого объекта и анализ данных задачи

На данном этапе производится анализ условий ФТЗ, выделяются и описываются важные для задачи характеристики действующих объектов, а также закладывается фундамент для будущей модели.

1.1 Анализ поставленной задачи с опорой на знания предметной области: объективизация и параметризация задачи; изучение известных данных о моделируемых объектах.

В данной задаче можно выделить следующие объекты и их параметры:

- Объект 1 – мяч: форма объекта (шар), масса, диаметр, плотность, координаты *x*, *y*, скорость *v*, ускорение *a*.
- Объект 2 – поверхность: профиль (задан таблично).

Тип взаимодействия объектов – столкновение, характер взаимодействия – упругое, повторяющееся.

Силы, воздействие которых на объект 1 может быть учтено: сила тяжести $F_т$, сила реакции опоры *N*, сила трения $F_{тр}$, атмосферное сопротивление $F_{атм}$.

Системные параметры: ускорение свободного падения *g*, трёхмерная декартова система координат.

1.2 Определение целей моделирования

Целью моделирования является построение координат мяча до точки останова – при выполнении условии $v_x \le v_{x0} \cdot 5\%$.

1.3 Оценка полноты и точности имеющихся данных для достижения поставленной цели. Доопределение неизвестных данных и выбор степени детализации задачи.

Ввиду того, что величина значений переменных (скорость, координаты) не будет большой, будем считать $g = const, F_{атм} = 0, F_{тр} = const$, а плотности тел будем учитывать в качестве задаваемого коэффициента отскока *k*. Остальные величины представим в качестве задаваемых исследователем.

2.1.1 ДЕКОМПОЗИЦИЯ ЗАДАЧИ И СИНТЕЗ СТРУКТУРЫ МНОГОУРОВНЕВОЙ КОМПЬЮТЕРНОЙ МОДЕЛИ

Специфика системы не исчерпывается особенностями составляющих её элементов, а определяется, прежде всего, характером связей и отношений между отдельными элементами системы [Дмитриев, Филиппов, Шарова, 2004]. Под

системой понимают упорядоченную совокупность взаимосвязанных элементов. Основными элементами ФТЗ как системы можно считать физические сущности – понятия и суждения (явления, процессы, физические законы, свойства тел и т.п.), а также технические (искусственные) объекты, являющиеся в свою очередь тоже системами.

На этапе 2 алгоритма многоуровневого моделирования ФТЗ осуществляется декомпозиция задачи. Под декомпозицией будем понимать «операцию разделения целого на части с сохранением признака подчинённости, принадлежности» [Перегудов, 1989, с. 354]. Эта процедура осуществляется с опорой на методику многоаспектного анализа, предложенного в работах [Дмитриев, Филиппов, Шарова, 2005] и развиваемого в данной работе в рамках развития языка ММКЦ (см. ниже).

Формализация ФТЗ осуществляется в соответствии с выделенными аспектами и субаспектами, в результате чего создаётся формальная информационная модель задачи (модель задачи на математическом языке). В работе [Анфилатов и др., 2002] процедура формализации описывается как процесс замены содержательного описания задачи формальным. Таким образом, происходит процедура замены элементов концептуальной модели формальными элементами (алгоритмическими – например, диаграммы состояний, математическими – дифференциальные уравнения физических процессов и пр.).

Структура представления условий физических и физико-технических задач

Как представлено в классификации (в пункте 1.1.2), ФТЗ по исходному представлению делятся на:

1. Задачи с хорошо структурированными исходными данными (ХСИД) – представленные почти в формальном виде (известны всё её элементы и связи между ними), например, в виде функциональных схем, рисунков (гидравлика, электротехника и др.). Внешний вид многоуровневой компьютерной модели на объектном уровне (С-слое) такой задачи явно и однозначно соответствует самой

задаче (например, схеме или рисунку). В качестве примера такой задачи можно привести задачу о проектировании системы водоснабжения, для которой заранее задана схема (чертёж).

2. Задачи со слабо структурированными исходными данными (ССИД) [Дмитриев, Филиппов, Шарова, 2005] – задачи, структура которых скрыта или задана неявно (чаще всего представлены на естественном языке – в виде текста). В таких задачах бывает трудно выделить значимые элементы, факторы и критерии, в связи с этим заранее невозможно определить состав и конечный вид компонентов задачи [Филиппов, 2007], а внешний вид МКМ такой задачи может быть вариативен.

Для ФТЗ с ССИД, а также избыточными или недостаточными данными возникает необходимость разработки чёткого аппарата формализации [Кочергин, Кочергина, 2016], позволяющего перейти от формулировки задачи к её МКМ. Автоматизация процедуры формализации позволит исследователю акцентировать свою деятельность на моделировании и дальнейшей работе с задачей, а не предварительной обработке данных, а также избавит представление задачи от избыточных данных и дополнит недостающими. Подобный инструмент может использоваться и при обучении моделированию. Исследование возможности автоматизации формализации ФТЗ представлено в параграфе 3.2.

Методика многоаспектного анализа условий физико-технических задач

Для того, чтобы осуществить процесс формализации словесного портрета задачи, представленной ССИД с целью создания её МКМ, необходимо осуществить многоаспектный анализ задачи. Разрабатываемая методика многоаспектного анализа базируется на подходе, предложенном в [Дмитриев, Филиппов, Шарова, 2005] и предназначенном для формализации условий ФЗ, и обусловливает процесс перевода ФТЗ (её словесного портрета) к виду МКМ согласно формализму ММКЦ, а также формализовать саму процедуру такого перевода.

Выделим 4 взаимосвязанных аспекта рассмотрения ФТЗ. Каждый из них может быть представлен как совокупность субаспектов.

1) Физико-математический аспект описывает моделируемые в ФТЗ физические процессы, степень их детализации исходя из целей моделирования и требований к точности модели, переменные и параметры (имеющие физический смысл), участвующие в описании физического процесса, а также задает математическое описание моделируемых процессов, ограничений на значение переменных и параметров модели, математических соотношений.

- *Математический субаспект* предназначен для определения способов описания непрерывного поведения объектов, описания математических связей между параметрами, переменными и их зависимостями;
- *Геометрический субаспект* обусловливает выделение и описание информационных элементов (ИЭ) словесного портрета задачи геометрического и схемного характера, установление информации о системе координат в ФТЗ и геометрических свойствах межобъектных связей.
- *Физический субаспект* включает в себя классификацию предметной области (например, механика, баллистика, гидравлика и пр.) ФТЗ, а также для определение требуемой степени детализации (абстракции) модели в зависимости от целей моделирования.
- *Объектно-параметрический субаспект* используется для установления соответствий ИЭ задачи (имеющейся информации об описываемых физических процессах) объектным моделям и их параметрам. Также этот аспект используется для определения размерностей физических величин.
- *Модельный субаспект* используется для установления соответствий между выделенными парами «объект-параметры» (характеризующими физический объект или процесс) и математическими моделями, описывающими их (выделенные объекты или их межобъектное взаимодействие).

3) Алгоритмический аспект определяет схему поведения объекта в течение его времени функционирования в ФТЗ, а также сценарий проведения эксперимента, порядок обработки результатов моделирования.

- *Поведенческий субаспект* характеризуется диаграммой состояний объектов в задаче.

- *Исследовательский субаспект* предназначен для разделения выполняемых алгоритмических предписаний по проведению вычислительного (виртуального) эксперимента, а также позволяет определить способы исследования модели, фиксации результатов моделирования и их визуализации.

4) Компонентный аспект задаёт соответствие между элементами формализованного представления задачи (её математической моделью) и их компонентными отображениями в рамках формализма метода многоуровневых компонентных цепей. Компонентный аспект является конкретизированным воплощением совокупности первых трёх аспектов.

- *Декомпозиционный субаспект* обусловливает декомпозицию задачи на три уровня: объектный, алгоритмический и визуальный.

- *Лексический субаспект* определяет набор языковых средств, используемых для описания ФТЗ и ставит в соответствие каждому элементу математической модели задачи конкретный компонент – единицу языка ММКЦ.

- *Топологический субаспект* предназначен, для построения схемы задачи, определения межкомпоненнтых связей.

Методика многоаспектного анализа ФТЗ заключается в построении информационной модели ФТЗ в соответствии с вышеописанными аспектами, каждый из которых (в отличии от метода в [Дмитриев, Филиппов, Шарова, 2005]) дополняет информационную модель, а не заменяет её.

Анализ ФТЗ с опорой на методику многоаспектного анализа позволяет привести рассматриваемую задачу к формализму ММКЦ – так при аспектном описании задачи исследователем формулируются все шаги (декомпозиция на уровни, определение компонентного состава КЦ и пр.), необходимые для построения модели в СМ МАРС.

2.1.2 ПОСТРОЕНИЕ МНОГОУРОВНЕВОЙ КОМПЬЮТЕРНОЙ МОДЕЛИ

В данном пункте представлено описание **этапа 3** алгоритма моделирования ФТЗ – **построение компьютерной модели**.

Первым шагом является **определение компонентного базиса будущей модели** – то есть набора компонентов, используемых для формирования КЦ ФТЗ.

Выбор компонентного базиса осуществляется из четырёх подмножеств:

1. ***K_W – компоненты источники,*** позволяющие задать значения какой-либо величине:

На *C*-слое представлены: источник физической величины 

, установка начального значения 

(имеет отображение на L-слое, выполняющее функции измерителя 

), начальное значение 

.

На *L*-слое представлены: регуляторы 
, 
, 
(имеют отображения на *V*-слое), источники значений различных типов (целочисленного 

, вещественного 

, логического 

), источник времени (

или 

), измерители информационных и энергетических переменных 

и 

(получающие данные с *C*-слоя), геометрические компоненты (источник фигуры), источники данных из баз данных и пр.

V-слой: регуляторы, например, 

или 

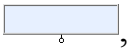
(имеют отображения на *L*-слое: 
, 
.), кнопки и пр.

2. ***K_P – компоненты-преобразователи***, использующие математические модели.

На *C-слое* представлены: готовые объектные модели физических процессов из библиотеки моделей компонентов (в виде алгебро-дифференциальных уравнений), ИМП, геометрические компоненты (профиль), арифметические операторы, специализированные компоненты преобразователи (демпфер, клапан, ёмкость и пр.).

На *L-слое* представлены: интерактивные математико-алгоритмические панели (ИМАП), арифметические операторы, блоки обработки данных (среднее значение, наибольшее значение и пр.), вспомогательные компоненты (перевод из градусов в радианы, расчёт проекции вектора и пр.), компоненты для обработки табличных данных (аппроксиматор, интерполятор и пр.), компоненты-оптимизаторы.

3. ***K_Z – компоненты-измерители***, позволяющие измерять на схеме результаты изменения физической величины.

На *C-слое* представлены: измеритель  , измерители потенциальной и потоковой переменной *Vn*  и *Vb*  (имеют отображения-источники на *L*-слое).

На *L-слое* представлены: компоненты-атрибуты  для любых компонентов, установка начального значения  (источник для *C*-слоя), график  , табло  (имеет отображение на *V*-слое).

На *V-слое* представлены: табло  (имеет отображение  на L-слое), график, и пр.

4. K_U – компоненты управления (алгоритмические конструкции)

Все компоненты данной категории относятся к *L*-слою. Их можно разделить на следующие типы:

1) компоненты для моделирования дискретного поведения объектов (старт  , окончание  , Состояние  , Событие );

2) алгоритмические компоненты: накопитель, логический мультиплексор, ветвление (if-else), информационный (логический) ключ и пр.;

3) логические операторы, операторы сравнения пр. базовые операции;

4) компоненты для управления моделированием: стоп, источник времени, установка шага моделирования и пр.

Следующим шагом данного этапа является **формирование дискретной структуры задачи из компонентов языка МАК** с использованием *компонентов управления* (алгоритмических конструкций, описанных выше) с

учётом *алгоритмического аспекта* анализа задачи. Формируемая диаграмма поведения строится из компонентов «Состояние» и «Событие» описывающими дискретное поведение объекта.

Далее осуществляется **формирование КЦ на С-слое для описания непрерывного поведения в задаче и пространственно-геометрической составляющей задачи**. На С-слое модели располагаются источники для управляемых величин, измерители – для наблюдаемых и преобразователи (ИМП или готовые блоки, например, «Модель твёрдого тела») для математических моделей описываемых физических процессов.

После этого осуществляется пробный запуск и **оценка корректности модели.** В случае некорректной работы модели производится отладка модели и поиск ошибок.

В случае корректной работы модели производится **формирование КЦ на L-слое для реализации сценария исследования модели, обработки и визуализации результатов.**

Заключительным этапом построения модели является **тестирование сценария исследования**, после которого непосредственно производится вычислительный эксперимент и анализ результатов моделирования.

Обобщённая структура многоуровневой модели.

Приведём обобщённую структуру многоуровневой компонентной цепи ФТЗ (рис. 2.4), включающую в себя вышеописанные компоненты. Входные данные с V-слоя передаются на L-слой, где проходят преобразование (например, в ИМАП), и согласно управляющим конструкциям (например, диаграммам состояний) передаются на С-слой. Далее происходит решение системы уравнений, записанных в ИМП или в физических компонентах, результаты которого передаются обратно на L-слой, а затем на V-слой и визуализируются пользователю.

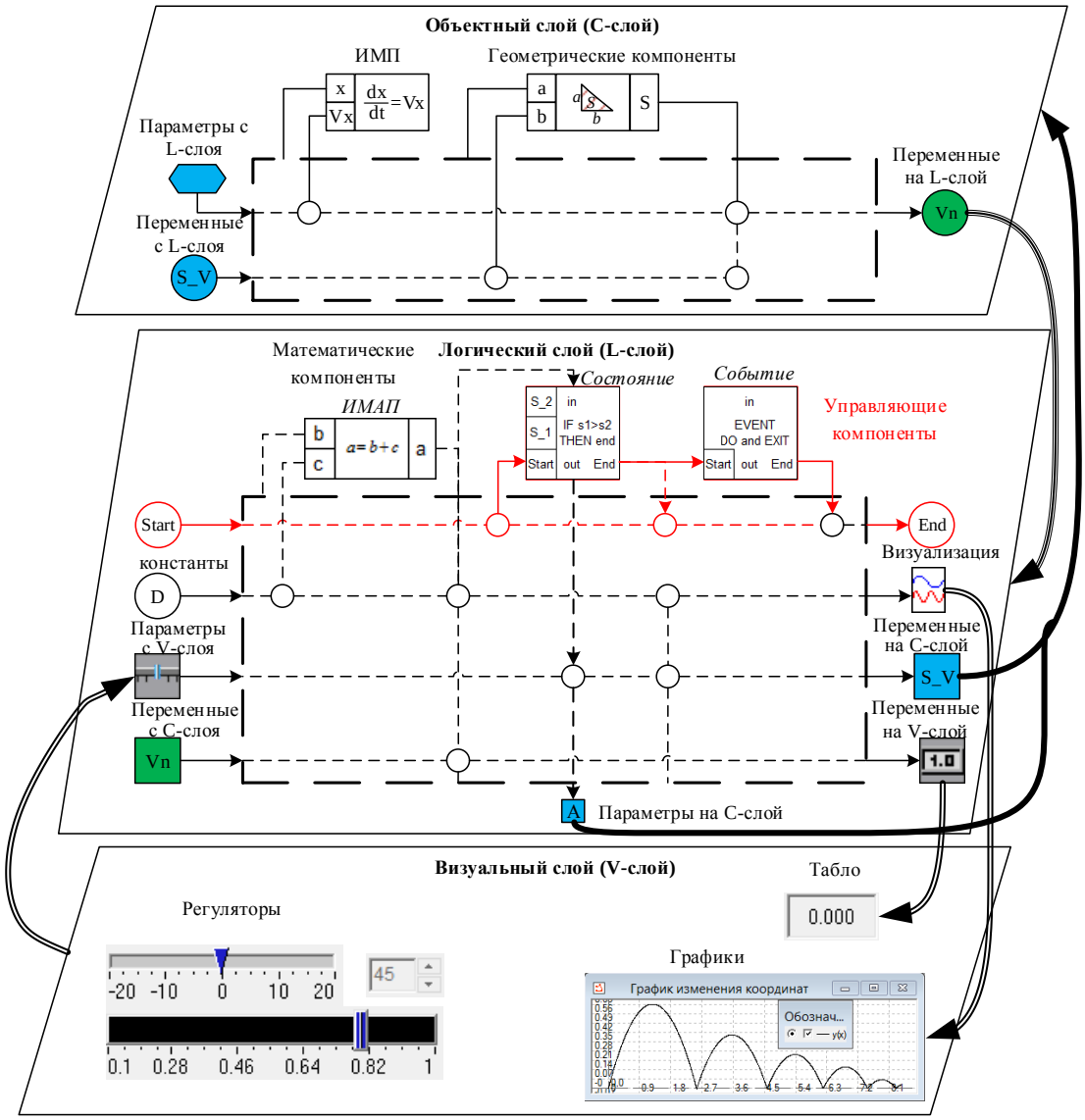


Рисунок 2.4 – Обобщенное представление структуры задачи

2.2 МОДЕЛИРОВАНИЕ ДИСКРЕТНОГО ПОВЕДЕНИЯ ОБЪЕКТОВ В ФИЗИКО-ТЕХНИЧЕСКИХ ЗАДАЧАХ

2.2.1 ТРАДИЦИОННЫЕ ПОДХОДЫ К МОДЕЛИРОВАНИЮ ДИСКРЕТНО-НЕПРЕРЫВНЫХ СИСТЕМ

Для качественного описания большого класса практических задач необходимо учитывать не только непрерывное, но и дискретное поведение систем. Системы, имеющие помимо своего непрерывного поведения (например, физико-химического) ещё и дискретное поведение (алгоритм функционирования), определяющее алгоритм поведения объекта, как уже

упоминалось, называются дискретно-событийными системами или гибридными системами [Колесов, 2012]. Для описания гибридных систем используются различные подходы: конечные автоматы, сети Петри, диаграммы состояний (UML), карты состояний Д. Харела.

Рассмотрим далее традиционные подходы к моделированию дискретного поведения дискретно-непрерывных систем, а именно, конечные автоматы, диаграммы состояния языка UML, гибридные автоматы.

Конечные автоматы

Конечный автомат – абстрактный автомат, число возможных внутренних состояний которого конечно.

Абстрактные автоматы (в теории алгоритмов) – математические «абстракции реально существующих дискретных (цифровых) автоматов, на которых основана современная цифровая вычислительная техника» [Глушков, 1961. С. 4], представляющие собой модели дискретных устройств с одним входом и выходом, находящиеся в каждый момент времени в одном состоянии из конечного множества возможных. На вход абстрактному автомату поступают символы входного алфавита, на выходе – символы выходного алфавита. Теория автоматов была заложена в работе Д. Хаффмана [Huffman 1954] и развита в работах Дж. Мили [Mealy, 1955] и Э. Мура [Moore, 1956], получивших названия *автомата Мили* и *автомата Мура* соответственно. Разница между автоматами Мили и Мура заключается в определении функции выхода, а именно тем, что «функция выходов не зависит от входного символа, т.е. представляет собой отображение $\Phi: Q \to B$» [Закревский, Поттосин, 2007. С. 444], где Q – множество состояний, а B – множество выходных символов.

Представим следующую модель *конечного автомата* (автомата Мили) $M = (A, B, Q, \Psi, \Phi)$, представляющей собой совокупность объектов:

- $A = \{a_1, a_2, \ldots, a_\alpha\}$ – множество входных символов (входной алфавит),
- $B = \{b_1, b_2, \ldots, b_\beta\}$ — множество выходных символов (выходной алфавит),

- $Q = \{q_0, q_1, \ldots, q_\gamma\}$ – множество состояний (внутренний алфавит),
- $\Psi : A \times Q \to Q$ – функция переходов,
- $\Phi : A \times Q \to B$ – функция выходов.

Конечный автомат начинает работу в начальном состоянии q_0, затем считывает по одному символу входного слова, который переводит автомат в новое состояние в соответствии с функцией переходов Ψ с формированием выходного символа (сигнала) в соответствии с функцией выходов Φ.

В литературе [Андерсон, 2004; Закревский, 2007] рассматриваются два способа представления конечного автомата:

- с помощью *графа переходов* (диаграмма состояний);
- с помощью *таблицы переходов* и *таблицы выходов*.

Важным замечанием будет то, что классическая модель конечного автомата игнорирует понятие физического времени, поэтому использование такой модели не является приемлемой для моделирования ФТЗ с дискретно-непрерывным поведением объектов.

Нотации UML для представления поведения гибридных систем

Диаграммы состояний языка (*statechart* diagram) *UML* – диаграммы конечного автомата *state machine diagram*) в *UML* 2.0 – основаны на картах состояния Д. Харела и представляют собой концептуальную модель объекта в виде конечного автомата, характеризующую поведение объекта в форме последовательности его состояний, сменяющих друг друга при выполнении определённых условий.

Конечный автомат (state machine) представляет собой некоторый формальную модель для представления поведения отдельных элементов системы или её целиком. **Поведение** (**behavior**) – последовательность состояний, иллюстрирующая то, как система изменяет значения отдельных характеристик в течение времени своего функционирования.

Под *состоянием* (state) объекта понимается элемент поведения объекта, в течение которого имеет место выполнение некоторого условия [Леоненков,

2004]. Состояние конечного автомата активно, если инициирован переход в него или неактивно в обратном случае (если, например, был инициирован переход из этого состояния в другое). Условное обозначение состояние на диаграмме конечного автомата в UML – прямоугольник с округлёнными краями.

В состоянии определяются:

- действие по входу (*entry action*) – действие, однократно выполняемое в момент входа в данное состояние (входное действие),
- действие по выходу (*exit action*) – действие, однократно выполняемое в момент выхода из данного состояния (выходное действие);
- активность в состоянии (*do activity*) – повторяющееся действие, многократно выполняемое в течение всего времени, данное состояние объекта активно.
- множество откладываемых событий (deferrable events).

Выделяются специальные состояния (**псевдосостояния**): **начальное** (initial state) и **конечное** (final state), – предназначенные для обозначения момента начала и конца процедуры смены состояний.

Начальное состояние (initial state) – частный случай состояния, не содержащего никаких внутренних действий, то есть псевдосостояния, в котором объект находится в начальный момент.

а б

Рисунок 2.5 – Условные обозначения начального и конечного состояний

а – начальное состояние, б – конечное состояние

Сторожевое условие (guard condition) – некоторый предикат, проверяемый на истинность. Записывается в квадратных скобках. Выполнение сторожевого условия инициирует некоторое событие, знаменующее переход в новое состояние [Леоненков, 2004].

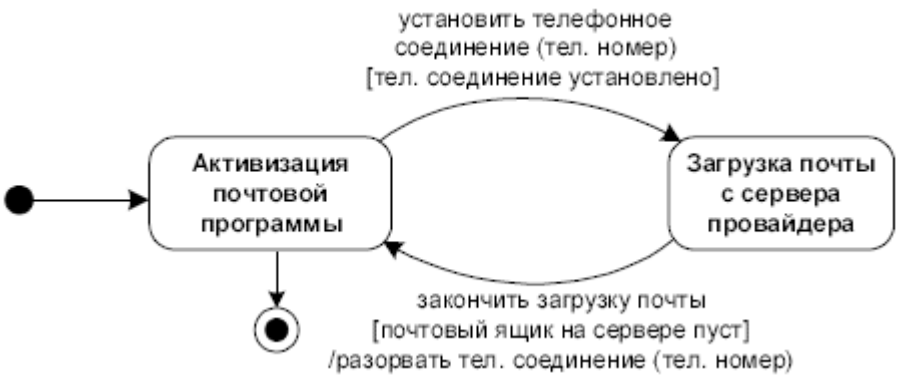


Рисунок 2.6 – Пример диаграммы состояний

Событием называется наступление некоторого факта, инициирующего ***переход*** – изменение состояния объекта. Простой переход (*simple transition*) – отношение между двумя состояниями, указывающее на факт выстраивания их в последовательность. Переходы могут быть **триггерными** (условными) и **нетриггерными** (безусловными).

Перечислим следующие недостатки языка UML:

1) он представляет собой стандарт, использующий графические обозначения для создания моделей системы, поэтому не имеет готовых интерпретаторов;

2) диаграммы состояний не подразумевают наличия непрерывного поведения, задаваемого в явном виде и отделяемого от самой диаграммы: реализация возможна только за счёт записи алгебраической системы уравнений в явном виде (т.е. преобразованной вручную из системы дифференциальных уравнений) в блоке *do activity* – именно такой вариант используется также в диаграммах состояний *stateflow* среды *Simulink* (*Matlab*);

3) модели непрерывного поведения могут задаваться только в символьном виде, что ограничивает исследователя в выборе инструментов для реализации непрерывной модели задачи.

Гибридные автоматы для моделирования дискретно-непрерывных систем

В середине 90-х годов было предложено использование ГА для визуального моделирования и численного исследования сложных динамических систем, которое нашло отражение в работах Колесова Ю.Б., Сениченкова Ю.Б.

ГА представляют собой расширение языка *UML* и сочетают дискретное поведение системы, представляемое в виде конечного автомата с его непрерывным поведением, характерным динамической системе.

Гибридным автоматом H называют кортеж $H=(Q, X, Init, f, Inv, E, G, R)$ [Янченко и др., 2012], где

- **Q** - конечное множество дискретных, а X - непрерывных переменных;
- $Init \subseteq \mathbf{Q}\times\mathbf{X}$ – множество начальных состояний;
- $f : \mathbf{Q}\times\mathbf{X}\rightarrow\mathbf{X}$ – правая часть системы обыкновенных дифференциальных уравнений первого порядка относительно $x\in X$;
- Inv: $\mathbf{Q}\rightarrow 2^{\mathbf{X}}$ множество инвариантов (неизменных на некотором промежутке свойств объекта), связанных с каждым значением переменной $q \subset \mathbf{Q}$;
- $E \subset \mathbf{Q}\times\mathbf{Q}$ – множество дискретных переходов;
- G: $E \rightarrow 2^{\mathbf{X}}$ – множество охранных предикатов, соответствующих каждому переходу $e = (q,q')\in E$;
- R: $E \times\mathbf{X}\rightarrow 2^{\mathbf{X}}$ – множество правил переопределения начальных условий, заданных на каждой дуге $e = (q,q')\in E$ для непрерывных переменных $x\in\mathbf{X}$.

Переменные $x\in\mathbf{X}$ – непрерывные, так как они являются решением дифференциальных уравнений, а переменные $q \subset \mathbf{Q}$ – дискретные, так как определяют конечное множество состояний автомата. Под состоянием гибридного автомата H понимается пара $(q, x)\in\mathbf{Q}\times\mathbf{X}$ [Янченко, 2011].

ГА может быть представлен в виде графа (рис. 2.7).

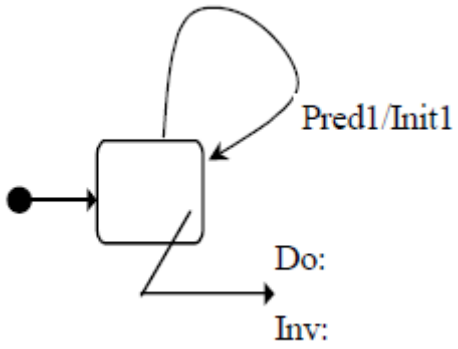


Рисунок 2.7 – Изображение гибридного автомата

Представим **алгоритм работы примитивного автомата** [Сениченков, 2004]:

1. Инициализация.

2. Начало текущего промежутка гибридного времени. Вычисление новых начальных условий и проверка истинности предиката. Если предикат истинен – переход к 4, иначе – начало текущего непрерывного поведения.

3. Решение уравнений непрерывной модели до выполнения условия истинности предиката.

4. Завершающее мгновенное поведение. Инициализация начальных условий для следующего интервала. Переход к 2.

Конец Алгоритма работы примитивного автомата.

2.2.2 ИНТЕРПРЕТАЦИЯ ДИАГРАММ СОСТОЯНИЙ В ЯЗЫК МЕТОДА МНОГОУРОВНЕВЫХ КОМПОНЕНТНЫХ ЦЕПЕЙ

Для моделирования задач с дискретно-непрерывным поведением в инструментальных средах используются следующие подходы: 1) в Rand Model Designer – гибридные автоматы [Сениченков, 2010]; 2) в AnyLogic – диаграммы состояний (statechart) [Стукалин, 2018] идентичные одноимённым диаграммам языка UML; 3) в Simulink – диаграммы состояний (stateflow) – графические и табличные интерфейсы для моделирования логики системы с использованием машин состояний [Десятов, Сирота, 2006]; 4) в SimInTech – конечные автоматы [Baum и др., 2013], 4) в СМ МАРС – общеалгоритмический подход к

моделированию дискретного поведения объектов и систем, заложенный в язык МАК [Дмитриев, Ганджа, 2015].

Представим выполненную интерпретацию диаграмм состояний язык МАК метода МКЦ, направленную на создание инструментов моделирования дискретного поведения гибридных систем более высокого уровня абстракции, чем имеющиеся в СМ МАРС.

Математическое описание диаграмм состояний в языке метода ММКЦ

Диаграммы состояний языка ММКЦ представляют собой кортеж

$$SD = (S, T, P(V), R, D), \text{ где}$$

- $S = \{S_1, S_2, ...S_n\}$ – множество дискретных состояний (соответствующее вершинам графа дискретного поведения);
- $T = \{(S_1,S_2),(S_2,S_3),...\}$, $T \subseteq S \times S$ – множество дискретных переходов (соответствующих ветвям графа – отношению, заданному на множеств S);
- P(V) – множество предикатов, инициирующих дискретный переход;
- $R \subseteq S \times D \times C$, $R = S \rightarrow C$ – соответствие из множества состояний во множество компонентов модели, определяющее каждому дискретному состоянию одну или несколько математических моделей непрерывного поведения;
- $C=\{C_1,C_2,...C_n\}$ – множество компонентов модели, представляющих математическую модель объекта;
- D – множество параметров модели, принятых за *const* во время состояния S_i и передаваемых на вход компонентам C_j.

Диаграммы состояний располагаются на L-слое и являются управляющими конструкциями для КЦ С-слоя, описывающих непрерывное поведение системы, которое может задаваться:

1) схемотехническими моделями, состоящими из блоков низкого уровня абстракции;

2) структурными (физическими) моделями, состоящими из блоков более высокого уровня абстракции;

3) аналитическими моделями, которые задаются в блочно-символьном виде (с помощью ИМП):

- системами обыкновенных дифференциальных уравнений первого порядка, как разрешённые относительно производных $\frac{dx}{dt} = F(x,t)$, так и не разрешённые относительно производных $G(x,t) \cdot \frac{dx}{dt} = F(x,t)$,
- системы алгебраических уравнений $F(x,t) = G(x,t)$.

Базовым элементом (единицей) дискретного поведения объектов в имитационной модели ФТЗ являются: 1) «состояние», характеризующееся наличием одновременного непрерывного поведения и 2) «событие», характеризующееся мгновенными качественными или количественными изменениями во всей модели задачи или её части. Под *состоянием* объекта понимается элемент поведения объекта, в течение которого выполняется некоторое условие, заданное предикатом. Состояние может быть описано через совокупность свойств (связей и атрибутов) и их текущих значений.

Термин «событие» характерен только для дискретного поведения, а «состояние» как для дискретного (состояние конечного автомата), так и для непрерывного (классическое понятие состояния динамической системы в момент времени t), поэтому в контексте языка ММКЦ будем разделять *дискретное* и *непрерывное* состояния. Под ***непрерывным*** *состоянием* будем понимать совокупность всех атрибутов объекта (параметров и переменных, т.е. вектора переменных состояния) в текущий момент времени и его математической модели (закона «эволюции»). Под ***дискретным*** *состоянием* будем понимать совокупность параметров (но не переменных) объекта и его математической модели непрерывного поведения, имеющего место на дискретном интервале времени, в течение которого эти параметры считаются неизменными и выполняется некоторое ***условие***. Дискретное состояние объекта можно рассматривать как некоторую абстракцию, накладываемую на фрагмент его непрерывного поведения.

На поведенческой схеме (*L*-слое) компонентная схема дискретного состояния посредством своих структурных связей обуславливает описание компонентной схемы непрерывного поведения объекта той или иной математической моделью С-слоя при заданных параметрах и режиме работы. Так дискретное состояние определяет не только количественный (значение параметров) аспект в поведении объекта, но и качественный (математическая модель или изменение структуры КЦ, например, за счёт переключения ключей/клапанов). Так называемые начальное и конечное состояния будем рассматривать в качестве сигнала к началу и завершению работы компьютерной модели, относя их к категории событий. Под *событием* будем понимать изменение значения внутреннего (атрибута свойственного непосредственно объекту) или внешнего (несвойственной объекту) параметра моделируемой системы, инициирующее переход от одного дискретного состояния к другому. Будем рассматривать только триггерные переходы, считая все переходы условными.

Модели компонентов диаграмм состояний

1. Компонент «Начало» (рис. 2.8) соответствует начальному состоянию в диаграмме состояний и выполняет функцию запуска диаграммы состояний путём передачи значения «истина» при начале вычислительного эксперимента.

Рисунок 2.8 – Компонент «Начало»

Связью компонента «Начало» является располагаемая на L-слое МКМ $S_1 = b_1 \eta^+ n_1 \rightarrow start$, предназначенная для передачи компонентом значения «истина» по ветви b_1 в КЦ.

2. Компонент «Окончание» (рис. 2.9) соответствует конечному состоянию в диаграмме состояний и останавливает проведение вычислительного эксперимента при получении значения «истина» на вход.

Рисунок 2.9 – Компонент «Окончание»

Связью компонента «Окончание» является располагаемая на L-слое МКМ $S_1 = b_1\eta^- n_1 \rightarrow end$, предназначенная для получения компонентом значения типа булевой переменной по ветви b_1 из КЦ.

3. Компонент «Состояние» (рис. 2.10) соответствует дискретному состоянию в диаграмме состояний.

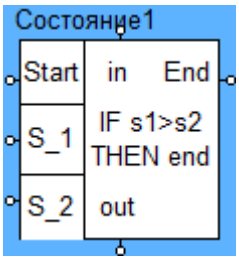


Рисунок 2.10 – Компонент «Состояние»

Связями компонента «Состояние» являются следующие:

- $S_1 = b_1\eta^- n_1 \rightarrow start$, предназначенная для инициализации компонента (активации текущего состояния);
- $S_2 = b_2\eta^- n_2 \rightarrow s1$, $S_3 = b_3\eta^- n_3 \rightarrow s2$ (и последующие динамические добавляемые связи), предназначенные для передачи в компонент значений переменных, используемых в проверке истинности выражении (предиката), инициализирующего переход в следующее состояние;
- $S_4 = b_4\eta^- n_4 \rightarrow in$, предназначенная для передачи в компонент значения параметра, которое будет использоваться для параметризации модели на С-слое;
- $S_5 = b_5\eta^+ n_5 \rightarrow out$ – для передачи в дальнейшую КЦ полученного компонентом на вход S_4 значения параметра;

4. Компонент «Событие» (рис. 2.11) соответствует одноимённому элементу дискретного поведения объекта. Его функция заключается в

однократной передаче значения, полученного на вход in, через вход out, при условии активизации компонента (получения значения «истина» на вход Start.

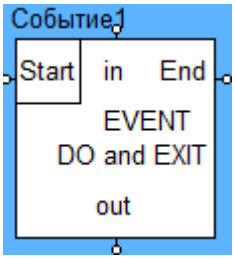


Рисунок 2.11 – Компонент «Событие»

Связями компонента «Событие» являются следующие:

- $S_1 = b_1\eta^- n_1 \rightarrow start$, предназначенная для инициализации компонента (активации текущего состояния);
- $S_2 = b_2\eta^- n_2 \rightarrow in$, предназначенная для передачи в компонент значения параметра, которое будет использоваться для параметризации модели на С-слое;
- $S_3 = b_3\eta^+ n_3 \rightarrow out$ – для передачи в дальнейшую КЦ полученного компонентом на вход S_4 значения параметра;

Пример модели с использованием разработанных компонентов для моделирования дискретного поведения

На рис. 2.12 представлена рассматриваемая ранее модель движения материальной точки с использованием разработанных компонентов. Ранее в пункте 1.1.3 на рис. 1.19 была представлена аналогичная схема, выполненная стандартными компонентами языка МАК. Несмотря на то, что диаграмма состояний состоит из 3 блоков «Состояние», они все управляют одной аналитической моделью, представленной системой дифференциальных уравнений $\frac{dx}{dt} = V, \frac{dV}{dt} = a$, записанном в одном (а не в нескольких, как например в *Simulink* и *Rand Model Designer*) компоненте – ИМП. Это возможно, за счёт разделения дискретной модели от непрерывной и позволяет, в частности, экономить машинные ресурсы при работе модели.

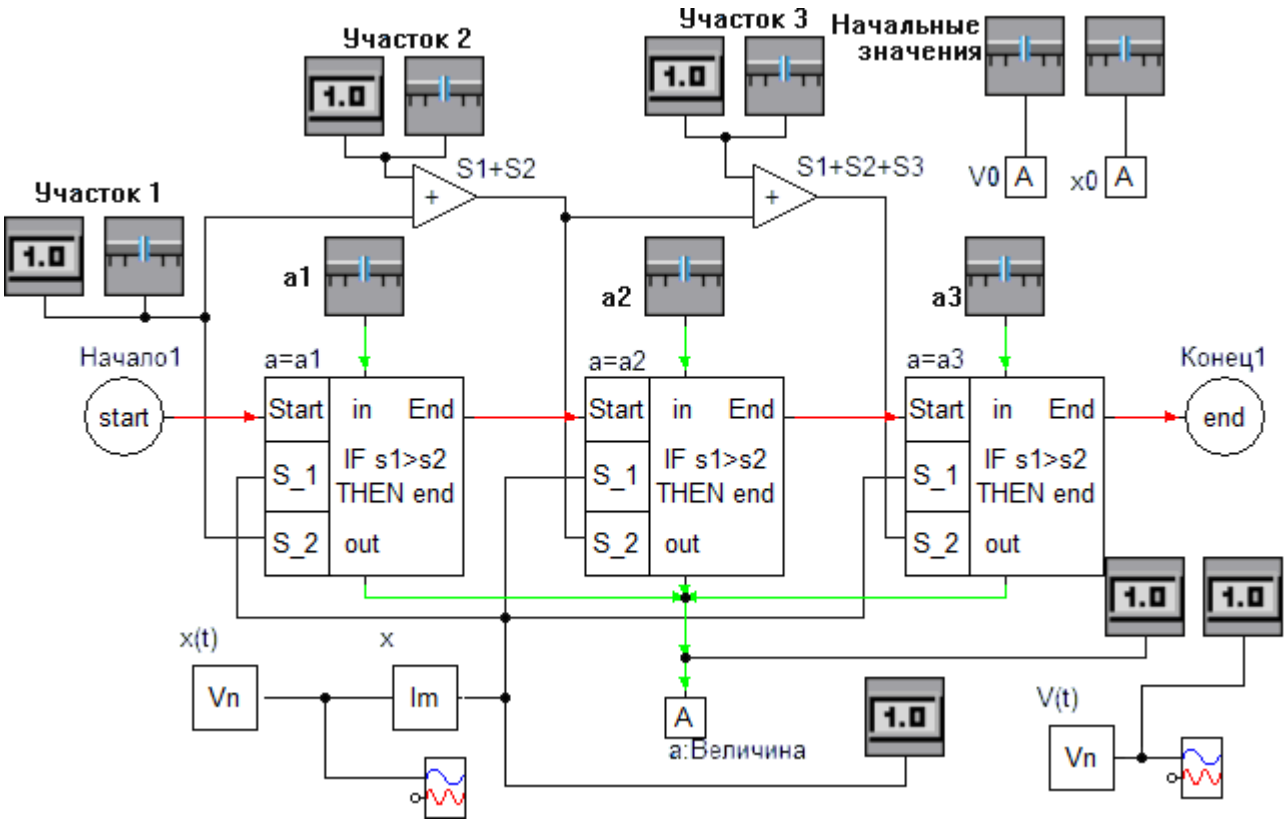


Рисунок 2.12 – КЦ L-слоя с использованием компонента «Состояние»

Опишем подробнее алгоритм работы компонента «Состояние». При подаче на узел *Start* значения «истина» компонент начинает работу: он передаёт поступающие на входы *In* (сверху) значения на соответствующие выходы *Out*, которые затем передаются через компоненты-атрибуты на С-слой в соответствующую математическую модель.

На каждой итерации работы модели выполняется проверка условия <event>, которое записывается пользователем в символьной форме. Если условие выполняется, текущий компонент прекращает свою работу и передаёт значение «истина» на выход *End* – где оно поступает на вход *Start* следующего компонента («Состояние» либо «Конец1» – для остановки работы модели).

Такая КЦ в явном виде изображает количество дискретных состояний объекта и условия перехода между ними в отличие от используемого в МАК общеалгоритмического подхода (рис. 1.19) более сложного и менее наглядного.

В обобщённом виде структура МКМ ФТЗ с применением диаграмм состояний представлена на рис. 2.13.

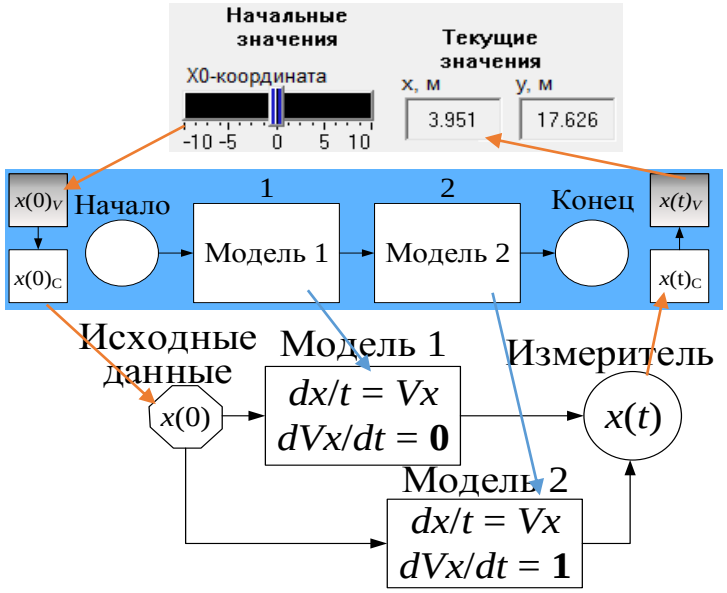


Рисунок 2.13 – Обобщённая МКМ дискретно-непрерывной ФТЗ

Использование ДС языка ММКЦ позволяет:

- компактно и наглядно описывать мгновенные количественные изменения значений переменных МКМ и качественные изменения непрерывного поведения объекта в ФТЗ;
- задавать последовательности дискретных операций аналогичных машине состояний UML;
- задавать компонентную структуру модели в ветвящихся КЦ;
- явно отделять непрерывное поведение объекта от дискретного, что облегчает работу численных методов (КЦ L-слоя рассчитываются имитационным ядром СМ МАРС, а С-слоя – вычислительным).

Основное отличие предлагаемых **диаграмм состояний** языка МАК от **гибридных автоматов**, реализованных, например, в *Rand Model Designer*, и от диаграмм *stateflow* (*Simulink*) состоит в следующем:

1) в теории гибридных автоматов непрерывное состояние «интегрировано» в дискретное (системы уравнений мат. модели прописываются внутри блока «состояние»), в то время как диаграммы состояний языка ММКЦ **представляют собой управляющую конструкцию L-слоя**, которая надстраивается над одной или несколькими математическими моделями С-слоя. Как следствие несколько «состояний» в языке МАК могут соответствовать одной математической модели,

но с разными параметрами, что по сравнению с гибридными автоматами позволит сократить объём машинных ресурсов, требуемых для работы модели (т.к. в каждое «состояние» гибридного автомата вносится в модель заново и хранится отдельно);

2) диаграммы состояний языка МАК **могут взаимодействовать с моделями С-слоя, представленными в структурном виде** (например, бак, труба и пр. – при использовании подхода «физического» моделирования) в отличие от гибридных автоматов, которые работают только с аналитическими моделями;

3) диаграммы состояний как единицы языка МАК **могут сочетаться с его другими элементами** (например, с сетями Петри), в то время как гибридный автомат среды *Rand Model Designer* является единственным средством моделирования дискретного поведения систем;

4) диаграммы состояний **могут использоваться в качестве управляющей конструкции** не только для компонентов С-слоя (ИМП, структурно-функциональные блоки), но и **для компонентов L-слоя** (запись/чтение из файлов, баз данных и пр.).

5) диаграммы состояний языка ММКЦ реализуют механизм **компенсации амплитудно-временной погрешности**, который представим далее.

Описание программной реализации диаграмм состояний представлено в пункте 3.1.1 данной работы.

2.2.3 КОМПЕНСАЦИЯ АМПЛИТУДНО-ВРЕМЕННОЙ ПОГРЕШНОСТИ

Ввиду дискретного характера машинных вычислений непрерывные модели систем решаются с некоторым (выбранным исследователем) шагом дискретизации, что требует от исследователя (для сохранения точности вычислений и избегания ошибок) учёта некоторых особенностей, например, [Машинная арифметика …]:

1) исчезновение порядка (потеря значимости – *underflow*), возникающее, например, при сложении чисел с плавающей запятой разного порядка (результат может быть равен большему из слагаемых);

2) арифметическое переполнение (*overflow*) при превышении значением результата арифметической операции максимально допустимого значения для переменной;

3) возможный некорректный результат проверки значений погрешности больших значений на абсолютную малость (поэтому рекомендуется использовать проверки на относительную малость);

4) возможный некорректный результат сравнения вещественных чисел на равенство (в связи с этим рекомендуется использовать процедуру приближённого сравнения) и пр.

Использование диаграмм состояний в дискретно-непрерывных моделях предполагает проверку значения некоторого значения переменной на каждой итерации (что может возвращать некорректный результат). Такой проверкой может быть, например, проверка на равенство нулю координаты *y* положения тела при моделировании его падения. Ввиду дискретности шага моделирования такое условие формулируется как $y \le 0$, а не $y = 0$. При этом срабатывание условия (смена режима гибридной системы) произойдёт в момент $y = -\Delta_y$ (где Δ_y – некоторое малое число), что может привести к появлению накапливаемой (в случае использования полученного значения в дальнейших расчётах) погрешности результатов моделирования. Обозначим эту погрешность как **амплитудно-временную погрешность** – погрешность как значений переменных вектора состояния динамической системы, так погрешность по времени (смещение вектора состояния динамической системы во временном пространстве), – и рассмотрим её на следующих примерах.

Имеем модель штангового глубинного насоса, рассматриваемую в п. 4.2.1 Главы 4, представленную в виде системы последовательно соединённых элементов: штока, колонны штанг и плунжерной пары. Возвратно-поступательное перемещение этой системы описывается дифференциальным уравнением продольных колебаний однородного стержня. Шаг времени моделирования выбран следующий $\Delta t = 0.01$ с. В модели выделяется непрерывное поведение – движение тел вверх или вниз под воздействием сил, – и дискретное

– смена направления движения. Дискретное поведение модели привязано к значению скорости штока v_{Stock}. Условием перехода является $v_{Stock} = 0$, которое с учетом дискретности машинной арифметики [Машинная арифметика…] формулируется как $v_{Stock} \geq 0$ или $v_{Stock} < 0$. Дискретная модель в рассматриваемой задаче определяет выбор выражения для расчёта скорости плунжера v_1, исходя из значения скорости штока v_{Stock} (тем самым реализуется запаздывание в движении плунжера в моменты смены знака скорости штока) по следующему выражению (2.1):

$$v_1(t) = \begin{cases} 0, if\ (P_1(t) < P_{ж} + P_{mp.c.n.} + P_{mp.c.n.} - P_{np} \cdot f) \wedge (v_{stock}(t) \geq 0) \\ \dfrac{P_1(t) - (P_{ж} + P_{mp.c.n.} + P_{mp.c.n.} - P_{np} \cdot f)}{\beta}, if\ (P_1(t) \geq P_{ж} + P_{mp.c.n.} + P_{mp.c.n.} - P_{np} \cdot f) \wedge (v_{stock}(t) \geq 0) \\ 0, if\ (P_1(t) \geq P'_{mp.c.n.} + P'_{mp.c.n.}) \wedge (v_{stock}(t) < 0) \\ \dfrac{P_1(t) - P'_{mp.c.n.} + P'_{mp.c.n.}}{\beta}, if\ (P_1(t) < P'_{mp.c.n.} + P'_{mp.c.n.}) \wedge (v_{stock}(t) < 0) \end{cases} \quad (2.1)$$

(2.1)Таким образом движение плунжера (вверх и вниз) начинается только тогда, когда усилие, приложенное к нему, превысит некоторое значение. График зависимости $v_{Stock}(t)$ представлен на рис. 2.14.

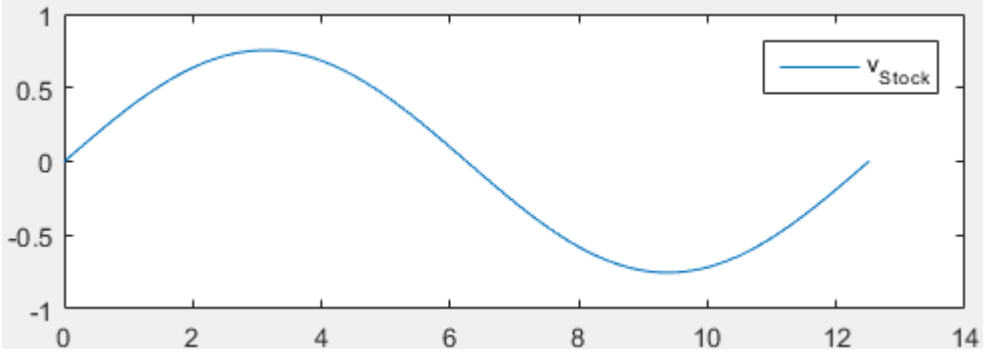
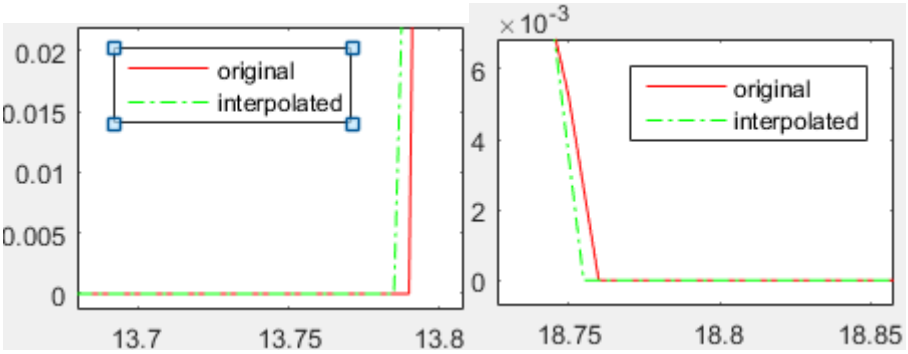


Рисунок 2.14 – График зависимости скорости штока от времени

Первое срабатывание алгоритма (1) происходит при $v_{Stock}(6.26) = -0.0019$, при этом предыдущее значение $v_{Stock}(6.25) = +0.0019$. Очевидно, что срабатывание алгоритма должно было произойти при значении $v_{Stock}(6.255) = 0$. Таким образом, значение всех переменных, рассчитанных на этой итерации работы модели, будет присвоено с незначительной погрешностью, обусловленной разницей времени $\Delta t = 0.005$ равной половине шага моделирования (имеем погрешность по времени – «временну́ю» составляющую погрешности – рис. 2.15), а также все переменные в расчёте которых участвует v_{Stock} и v_1 получат

незначительно большее значение, исходя из того, что как минимум входное (для этих переменных) значение v_{Stock} равно не 0, а -0.0019 (имеем погрешность по значению – «амплитудную» составляющую). Значения данных погрешностей могут быть уменьшены за счёт понижения шага моделирования, однако совсем избавиться от них при стандартном подходе невозможно – при этом эти погрешности накапливаются в модели с течением времени и в таком случае уже могут оказать ***значительное*** влияние на корректность результатов моделирования. Для компенсации такой амплитудно-временной погрешности в данной работе предлагается алгоритм на основе методов решения задачи обратной интерполяции, который будет описан далее. На следующих рисунках красной сплошной линией представлены результаты моделирования без использования данного алгоритма, зелёной штрихпунктирной – с использованием.

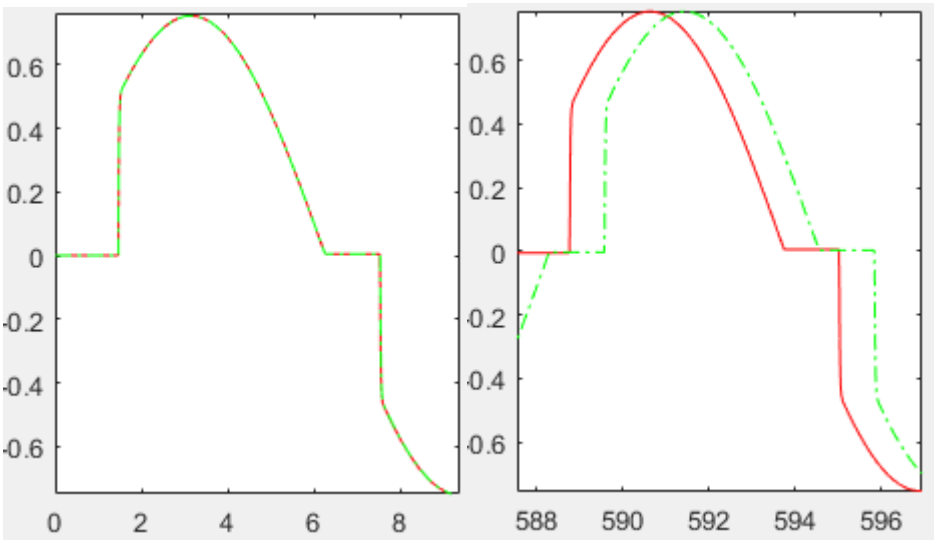


а *б*

Рисунок 2.15 – Иллюстрация погрешности по времени

а – на интервале t = [13.7, 13.8] с, *б* – на интервале t = [18.75, 13.85] с

Продемонстрируем накопление погрешности на примере смещения графика зависимости скорости штока от времени (рис. 2.16) в отрезки времени t = [0, 9] с и t = [588, 597] с соответственно.

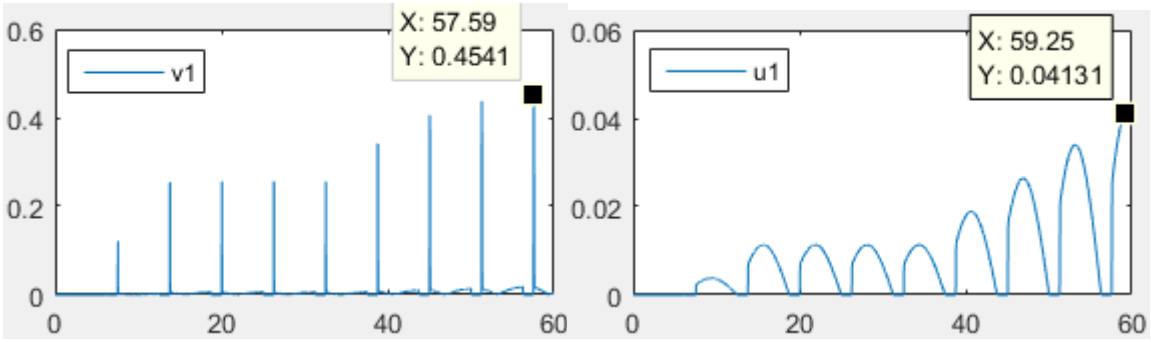


а *б*

Рисунок 2.16 – Смещение графика зависимости скорости штока от времени

а – на интервале t = [0, 10] с, *б* – на интервале t = [588, 598] с

На рис. 2.17 иллюстрируется рост разности между значениями скорости $v_1(t)$ (а), перемещения $u_1(t)$ (б) плунжера в одноимённых узлах t_i при использовании диаграмм состояний с механизмом повышения точности и без него.



а *б*

Рисунок 2.17 – Графики разности между значениями

а –скорости плунжера, *б* – положения плунжера

Гармонический характер графика разности значения объясняется выражением (2.1), в котором переменной v_1 (скорость плунжера) присваивается 0 в том случае, если значение приложенных к нему усилий не превышает некоторого значения. В эти промежутки времени (с учётом смещения) значения скорости плунжера в обоих случаях совпадают.

Отметим, что рост погрешности затрагивает и другие переменные, зависящие от v_1, например, значение нагрузки *P(t)* на шток (рис. 2.18).

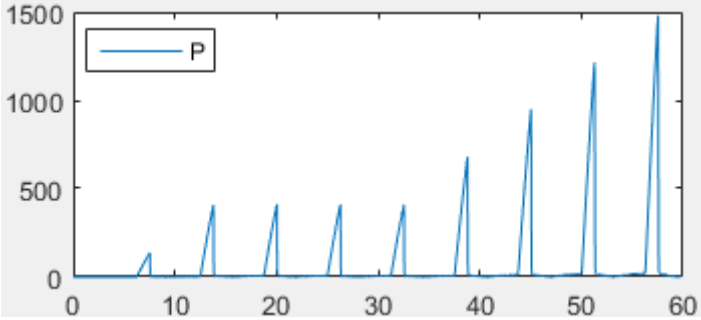


Рисунок 2.18 – График роста разности значения нагрузки на шток

Реализованный механизм компенсации амплитудно-временной погрешности при смене состояний дискретного поведения заключается в следующем. Напомним, что согласно графику зависимости скорости штока от времени (рис. 2.14, представленные ранее) срабатывание перехода в диаграммах состояний по выражению (2.1) происходит при $v_{Stock}(6.26) = -0.0019$, при этом предыдущее значение $v_{Stock}(6.25) = +0.0019$, в то время как должно было произойти при значении $v_{Stock}(6.255) = 0$. При использовании алгоритма компенсации амплитудно-временной погрешности в момент ($t = 6.26$) поступления на вход S_i значения $v_{Stock}(6.26) = -0.0019$, компонент «Состояние» осуществляет перерасчёт всех переменных вектора решений по интерполяционной формуле Лагранжа (2.2) [Press, 1997], для узла $v_{Stock}(t) = 0$:

$$x = \sum_{k=0}^{n} \frac{\prod_{j \neq k} (y - y_i)}{\prod_{j \neq k} (y_k - y_i)} \cdot x_k \quad (2.2),$$

где x – узел интерполяции, y – значение функции в соответствующем узле.

Таким образом решается задача обратной интерполяции и вектор состояния динамической системы возвращается на некоторую часть шага моделирования назад, после чего работа модели продолжается. В табл. 2.1 представлены результаты пересчёта значений некоторых переменных.

Таблица 2.1 – Результаты пересчёта переменных при t = 6.26

	№ итерации	t	y_{stock}	v_{stock}	u_1	v_1	P_1
до пересчёта	627	6.260	2.999981	-0.00189	2.613814	0	23 099.778
после пересчёта	627	6.255	2.999990	0	2.613788	0	23 101.918

Как видно из таблицы величина погрешностей не велика, однако их аддитивность с течением времени приводит к снижению точности модели (см. рис. 2.15–2.18 выше). С момента пересчёта происходит сдвиг расчётной сетки [Thompson, 1985] модели (изменение её шага), построение которой осуществляется алгебраически, за счёт однократного изменения шага моделирования (рис. 2.19).

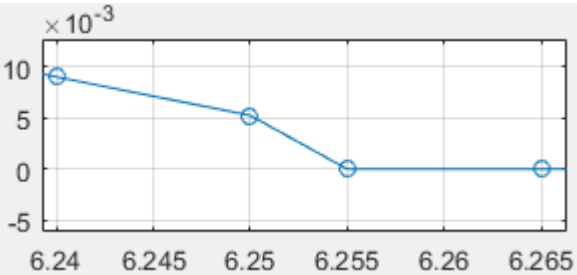


Рисунок 2.19 – Сдвиг расчётной сетки модели

Алгоритм компенсации амплитудно-временной погрешности подразумевает следующие опциональные варианты проведения дальнейших расчётов:

а) продолжение расчёта по смещённой сетке (после однократного изменения шага) с прежним шагом *dt* (параметры *reCompute*, *goBack* алгоритма равны 0) – см. рис. 2.19 выше,

б) расчёт следующей итерации в узле старой сетки с сохранением значений в промежуточном узле (*goBack* = 0, *reCompute* = 1), в таком случае имеем смешанную сетку с переменным шагом вокруг узлов срабатывания переходов из одного дискретного состояния в другое (рис. 2.20);

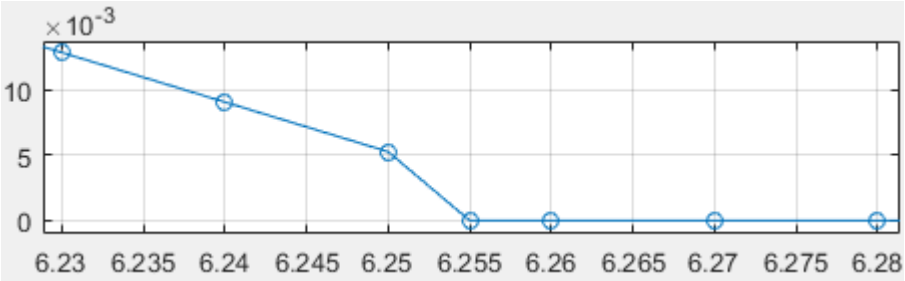


Рисунок 2.20 – Переход к смешанной расчётной сетке

в) расчёт следующей итерации работы модели с дальнейшей заменой (перерасчётом) вектора решения в смещённом узле с целью возврата к исходной расчетной сетке (*goBack* = 1, *reCompute* = 1); в этом случае новая расчётная сетка в точности идентична оригинальной – однако значения вектора решения модели в этих узлах пересчитаны (рис. 2.21).

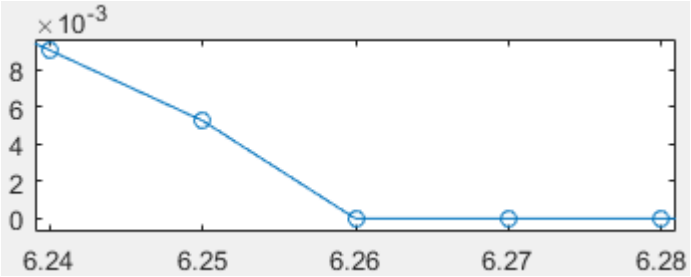


Рисунок 2.21– Возвращение к исходной расчётной сетке

Алгоритм компенсации погрешности представлен на рис. 2.22.

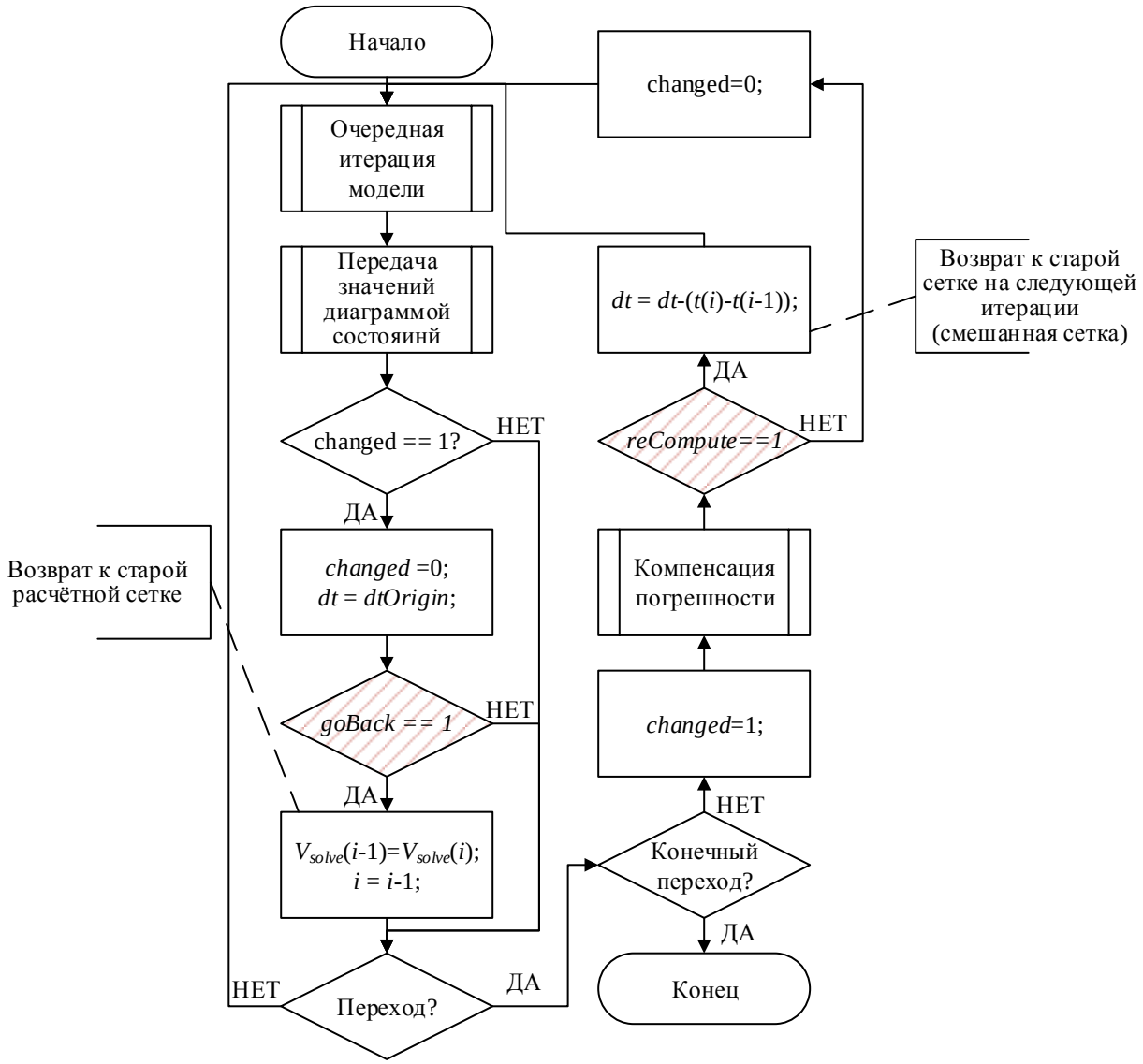


Рисунок 2.22 – Алгоритм компенсации амплитудно-временной погрешности

В приведённом выше примере можно наблюдать только временну́ю составляющую погрешности, так как значение v_{Stock} скорости штока не участвует в расчёте других значимых переменных модели. Для иллюстрации амплитудной составляющей представим следующий пример.

Рассмотрим движение тела массы m = 1 кг, брошенного под углом $\alpha = 45^{\circ}$ к горизонту с начальной скоростью v_0 = 5 м/с, совершающего отскоки после вязкого удара о землю (y = 0). Подробно данная задача рассматривается в пункте 4.2.2. Исследователи отмечают, что «задача прыгающего мячика является одним из самых распространённых примеров гибридного поведения и иллюстрации смены состояний» гибридной системы (ГС) [Шорников, 2009]. Из словесного

портрета задачи видны две чередующиеся фазы – полёт и отскок. Непрерывное движение тела во время полёта может быть описано системой уравнений (2.3):

$$\frac{dx}{dt} = V_x, \frac{dy}{dt} = V_y, \frac{dV_y}{dt} = -g, \frac{dV_x}{dt} = 0 \quad (2.3).$$

Дискретное событие (отскок) представляет собой мгновенное дискретное действие (длительностью отскока можно пренебречь), в результате которого происходит изменение знака вертикальной составляющей скорости на противоположный (2.4):

$$v_y(t) = \begin{cases} -v_y(t) \cdot k, & if\ (v_y(t) < 0) \wedge (y(t) \le 0) \\ v_y(t), else \end{cases} \quad (2.4),$$

где k – коэффициент затухания, $y(t)$, $v_y(t)$ – соответственно высота на которой находится тело и проекция скорости тела на ось OY в момент времени t.

Таким образом дискретно-непрерывное поведение тела в данной задаче представляет собой последовательность фаз непрерывного поведения, с рассчитываемыми начальными (граничными) условия. При построении модели такой дискретно-непрерывной задачи с использованием как диаграмм состояний, так и гибридных автоматов можно обнаружить накапливающуюся амплитудно-временную погрешность (рис. 2.23).

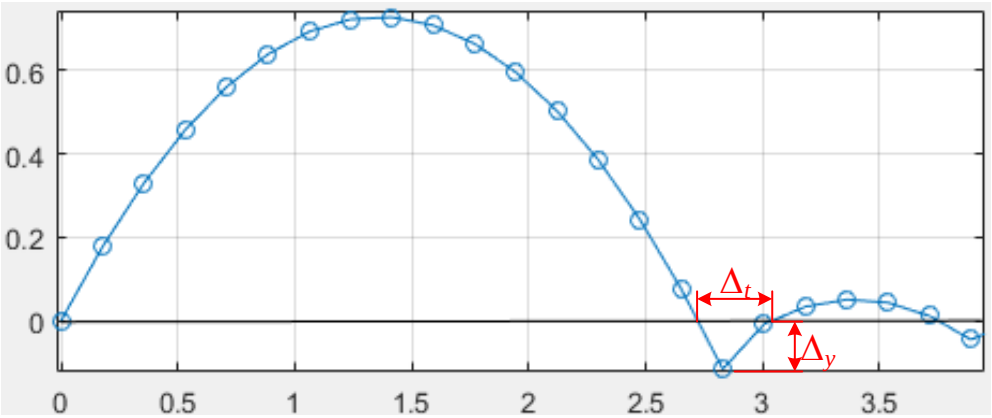


Рисунок 2.23 – Траектория движения тела при шаге dt = 0.05

На рисунке 2.23 амплитудная составляющая обозначена как Δ_y, а временна́я – Δ_t. Источником возникновения данной погрешности являются особенности машинных вычислений, рассмотренные ранее, вследствие которых срабатывание условия перехода (2.4) происходит в момент времени t = 0.85,

когда высота положения тела $y(0.85)$ равна -0.1157, а не 0 (имеем дело с так называемой «ложной точкой переключения» [Шорников, 2009]). Данная разность и составляет **амплитудную** составляющую погрешности Δ_y, которая увеличится на некоторую величину после следующей смены дискретного состояния. В случае, когда коэффициент затухания k=1 (идеальный незатухающий отскок), данная погрешность приведёт к постепенному снижению амплитуды отскоков тела, чего в принципе не должно быть (ввиду упрощения модели). В случае k<1 данная погрешность может быть ошибочно принята за составляющую затухания (ввиду малости своего значения при достаточно маленьком шаге моделирования), однако она оказывает существенное влияние на результаты моделирования при продолжительной работе модели (см. пример движения штангового глубинного насоса выше или задачу о падающем теле по ступенчатой поверхности – Глава 4).

Временна́я составляющая погрешности обусловлена тем, что движение тела вверх выше оси OY начинается не в точке приземления, а дальше на величину Δ_t. Эта погрешность также обладает аддитивным характером (рис. 2.24).

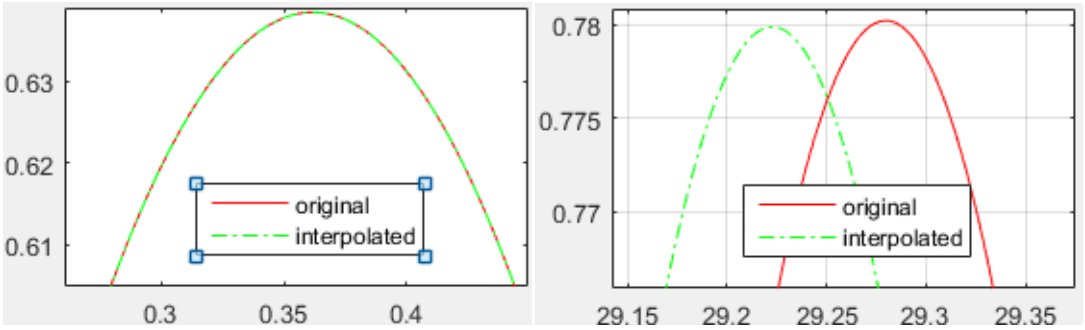


Рисунок 2.24 – Графики y(x) при k=1 и dt = 0.001 с

В работе [Колесов, Сениченков, 2016] при использовании гибридных автоматов для моделирования подобной задачи предлагается присвоить переменной $y(t)$ значение 0 в момент отскока с целью устранения данной погрешности. Однако, по нашему мнению, данная процедура не только снизит погрешность, но наоборот приведёт к «рассинхронизации» непрерывного состояния системы и росту погрешности, т.к. компенсация таким образом осуществляется только для одной переменной, а не всего вектора решения.

Продемонстрируем это на примере сравнения моделей, построенных с шагом моделирования $dt = 10^{-3}$ и коэффициентом $k = 0.6$ с эталонной моделью. В качестве эталона возьмём модель без компенсации погрешностей большей точности – с шагом моделирования $dt = 10^{-6}$. На рис. 2.25 представлены следующие графики: 1) без компенсации погрешностей с точностью $dt = 10^{-6}$ – синяя пунктирная линия, 2) с компенсацией по предложенному в данной работе алгоритму – красная сплошная линия, 3) с подходом предложенным в работе [Колесов, Сениченков, 2016] – зелёная штрихпунктирная линия.

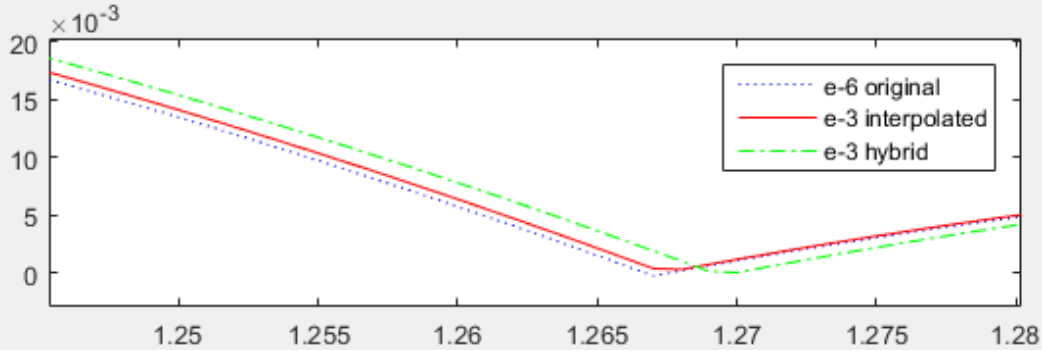


Рисунок 2.25 – Сравнение зависимостей y(t) после 2 отскока

Сравним точность результатов моделирования посредством сравнения вектора состояния системы. В качестве критериев сравнения возьмём среднее абсолютное значение отклонения (САО) в узлах зависимостей y, v_y и среднеквадратичное отклонение (СКО) в этих же узлах (на интервале соответствующему после первого и после второго отскока). Для выполнения данной процедуры из табличной зависимостей, построенных с шагом $dt = 10^{-6}$ были удалены лишние (промежуточные узлы). Результаты сравнения представим в табл. 2.2, округлив значения до 9 знака.

Таблица 2.2 – Сравнение погрешности моделей после первого отскока

Модель	Δy, м	Δv_y, м/с	δy	δv_y
Среднее абсолютное отклонение				
Алгоритм компенсации	**0.002938352**	**0.024430759**	0.693%	1.384%
Гибридный автомат	0.002939542	0.024440639	0.693%	1.384%
Без компенсации	0.002941918	0.024440639	0.694%	1.384%
Среднеквадратичное отклонение				
Алгоритм компенсации	**0.000131087**	**0.010356617**		
Гибридный автомат	0.000131136	0.010356654		
Без компенсации	0.000131267	0.010356654		

В таблице выделены полужирным наименьшие значения погрешностей и подчёркнуты первые отличающиеся в них значащие цифры.

Поясним, что на данном участке оцениваемые параметры характеризуются следующим образом – табл. 2.3.

Таблица 2.3 – Характеристика значений рассматриваемых переменных

Значение	y, м	$\|v_y\|$, м/с
Минимальное	8.522298402e-04	0.002535716
Максимальное	0.636599089	3.529064284
Среднее	0.424115676	1.765795186

Сравним погрешности на 2 участке – после первого отскока и до приземления (табл. 2.4).

Таблица 2.4 – Сравнение погрешности моделей после второго отскока

	y, м	v_y, м/с	δy	δv_y
Среднее абсолютное отклонение				
Алгоритм компенсации	**0.002921867**	**0.049572730**	2.760%	2.8%
Гибридный автомат	0.003634378	0.061190743	3.432%	3.5%
Без компенсации	0.003281983	0.056537486	3.100%	3.2%
Среднеквадратичное отклонение				
Алгоритм компенсации	**0.000184156**	**0.012177116**		
Гибридный автомат	0.000228907	0.013512609		
Без компенсации	0.000199540	0.012182951		

Также поясним, что на данном участке оцениваемые параметры характеризуются следующим образом – табл. 2.5.

Таблица 2.5 – Характеристика значений рассматриваемых переменных

Значение	y, м	$\|v_y\|$, м/с
Минимальное	9.773976672e-05	0.003470203
Максимальное	0.159150476	1.762329797
Среднее	0.105883542	0.881671989

Как видно из сопоставления таблиц после третьего дискретного состояния общая погрешность выросла с δ_y=0.69% до δ_y=3.10% без использования алгоритма компенсации и до δ_y=2.76% – с использованием. Эта разница (0.34%) и представляет собой амплитудно-временну́ю погрешность, которую компенсирует предложенный алгоритм (после 1 отскока разница составляла 0.001%, что свидетельствует о её росте с течением времени). Общий рост погрешности связан с разницей шага моделирования (интегрирования) рассматриваемой модели и эталонной.

Необходимо отметить в работах зарубежных [Mosterman, 1999, Park, 1996, Cellier, 1979, Gear, 1981] и отечественных [Сениченков, 2004, Шорников, 2009] исследователей подобная проблема ставилась более широко – как корректное обнаружение событий гибридных систем. Предлагаемые решения в целом сводятся к 2 вариантам:

1) замедление шага моделирования (интегрирования) при приближении к границе режима гибридной системы – в частности в работе [Шорников, 2009] предлагается алгоритм управления шагом интегрирования, обеспечивающий экспоненциальное асимптотическое приближение к границе режима ГС, а также приводится его сравнение с другими аналогичными методами;

2) использование методов поиска корней нелинейных уравнений вида $f(x) = 0$ (обычно метод половинного деления) при пересечении границы режима гибридной системы. В этом случае задача обнаружения события гибридной системы обычно [Bruck, 1996, Lui, 1999] делится на 2 этапа: фаза обнаружения (пробный расчёт следующей итерации и в случае срабатывания условия перехода – переход к следующей фазе) и фаза локализации (применение численного метода для уточнения точки смены режима).

Большинство методов, осуществляющих замедление при приближении к границе смены режима ГС, не обеспечивает выбор величины шага, предотвращающей пересечение границы события, при этом их использование способно сильно влиять на скорость расчёта компьютерной модели.

Подходы второй группы при использовании стандартных численных методов имеют опасность пропуска точки переключения для немонотонных функций [Колесов, 2004]. Некоторые предлагаемые специальные алгоритмы [Сениченков, 2004] позволяют обнаруживать многократные переходы, но приводят к большим вычислительным затратам.

Таким образом для решения задачи компенсации амплитудно-временной погрешности наиболее целесообразным (с точки зрения соотношения вычислительных затрат и поддерживаемой точности) является использование

предлагаемого в данной работе алгоритма компенсации на основе решения задачи обратной интерполяции.

2.3 МОДЕЛИРОВАНИЕ НЕПРЕРЫВНОГО ПОВЕДЕНИЯ ОБЪЕКТОВ В ФИЗИКО-ТЕХНИЧЕСКИХ ЗАДАЧАХ

2.3.1 ОТОБРАЖЕНИЕ ГЕОМЕТРИЧЕСКИХ СВОЙСТВ ОБЪЕКТОВ В МНОГОУРОВНЕВОЙ КОМПЬЮТЕРНОЙ МОДЕЛИ

Согласно методике многоаспектного анализа ФТЗ (описанной в п. 2.1.1 данной работы) при её формализации можно выделить различные аспекты (физико-математический, алгоритмический, компонентный) и субаспекты (например, такие субаспекты физико-математического аспекта, как математический, пространственно-геометрический, физический и пр.). *Физико-геометрическая* задача как разновидность ФТЗ характеризуется явно выраженным *пространственно-геометрическим субаспектом*, который обусловливает структуру её модельного представления. Можно выделить следующие существенные геометрические свойства объектов в ФТЗ: а) форма объекта, б) траектория движения объекта, в) межобъектные связи (соединение фигур), г) положение объекта в пространстве. Эти свойства влияют как на значения физических величин (например, силы атмосферного сопротивления $F_{атм}$ или силы трения $F_{тр}$), так и на алгоритм поведения объекта (например, при построении траектории движения). Вследствие этого **актуальным** является совершенствование существующих инструментальных средств моделирования ФТЗ с целью формирования комплекса инструментов, позволяющих эффективно и точно реализовывать пространственно-геометрическую составляющую ФЗ и ФТЗ.

Проведённый обзор программно-инструментальных средств для моделирования ФТЗ показывает почти полное отсутствие специализированных геометрических компонентов.

Для отображения геометрических свойств компонентов в **LabView** используются следующие группы компонентов:

а) Geometry VIs – компоненты для сдвига, вращения, преобразования системы координат и преобразования углов Эйлера в матрицу направляющих косинусов и обратно.

б) Angle VIs – компоненты для перевода, расчёта, поворота значений углов.

в) Computational Geometry VIs – вычислительные геометрические компоненты (компоненты-решатели) для построения контурных графиков, расчёта пересечения многоугольников, центра тяжести фигуры и пр.

Краткое описание объектов представлено в табл. 1.

Таблица 1 – Геометрические компоненты LabView

Группа компонентов Geometry VIs	
2D Cartesian Coordinate Rotation VI	Вращает двумерную декартову систему координат в направлении против часовой стрелки.
2D Cartesian Coordinate Shift VI	Смещает двумерные декартовы координаты вдоль осей x и y.
3D Cartesian Coordinate Rotation (Direction)	Вращает трехмерную декартову координату в направлении против часовой стрелки, используя метод направляющих косинусов
3D Cartesian Coordinate Rotation (Euler)	Поворот трехмерной декартовой координаты в направлении против часовой стрелки с использованием метода углов Эйлера
3D Cartesian Coordinate Shift	Смещает трехмерные декартовы координаты вдоль осей x, y и z
3D Coordinate Conversion	Преобразует координаты между декартовой, сферической и цилиндрической системами координат.
Cross Product VI	Вычисляет перекрестное произведение вектора A и вектора B.
Direction Cosines To Euler Angles VI	Преобразует матрицу 3-x-3-x направляющих косинусов в углы Эйлера
Euler Angles To Direction Cosines VI	Преобразует углы Эйлера в матрицу 3 × 3 направляющих косинусов.
Группа компонентов Angle VIs	
Absolute Angle Difference	Вычисляет абсолютную разницу между двумя входными углами или несколькими парами входных углов.
Angle Rotation	Поворот входного угла против часовой стрелки.
Bisect Angle	Вычисляет угол, который делит начальный угол и конечный угол на две равные части.
Check for Included Angle	Проверяет, находится ли один или несколько входных углов в диапазоне против часовой стрелки
Complementary Angle	Вычисляет дополнительный (до 90) угол входного угла
Supplementary Angle	Вычисляет дополнительный (до 180) угол входного угла.
Wrap Angle	Переносит входной угол на значение в пределах диапазона, указанного вами в диапазоне углов.

Группа компонентов Computational Geometry VIs	
Contour Line	Вычисляет и возвращает контурные линии поверхности и контурный график.
Convex Hull	Вычисляет вершины выпуклого многоугольника, охватывающего (содержащего) все указанные точки.
Convex Polygon Intersection	Вычисляет пересечение двух выпуклых многоугольников.
Delaunay Triangulation	Вычисляет триангуляцию Делоне для указанных точек на плоскости.
Point in Polygon	Проверяет, находится ли точка внутри многоугольника.
Polygon Area	Вычисляет площадь простого многоугольника.
Polygon Centroid	Вычисляет центр тяжести многоугольника.
Voronoi Diagram VI	Вычисляет диаграмму Вороного указанных точек на плоскости.

Как видно из представленной таблицы можно отметить, что в LabView присутствует универсальные компоненты для расчёта площади, центра тяжести многоугольников, заданных по точкам, которые могут использоваться при моделировании физико-геометрических задач, но отсутствуют компоненты для расчёта других характеристик фигур и, что более важно, геометрические фигуры не рассматриваются как *объекты*, вследствие чего все компоненты являются *преобразователями*, а не *источниками*, что ограничивает их использование и организации межкомпонентного взаимодействия:

В среде Simulink (надстройке Matlab) отсутствуют специализированные геометрические компоненты – учёт геометрических свойств осуществляется за счёт параметрирования блоков типа «Модель тела» (body), «Соединители тел» (jointer) и пр.

Отметим следующие недостатки подходов к отображению геометрических свойств:

1) модель непрерывного поведения (физико-математическая составляющая задачи) и модель дискретного поведения (сценарий проведения эксперимента и алгоритм поведения объектов) находятся на одной схеме;

2) геометрические свойства либо инкапсулированы в компонент-объект (Simulink), либо задаются в символьной форме (Rand Model Designer);

3) схема компонентой модели в рассмотренных средах не отвечает концепции «физического» моделирования (1 объект = компонент) и визуально не соответствует структурной схеме реального объекта.

Предлагаемый подход к моделированию ФТЗ

В качестве комплекса инструментов для моделирования пространственно-геометрической составляющей ФЗ и ФТЗ предлагается разработка библиотеки моделей геометрических компонентов следующих типов:

1) *геометрические примитивы* (отрезок, треугольник, четырёхугольник и пр.) – набор базовых геометрических фигур, лежащий в основе всех графических построений и предназначенный для расчёта их характеристик;

2) *геометрические преобразователи*, предназначенные для расчёта проекций вектора на плоскость, расчёта вектора по его проекциям и пр.;

3) *геометрические решатели*, предназначенные для решения базовых геометрических задач (поиск точки пересечения фигур, оценка факта попадания точки в фигуру);

4) *кривые* – для расчёта профилей поверхностей, определения траекторий движения и т.д., задаваемых уравнениями кривых различных порядков или дискретным набором точек (табличной функцией).

Данный набор компонентов может быть расширен и позволяет покрыть основные потребности исследователей при построении ФТЗ с геометрическим содержанием, в случае необходимости могут также использоваться ИМП и ИМАП.

Обобщённая структура моделей физико-геометрических задач с использованием геометрических компонентов представлена на рис. 2.26.

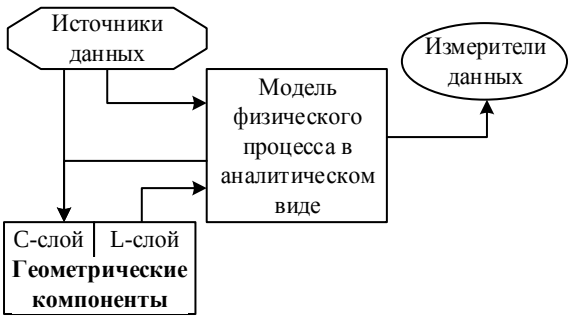


Рисунок 2.26 – Обобщённая структура модели физико-геометрической задачи

На рис. 2.26 модель физического процесса задаётся с помощью ИМП, позволяющих в аналитическом виде представлять математические модели и включать их в компонентную цепь, состоящую из графических блоков (компонентов). Такая структура модели предполагает лёгкую её модификацию или детализацию, что особенно актуально при использовании подхода итерационного моделирования.

Математическое описание разработанных геометрических компонентов

1. Группа компонентов **«Геометрические примитивы»:**

1.1 Компонент **«Многоугольник по точкам»** задаёт фигуру произвольного вида через набор точек, последовательно соединяемых кривыми (прямыми) выбранного порядка. Задаваемые параметры: координаты X и Y, тип кривой – целое число, определяющее порядок функции кривой, соединяющей точки $(x_i, y_i) \rightarrow (x_{i+1}, y_{i+1})$. Выход: вершины фигуры $\{x_i\}, \{y_i\}$. Для задания замкнутой фигуры необходимо, чтобы первая и последняя пара (x, y) совпадали.

Компонент имеет параметр «Тип соединительной» линии, выбирая значение которого между «прямая» и «кривая n-го порядка» пользователь определяет тип требуемой фигуры: ромб, окружность (эллипс) – рис. 2.27.

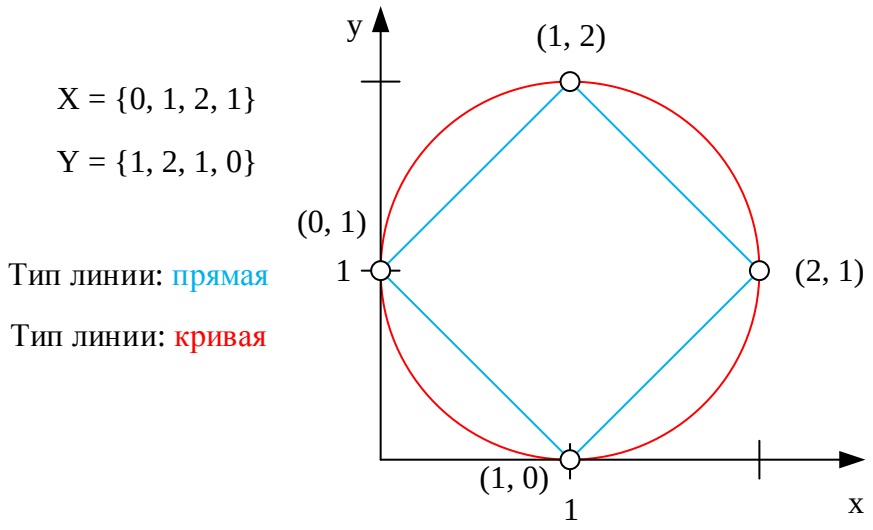


Рисунок 2.27 – Иллюстрация параметра «Тип соединительной линии»

1.2 Компонент **«Площадь фигуры»** производит расчёт площади выпуклой фигуры по данным, полученным от компонента «Многоугольник по точкам».

Входные данные: вершины фигуры $\{x_i\},\{y_i\}$. Выходные данные: значение площади фигуры S, рассчитываемое по формуле: $S = |\frac{1}{2}\sum_{i=0}^{n}(x_i \cdot y_{i+1} - x_{i+1} \cdot y_i)|$. В случае фигуры, границы которой – кривые, используется формула интегрирования Симпсона (парабол).

1.3 Компонент **«Периметр фигуры»** производит расчёт периметра выпуклой фигуры по данным, полученным от компонента «Многоугольник по точкам». Входные данные: вершины фигуры $\{x_i\},\{y_i\}$. Выходные данные: значение периметра фигуры P, рассчитываемое по формуле: $P = \sum_{i=0}^{n}\sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2}$.

1.4 Компонент **«Длина n-й стороны»** производит расчёт длины *n*-й стороны (между *n*-1 и *n* точками) фигуры по данным, полученным от компонента «Многоугольник по точкам». Входные данные: вершины фигуры $\{x_i\},\{y_i\}$, целое число *n* – номер стороны. Выходные данные: длина *l*.

2 Геометрические преобразователи:

2.1 Компонент **«Перевод из градусов в радианы»** производит перевод значения, получаемого на вход слева, из градусов в радианы и передаёт его на узел справа.

2.2 Компонент **«Перевод из радиан в градусы»** производит перевод значения, получаемого на вход слева, из радиан в градусы и передаёт его на узел справа.

2.3 Компонент **«Расчёт проекции вектора на ось»** производит расчёт проекции вектора на ось, используя значения длины вектора и угла наклона к оси, получаемых на вход слева, и передаёт его на узел справа.

2.4 Компонент **«Расчёт вектора по его проекциям»** производит расчёт длины вектора, используя значения величины его проекций на оси (подаётся слева на вход).

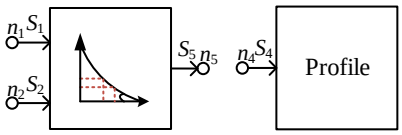
3. Группа компонентов «**Геометрические решатели»:**

3.1. Компонент **«Поиск точки пересечения фигур»** возвращает точки пересечения фигур. Входные данные: вектор координат X фигуры 1, вектор координат Y фигуры 1, вектор координат X фигуры 2, вектор координат Y фигуры 2.

3.2. Компонент **«Проверка попадания точки в фигуру»** проверяет, находится ли точка внутри многоугольника. Входные данные: вектор координат X, вектор координат Y. Выходные данные: идентификатор положения точки (внутри, на границе, снаружи).

4. Группа компонентов **«Кривые»:**

4.1 Компонент **«Кривая (аналитически)»** – компонент, позволяющий задавать профиль поступательного движения (скатывания) тела по поверхности, задаваемой аналитическим выражением.



а б

Рис.

Связями компонента «Профиль» являются следующие:

а) располагаемые на L-слое многоуровневой компьютерной модели:

- $S_1 = b_1\eta^- n_1 \rightarrow x_0$, $S_2 = b_2\eta^- n_2 \rightarrow y_0$ – предназначенные для инициализации компонента (получения начальных координат тела с V-слоя);
- $S_3 = b_3\eta^+ n_3 \rightarrow \alpha$ – для передачи рассчитанного значения угла между ускорением центра масс тела и осью OX, которое зависит от профиля поверхности скатывания;

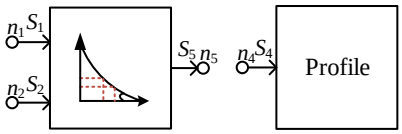
б) располагаемая на C-слое многоуровневой компьютерной модели:

- $S_4 = b_4\eta^- n_4 \rightarrow x, y$ – для получения компонентом текущих координат тела.

Математическая модель компонента «Профиль», рассчитываемая имитационным, а не вычислительным ядром, описывается выражениями:

$$x_{i+1} = x_i + \Delta x, y_{i+1} = f(x_{i+1})$$
$$a = |y_i - y_{i+1}|, b = |x_i - x_{i+1}|, c = \sqrt{a^2 + b^2}$$
$$\alpha = \arccos\frac{b}{c}$$

4.2 Компонент **«Кривая (по точкам)»** – компонент, позволяющий задавать профиль поступательного движения (скатывания) тела по поверхности, задаваемой последовательностью точек (узлов), промежуточные значения которых рассчитываются посредством кусочной интерполяции (квадратичной, линейной – опционально).



а б

Связями компонента «Профиль» являются следующие:

а) располагаемые на L-слое многоуровневой компьютерной модели:

- $S_1 = b_1\eta^- n_1 \rightarrow x_0$, $S_2 = b_2\eta^- n_2 \rightarrow y_0$ – предназначенные для инициализации компонента (получения начальных координат тела с V-слоя);
- $S_3 = b_3\eta^+ n_3 \rightarrow \alpha$ – для передачи рассчитанного значения угла между ускорением центра масс тела и осью OX, которое зависит от профиля поверхности скатывания;

б) располагаемая на C-слое многоуровневой компьютерной модели:

$S_4 = b_4\eta^- n_4 \rightarrow x, y$ – для получения компонентом текущих координат тела.

Описание программной реализации геометрических компонентов представлено в Приложении Б.

2.3.2 ОТОБРАЖЕНИЕ ФИЗИЧЕСКИХ СВОЙСТВ В КОМПЬЮТЕРНЫХ МОДЕЛЯХ

Все компоненты используют неоднородные векторные связи (НВС) следующего вида (рис. 2.28) на С-слое. Каждая такая связь содержит 2 энергетические связи, содержащие потоковые (проекция силы *F* на одну из осей) и потенциальные (проекции скорости *V* на одну из осей) переменные. Связи геометрического компонента на С-слое содержат 2 потенциальных переменных (*x*, *y*).

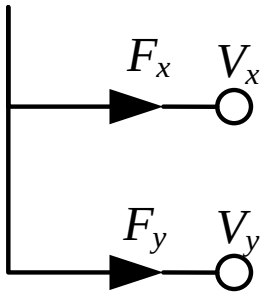
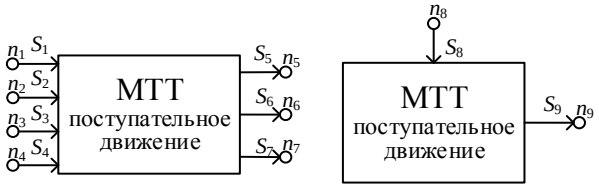


Рисунок 2.28 – Иллюстрация неоднородной векторной связи

Компонент «**Твёрдое тело в поступательном движении**» (рис. 2.29) – компонент, позволяющий моделировать движение тела в поступательном движении (например, скатывание тела по наклонной поверхности).



а) б)

Рисунок 2.29 – Вид компонента «Твёрдое тело в поступательном движении»
а – на L-слое, б – на С-слое

Связями компонента «Модель твёрдого тела в поступательном движении» являются следующие:

а) располагаемые на L-слое многоуровневой компьютерной модели:

- $S_1 = b_1\eta^- n_1 \rightarrow x_0$, $S_2 = b_2\eta^- n_2 \rightarrow y_0$, $S_3 = b_3\eta^- n_3 \rightarrow \vec{V}_0$ – предназначенные для инициализации компонента (получения начальных координат тела и значения начальной скорости с V-слоя);

- $S_4 = b_4\eta^- n_4 \rightarrow \alpha$ – для получения значения угла между ускорением центра масс тела и осью OX, которое зависит от профиля поверхности скатывания;

- $S_5 = b_5\eta^+ n_5 \rightarrow x$, $S_6 = b_6\eta^+ n_6 \rightarrow y$, $S_7 = b_7\eta^+ n_7 \rightarrow \vec{V}$ – для вывода соответствующих значений координат тела и текущей скорости его поступательного движения относительно начала координат;

б) располагаемые на C-слое многоуровневой компьютерной модели:

- $S_8 = b_8\eta^- n_8 \rightarrow F_x, F_y$ – по этой НВС поступают значения потоковых переменных $\sum_i F_i$ для дальнейшего расчёта системы уравнений модели;

- $S_9 = b_9\eta^+ n_9 \rightarrow x, y$ – для передачи рассчитанных координат тела.

Математическая модель компонента «Твёрдое тело в поступательном движении» описывается следующими выражениями, расчёт которых производится имитационным ядром среды:

$$\begin{aligned} V_{0_x} &= V_0 \cdot \cos\alpha \\ V_{0_y} &= V_0 \cdot \sin\alpha \\ \vec{V} &= \sqrt{V_x^{\,2} + V_y^{\,2}} \end{aligned},$$

и следующей системой уравнений, решаемой вычислительным ядром среды:

$$\frac{dx}{dt} = V_x, \frac{dy}{dt} = V_y, m \cdot \frac{dV_x}{dt} = \sum F_x, m \cdot \frac{dV_y}{dt} = \sum F_y ,$$

где x_0, y_0 – начальные координаты тела, V_0 – начальная скорость тела (передаются в компонент единожды); F_x, F_y – проекции суммы действующих сил на проекции OX и OY.

Источник силы трения (рис. 2.30) – компонент, позволяющий задавать силу трения $F_{тр}$, воздействующую на тело при скатывании с наклонной

плоскости. Параметры α, μ, m, g задаются в параметрах объекта. При необходимости передачи данных с L-слоя обмен данных осуществляется через соответствующий компонент-атрибут.

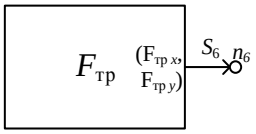


Рисунок 2.30 – Компонент «Источник силы трения»

Связями компонента «Источник силы трения» являются следующие, располагаемые на С-слое многоуровневой компьютерной модели:

- $S_6 = b_1 \eta^+ n_1 \rightarrow F_{mp_x}, F_{mp_y}$ – для передачи рассчитанных проекций вектора $\vec{F}_{mp}$ по НВС в непрерывную модель на С-слое.

Математическая модель «Источника силы трения» описывается выражениями, рассчитываемыми имитационным ядром среды:

$$\begin{aligned} \vec{F}_{mp} &= \mu \cdot m \cdot g \cdot \cos\alpha \\ F_{mp_x} &= \vec{F}_{mp} \cdot \cos\alpha \\ F_{mp_y} &= \vec{F}_{mp} \cdot \sin\alpha \end{aligned} ,$$

где α – угол для расчёта проекций силы на оси, μ – коэффициент трения, m – масса тела, g – ускорение свободного падения, $\vec{F}_{mp}$ – сила трения;

«Источник силы реакции опоры» (рис. 2.31)– компонент, позволяющий задавать силу реакции опоры $\vec{N}$, воздействующую на тело. Параметры α, m, g задаются в параметрах объекта. При необходимости передачи данных с L-слоя обмен данных осуществляется через соответствующий компонент-атрибут.

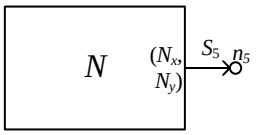


Рисунок 2.31 – Компонент «Источник силы реакции опоры»

Рис.

Связями компонента «Источник силы реакции опоры» являются следующие располагаемые на С-слое многоуровневой компьютерной модели:

- $S_5 = b_5\eta^+ n_5 \rightarrow N_x, N_y$ – для передачи рассчитанных проекций вектора $\vec{N}$ по НВС в непрерывную модель на С-слое.

Математическая модель «Источника силы трения» описывается выражениями, рассчитываемыми имитационным ядром среды:

$$\vec{N} = m \cdot g \cdot \cos\alpha$$
$$N_x = \vec{N} \cdot \cos\alpha$$
$$N_y = \vec{N} \cdot \sin\alpha$$

«Источник силы тяжести» (рис. 2.32) – компонент, позволяющий задавать силу тяжести $\vec{F}_m$, воздействующую на тело. Параметры α, m, g задаются в параметрах объекта. При необходимости передачи данных с L-слоя обмен данных осуществляется через соответствующий компонент-атрибут.

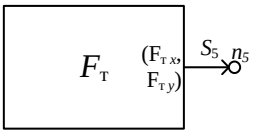


Рисунок 2.32 – Компонент «Источник силы тяжести»

Связями компонента «Источник силы тяжести» являются следующие располагаемые на С-слое многоуровневой компьютерной модели:

- $S_5 = b_5\eta^+ n_5 \rightarrow F_{m_x}, F_{m_y}$ – для передачи рассчитанных проекций вектора $\vec{N}$ по НВС в непрерывную модель на С-слое.

Математическая модель «Источника силы трения» описывается выражениями, рассчитываемыми имитационном ядром среды:

$$\vec{F}_m = m \cdot g \cdot \cos\alpha$$
$$F_{m_x} = \vec{F}_m \cdot \cos\alpha$$
$$F_{m_y} = \vec{F}_m \cdot \sin\alpha$$

Пример задачи. Представим вид компонентной цепи на L- (рис. 2.33) и С-слоях (рис. 2.34) для моделирования скатывания тела по наклонной поверхности произвольного профиля.

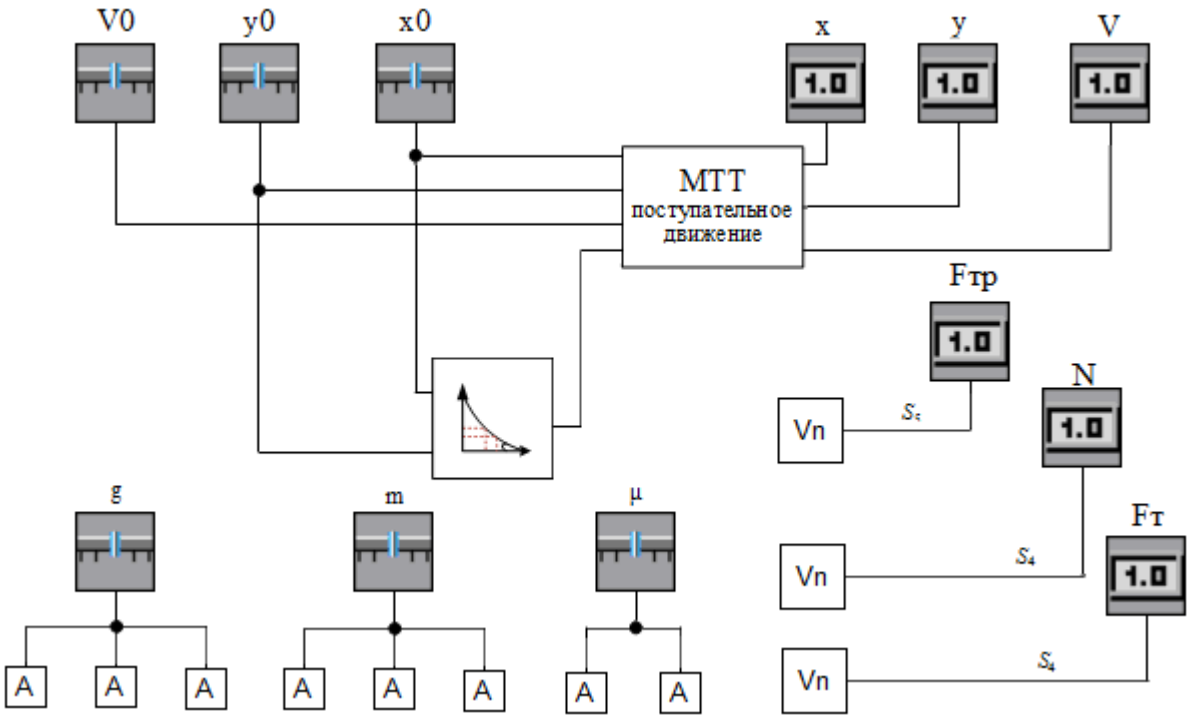


Рисунок 2.33 – Вид модели на L-слое

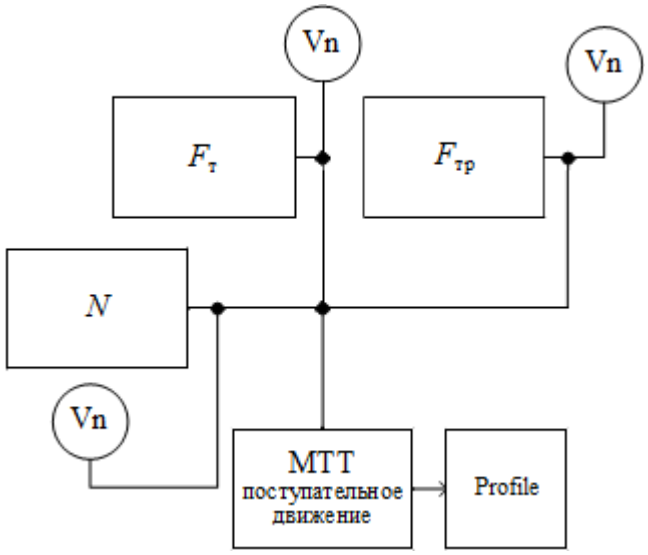


Рисунок 2.34 – Вид модели на C-слое

Опишем алгоритм работы, представленной выше МКМ скатывания тела по наклонной плоскости:

1. Компонент «Профиль» на *L*-слое получает значения x_0, y_0 и передаёт рассчитанное значение α_0 компонентам: МТТ, $F_т$, $F_{тр}$, N.

2. Компоненты F_m, $F_{тр}$, N получают с *V*-слоя оставшиеся исходные данные;

3. Компоненты $F_т$, $F_{тр}$, N передают рассчитанные проекции вектора сил Fi через выходные узлы своих отображений на *C*-слое.

4. Компонент МТТ на *L*-слое рассчитывает проекции вектора скорости V и передаёт на измерители *V*-слоя значения, *V*, *x*, *y*.

5. Компонент МТТ на *C*-слое осуществляет решение системы дифференциальных уравнений, рассчитанные значения *x*, *y* передаются в отображение «Профиля» на *C*-слое.

6. Компонент «Профиль» рассчитывает новое значение угла α и передаёт его компонентам МТТ, $F_т$, $F_{тр}$, *N* на *L*-слое.

7. Переход к этапу 3 (повторяется до окончания работы модели).

Программная реализация физических компонентов описана в п. 3.1.3.

Компоненты для моделирования одномерной механики

Далее представим компоненты, разработанные для построения модели штангового глубинного насоса (см. параграф 4.2):

1. Компонент «Источник скорости» (рис. 2.35), математическая модель которого имеет вид (2.10):

$$B_1 \cdot V_p = S(t) \quad (2.5),$$

где B_1 – некоторый коэффициент (константа или функциональный параметр), V_p – потенциальная переменная (в нашем случае – скорость), $S(t)$ – некоторая функциональная зависимость.

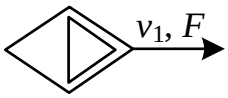


Рисунок 2.35 – Компонент «Источник скорости»

Будем рассматривать насосные штанги и плунжер в качестве моделей твёрдых тел, учитывая такие их параметры, как площадь сечения и плотность материала. Математическая модель инерционного звена (рис. 2.36) имеет вид (2.11):

$$A_1 \cdot \frac{dV_p}{dt} + B_2 \cdot V_f = 0 \quad (2.6),$$

где V_f – потоковая переменная (сумма сил *F*, воздействующих на тело), V_p – потенциальная переменная (скорость тела *V*), A_1 – параметр модели (масса тела *m*), B_2 – параметр модели (здесь равен 1), *t* – время.

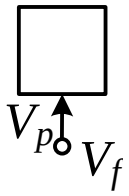


Рисунок 2.36 – Компонент «Инерционное звено»

Деформация (растяжение и сжатие) штанг во время их движения может быть смоделирована посредством компонентов типа «Пружина» (рис. 2.37), имеющими (как и остальные рассматриваемые нами компоненты) 2 элементарные связи с потенциальными переменными v_1, v_2 (скорости) и потоковой переменной F (сила). Типовая математическая модель компонента имеет вид (2.12):

$$\frac{dF}{dt} = k \cdot (v_1 - v_2) \quad (2.7),$$

в которой коэффициент k может быть как постоянным, так и функциональным параметром. Величину растяжения штанг в таком случае можно рассчитать исходя из разницы значений скоростей v_1 и v_2 на узлах компонента.

Рисунок 2.37 – Компонент «Пружина»

Силы трения, воздействующие при совершении движения на шток, штанги и плунжер, могут быть смоделированы с помощью компонента «Демпфер» (рис. 2.38). Математическая модель этого компонента имеет вид (2.13):

$$F = R \cdot (v_1 - v_2) \quad (2.8),$$

где коэффициент R аналогичным образом может представлять собой константу или функциональный параметр.

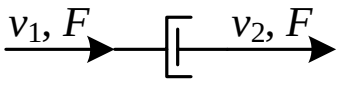


Рисунок 2.38 – Компонент «Демпфер»

Давление газожидкостной смеси на плунжер может быть смоделировано с помощью компонента «Источник потоковой переменной», аналогичного «Источник скорости» и имеющему математическую модель (2.14):

$$B_2 \cdot V_f = C(t) \quad (2.9),$$

где B_2 – некоторый коэффициент (константа или функциональный параметр), V_f – потоковая переменная (в нашем случае – сила), $C(t)$ – некоторая функциональная зависимость.

2.3.3 ОБРАБОТКА ТАБЛИЧНЫХ РЕЗУЛЬТАТОВ МОДЕЛИРОВАНИЯ

Результатом компьютерного моделирования часто становятся массивы численной информации. Эта информация носит приближенный характер ввиду случайных ошибок, погрешностей используемых численных методов, неточности измерительных приборов (используемых, например, в качестве начальных данных) и пр. Такие данные подлежат обработке, целью которой может быть:

1) удаления шума, сглаживание функции;

2) поиск закономерностей в данных, нахождение аналитической формы исследуемой зависимости;

3) сжатие информации для экономии памяти или повышения скорости передачи;

4) определение промежуточных значений зависимости (внутри или вне имеющегося дискретного набора).

Для решения таких задач используют методы вычислительной математики (а именно теории приближений), заключающиеся в замене исходной табличной (дискретной) функции $f(x) = (x_i, y_i)$ приближенной непрерывной $\varphi(x)$. В общем случае выделяются 2 типа задач аппроксимации [Турчак, Плотников, 2003]:

1) точечная аппроксимация – построение приближения на дискретном множестве точек $\{x_i\}$, которая включает в себя задачи:

а) интерполяции – соединение точек внутри рассматриваемого отрезка (дискретного набора точек $\{x_i\}$) кривыми выбранного порядка (локальная и глобальная интерполяция);

б) экстраполяции – приближенное вычисление функции вне рассматриваемого отрезка;

в) аппроксимации (сглаживание или среднеквадратичное приближение) – замена табличной функции (x_i, y_i) непрерывной функцией заданного вида, проходящей «близко» (в соответствии с выбранным критерием) от данных точек.

2) непрерывная (интегральная) аппроксимация построение приближения на непрерывном множестве точек (например, равномерное приближение, накладывающее более жёсткое условие для всех точек отрезка [a, b] – $|f(x)-\varphi(x)<\varepsilon|, a\le x\le b)$. Примером такой задачи может быть замена некоторой непрерывной функции степенным многочленом.

Для аппроксимации табличных результатов моделирования ФТЗ в данной работе представляет интерес точечная аппроксимация. При необходимости учитывать погрешность расчёта данных и «при наличии значительного числа экспериментальных точек сглаживание с помощью многочленной интерполяции не имеет смысла не только из-за неустойчивости (локальных выбросов) интерполирующей функции, но и из-за сильного колебания заданных точек» [Формалёв, 2004]. В таком случае имеющуюся табличную функцию сглаживают «в среднем», многочленом выбранного вида, коэффициенты которого находят по некоторому критерию для оценки отклонения аппроксимирующей функции от заданных точек.

Анализ существующих подходов к аппроксимации табличных данных

Одним из основных математических методов обработки опытных данных является метод наименьших квадратов (МНК) [Голованчиков, 2019], который сводится к решению задачи оптимизации суммы квадратов отклонений [Линник, 1958]: $Q(a_0,a_1,...,a_m)=\sum_{i=0}^{n}\varepsilon_i^{\,2}=\sum_{i=0}^{n}(\varphi(x_i,a_0,a_1,...,a_m)-y_i)^2\to\min$, где $a_0,a_1,...,a_m$ –

коэффициенты аппроксимирующей функции (полинома), $i = 0,1...n$ (n – число точек табличной функции).

С опорой на необходимое условие экстремума $\frac{\partial Q}{\partial a_0} = 0, \frac{\partial Q}{\partial a_1} = 0, ..., \frac{\partial Q}{\partial a_m} = 0.$ получаем системе линейных[1] алгебраических уравнений (СЛАУ):

$$\sum_{i=0}^{n} (\varphi(x_i, a_0, a_1, ..., a_m) - y_i) \cdot \varphi'_{a_0}(x_i, a_0, a_1, ..., a_m) = 0$$

$$\sum_{i=0}^{n} (\varphi(x_i, a_0, a_1, ..., a_m) - y_i) \cdot \varphi'_{a_1}(x_i, a_0, a_1, ..., a_m) = 0$$

$$\vdots$$

$$\sum_{i=0}^{n} (\varphi(x_i, a_0, a_1, ..., a_m) - y_i) \cdot \varphi'_{a_m}(x_i, a_0, a_1, ..., a_m) = 0$$

Решением приведённой СЛАУ и являются коэффициенты приближающей функции заданного вида, обеспечивающие минимальное среднеквадратичное отклонение в дискретных узлах.

На практике выбор приближающей функции осуществляется исследователем исходя из некоторых соображений. Однако, как правило, в инструментальных средствах доступен лишь фиксированный список доступных (заранее внесённых) аппроксимирующих функций. В качестве примера программной реализации МНК приведём аппроксиматор СМ МАРС, интерфейс которого представлен на рис. 2.39.

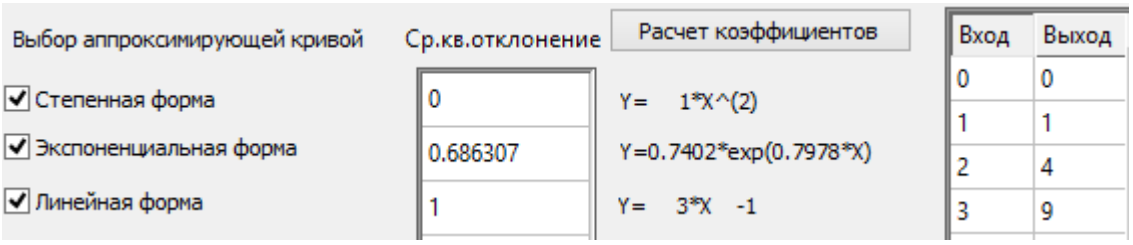


Рисунок 2.39 – Фрагмент окна аппроксиматора в СМ МАРС

При этом не всегда погрешность лучшей (из имеющегося списка) приближающей функции является приемлемой, а использование МНК для аппроксимации данных «пользовательской» функцией требует ручной процедуры нахождения частных производных этой функции. Альтернативным

[1] линейных относительно параметров $a_0, a_1, ..., a_m$

решением может служить переход от построения СЛАУ с применением аналитического дифференцирования к решению задачи многомерной оптимизации целевой функции, которая представляет собой критерий оценки погрешности приближения.

Метода Левенберга-Маркгвардта [Levenberg, 1944; Marquardt, 1963]. Данный метод является наиболее распространенным (для решения задач оптимизации в целом и задач о минимизации суммы квадратов в частности) на сегодняшний день – он используется для аппроксимации табличных зависимостей функциями произвольного вида, для обучения нейронных сетей [Пархоменко, Леденёва, 2014] и пр.

Задача о наименьших квадратах имеет вид: $F(x) = \sum_{i=1}^{m} (\varphi_i(\vec{x}) - f_i)^2 \to \min$,

где f_i – значения табличной (аппроксимируемой) функции, $\varphi(x)$ – аппроксимирующая функция.

В основу метода Левенбрга-Марквардта положено направление поиска, которое находится при решении системы линейных уравнений $(J(x_k)^T \cdot J(x_k) + \lambda_k \cdot I) = -J(x_k) \cdot F(x_k)$, где $J(x)$ – матрица Якоби функции $F(x)$, x_k – вектор коэффициентов аппроксимирующей функции, I – единичная матрица, скаляр λ_k задает как величину, так и направление параметра d_k, отвечающего за направление поиска. При λ_k=0 имеем направление поиска аналогичное направлению в методе Гаусса-Ньютона [Ловецкий и др., 2008].

Метод является поисковым и представляет собой комбинацию метода наискорейшего спуска с методом Ньютона. Он превосходит по производительности метод наискорейшего спуска и другие методы сопряженных градиентов в различных задачах. В качестве недостатков необходимо отметить высокую чувствительность данного метода к точности начального приближения и ограничения на целевую функцию.

Рассмотрим задачу аппроксимации зависимости ускорения штока (моделируемой компонентом «Источник скорости» в задаче из параграфа 4.1) от времени.

В качестве приближающей функции выберем следующую – $f(x) = C_1 \cdot \cos(C_2 \cdot t + C_3) + C_4$. Решением задачи будем считать следующие значения параметров функции $C_1 = 3.773, C_2 = 0.503, C_3 = 0.004, C_4 = 1.029$ (с округлением до третьего знака). Для иллюстрации чувствительности метода Левенберга-Марквардта проведём серию вычислительных экспериментов из различных точек начального приближения, положив $C_1 = 4 \cdot 10^{-5}; C_2 = 0.3 + i/20, i = 1,...10; C_3 = 0; C_4 = 0$, т.е. изменяя значения одного из коэффициентов с шагом 0.05 (рис. 2.40).

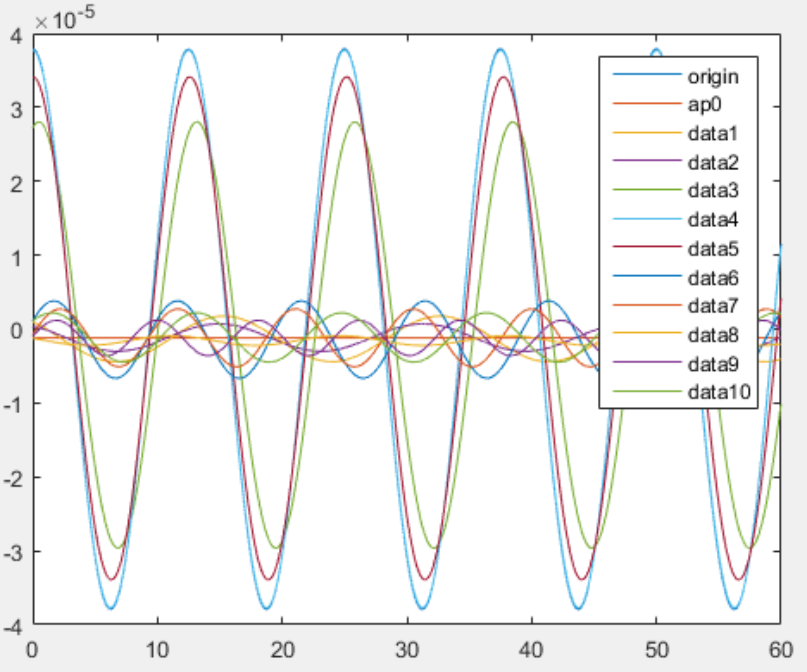


Рисунок 2.40 – Поиск решения из различных начальных точек

Как видно из рисунка выше только 3 кривые дают удовлетворительный результат при $\delta_3 = 1.49 \cdot 10^{-10}, \delta_4 = 3.32 \cdot 10^{-14}, \delta_5 = 1.56 \cdot 10^{-11}$. При изменении шага изменения параметра C_2 с 0.05 до 0.1 остаётся только одна удовлетворяющая точка, а при шаге 0.5 – 0 точек.

Метод аппроксимации на базе поисковых методов оптимизации

Целью данного пункта работы является представление алгоритма предлагаемого метода аппроксимации на основе методов многомерной оптимизации по критерию среднеквадратичного отклонения приближающей функции в дискретных узлах. Данный метод не требует информации о производных приближающей функции, а, следовательно, позволяет в

автоматизированном режиме использовать любые функции, введённые пользователем в ИМП среды моделирования МАРС в качестве приближающих.

Алгоритм численной аппроксимации на основе методов многомерной оптимизации. В качестве критерия качества приближения предлагается использовать среднеквадратичное отклонение в узлах табличной функции. В результате получаем задачу многомерной оптимизации целевой функции $f(x, a, b, c)$ по критерию $\sqrt{\sum_i (y_{табл_i} - f(x_i, a, b, c))^2} \to \min$ (где x_i – аргументы табличной функции; a, b, c – параметры приближающей функции), для решения которой используем модификацию метода покоординатного спуска, предполагающую применение метода золотого сечения для оптимизации функции одной переменной на каждой оси поочерёдно. Метод покоординатного спуска является простым неградиентным методом локальной оптимизации, метод золотого сечения также не требует информации о производной оптимизируемой функции и имеет более высокую сходимость по сравнению с аналогичными методами. Начальное приближение для определения начальной точки покоординатного спуска и интервала поиска минимального значения функции по одной из координат определяется методом сеток с некоторым фиксированным шагом. Комбинация методов локальной оптимизации позволяет (в первую очередь вследствие использования метода сеток) решить задачу глобальной оптимизации целевой функции $f(x, a, b, c)$. Невысокая «стоимость» одной итерации поискового метода позволяет осуществлять исследование пространства решений из нескольких точек.

Представим алгоритм численной аппроксимации на рис. 2.41:

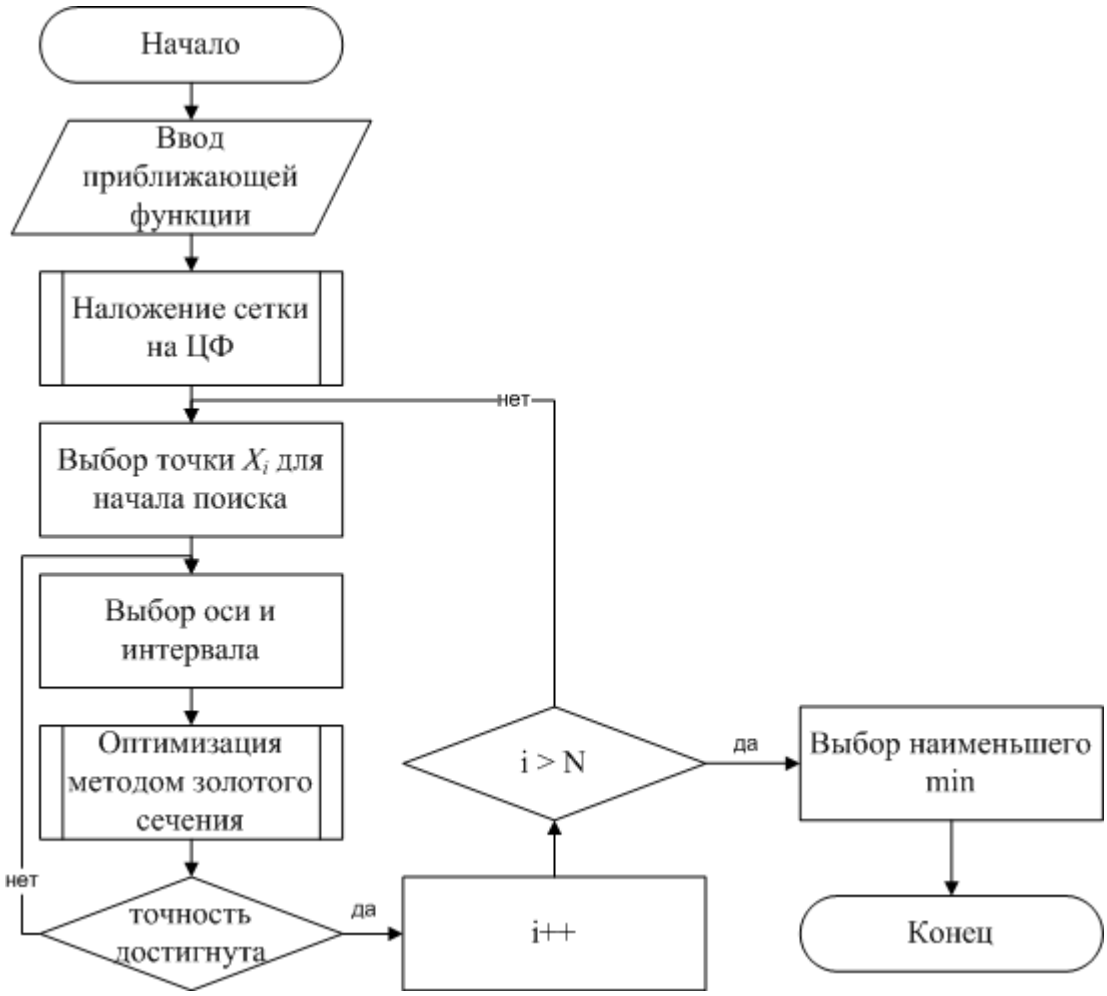


Рисунок 2.41 – Алгоритм численной аппроксимации поисковыми методами

Все найденные локальные минимумы также могут быть проверены на устойчивость путём исследования их окрестностей, что позволит выбирать не только наименьший из них, но и наиболее устойчивый.

Сравнение результатов работы метода. Сравним результаты применения предлагаемого метода и МНК для аппроксимации траектории полёта тела в атмосфере, полученной в результате компьютерного моделирования данной ФТЗ с шагом Δt=0.1 при начальной скорости v_0=50 м/с, массе m=10 кг, углом полёта α=45˚ и силе атмосферного сопротивления F_a пропорциональной скорости тела v. Таким образом имеем 697 узлов табличной функции.

Для обоих методов в качестве приближающей функции выбрана квадратичная функция $f(x) = a \cdot x^2 + b \cdot x + c$. Для решения СЛАУ в МНК использован метод Гаусса. Для предлагаемого метода аппроксимации были

заданы следующие параметры: шаг метода сеток – 0.25, для начального приближения по всем трём координатам выбран интервал [-10, 10], заданная точность для метода золотого сечения – 10^{-6}. Результаты аппроксимации зависимостей $y(x)$ и $y(t)$ представлены в табл. 2.6 с округлением до 5-х знаков.

Таблица 2.6 – Сравнение результатов аппроксимации

Метод		Коэффициенты приближающей функции			Средне-квадратичное отклонение
		a	b	c	
МНК		-0.00473	1.07290	-1.36893	0.50734
Предложенный метод		-0.00473	1.07260	-1.35730	0.50753
	δ,%	0.0279%	0.0281%	0.8497%	0.0373%

Приведём ещё один пример работы алгоритма – на других данных, с точностью до 4-х знаков (табл. 2.7).

Таблица 2.7 – Другое сравнение результатов аппроксимации

Метод		Аппроксимация зависимости y(x)				Аппроксимация зависимости y(t)			
		Значения коэффициентов			σ	Значения коэффициентов			σ
		a	b	c		a	b	c	
МНК		-0.1088	1.1122	-0.0317	0.016285	-4.9402	8.4129	-0.7737	0.009341
Предложенный метод		-0.1087	1.1117	-0.0308	0.016291	-4.9388	8.4103	-0.7728	0.009346
	δ, %	0.0417	0.0450	2.8067	0.0368	0.0289	0.0311	0.1110	0.0535

Таким образом, при заданных условиях описываемый метод аппроксимации обеспечивает точность, сравнимую с точностью метода наименьших квадратов. При этом применение данного метода **не ограничивает пользователя списком встроенных приближающих функций** – пользователь может сам ввести вид требуемой функции (в аналитическом виде в ИМП) получить значения её параметров, обеспечивающие минимальное среднеквадратичное отклонение.

Использование метода наименьших квадратов для аппроксимации табличных функций требует наличия информации о частных производных рассматриваемой приближающей функции. При необходимости аппроксимации исследуемой зависимости функцией введённой пользователем, информация о её производных может отсутствовать в аппроксиматоре. Использование существующих методов оптимизации, например, метода Левенберга-

Марквардта, не обеспечивает нахождение глобального минимума, к тому же метод является чувствительным к выбору начальных условий – в некоторых случаях отклонение на 0.1 от начальной точки может вернуть результат с большим значением СКО. Описываемый метод аппроксимации на основе методов многомерной оптимизации позволяет подобрать коэффициенты приближающей функции, не требуя информации о её производных. Метод предполагает решение задачи аппроксимации через задачу многомерной оптимизации целевой функции – среднеквадратичного отклонения в узлах табличной функции. Для решения задачи оптимизации предлагается использование метода сеток для нахождения начального приближения и комбинации метода покоординатного спуска и метода золотого сечения для решения задачи локальной оптимизации. Применение метода сеток позволяет находить несколько локальных минимумов, что позволяет решать задачу глобальной оптимизации. Рассмотренный пример аппроксимации траектории полёта тела в атмосфере иллюстрирует сопоставимость результатов предлагаемого метода и метода наименьших квадратов ($\delta max < 1\%$). Актуальной остаётся задача повышения скорости сходимости предложенного метода.

ВЫВОДЫ ПО ГЛАВЕ 2

1. Разработанный алгоритм многоуровневого компьютерного моделирования ФТЗ, включает этапы формализации моделируемой задачи и её многоуровневой декомпозиции на объектный, логический (алгоритмический) и визуальный уровни и позволяет составлять модели ФТЗ из готовых блоков (компонентов), разделяя непрерывное (физическое) поведение объектов от алгоритма их функционирования (дискретное поведение).

2. Разработанные геометрические, физические компоненты, а также компоненты для моделирования дискретного поведения объектов (диаграммы состояний) направлены на отражение в многоуровневых компьютерных моделях ключевых особенностей ФТЗ. Диаграммы состояний поддерживают

разработанный алгоритм компенсации накапливаемой амплитудно-временной погрешности, что позволяет повышать точность моделей ФТЗ.

3. На базе поисковых методов оптимизации разработан алгоритм численной аппроксимации, осуществляющий приближение табличных результатов моделирования функцией произвольного вида с погрешностью сопоставимой с результатами аппроксимации существующих методов и обеспечивающий выбор глобального минимума из ряда локальных.

ГЛАВА 3. КОМПЛЕКС ПРОГРАММ ДЛЯ КОМПЬЮТЕРНОГО МОДЕЛИРОВАНИЯ ФИЗИКО-ТЕХНИЧЕСКИХ ЗАДАЧ

В главе представлено описание структуры комплекса программ многоуровневого компьютерного моделирования физико-технических задач, предназначенного для построения МКМ ФТЗ и применения в образовательной деятельности. Приводится описание библиотеки компонентов для моделирования физико-технических задач, программного модуля для обучения многоуровневому компьютерному моделированию ФТЗ на основе анализа словесного портрета задачи и информационной системы управления лабораторией моделирования физико-технических задач. Разработанный комплекс программ функционирует под управлением операционных систем семейства Windows. Для разработки комплекса использовался объектно-ориентированный язык программирования Microsoft Visual C++ с библиотекой классов Microsoft Foundation Classes (MFC) [MFC …], а также C# с библиотекой классов NetFramework.

На рис. представлена структурная схема разработанного комплекса программ. Разработанные блоки выделены штриховкой. Порядок взаимодействия с комплексом следующий:

1. Студент во время аудиторного занятия запускает Информационную систему управления лабораторией (ИСУЛ), которая предоставляет доступ к электронному курсу дисциплины в Moodle.

2. Ознакомившись на аудиторном занятии с требуемыми учебно-методическими материалами (лекция, входной контроль, методические указания к выполнению работы) студент запускает необходимую модель в СМ МАРС через ИСУЛ.

3. Далее студент осуществляет выполнение необходимых заданий по построению моделей ФТЗ (или их систем управления) в СМ МАРС с использованием библиотеки моделей компонентов.

4. Система интерактивного документирования при непосредственном участии студента фиксирует результаты выполнения работы в шаблоне отчёта.

5. По окончании работы студент через ИСУЛ открывает текст отчёта, вносит необходимые изменения и отправляет его в Moodle для проверки преподавателю. Во время самостоятельной (внеаудиторной) работы студенту через *Moodle* доступен модуль обучения, направленные на закрепление навыком анализа ФТЗ и приведения их описания к формализму ММКЦ.

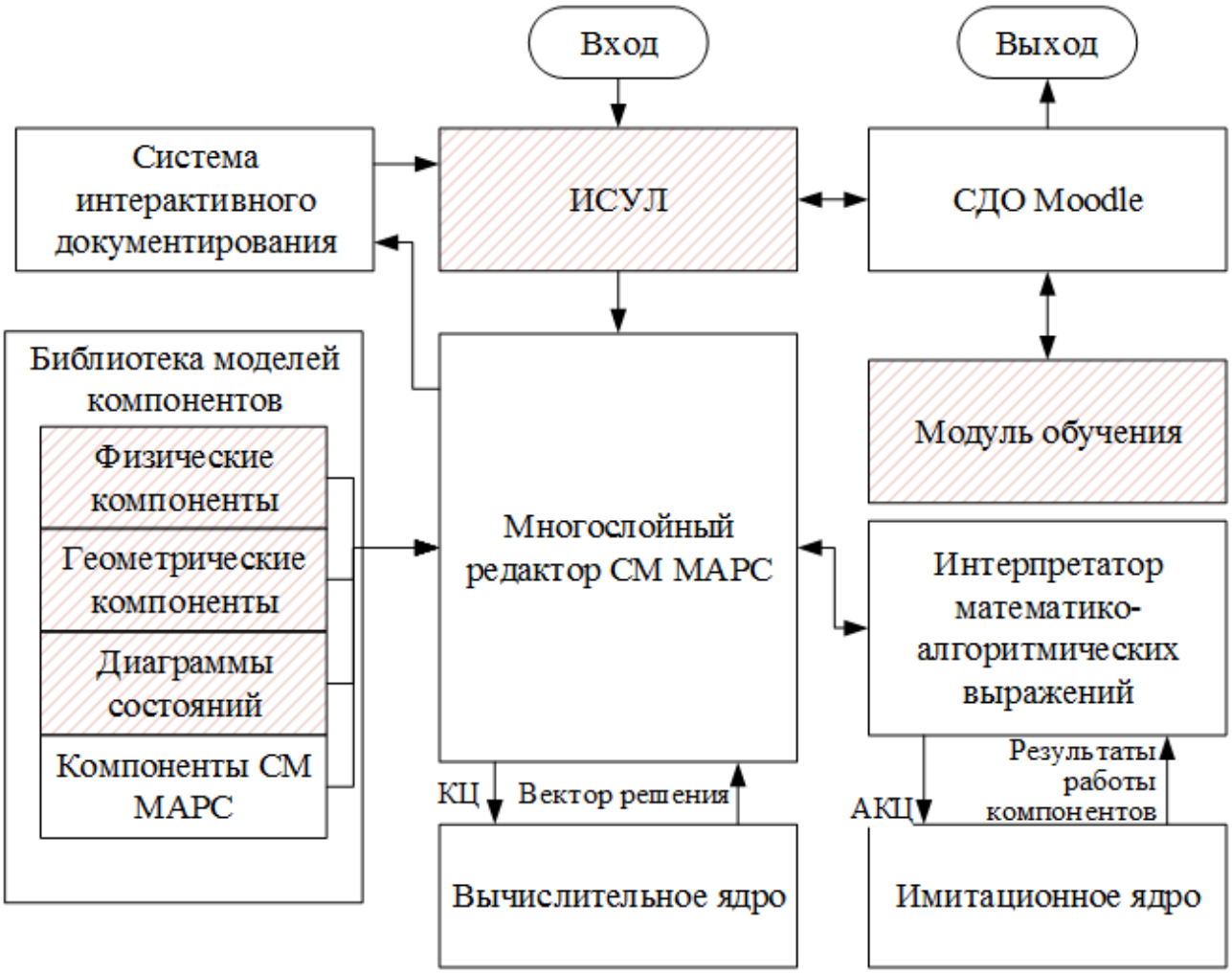


Рисунок 3.1 – Структура комплекса программ

3.1 БИБЛИОТЕКА МОДЕЛЕЙ КОМПОНЕНТОВ ДЛЯ МОДЕЛИРОВАНИЯ ФИЗИКО-ТЕХНИЧЕСКИХ ЗАДАЧ

«Библиотека моделей компонентов для многоуровневого компьютерного моделирования физико-технических задач в среде моделирования МАРС»[2],

[2] Свидетельство о государственной регистрации программы для ЭВМ №2019660693. Библиотека моделей компонентов для многоуровневого компьютерного моделирования физико-технических задач в среде моделирования МАРС / Кочергин М.И. – 12.08.2019. – М.: Роспатент, 2019.

предназначена для многоуровневого компонентного моделирования ФТЗ в рамках формализма ММКЦ в СМ МАРС и состоит из трёх блоков, охватывающих их отличительные особенности:

1. Компоненты для моделирования гибридного поведения, позволяющие учитывать сложное (гибридное) поведение объектов в задачах (см. параграф 2.2 Главы 2) – рис. 3.2.

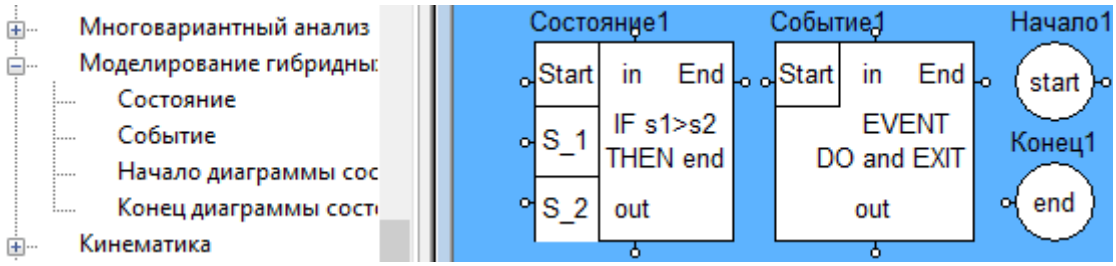


Рисунок 3.2 – Компоненты для моделирования гибридного поведения

2. Геометрические компоненты, позволяющие отображать геометрические свойства объектов, обусловливающие структурную сложность объектов в задачах: 1) *геометрические примитивы* – наборы базовых геометрических фигур, 2) *геометрические преобразователи* – компоненты для выполнения геометрических преобразований, 3) *геометрические решатели* – компоненты для решения базовых геометрических задач, 4) *кривые* – компоненты, задающие профили поверхностей.

На рис. 3.3 представлены некоторые примеры геометрических компонентов.

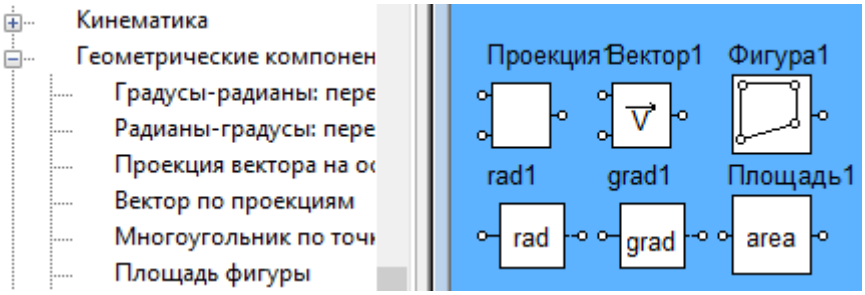


Рисунок 3.3 – Пример геометрических компонентов

3. Физические компоненты, позволяющие отображать физические свойства и явления в задачах с помощью готовых типовых компонентов: 1) источники физических величин; 2) модели физических тел; 3) физические эффекты, – и представлять компьютерные модели непрерывного (физического)

поведения объектов в виде цепи связных компонентов, непосредственно соответствующих взаимодействующим в задаче объектам, например, «твёрдое тело – атмосфера» или «твёрдое тело – поверхность».

На рис. 3.4 представлены некоторые примеры физических компонентов.

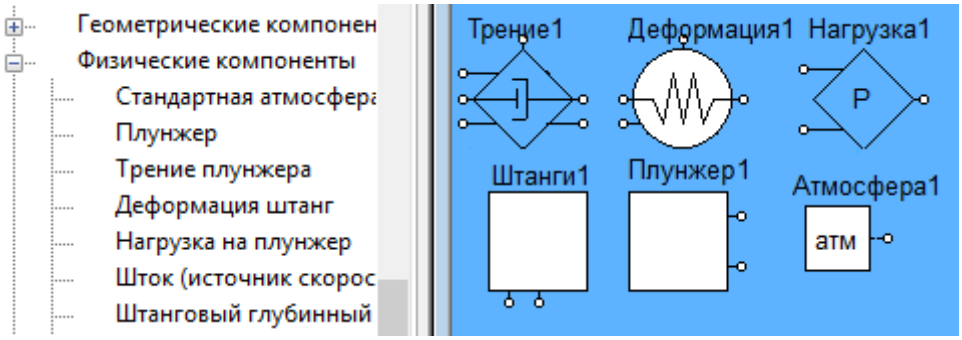


Рисунок 3.4 – Пример физических компонентов

Вышеописанные компоненты предназначены для моделирования класса ФТЗ и, в частности, конкретной модели – штангового глубинного насоса, рассматриваемого в параграфе 4.2. Использование представленных компонентов предоставляет возможность быстрой разработки компьютерных моделей с высокой степенью визуализации и проведения виртуальных экспериментов, не требующих написания программного кода на языках программирования. Отмечается, что применение таких инструментальных средств как среды визуального моделирования в образовательной деятельности на аудиторных занятиях различного вида и во время самостоятельной работы студентов позволяет повысить степень их вовлеченности в работу, интерактивности и наглядности, а как следствие, качество преподавания и результаты учебной деятельности [Королёв, 2013].

3.2 ПРОГРАММНЫЙ МОДУЛЬ ОБУЧЕНИЯ КОМПЬЮТЕРНОМУ МОДЕЛИРОВАНИЮ ФИЗИКО-ТЕХНИЧЕСКИХ ЗАДАЧ НА ОСНОВЕ АНАЛИЗА СЛОВЕСНОГО ПОРТРЕТА

«Программный модуль для обучения компьютерному моделированию физико-технических задач»[3] предназначен для обучения студентов многоуровневому компьютерному моделированию ФТЗ, в том числе для обучения анализу задач со ССИД [Kochergin, 2017]. Работа модуля возможна как в режиме *иллюстрации*, так и в режиме *проверки*. В режиме *иллюстрации* программа осуществляет автоматический анализ русскоязычного текста условий введённой задачи, выделяет список взаимодействующих объектов, их параметры, характер взаимодействия и другие значимые элементы с целью формирования справочных инструкций, содержащих указания по анализу условий задачи, выбору компонентов из библиотеки моделей компонентов и порядку построения компонентной модели. В режиме *проверки* осуществляется проверка правильности анализа задачи в вопросно-ответной форме. Комбинация двух режимов позволяет достичь образовательного эффекта.

3.2.1 НАЗНАЧЕНИЕ ПРОГРАММНОГО МОДУЛЯ

Программный модуль представляет собой часть Системы обучения компьютерному моделированию ФТЗ [Кочергин, 2016], ориентированную в первую очередь на самостоятельную работу студентов. Другими элементами разрабатываемой системы являются «Интерфейс виртуальной лаборатории», описываемый в параграфе 3.3, электронный курс в СДО *Moodle*, генератор текстов задач, модуль проверки схемы задачи. Отличительной чертой такой системы должна быть возможность демонстрации процесса моделирования задачи на любом примере, а не только из фиксированного набора. Это позволит повысить автономность работы обучающей системы и осуществлять поддержку

[3] Свидетельство о государственной регистрации программы для ЭВМ №2019660759. Программный модуль для обучения компьютерному моделированию физико-технических задач / Кочергин М.И. – 13.08.2019. – М.: Роспатент, 2019.

исследования учащимся задач любого типа. Определим необходимый функционал такой системы обучения в целом (полужирным выделены функции, реализуемые в разрабатываемом модуле):

1) предоставление теоретического материала по предметной области и компьютерному моделированию;

2) иллюстрация процесса построения модели задачи от этапа анализа условий до построения компьютерной модели в СМ МАРС на конкретном примере любой ФЗ или ФТЗ (даже введённой учащимся), а не только ограниченного фиксированного набора;

3) проверка корректности составленной пользователем модели задачи, информирование пользователя о характере произведённых им ошибок для последующего их исправления;

4) генерация и подбор задач для решения в соответствии с ростом уровня их сложности;

5) ведение статистики успешности прохождения обучения пользователем;

6) выработка рекомендаций учащемуся к дополнительной проработке теоретического материала и решению новых наборов задач по тем разделам физики, где им были совершены ошибки.

Функционал, указанный в п. 1, 5, 6, может быть реализован в существующих средах для сопровождения обучения, например, в СДО Moodle. Таким образом, в рамках данной работы остановимся на п. 2, 4.

3.2.2 СТРУКТУРА И ФУНКЦИИ ПРОГРАММНОГО МОДУЛЯ ОБУЧЕНИЯ МОДЕЛИРОВАНИЮ

Под *формализацией* будем понимать процесс выявления в предметной области (ПО) элементов (действующих объектов более низкого уровня), их важных характеристик и параметров, под *структуризацией* – процесс моделирования ПО или её экземпляра, выделения отношений между такими объектами и их параметрами. Полученное в ходе указанных процедур

формализованное и структурированное представление может быть переведено в компьютерную модель.

Для любой предметной области можно выделить три уровня автоматизированной формализации текстовых описаний [Кочергин, Кочергина, 2016]:

1) формализация описания на уровне текста – выделение из текстового описания сущностей, их признаков, действий, а также отношений с другими сущностями (получаемое при этом представление имеет вид семантического графа, где вершинами являются слова, а дугами – связывающие их семантические отношения, например, «часть-целое», «агент-реципиент» и др.),

2) формализация описания на предметном (аспектном) уровне – объективизация и параметризация полученного представления методикой многоаспектного анализа с использованием знаний из предметной области, в ходе которого оно получает вид отдельных взаимосвязанных наборов вида «объект-действие-параметр»,

3) формализация описания на поведенческом (модельном) уровне – структуризация полученного представления (выявление неявных и математическое описание найденных ранее межобъектных связей), в ходе которой оно приобретает более сложную единую форму. Так ФЗ и ФТЗ отображаются с помощью диаграмм поведения [Кочергин, 2016], характеризующих связи объектов в задаче и смены их поведений. Вводимые на данном уровне модели поведения и отношений объектов описывают их действия, состояние и взаимосвязи с помощью конкретных физических и математических закономерностей.

ФЗ и ФТЗ как объект формализации

Одним из простых примеров описаний физико-химических процессов являются тексты ФЗ и ФТЗ: 1) в них содержится описание некоторого физико-химического процесса (условие задачи), 2) в них ставится некоторая проблема, которую необходимо решить (требования задачи), 3) для их решения необходимо знание предметной области – физических законов.

Другими словами, решение задачи по физике можно рассматривать как процедуру исследования некоторого явления или процесса, требующего экспертных знаний в соответствующей предметной области (физика) и поиска подходящего решения поставленной проблемы (нахождение значения искомого параметра).

Автоматизированная формализация задачи производится модулем формализации задачи, который состоит из:

1) *лингвистического блока*, осуществляющего предобработку текста задачи [Huffman, 1995; Палагин, 2009]; данный модуль осуществляет анализ на уровне естетсвенноязыкового представления задачи;

2) *блока логической обработки*, производящего построение информационной модели задачи по его формализованному (размеченному) текстовому описанию;

3) *блока формирования* инструкции [Суранова, 2013], осуществляющего формирование пользовательской инструкции-алгоритма. Структурная схема этого модуля представлена ниже на Рисунке 3.5.

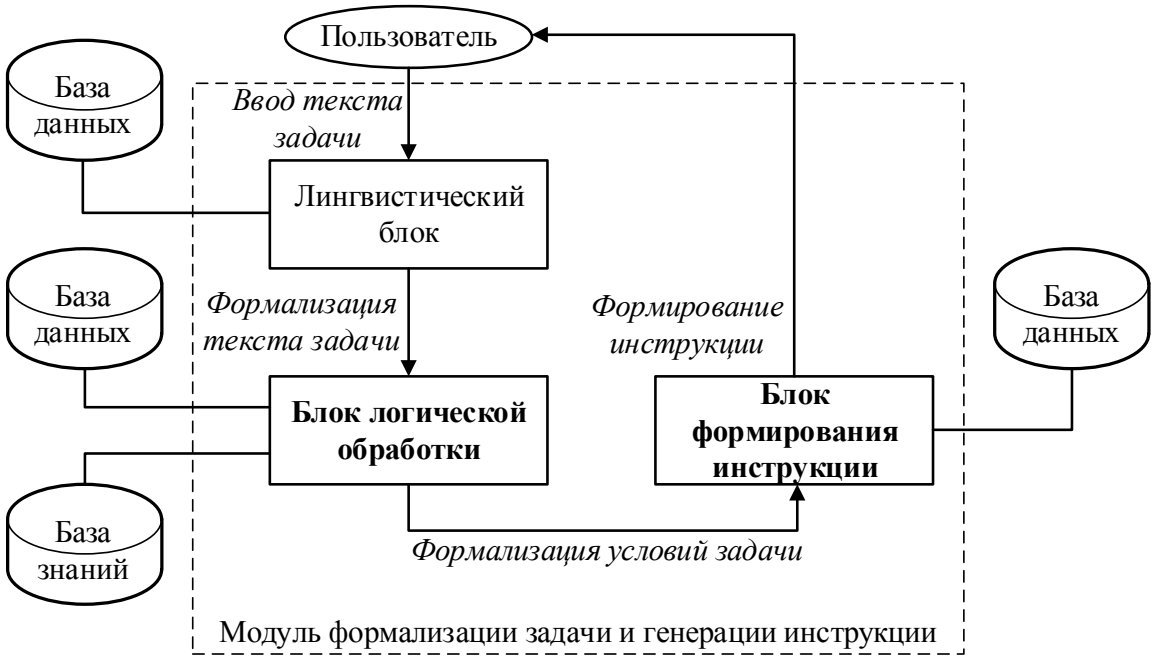


Рисунок 3.5 – Структура модуля формализации задачи

Автоматизированная формализация физических задач состоит из двух шагов:

1) предобработка текстовых условий задачи – построение их предварительного формализованного представления (формализация на уровне текста), минимально иллюстрирующего содержание задачи с помощью семантических отношений между выделенными в ней сущностями,

2) непосредственно интеллектуальная обработка полученного графа – извлечение из него необходимой полезной информации и приведение самой задачи к формализованному виду (сначала на предметном уровне, затем на поведенческом).

На первом этапе применяются методы компьютерной лингвистики, производится автоматический анализ текстовых условий задачи, в результате которого строится семантический граф (такое представление может быть представлено и в другом виде, например, в предикатном). Минимальной (атомарной) единицей формализуемого представления на данном этапе является слово.

На втором этапе производится интеллектуальная обработка полученного представления, заключающаяся в его объективизации, параметризации и структуризации. Результат данного этапа – формальное представление (информационная модель) описываемого в тексте процесса или объекта (системы объектов), пригодное для дальнейшей постобработки. Минимальной единицей такого представления является уже не слово, а модель объекта или его фрагмента, выраженная на математическом языке и описывающая его. Так ФЗ и ФТЗ на данном этапе представляется в виде диаграммы, иллюстрирующей смену поведения (описываемого законами физики) действующими объектами и взаимных качественных и количественных связей этих объектов. Такое представление, на наш взгляд, удобно описывать терминами теории моделирования: с помощью объектных моделей (ОМ) – физических законов, характеризующих поведение или состояние объекта в определённый момент времени, и моделей отношений (МО) – физико-математических законов, характеризующих взаимосвязи параметров различных объектов.

Перевод текстовых описаний на формальный язык

Предобработка текстовых условий задачи. Предобработка текстовых описаний задач по физике осуществляется разработанным модулем формализации текстовых условий. Содержание исполняемых процедур на шаге предобработки представлено на рис. 3.6.

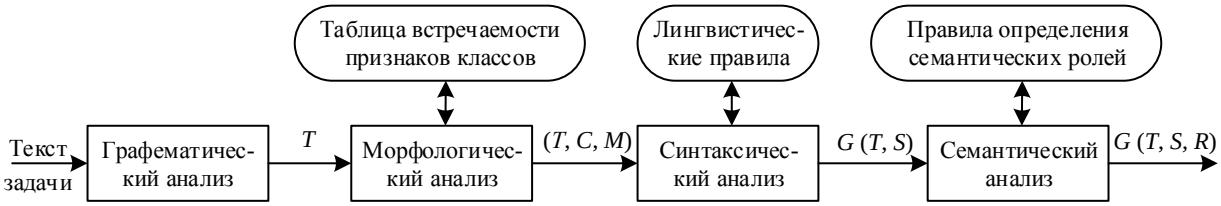


Рисунок 3.6 – Схема процедуры предобработки текстовых условий задачи

Так на первом этапе предобработки (графематический анализ) происходит простое разбиение текстового потока на минимальные единицы, разделённые пробелом, и их группировка (например, обозначение физической величины латинскими буквами группируется вместе с числовым значением и единицами измерения). Выходные данные этого этапа – множество T, представляющее последовательность текстовых единиц (ТЕ). ТЕ представляет собой абстрактную структуру данных с полями знаковой формы (слова в предложении) и набора числовых параметров.

На втором этапе (морфологический анализ) происходит классификация ТЕ (определение частеречной принадлежности слов к одному из классов множества C) на основании комбинированного (статистического и словарного) методов. Словарный метод используется для определения слов-исключений и условных обозначений (единицы измерения, латинские обозначения величин и т.д.), статистический – в остальных случаях. В соответствии с тем, к какому классу отнесён объект, ему присваиваются значения морфологических параметров (вектор M) из соответствующих составленных таблиц. Эти параметры несут информацию о характере рассматриваемого слова для определения его связи с остальными словами в предложении, а затем и в тексте задачи. Выходными данными этого этапа являются тройки (T, C, M), устанавливающие гомоморфное

отображение каждой ТЕ в некоторый класс (часть речи) и вектор числовых (морфологических) параметров.

На третьем этапе (синтаксический анализ) происходит анализ морфологических характеристик ТЕ для построения ориентированного графа *G*, отображающего их синтаксические связи *S* в предложении, что даёт первоначальное представление о семантике слов. Выходными данными этого этапа является граф *G* (*T, S*) синтаксических связей ТЕ.

На четвертом этапе (поверхностный семантический анализ), происходит распределение семантических ролей *R* по словам с использованием данных, полученных с предыдущих этапов. Семантические роли – минимальные смысловые категории слов, более полно характеризующие их семантическое значение (чем синтаксические функции), подобранные в ходе исследования на основе семантических падежей Ч. Филмора [Filmor, 1985]. Выходные данные этого этапа – взвешенный ориентированный семантико-синтаксический граф *G* (*T, S, R*), где веса *R* синтаксического графа *G* (*T, S*) – семантические роли ТЕ.

Этап предобработки итеративен для каждого предложения в тексте задачи. Для исключения ошибок классификации или взвешивания графов, на каждом этапе выполняется проверка непротиворечивости данных с предыдущего этапа, при необходимости инициируется перезапуск предыдущего этапа с пересчётом результата.

Отметим, что для предобработки текстовых описаний также применимы и другие существующие программные средства (синтактико-семантические парсеры), но при этом требуется их некоторая адаптация и дополнительная обработка полученного ими результата для корректной параметризации и структуризации формального представления, а также не гарантируется высокая точность предобработки.

Предобработка текстовых условий ФЗ и ФТЗ на предметном уровне

На данном этапе производится объективизация и параметризация строящегося формализованного представления ФЗ или ФТЗ. Для этого производится упрощение сформированного синтактико-семантического графа и

приведение его к структуре «объект-действие-параметр-значение». Указанная процедура осуществляется посредством поиска ключевых слов, занесённых в базу терминов предметной области, среди вершин графа. Фрагмент базы данных представлен в табл. 3.1.

Таблица 3.1 – Фрагмент базы ключевых слов в ФЗ

Слово	**Роль**	**Категория**
ехать	Действие	Движение
скорость	Параметр	Параметр действия
км/ч	Единица измерения	Скорость
равномерно	Характер	Действие

После того как найдено ключевое слово, ему присваивается новая функциональная роль (вместо семантической) из соответствующего (одноимённого) столбца. Данные из столбца «категория» являются вспомогательными, и используются для проверки адекватности формализованного представления и интерпретируются в зависимости от роли ключевого слова. Так для роли «действие» они сообщают о характере действия, для «параметра» – о принадлежности объекту или действию (например, «масса» – параметр объекта, а «пройденный путь» – действия), для «единицы измерения» – о параметре, который измеряется в этих единицах.

После того, как всем ключевым словам присвоены новые роли, производится построение нового графа с исключением вершин, не содержащих ключевых слов, кроме имеющих роль «агент» или «реципиент» или содержащих признаки объектов, необходимые для их дифференциации.

Логическая обработка формализованного представления условий задачи

Логическая обработка заключается в применении набора правил к полученному формализованному представлению [Рассел, 2006]. Применение таких продукционных правил позволяет определять, как тип задачи (раздел физики), так и объектные и системные модели, и их связи, что является

необходимым и достаточным условием для построения модели задачи в формате МКЦ. Используемый язык правил представлен ниже:

<правило>::= ЕСЛИ <антецедент> ТО <консеквент> КОНЕЦЕСЛИ;

<антецедент>::= (И {<условие>}) | (ИЛИ {<условие>});

<условие>::= <переменная> <оператор> <значение>;

<консеквент>::={<действие>};

<действие>::= (<переменная> = <значение>);

Примеры правил, написанные на приведённом формальном языке представлены в Таблице 3.2.

Таблица 3.2 – Примеры правил логической обработки

Тип правила	Правило
Правило определения раздела	ЕСЛИ (ПараметрДействия = «ускорение») ИЛИ (ПараметрДействия = «скорость») И (ПараметрДействия = «расстояние») И (ПараметрДействия = «время») И (ПараметрДействия != «импульс») И (ПараметрДействия != «масса») И (ПараметрДействия != «кинетическая_энергия») И (ПараметрДействия != «потенциальная_энергия») ТО (Раздел = «кинематика») КОНЕЦЕСЛИ;
Правило определения подраздела	ЕСЛИ (Раздел = «кинематика») И (ПараметрДействия != «ускорение») ТО (Подраздел = «равномерное прямолинейное движение») КОНЕЦЕСЛИ;
Правило определения объектной модели	ЕСЛИ (Подраздел = «равномерное прямолинейное движение») И (ПараметрДействия != «начальное_положение») И (ПараметрДействия != «текущее_положение») ТО (Модель = «путь») КОНЕЦЕСЛИ;

Формальное представление задачи хранится в специально разработанном формате, структура которого задаётся на программном уровне классом *FormalObject*. Структура представлена на рис. 3.7.

Рисунок 3.7 – Структура формального представления задачи

Класс *FormalObject* включает в себя описание новых типов данных.

1. Структурный тип *parameter* для описания параметра действия, включающий в себя следующие подтипы данных: а) строковая переменная *TextForm* для хранения названия параметра действия объекта; б) строковая переменная *Value* для хранения значения этого параметра; в) строковая переменная *Measure* для хранения размерности параметра; г) вектор строковых переменных *Attribute* для хранения значимых характеристик параметра.

2. Структурный тип *action* для описания действия объекта, включающий в себя следующие подтипы данных: а) строковая переменная *TextForm* для хранения названия действия; б) строковая переменная *Character* для хранения описания характера действия; в) вектор структурных переменных типа *parameter – Parameter* для хранения параметров действия; г) строковая переменная *Section* для описания раздела физики, к которому можно отнести совершаемое объектом действие; д) строковая переменная *Subsection* для описания подраздела физики; е) строковая переменная *Model* для описания модели поведения объекта.

3. Структурный тип *link* для хранения описания связи различных объектов в задаче, включающий в себя следующие подтипы данных: а) числовую переменную *LinkCode* для хранения ссылки на связанный объект; б) строковую переменную *LinkerTextForm* для хранения имени этого объекта; в) числовую переменную *FormCode* для хранения численного кода типа связи; г) строковую переменную *Form* для названия типа связи.

4. Структурный тип *sector* для хранения разметки правила (которая используется обработчиком логических правил), включающая в себя две переменные: *begin* и *end* для хранения ссылки на начало и конец участка соответственно.

Класс *FormalObject* включает в себя переменные: 1) вектор переменных типа *action – Action* – действия объекта в задаче; 2) вектор переменных типа *link – Link* – связи между объектами; 3) строковую переменную *ObjTextForm* – название объекта; 4) переменную *ObjNo* – ссылку на лингвистические параметры слова-объекта; 5) строковую переменную *Attribute* – характеристика объекта (для различения объектов, например, «первый» или «второй» велосипедист; а также следующие методы: 1) *Split* – для разбиения формализованного представления текста на меньшие участки с целью их дальнейшего анализа и заполнения формализованного представления задачи. Так семантический граф сложного предложения разбивается на несколько простых, вследствие чего подлежащее в каждом из предложений будет рассматриваться как новый объект. Но на этапе синтеза КЦ будет проводиться проверка полноты представления параметров формального объекта, так один и тот же объект может описываться в разных предложениях, например: «Поезд ехал 30 секунд. За это время он проехал 500 м»; 2) *Fill* – для заполнения формализованного представления задачи на основе анализа текстового представления. В качестве источника данных также используются размерности встречающихся в задаче физических величин (метры, килограммы и так далее), которые с некоторой однозначностью на следующих этапах помогут определить раздел физики, к которому относится задача, и даже модель поведения объекта.

Схема перехода от синтаксической (лингвистической) структуры представления текста задачи к формальному представлению её условий, реализованная в методе *Fill*, представлена на рис. 3.8.

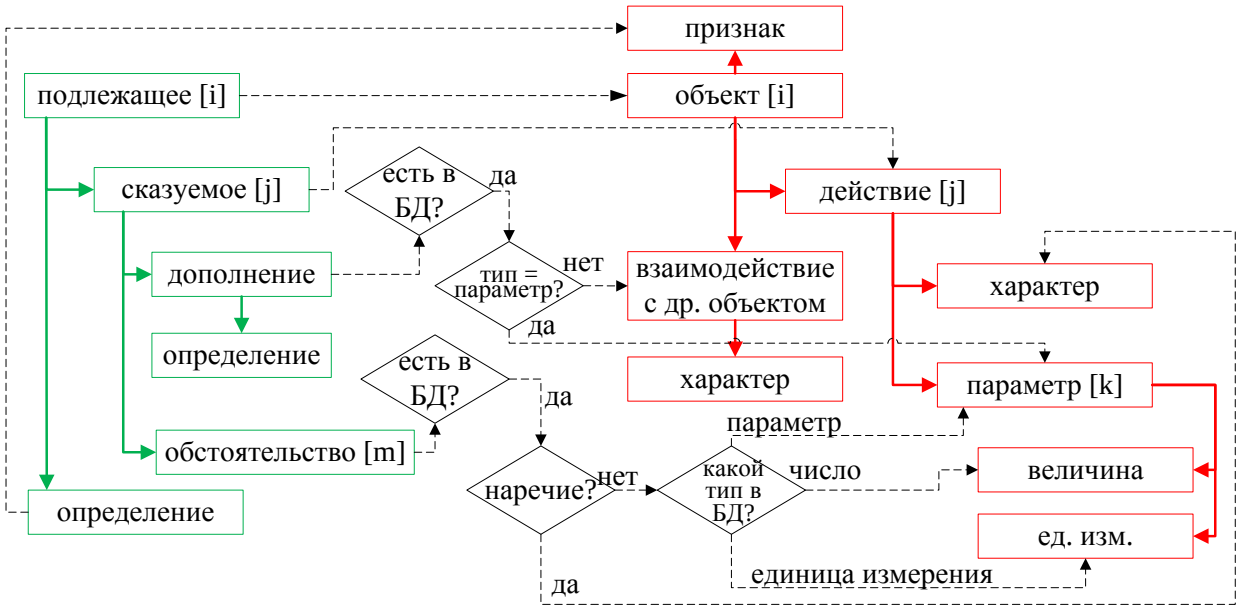


Рисунок 3.8 – Переход от текста к формализованной структуре

При переходе к формализованному представлению задачи (формальному виду) используется информация о значимых словах из таблицы маркёров *Markers* базы данных. После формирования формальной структуры условий задачи запускается процедура анализа продукционных правил.

Чтение и анализ продукционных правил производится с помощью разработанного логического обработчика правил. Обработчик считывает правило из базы знаний и последовательно проверяет выполняемость условий, записанных в антецеденте [Ясницкий, 2008]. В случае истинности антецедента запускается выполнение действий, записанных в консеквенте, в случае ложности выполняется проверка следующего подходящего правила. Подходящим считается то правило, которое соответствует типу текущего рассматриваемого правила. Так для каждого правила указано, какую переменную оно определяет: раздел, подраздел или модель поведения. Порядок хода обработки следующий:

1) сначала обрабатываются все правила, определяющие раздел физики, к которому можно отнести действия объекта (физического тела) в задаче, например, кинематика, динамика оптика и так далее (при этом остальные правила не рассматриваются);

2) затем обрабатываются правила, определяющие подраздел – категорию моделей, например, равномерное прямолинейное движение или равноускоренное прямолинейное движение;

3) после этого обрабатываются правила, определяющие модель объекта. После определения объектных моделей для всех объектов и их действий работа обработчика прекращается. Алгоритм работы обработчика представлен на рис. 3.9.

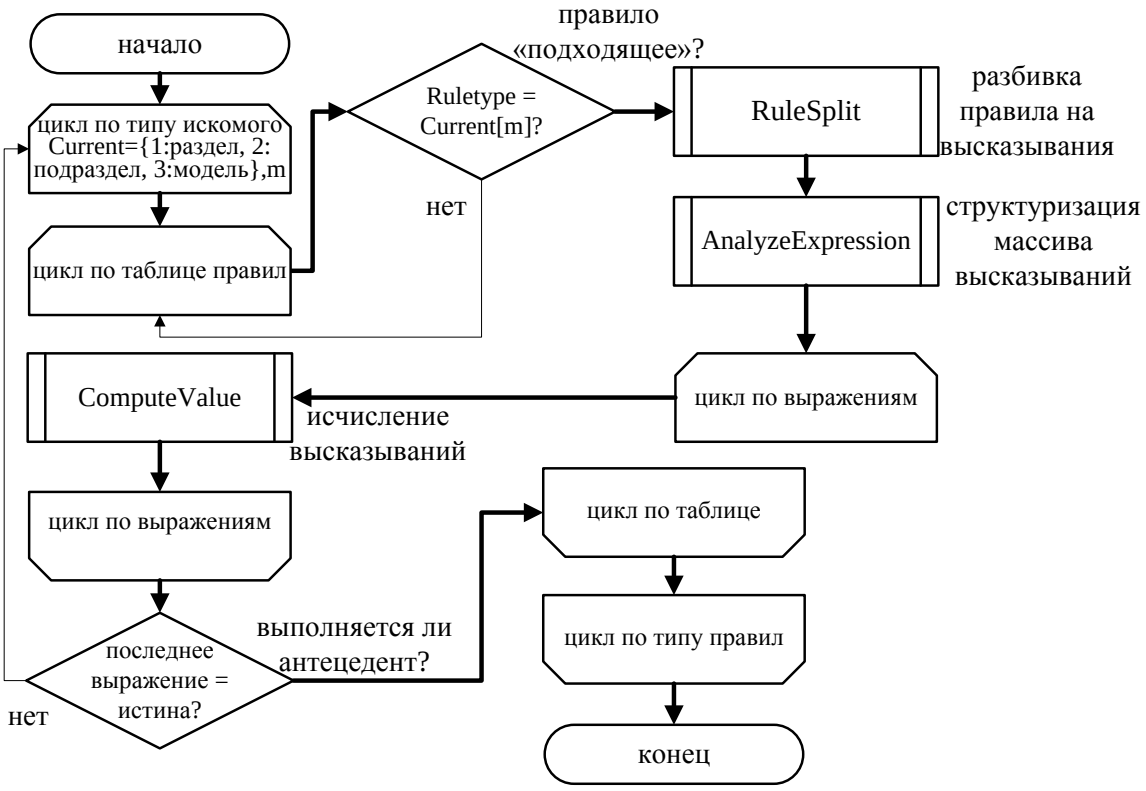


Рисунок 3.9 – Алгоритм работы обработчика правил

Логический обработчик продукционных правил реализован в классе *Rule_analysis*. Этот класс определяет следующие переменные: 1) вектор классового типа *Logic_expression – Expression* для хранения вычисленных логических высказываний в формате булевой алгебры; 2) вектор строковых переменных *PropositionsArray* для хранения массива высказываний и операторов; 3) счётчик простых высказываний *OriginPropCount*; 4) счётчик составных высказываний *PropCount;* 5) переменная структурного типа proposition – *FinProp* для хранения консеквента; а также следующие методы: 1) *RuleSplit* для разделения рассматриваемого правила на высказывания,

заполнения вектора строковых переменных *PropositionsArray* и подсчёта числа высказываний. Так, например, правило «ЕСЛИ (ПараметрДействия != «начальное положение») И (ПараметрДействия != «текущее положение») ТО (Модель = «путь») КОНЕЦЕСЛИ;» будет разделено на массив строковых переменных (простых высказываний и бинарных операторов в консеквенте и антецеденте): 1_A: «(ПараметрДействия != «ускорение»)»; 2_O(A1, A2): И; 2_A: (ПараметрДействия != «текущее положение»); 3_K: (Модель = «путь»)». Здесь обозначения в индексе «А», «К», «О» означают антецедент, консеквент и бинарный оператор соответственно; 2) *FillExpression* для заполнения вектора выражений данными. Этот метод осуществляет структуризацию массива высказывания, полученного разделением из продукционного правила. Так составные высказывания антецедента распределяется по вектору *Expression*, а консеквент записывается в *FinProp*; 3) *Clear* для очищения вектора выражений и обнуления всех переменных.

Приведённый выше класс *Logic_expression* предназначен для определения истинности условий антецедента. Данный класс содержит переменные: 1) вектор высказываний *Proposition*; 2) вектор логических операторов *Operator*; 3) булеву переменную *Value*, хранящую информацию об истинности или ложности условия; и методы: 1) *FillExpression* для заполнения экземпляра класса данными; 2) *AnalyzeExpression* для приведения антецедента к линейной структуре и определения порядка исчисления высказывания. Так порядок исчисления для правила «ЕСЛИ (ПараметрДействия = «ускорение») ИЛИ (ПараметрДействия = «скорость») И (ПараметрДействия = «расстояние») И (ПараметрДействия = «время») ТО (Раздел = «кинематика») КОНЕЦЕСЛИ;» приведён в табл. 3.3.

Таблица 3.3 – Иллюстрация порядка исчисления высказываний

Номер выражения	Тип выражения	Содержание выражения
1	простое	(ПараметрДействия = «ускорение»)
2	простое	(ПараметрДействия = «скорость»)
3	простое	(ПараметрДействия = «расстояние»)

4	простое	(ПараметрДействия = «время»)
5	составное	(Выражение$_1$) ИЛИ (Выражение$_2$)
6	составное	(Выражение$_5$) И (Выражение$_3$)
7	составное	(Выражение$_6$) И (Выражение$_4$)

3) *ComputeValue* для исчисления высказываний.

Формируемое на данном этапе представление является информационной моделью рассматриваемой задачи. Далее возможно применение средств постобработки, производящей, например, аналитические преобразования для вывода расчётной формулы или же формирования компьютерной модели задачи

Построение компонентного портрета задачи

По окончании формализации задачи запускается процедура формирования компонентного портрета задачи (формального описания её компьютерной модели). Структура используемого представления КЦ представлена ниже на рис. 3.10.

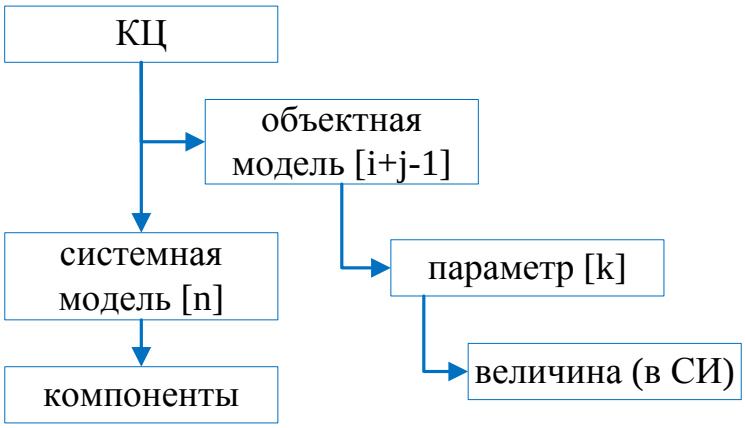


Рисунок 3.10 – Структура представление в формате КЦ

Процедура формирования КЦ реализована в классе *ComponentCircuit*. Этот класс объявляет переменные:

1) вектор структурных переменных типа *obj_model* – *ObjectModel* – объектные модели в задаче, они включают в себя переменные:

а) строковую переменную *TextForm* – название объектной модели,

б) строковую переменную *ObjectName* – имя объекта («велосипедист», «поезд»),

в) вектор параметров Parameter;

2) вектор структурных переменных типа *sys_model* – *SystemModel* – системные модели в задаче, они включают в себя переменные:

а) строковую переменную *TextForm* – название системной модели,

б) код системной модели *Code*,

в) строковую переменную *ObjectName* – имя связанного объекта,

г) ссылку на связанный объект – *ObjectCode*;

и методы:

1) *Unite* – производит слияние структур формальных объектов (то есть объектов в формализованном представлении задачи) в том случае, если они представляют собой единый объект, но разделены на несколько приложений, где название второго объекта выражено местоимением. Так, например, при построении формализованного представления условий в задаче «Поезд ехал 30 с. За это время он проехал 500 м. Определите, с какой скоростью ехал поезд» будет выделено 3 формальных объекта («поезд», «он», «поезд»), но затем они будут объединены в одну объектную модель, так как по сути представляют один объект.

2) *Fill* – производит заполнение переменных представления в виде КЦ данными на основе анализа формального представления задачи. Схема заполнения представлена на рис. 3.11.

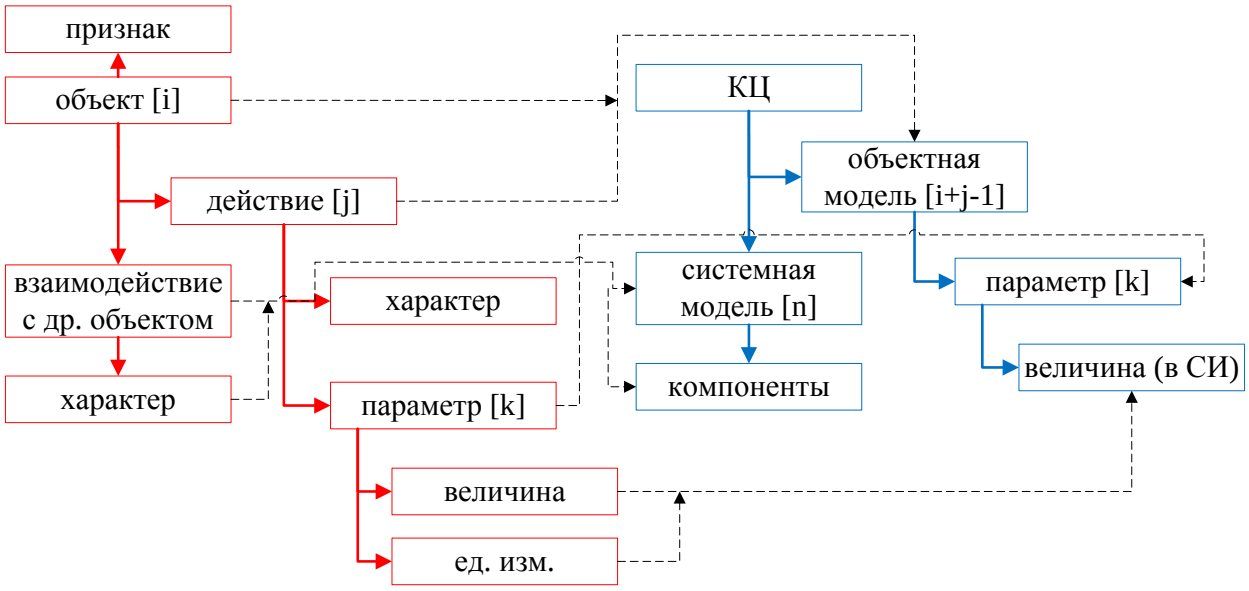


Рисунок 3.11 – Схема перехода от формального представления к КЦ

Пример формализации текстовых условий ФЗ

Рассмотрим простую физическую задачу из раздела кинематики: «Первую половину своего пути автомобиль двигался со скоростью 80 км/ч, а вторую – со скоростью 40 км/ч. Какова средняя скорость автомобиля на всем пути?» В данной задаче можно выделить один объект («автомобиль»), производящий смену поведения (изменение значения параметра «скорость» при равномерном прямолинейном движении). Проиллюстрируем поэтапно ход автоматической формализации рассматриваемой задачи. На рис. 3.12 изображен результат формализации задачи на уровне текста.

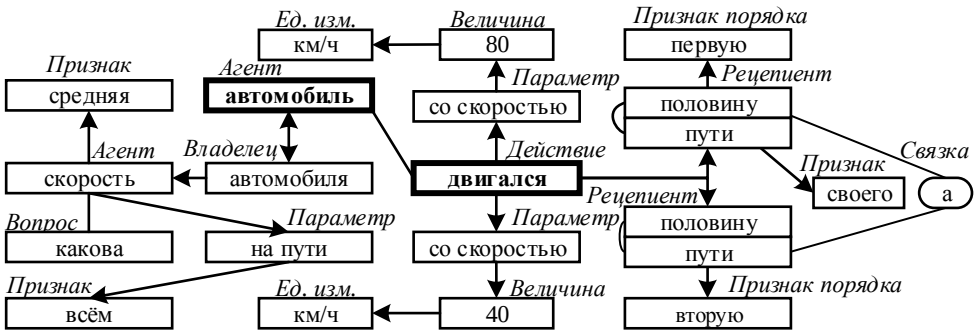


Рисунок 3.12 – Пример семантико-синтаксического графа задачи

Изображенный семантико-синтаксический граф поверхностно иллюстрирует смысловое содержание задачи только на уровне текста. Например, параметр «средняя скорость» во втором предложении приобретает роль «агента»

(действующего лица), так как является подлежащим. Данное представление нуждается в дальнейшей обработке.

Далее производится объективизация и параметризация полученного графа, результат которой представлен на рис. 3.13.

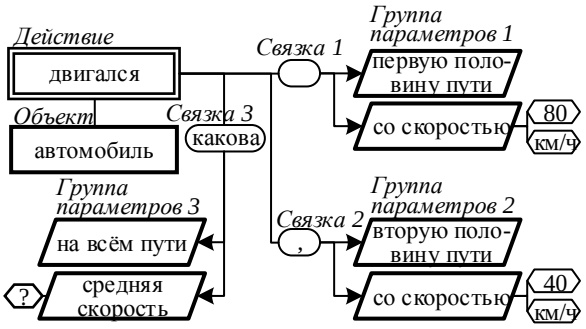


Рисунок 3.13 – Формализованное представление задачи на предметном уровне

На данном уровне выделяется объект – «автомобиль», совершающий одно действие, обладающее тремя сменяющимися наборами параметров, разделённых связками (выраженными синтаксически на уровне языка), позволяющими определить порядок смены поведения объектом.

Затем производится сопоставление выделенных групп параметров с физическими законами, описанными в базе знаний предметной области, по критерию характерности переменной закону и окончательная структуризация задачи. Результат выполнения указанных процедур представлен графически на рис. 3.14.

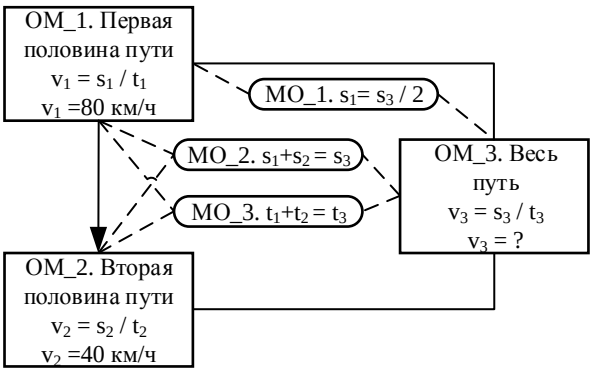


Рисунок 3.14 – Формализованное представление задачи на поведенческом уровне

В данной задаче выделяется три поведения объекта (объектные модели: ОМ_1, ОМ_2, ОМ_3) и три связывающих их математические зависимости

(модели отношений: МО_1, МО_2, МО_3). Полученное представление является окончательно формализованным и структурированным и пригодно для постобработки.

Разработанная система производит формализацию текстовых условий задач по физике. Формализация производится в три этапа: на уровне текста (с применением методов компьютерной лингвистики), на предметном уровне (посредством объективизации и параметризации условий задачи с использованием базы ключевых слов предметной области) и поведенческом уровне (посредством анализа и интерпретации продукционных правил из базы знаний предметной области). Предлагаемая схема является универсальной для различных предметных областей и обусловливается содержанием базы знаний и базы терминов (ключевых слов).

Формирование справочных материалов для моделирования физической задачи

На основе анализа полученной модели физической задачи производится построение связного текста на русском языке – справочных материалов, поясняющих алгоритм моделирования рассматриваемой задачи в СМ МАРС. Их текст строится по шаблону, структура и содержание которого варьируется в зависимости от типа рассматриваемой задачи.

Число базовых типов задач в каждом разделе конечно (их формальная составляющая). Остальные задачи являются их содержательными вариациями. Поэтому задача комплекса программ, направленного на обучение моделированию – наполнение разделов предметной области задачами базовых типов («формальными моделями») и сопровождение их справочными материалами!

Справочные материалы для задачи любого типа включают в себя:

1) краткие теоретические сведения по разделу физики, к которому относится моделируемая задача,

2) описание процедуры анализа условий рассматриваемой задачи: информация о том, как выделить и классифицировать ИЭ в данной задаче, какие действующие объекты выделять и как определять их параметры,

3) описание поведенческой структуры рассматриваемой задачи, списки объектных моделей и моделей отношений с указанием ИЭ, которые дают необходимую для этого информацию (указанной информации достаточно для самостоятельного построения компьютерной модели в СМ МАРС),

4) описание общей процедуры построения компьютерной модели в СМ МАРС – без указания состава компьютерной модели и порядка соединения компонентов для моделируемой задачи (данная информация в целях организации самостоятельной работы студентов выводится только для фиксированного набора задач),

5) сведения для интерпретации результатов, содержащие описание полученного решения.

Таким образом, формируемые системой обучения справочные материалы представляет собой не пошаговый алгоритм для моделирования физической задачи, а иллюстративный материал, предназначенный для разъяснения методики моделирования и демонстрации особенностей модельного представления задач на конкретном, интересующем учащегося примере. После ознакомления с такой инструкцией учащемуся предстоит самостоятельно перевести задачу из модельного представления в компьютерное, что способствует приобретению навыков моделирования и дополнительной проработке курса физики.

3.2.3 УКАЗАНИЯ К ПРИМЕНЕНИЮ ПРОГРАММНОГО МОДУЛЯ ОБУЧЕНИЯ МОДЕЛИРОВАНИЮ

Для работы с системой обучения предусмотрен следующий порядок. Работа с системой обучения может проводиться в двух режимах: в *режиме обучения* и *режиме проверки*. В режиме обучения пользователь может получить пошаговую инструкцию по построению КЦ конкретной задачи (также

предполагается возможность интерактивного взаимодействия в вопросно-ответной форме). Текст задачи может быть введён пользователем вручную или предложен ему из банка задач. Инструкция представляет собой пошаговый алгоритм, описывающий необходимые действия пользователя для построения модели задачи, и содержит указания по выбору необходимых компонентов из библиотеки компонентов, порядку их соединения, именованию, а также краткие пояснения к физическим законам, действующим в решаемой в системе задаче [Кочергин, 2015].

В режиме проверки пользователь может оценить свои навыки решения задач в СМ МАРС. Для этого пользователю из банка задач выводится одна из задач по физике, представленных в текстовом виде. Пользователь выполняет построение КЦ задачи в СМ МАРС и загружает полученный файл в систему. Далее модуль проверки задач осуществляет проверку правильности построения КЦ в данном файле, и пользователю выводится ответ (на русском языке), содержащий информацию о допущенных им ошибках при решении задачи.

Обучение моделированию заключается в комбинации указанных режимов работы системы: режима обучения для иллюстрации пользователю принципов моделирования в СМ МАРС и модельного представления в целом, а также получения им кратких пояснений к физическим законам; и режима проверки для оценивания приобретённых навыков моделирования у обучаемого.

Разрабатываемая система обучения моделированию задач может применяться как для самостоятельного обучения решению задач (так как в её структуру входят теоретические материалы, модуль обучения моделированию и модуль проверки навыков моделирования), так и для закрепления навыков решения на практике и проведения контроля знаний и умений.

Способ применения Системы обучения моделированию задач, состоящей из модулей построения модели задачи и формирования инструкции, представлен на рис. 3.15.

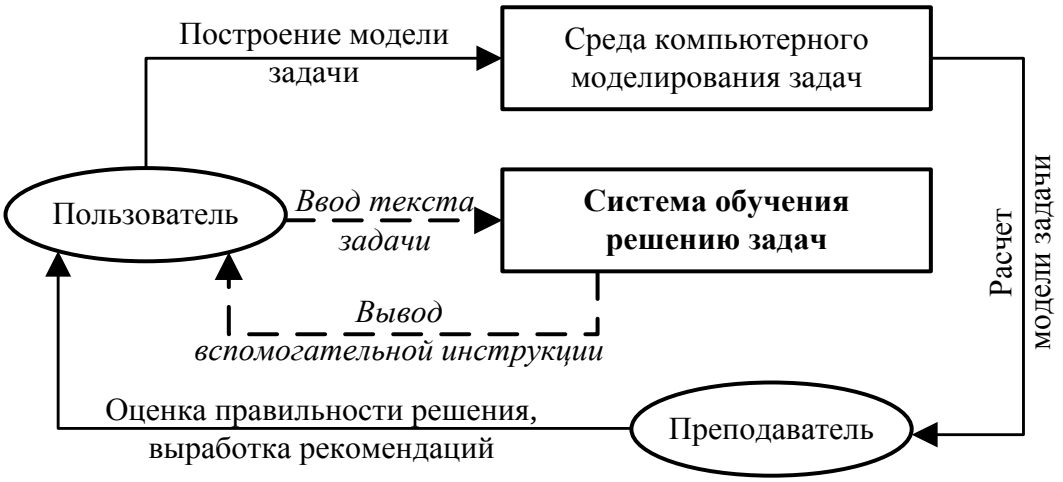


Рисунок 3.15 – Способ применения системы обучения

Интерфейс разработанного в рамках данной работы построения модели задачи и формирования пользовательской инструкции модуля представлен на рис. 3.16.

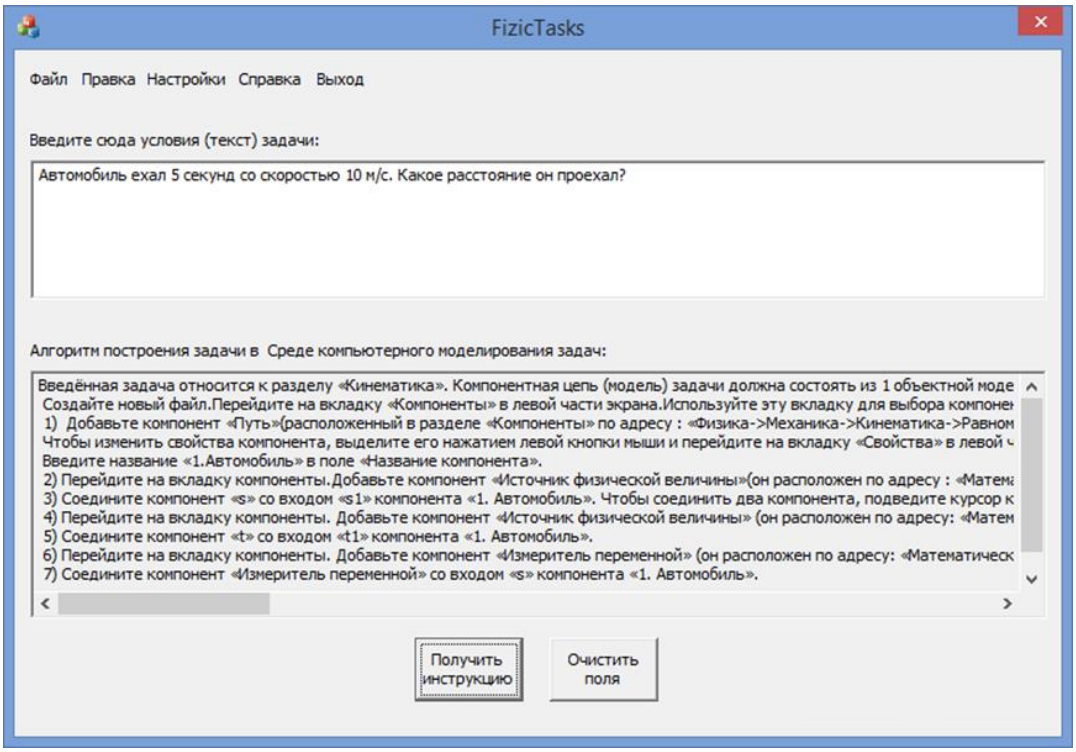


Рисунок 3.16 – Интерфейс модуля формирования инструкции

В верхнее окно вводится текст условий задачи. Для получения справочной инструкции необходимо нажать кнопку «Получить инструкцию». Инструкция, содержащая указания к порядку построения модели задачи в СМ МАРС появится в нижнем окне. Помимо указаний к выбору, расположению, наименованию и соединению компонентов, инструкция содержит краткие теоретические

сведения о физических законах, действующих в рассматриваемой задаче. Кнопка «Очистить поля» очищает окно ввода текста задачи и окно вывода инструкции. Кнопки «Файл», «Правка» предназначены для корректировки режима отображения текста в модуле, возможности сохранения текста задачи и инструкции или их открытия. Кнопка «Настройки» предназначена для доступа к базам данным и базе знаний. Справку о возможностях программы, порядок работы с ней можно посмотреть через меню «Справка». Выход из программы осуществляется нажатием кнопки «Выход».

Разрабатываемая система обучения моделированию задач предназначена для обучения студентов младших курсов вузов, ссузов и школьников моделированию методом ММКЦ. Особенностью системы является возможность иллюстрации учащимся процедуры моделирования любых ФЗ и ФТЗ, представленных в текстовом виде (а не только заранее внесённых примеров), что позволяет не только реализовать автоматизированную систему генерации задач и проверки навыков моделирования учащихся, но и даёт возможность самостоятельного обучения моделированию неограниченного круга задач.

3.3 ИНФОРМАЦИОННАЯ СИСТЕМА УПРАВЛЕНИЯ ВИРТУАЛЬНОЙ ЛАБОРАТОРИЕЙ МОДЕЛИРОВАНИЯ ФИЗИКО-ТЕХНИЧЕСКИХ ЗАДАЧ

При подготовке специалистов технического профиля важную роль играет проведение лабораторных работ и практических занятий, развивающих навыки применения экспериментального исследования реальных объектов и их компьютерных моделей [Кочергин, 2019]. Создание виртуальных, реально-виртуальных лабораторий позволяет проводить «эксперименты с оборудованием и материалом, которыми он не имеет возможности воспользоваться из-за отсутствия реальной лаборатории, получить практические навыки проведения экспериментов, ознакомиться детально с компьютерной моделью и процессом работы уникальной аппаратуры, исследовать опасные в реальной ситуации процессы и явления, не опасаясь за возможные последствия» [Черемисина,

Антипов, Белов, 2012; Саданова и др., 2016], наглядно визуализировать и эффективно обрабатывать результаты моделирования, организовывать взаимодействие реальных объектов и компьютерных моделей. Под виртуальной лабораторией понимаем вслед за В. В. Трухиным «программно-аппаратный комплекс, позволяющий проводить опыты без непосредственного контакта с реальной установкой или при полном отсутствии таковой». Как следствие, можно выделить дистанционные лаборатории (лаборатории с удаленным доступом) [The Virtual Lab …], которые могут быть как виртуальные, так и реально-виртуальные.

В качестве примеров виртуальных лабораторий приведём следующие:

- *STAR* (*Software Tools for Academics and Researchers*) [Software Tools …] – программное обеспечение, разработанное в Массачусетском технологическом институте для создания виртуальных лабораторий в исследовательских и образовательных целях в областях биологии, биохимии, генетики и пр., реализованных на *Java* или в *html*;
- *VirtuLab* [VirtuLab …] – проект по разработке с применением *Flash*-технологии виртуальных лабораторных работ по физике, химии и пр.;
- Виртуальные лаборатории *teachmen.ru* [Физикам …] – проект Челябинского государственного университета, содержащий виртуальные лабораторные работы по физике, а также интерактивные лекции;
- *Wolfram Demonstrations Project* [Wolfram …] – библиотека интерактивных демонстраций, содержащая более 12 000 интерактивных демонстраций, выполненных в математическом пакете *Wolfram Mathematica*.

Также необходимо отметить наличие разработок в области создания виртуальных лабораторий на базе среды *LabView* [Разработка школьных …]. Вышеперечисленные виртуальные лаборатории обладают существенным недостатком – они не опираются на инструментарий для компьютерного моделирования, что приводит к необходимости разработки каждой работы с ноля на языках программирования, а также созданию графических интерфейсов. Использование же среды моделирования, такой как СМ МАРС, позволяет при

создании лабораторных работ оперировать объектами более высокого уровня абстракции, создавать сценарии проведения виртуальных экспериментов, осуществлять автоматизированное документирование результатов моделирования [Дмитриев, Ганджа, Панов, 2014], составлять интерфейс работы из блоков панелей виртуальных приборов (ВИП) [Дмитриев, Ганджа, Панов, 2016]

Создание виртуальной (как и реально-виртуальной) лаборатории требует создания информационной системы управления, осуществляющей контроль и управление лабораторией, мониторинг работы студента и сопровождение образовательного процесса. В данном параграфе рассмотрим информационную систему управления виртуальной лабораторией моделирования физико-технических задач (ИСУЛ МФТЗ).

3.3.1 НАЗНАЧЕНИЕ И СТРУКТУРА ПРОГРАММНОГО МОДУЛЯ

ИСУЛ МФТЗ предназначена для сопровождения образовательного процесса при проведении лабораторных работ по таким дисциплинам как «Компьютерное моделирование физических задач», «Основы компьютерного моделирования физико-технических задач», «Моделирование систем». ИСУЛ взаимодействует с СДО Moodle как элементом Электронной информационно-образовательной среды Университета.

Схема взаимодействия ИСУЛ представлено на рис. 3.17.

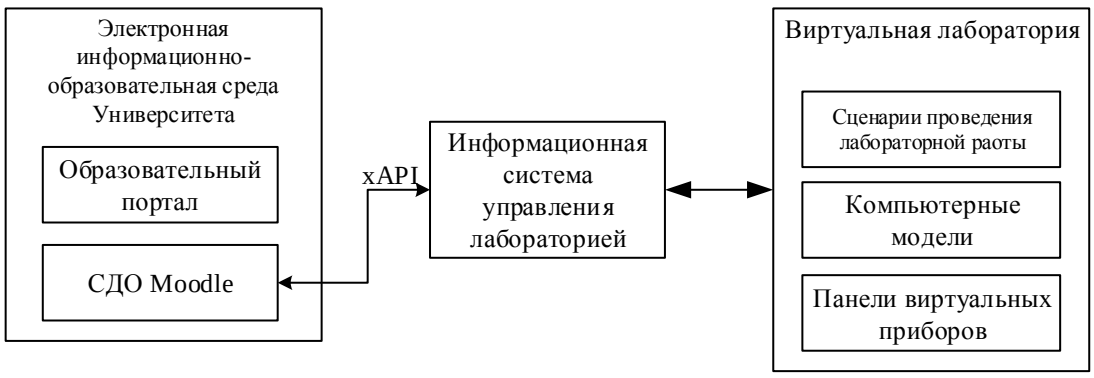


Рисунок 3.17 – Взаимодействие ИСУЛ с другими элементами

ИСУЛ представляет собой запускаемое оконное приложение, содержащее внутри *Web*-браузер для взаимодействия студента с СДО *Moodle*. ИСУЛ является связующим звеном между элементами курса (лабораторными и практическими работами) *Moodle* и соответствующими компьютерными моделями в СМ МАРС на локальном компьютере в учебной аудитории. Сценарии проведения лабораторных работ располагаются в *L*-слое МКМ моделей и могут быть как заданными изначально, так и составляться студентами в ходе выполнения лабораторной работы. ПВП располагаются на *V*-слое многоуровневых компьютерных моделей, при этом «содержимое» компьютерной модели может быть, как скрыто от студента (в случае учебно-иллюстративных моделей), так и доступно для модификации [Дмитриев, Ганджа, 2016].

ИСУЛ выполняет следующие функции:

1) взаимодействие с электронной средой (Moodle) – во время выполнения лабораторной работы используется студентами в качестве браузера;

2) обмен данными с Moodle – получение данных пользователя из Moodle, требуемых для автоматизированного заполнения отчёта, а также отправка файла отчёта в форму «загрузка файла» в Moodle.

3) предварительное заполнение отчёта данными пользователя из Moodle (непосредственно результаты моделирования записываются в отчёт согласно алгоритмическим конструкциям сценария проведения эксперимента).

4) интерфейс между электронным курсом в *Moodle* и компьютерной моделью – ИСУЛ задаёт соответствие между элементом курса (лабораторной или практической работой) и компьютерной моделью, что позволяет «связать» файл на локальном компьютере с элементом курса на сервере.

Рассмотрим подробнее типовой сценарий выполнения лабораторной работы:

1. Включение компьютера. Запуск ИСУЛ.

2. Изучение студентом методических материалов в СДО Moodle.

3. Прохождение студентом контрольных процедур в СДО Moodle для получения доступа к экспериментальной установке.

4. Генерация (представление) задания студенту в СДО Moodle.

5. Проведение лабораторного эксперимента.

6. Интерактивное документирование результатов экспериментирования, осуществляемое автоматически в многоуровневой компьютерной модели.

7. Протоколирование действий студента, автоматизированное формирование отчёта о лабораторной работе и его передача в СДО Moodle по протоколу xAPI [Романенко, 2019].

8. Проверка и оценивание работы преподавателем.

На этапах 2–4 преподаватель выполняет контролирующую функцию посредством мониторинга деятельности студентов через СДО Moodle.

Отметим другие функции, используемые в описываемой виртуальной лаборатории, но осуществляемые СДО Moodle:

1. Регистрация студентов, ведение их учётных записей.

2. Предоставление доступа к методическим материалам, в том числе с ограничениями (например, результат прохождения входного теста).

3. Проведение контрольных процедур на получение доступа к экспериментальной установке.

4. Мониторинг активности студента.

5. Организация предоставления преподавателю отчётов о лабораторных работах на проверку.

6. Организация офлайн-связи между преподавателем и студентом.

7. Формирование ведомости с оценками.

Отметим, что ИСУЛ может быть объединена с программным модулем обучения, описанным в параграфе 3.2 – в этом случае программный модуль будет использоваться при организации самостоятельной работы студента, в то время как ИСУЛ – при организации аудиторной (практической и лабораторной).

3.3.2 ОПИСАНИЕ ИНТЕРФЕЙСА ПРОГРАММНОГО МОДУЛЯ

Внешний вид ИСУЛ представлен на рис. 3.18. Кнопки в левом верхнем углу предназначены для навигации по электронному курсу в Moodle аналогично

соответствующим кнопкам в Web-браузере («Домой», «Назад», «Вперёд», «Обновить»).

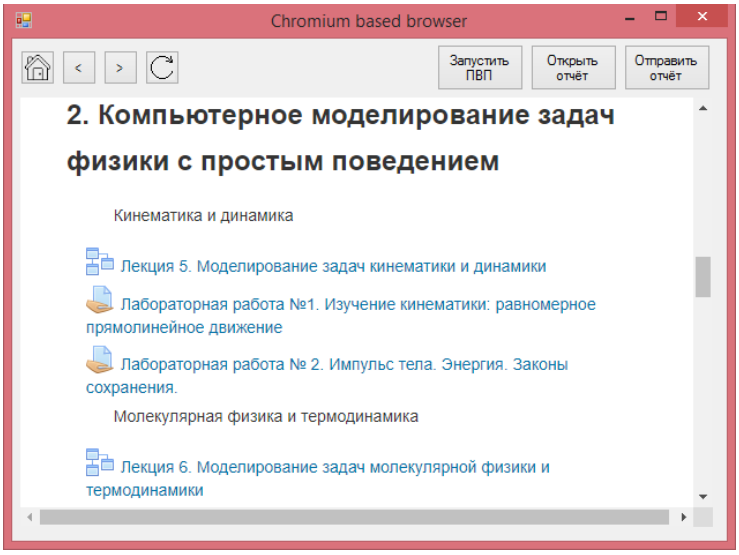


Рисунок 3.18 – Интерфейс ИСУЛ с открытой страницей курса в Moodle

Кнопки в правом верхнем углу имеют следующий функционал:

1) «Запустить ПВП» запускает компьютерную модель (или панель виртуальных приборов – ПВП – в случае реально-виртуальной лаборатории), соответствующую заданию, расположенному на открытой (в браузере) странице. Перед этим ИСУЛ запрашивает у сервера Moodle данные об авторизованном пользователе и заполняет титульный лист шаблона отчёта.

2) «Открыть отчёт» открывает студенту заполненный в ходе виртуального эксперимента отчёт в окне *Microsoft Word* для просмотра и редактирования. Нажатие этой кнопки производится студентом после завершения виртуального эксперимента.

3) «Отправить отчёт» отправляет отчёт в соответствующую форму открытой страницы с заданием для проверки преподавателем. Во время работы ИСУЛ соответствие между файлом отчёта на локальном компьютере и учётной записью студента в Moodle осуществляется по идентификатору, полученному от Moodle перед запуском модели.

3.4 СРАВНЕНИЕ С АНАЛОГАМИ

Разработанный комплекс программ состоит из: 1) библиотеки моделей компонентов для моделирования ФТЗ, 2) модуля обучения моделированию ФТЗ, 3) системы управления лабораторией моделирования ФТЗ, – и не имеет прямых полных аналогов. Представим сравнение результатов диссертационной работы с частичными аналогами по каждой составляющей программного комплекса.

1. Библиотека моделей компонентов.

1.1. В данной работе для моделирования дискретно-непрерывного поведения предлагаются диаграммы состояний языка ММКЦ. Их использование даёт положительный эффект не только по сравнению с СМ МАРС (использующим базовые универсальные компоненты низкого уровня абстракции – логические операторы), но и по сравнению с другими СМ, например, *Rand Model Designer* (использующим ГА), *Simulink* (использующим диаграммы *stateflow*). Более подробно данный вопрос рассмотрен в п. 2.2.2 работы. Здесь представим краткое описание основных преимуществ предлагаемого подхода:

1) сокращение объёма машинных ресурсов, требуемых для работы модели, в случае, когда одной непрерывной модели соответствует несколько дискретных – за счёт отделения дискретной модели объекта (диаграмм состояний L-слоя) от непрерывной (в *Rand model designer*, например, соответствие между дискретными и непрерывными моделями взаимно-однозначное);

2) диаграммы состояний языка ММКЦ могут взаимодействовать с моделями С-слоя, представленными в структурном виде, а не только в аналитическом (в отличие от *Rand Model Designer*, *Simulink* и пр.), что расширяет возможности их применения для моделирования ФТЗ;

3) диаграммы состояний языка ММКЦ реализуют механизм компенсации накапливаемой амплитудно-временной погрешности, что повышает точность моделей (подробнее см. в пункте 2.2.3 данной работы).

1.2. В данной работе для моделирования физических свойств объектов предложены физические компоненты, представляющие собой компоненты высокого уровня абстракции. Подобные компоненты отсутствуют в *Rand Model*

Designer, LabView, а от компонентов *Simulink* отличаются возможностью гибкого параметрирования (не только параметрами-константами, но и функциональными параметрами) за счёт встраивания в них ИМП, содержащих редактируемую формулу расчёта параметров. Использование таких компонентов повышает компактность и наглядность компонентной модели. Более подробно см. в пункте 2.3.2 данной работы.

1.3. Для моделирования геометрических свойств межобъектных отношений в ФТЗ предлагаются геометрические компоненты. Подобные компоненты отсутствуют в *Rand Model Designer, Simulink* и отличаются от блоков *LabView* тем, что позволяют решать более широкий круг задач и снижают размерность непрерывной модели (рассчитываемой вычислительным ядром) за счёт предварительного разрешения части уравнений, представленных в явном виде (рассчитываемых имитационным ядром). Использование геометрических компонентов также позволяет повысить наглядность компонентной модели ФТЗ за счёт отделения геометрической составляющей ФТЗ от физической. Подробнее см. в пункте 2.3.1.

2. Модуль обучения моделированию. Данный модуль формирует естественноязыковую справочную инструкцию пользователю по процедуре анализа текстовых условий ФЗ или ФТЗ, введённых пользователем в неё. Модуль состоит из блоков предобработки, логической обработки и формирования инструкции и опирается на методы автоматического анализа текста, методику многоаспектного анализа, предлагаемый в данной работе и формализм ММКЦ. Прямые аналоги данного модуля отсутствуют, а существующие программы, предназначенные для автоматической обработки текста (томита-парсер, GATE и пр.) осуществляют только предобработку текста (частеречную, синтаксическую и семантическую разметку). Предложенные в данной работе блоки логической обработки формализованного (предобработанного) представления ФТЗ являются оригинальными, а блок предобработки текста, учитывает специфику языка и стиля описания ФЗ и ФТЗ. Использование данного блока позволяет осуществлять поддержку самостоятельной (внеаудиторной) работы студента,

направленной на формирование навыков анализа ФЗ и ФТЗ, овладение основами и принципами ММКЦ. Более подробно см. в параграфе 3.2.

3. Информационная система управления виртуальной лабораторией моделирования ФТЗ. ИСУЛ представляет собой узкоспециализированное программное обеспечение и позволяет организовать аудиторную работу студентов. Отличительной особенностью ИСУЛ является осуществление взаимодействия студента с СДО Moodle и СМ МАРС, а также возможность автоматизированного заполнения отчёта о выполненной лабораторной работе данными студента и результатами моделирования. Более подробное описание представлено в параграфе 3.4.

ВЫВОДЫ ПО ГЛАВЕ 3

В данной главе рассмотрен и описан разработанный комплекс программ, состоящий из:

1) библиотеки моделей компонентов для моделирования физико-технических задач в СМ МАРС,

2) программный модуль для обучения компьютерному моделированию физико-технических (в том числе физических задач),

3) информационная система для управления лабораторией моделирования физико-технических задач.

Данный комплекс программ нацелен на формирование инструментария для моделирования физико-технических задач и обучения студента освоению этого инструментария.

Таким образом, в данной главе описаны следующие полученные результаты:

1. Разработана и реализована на языке C++ библиотека моделей компонентов для моделирования физико-технических задач, позволяющая отражать следующие особенности физико-технических задач:

а) дискретно-непрерывное (гибридное) поведение объектов в задаче, заключающееся в мгновенной смене одной непрерывной модели поведения

другой, посредством использования диаграмм состояний – КЦ, состоящих из компонентов «Состояние» и «Событие», соответствующих дискретному поведению;

б) физические свойства объектов посредством использования специализированных компонентов, инкапсулирующих внутри себя готовые модели, например, модели твёрдого тела, плунжера или уровней стандартной атмосферы;

в) геометрические свойства объектов посредством использования специальных компонентов – *геометрических примитивов* (фигур), *преобразователей*, *решателей*, *кривых*.

2. Разработан и реализован на языке C++ программный модуль для обучения компьютерному моделированию физико-технических задач, предназначенный для самостоятельной работы студентов, нацеленной на овладение навыками анализа, декомпозиции физических и физико-технических задач, синтеза их многоуровневых компьютерных моделей в СМ МАРС.

3. Разработана и реализована на языке C# информационная система управления лабораторией моделирования физико-технических задач, предназначенная для организации взаимодействия СДО Moodle и виртуальной лабораторией, реализованной в СМ МАРС, во время проведения аудиторных лабораторных работ по таким дисциплинам как «Компьютерное моделирование физических задач», «Основы компьютерного моделирования физико-технических задач», «Моделирование систем».

ГЛАВА 4. МОДЕЛИРОВАНИЕ ФИЗИКО-ТЕХНИЧЕСКИХ ЗАДАЧ МЕТОДОМ КОМПОНЕНТНЫХ ЦЕПЕЙ

В данной главе с целью демонстрации разработанного аппарата моделирования представлены единичные примеры многоуровневых компьютерных моделей ФТЗ, реализованных с применением разработанного подхода в целом и разработанной библиотеки компонентов в частности. Остальные примеры задач представлены в Приложении А.

4.1 НЕПРЕРЫВНАЯ МОДЕЛЬ ПОЛЁТА ТЕЛА В АТМОСФЕРЕ ЗЕМЛИ

Задачи внешней баллистики представляют особый интерес для их компьютерного моделирования в образовательных целях, так как:

1) являясь динамическими системами, позволяют разъяснить суть и особенности таких систем на своём примере,

2) помимо способствования углублению знаний в области физики и навыков моделирования, наглядно позволяют продемонстрировать процедуру итерационного моделирования – постепенного усложнения (детализации) модели и её отладки на каждом этапе.

Так **задача о движении в атмосфере Земли тела, брошенного под углом к горизонту**, может быть разделена на несколько уровней детализации:

1) идеальный полёт тела, 2) полёт тела с учётом сопротивления воздуха при его постоянной плотности, 3) полёт тела с учётом изменения его аэродинамических характеристик в зависимости от его скорости, 4) полёт тела с учётом изменения плотности воздуха с изменением высоты полёта, 5) полёт тела с учётом изменения воздействующей на него силы тяжести с изменением высоты полёта, 6) полёт тела с учётом кривизны Земли.

Рассмотрим поэтапную детализацию этой задачи и её построение в СМ МАРС. Эта задача рассматривалась также в работе. В случае полёта в безвоздушном пространстве (имеем «идеальный» полёт) движение тела можно

описать следующей системой уравнений (пригодной для итерационного решения):

$$\frac{dx}{dt} = v_x, \quad \frac{dv_x}{dt} = 0, \quad \frac{dy}{dt} = v_y, \quad \frac{dv_y}{dt} = -g \quad (4.1)$$

При таком представлении легко учесть новые физические эффекты, например, сопротивление воздуха. Реализуем данную модель в СМ МАРС с использованием ИМП. На рис. 4.1 представлен интерфейс модели, расположенный на V-слое.

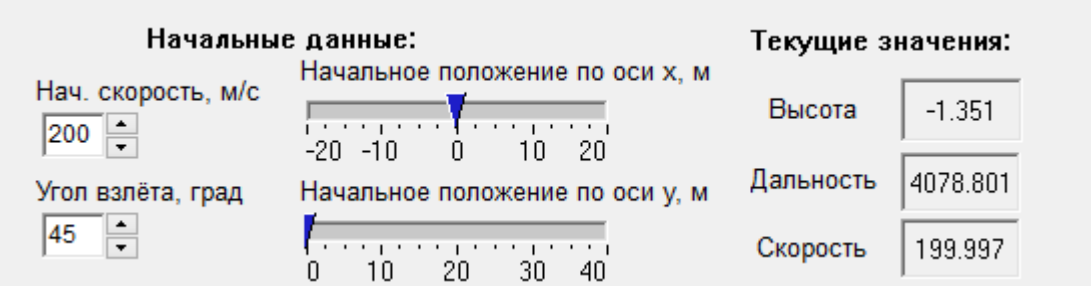


Рисунок 4.1 – Интерфейс для работы с моделью (V-слой)

На V-слое пользователь может ввести начальные данные для работы модели (начальная скорость тела, угол взлёта и координаты взлета тела по осям *X* и *Y*) и наблюдать за ходом работы модели. Измеряемыми данными являются текущие значения скорости тела, высоты полёта и пройденного расстояния. График изменения координат тела строится в «реальном времени» (с регулируемой скоростью) и отображается в отдельном окне (рис. 4.2).

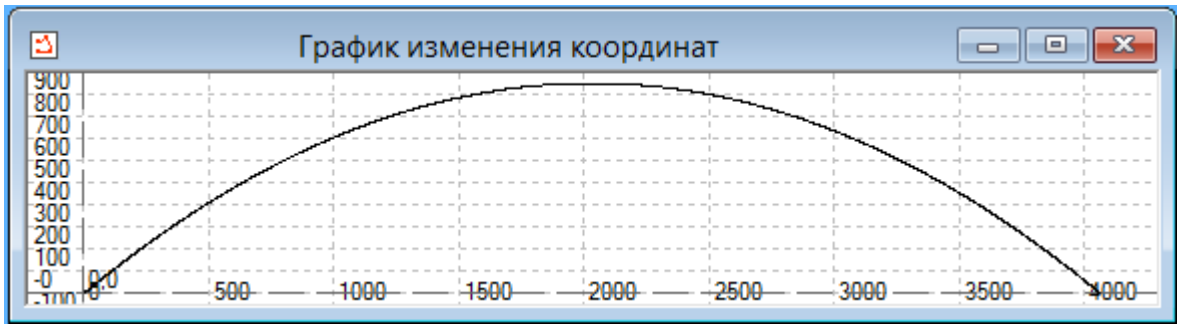


Рисунок 4.2 – График изменения координат тела *y*(*x*)

Как видно из графика тело, запущенное под углом 45° к земле при начальной скорости 200 м/с, не испытывая сопротивления атмосферы, пролетело 4 079 м. На рис. 4.3 представлен вид модели на *C*-слое, который непосредственно описывает физическое поведение тела, совершающего полёт.

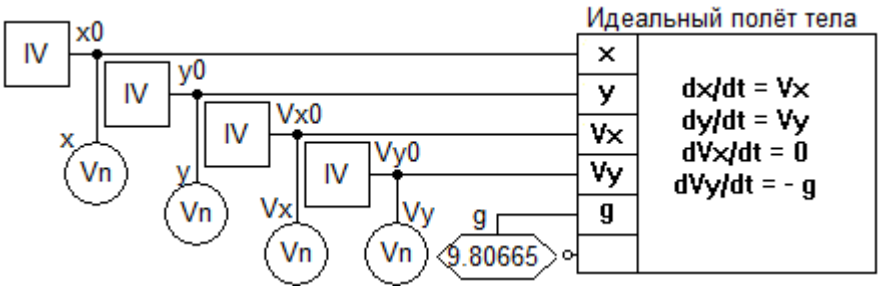


Рисунок 4.3 – Вид модели на объектном слое

ИМП «Идеальный полёт тела» содержит 4 дифференциальных уравнения, представленных в центре компонента. Также в КЦ *C*-слоя входят 4 источника начального значения (обозначены квадратами с надписью «*IV*» в центре): x_0, y_0, Vx_0, Vy_0, задающие начальные условия для решения системы уравнений. Эти компоненты имеют также отображение на *L*-слое, что позволяет получать значения в ИМП с *V*-слоя через *L*-слой. В КЦ также входят 4 измерителя (окружности с надписью «*Vn*» внутри), передающие рассчитанные значения на *L*-слой, откуда они поступают на *V*-слой и отображаются пользователю на соответствующих компонентах-визуализаторах. Значение ускорения свободного падения задаётся в компоненте-источнике g.

На рис. 4.4 представлены связующие КЦ, расположенные на L-слое.

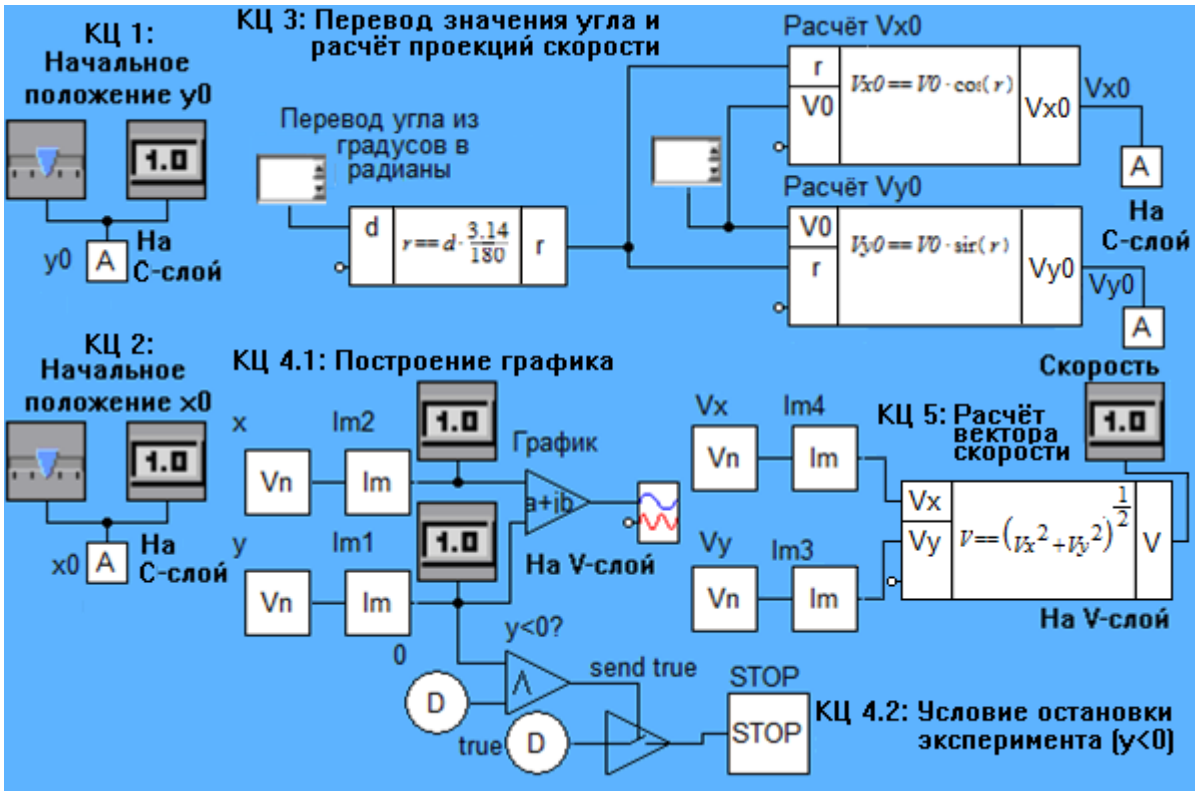


Рисунок 4.4 – Вид модели на L-слое

КЦ 1, 2 выполняют передачу значений x_0 и y_0 с визуального слоя на объектный, КЦ 3 производит перевод значения угла из градусов в радианы и расчет проекций вектора начальной скорости V_0. Подцепь 4.1 КЦ 4 предназначена для передачи данных с измерителей *x*, *y* *C*-слоя на табло *V*-слоя и график. Подцепь 4.2 КЦ 4 реализует механизм остановки работы модели в случае выполнения условия $y<0$, а КЦ №5 – расчёт вектора текущей скорости *V*.

Такое представление, обусловленное многослойной структурой СМ, позволят отделить математическую модель непрерывного физического поведения объекта (реализованную в виде КЦ на С-слое) от алгоритмов визуализации данных, вспомогательных расчётов, сценария проведения эксперимента или алгоритма поведения объекта (в виде отдельных КЦ на *L*-слое), а также от пользовательского интерфейса (расположенного на *V*-слое).

Произведём некоторое усложнение модели: начнём учитывать сопротивление воздуха. В модели появятся 2 новых параметра: масса тела m и сопротивление воздуха F_a (на данном этапе сделаем его константой, задаваемой пользователем). Математическая модель полёта снаряда примет вид:

$$\frac{dx}{dt} = v_x, \quad \frac{dv_x}{dt} = -\frac{F_{ax}}{m}, \quad \frac{dy}{dt} = v_y, \quad \frac{dv_y}{dt} = -g - \frac{F_{ay}}{m} \quad (4.2)$$

Для того чтобы внести данные изменения в описанную компьютерную модель, необходимо в редакторе схем СМ МАРС добавить компоненты-источники m и F_a, соединить их со свободными узлами ИМП «Полёт тела», присвоить им в настройках этого компонента имена и внести изменения в систему уравнений через редактор формул. Полученная компьютерная модель примет следующий вид (рис. 4.5). Согласно этой модели дальность полёта тела при тех же начальных данных и m = 20 кг, F_a = 100 Н составит 2 257 м.

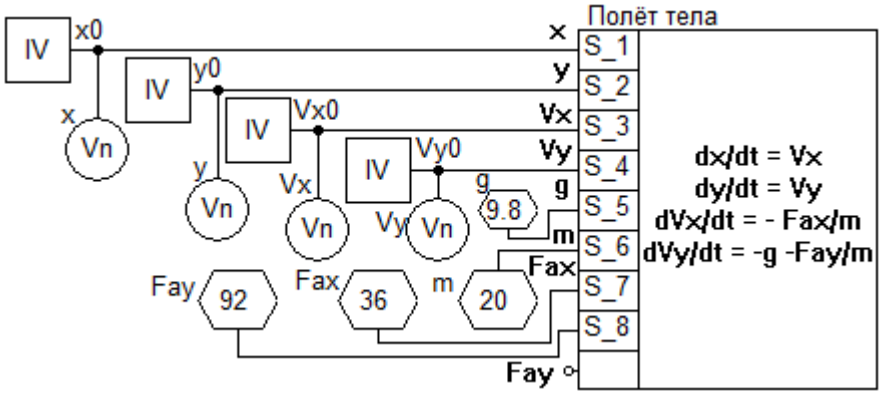


Рисунок 4.5 – Вид изменённой модели на C-слое

Для реализации возможности регулирования значений добавленных параметров (m и F_a) через интерфейс пользователя необходимо разместить соответствующие компоненты-регуляторы на *V*-слое. Эти компоненты также имеют отображение на *L*-слое, что позволяет производить обмен данными между слоями. Фрагмент изменённой модели на *L*-слое представлен на рис. 4.6.

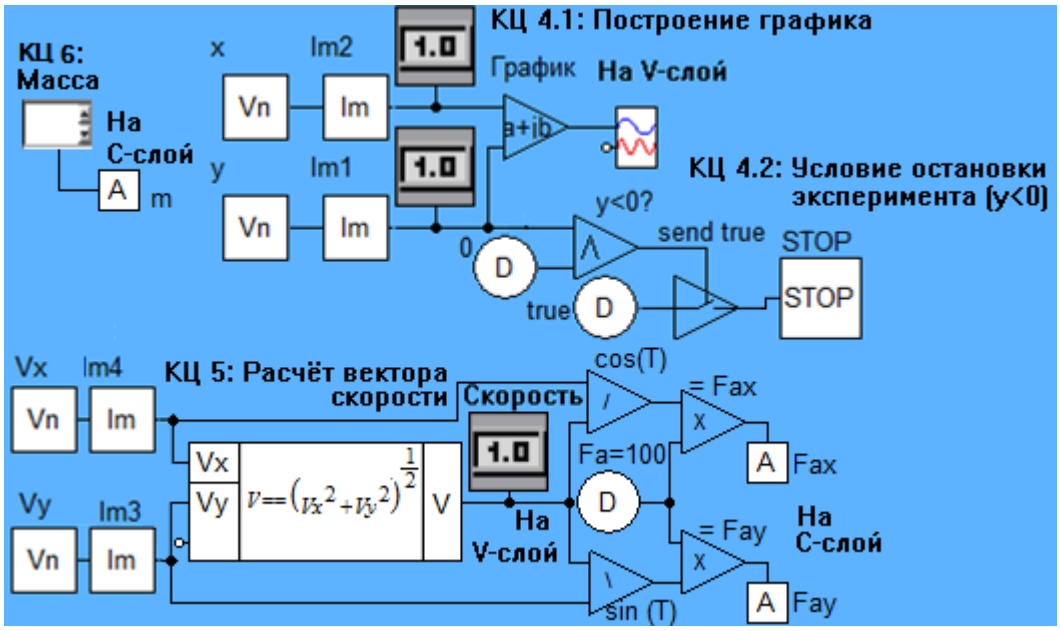


Рисунок 4.6 – Фрагмент изменённой модели на L-слое

Ещё усложним модель, сделав параметр F_a не константой, а переменной, зависящей от скорости тела и его аэродинамических характеристик. Для этого будем использовать зависимость $F_a = C_X \cdot S_M \cdot \rho \cdot V^2 / 2$, где C_X – безразмерный коэффициент силы аэродинамического лобового сопротивления (в данной модели является константой и равен 0.29), S_M – геометрическая характеристика

движущегося объекта (площадь «миделевого сечения» – в данной модели равняется 0.04 м^2), ρ – плотность воздуха (для данной модели взято значение плотности воздуха на уровне моря при температуре 15 °C равное 1.225 кг/м^3). Эту формулу можно добавить в новую ИМП, соответствующую объекту «Сопротивление атмосферы», в то время как первая ИМП будет соответствовать объекту «Тело». Вид модели на C-слое представлен на рис. 4.7. Изменение компьютерной модели на других слоях в данном случае не требуется (исключение может составить необходимость реализации ввода добавленных констант через интерфейс). Согласно этой модели дальность полёта тела при тех же начальных данных, но при условии изменяемости величины F_a, составит 2 067 м.

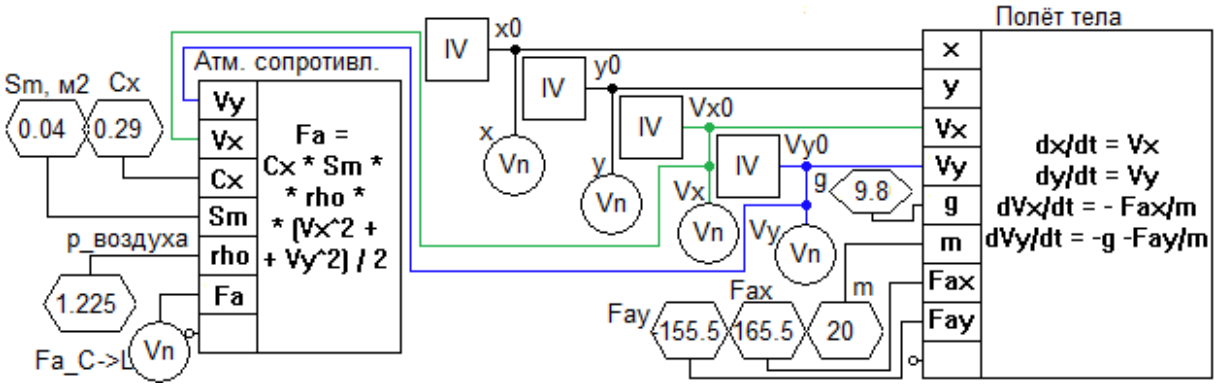


Рисунок 4.7 – Вид модели на C-слое

Усложнив модель ещё: будем учитывать изменение плотности атмосферы в зависимости от слоя стандартной атмосферы: 1) 0 – 11 000 м, 2) 11 000 – 25 000 м, 3) 25 000 – 46 000 м. Будем использовать формулы и соотношения для расчёта плотности воздуха, представленные в [Колесов, Сениченков, 2007, С. 18]. В таком случае эта задача может быть отнесена к классу дискретно-непрерывных (состояние = слой атмосферы), однако продолжим её рассматривать как непрерывную. Для моделирования изменения плотности атмосферы будем использовать компонент атмосфера, описанный в пункте 1.3.3. На рис. 4.8 представим вид модели на L-слое.

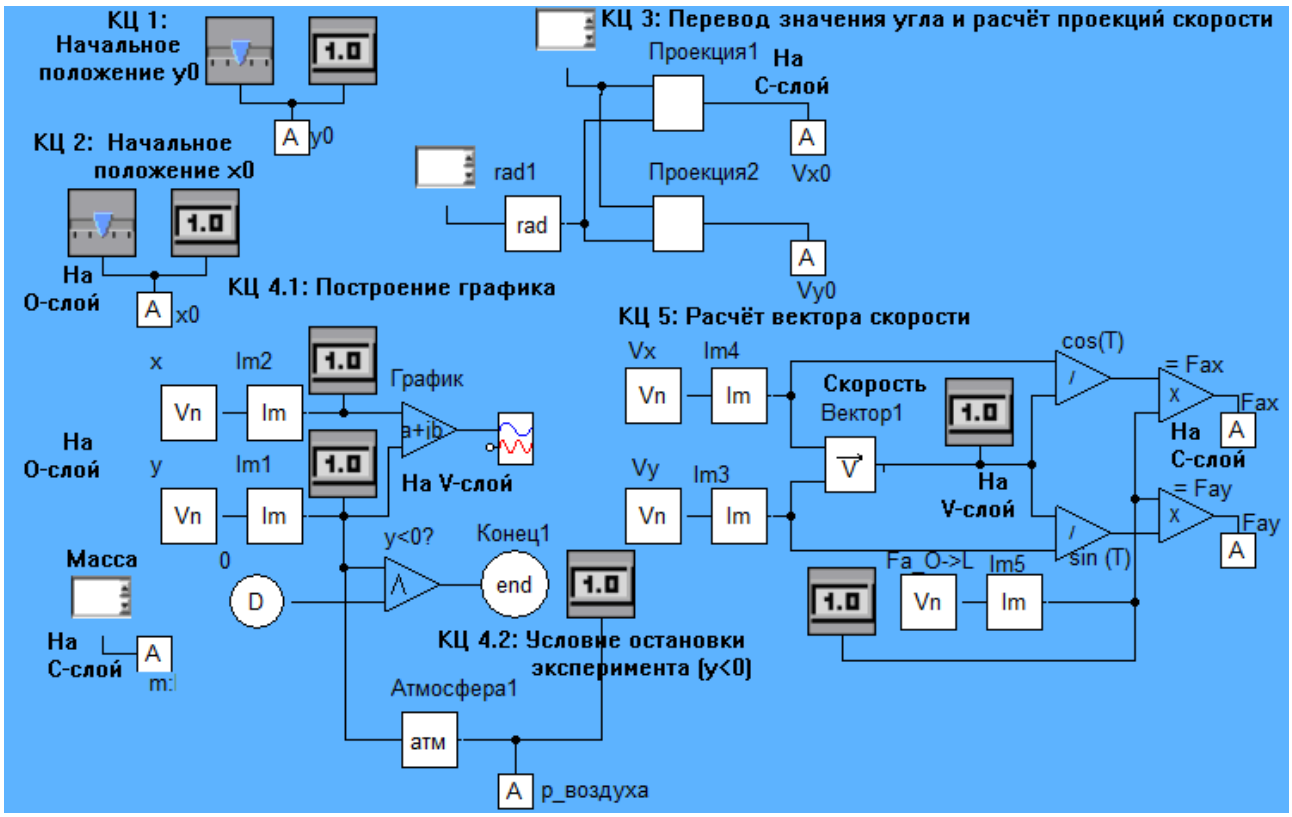


Рисунок 4.8 – Вид модели на L-слое

Ввиду того, что теперь плотность воздуха не является константой, несколько изменились результаты моделирования (рис. 4.9): так, например, дальность полёта составила 2 101 м.

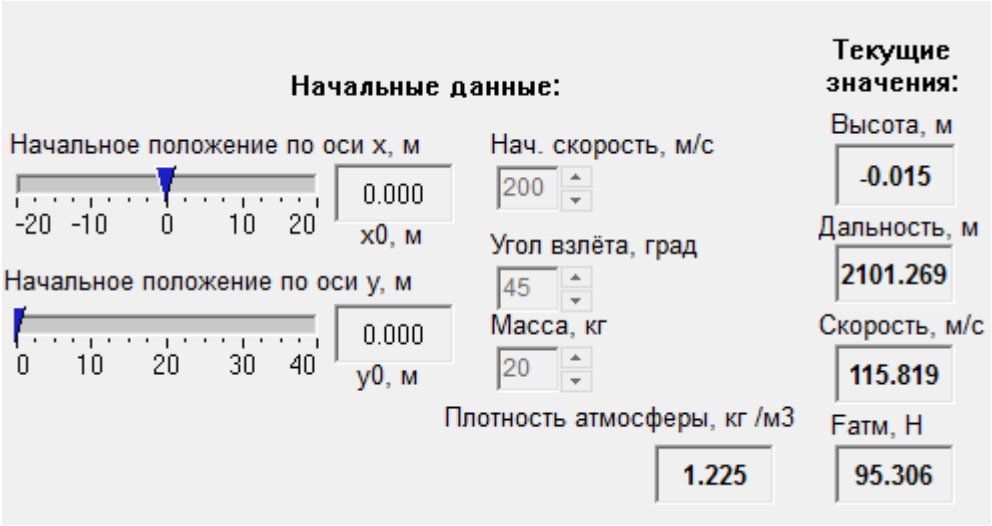


Рисунок 4.9 – Вид модели на V-слое

4.2 МОДЕЛИРОВАНИЕ УСИЛИЙ НА ПОЛИРОВАННОМ ШТОКЕ ШТАНГОВОГО ГЛУБИННОГО НАСОСА

Одним из примеров ФТЗ (как задач о техническом – искусственном – объекте, решаемой с помощью знаний из специализированных разделов физики) является задача моделирования усилия на полированном штоке штангового глубинного насоса (ШГН), структурная схема которого представлена на рис. 4.10 [Вакула, 2006].

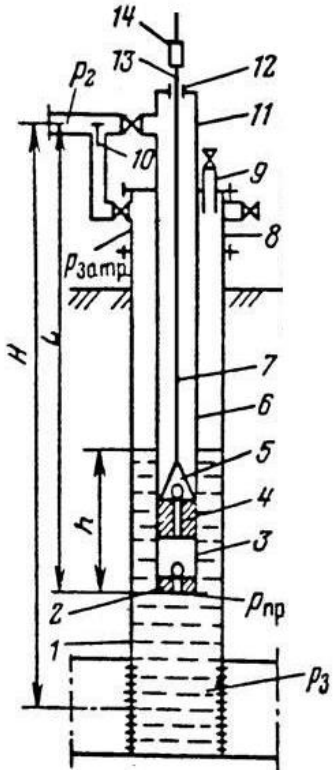


Рисунок 4.10 – Структурная схема штангового глубинного насоса

1 – эксплуатационная колонна; 2 – всасывающий клапан; 3 – цилиндр насоса; 4 – плунжер; 5 – нагнетательный клапан; 6 – насосно-компрессорные трубы; 7 – насосные штанги; 8 – крестовина; 9 – устьевой патрубок; 10 – обратный клапан для перепуска газа; 11 – тройник; 12 – устьевой сальник; 13 – устьевой шток; 14 – канатная подвеска [Рожкин, 2010]

Нефтедобывающие установки, оснащённые ШГН, являются наиболее распространённым видом добывающего оборудования. Несмотря на низкую (в сравнении с электроцентробежными насосами) производительность, данные

установки нашли широкое применение при добыче из низкодебитных скважин и скважин с различными осложнениями в добыче, такими как высокая обводнённость, пескопроявления, высокая вязкость или температура скважинной жидкости, наличие ароматических углеводородов, соле- и парафинообразование, то есть в тех случаях, когда применение центробежных насосов становится неэффективным [Ковшов, 2004]. Для определения оптимальной скорости откачивания скважинной жидкости, оптимального закона движения полированного штока, выявления различных неисправностей ШГН и пр. наиболее простым подходом является создание и исследование его математической модели.

Обычно при моделировании ШГН представляется в виде системы последовательно соединённых элементов: плунжерной пары, колонны штанг, штока [Светлакова, 2008]. Возвратно-поступательное перемещение колонны штанг описывается дифференциальным уравнением продольных колебаний однородного стержня. При этом заданными граничными условиями являются значения перемещений полированного штока и плунжера, а также усилий, приложенных в этих точках [Ивановский, 2013].

Охарактеризуем формальный портрет рассматриваемой задачи. Основным моделируемым процессом является возвратно-поступательное движение колонны штанг, поэтому в качестве основных взаимодействующих объектов выделим полированный шток (источник движения), плунжер (движущееся тело), насосные штанги (растяжимое тело). Будем считать полированный шток источником скорости в моделируемой системе, рассчитывая его перемещение $S(t)$ по некоторому гармоническому закону. Поставим данному объекту в соответствие компонент «Источник скорости», математическая модель которого имеет вид (1):

$$B_1 \cdot V_p = S(t) \quad (1),$$

где B_1 – некоторый коэффициент (константа или функциональный параметр), V_p – потенциальная переменная (в нашем случае – скорость), $S(t)$ – некоторая функциональная зависимость.

Будем рассматривать насосные штанги и плунжер в качестве моделей твёрдых тел, учитывая такие их параметры, как площадь сечения и плотность материала. Математическая модель инерционного звена имеет вид (4.1):

$$A_1 \cdot \frac{dV_p}{dt} + B_2 \cdot V_f = 0 \quad (4.1),$$

где V_f – потоковая переменная (сумма сил F, воздействующих на тело), V_p – потенциальная переменная (скорость тела V), A_1 – параметр модели (масса тела m), B_2 – параметр модели (здесь равен 1), t – время.

Деформация (растяжение и сжатие) штанг во время их движения может быть смоделирована посредством компонентов типа «Пружина», имеющими (как и остальные рассматриваемые нами компоненты) 2 элементарные связи с потенциальными переменными v_1, v_2 (скорости) и потоковой переменной F (сила). Типовая математическая модель компонента имеет вид (4.2):

$$\frac{dF}{dt} = k \cdot (v_1 - v_2) \quad (4.2),$$

в которой коэффициент k может быть как постоянным, так и функциональным параметром. Величину растяжения штанг в таком случае можно рассчитать исходя из разницы значений скоростей v_1 и v_2 на узлах компонента.

Силы трения, воздействующие при совершении движения на шток, штанги и плунжер, могут быть смоделированы с помощью компонента «Демпфер». Математическая модель этого компонента имеет вид (4.3):

$$F = R \cdot (v_1 - v_2) \quad (4.3),$$

где коэффициент R аналогичным образом может представлять собой константу или функциональный параметр.

Давление газожидкостной смеси на плунжер может быть смоделировано с помощью компонента «Источник потоковой переменной», аналогичного «Источник скорости» и имеющему математическую модель (4.4):

$$B_2 \cdot V_f = C(t) \quad (4.4),$$

где B_2 – некоторый коэффициент (константа или функциональный параметр), V_f – потоковая переменная (в нашем случае – сила), $C(t)$ – некоторая функциональная зависимость.

Построим компьютерную модель рассматриваемой задачи из вышеописанных компонентов, указывая на точки приложения сил (объекты их воздействия) путём формирования топологических связей компонентов, опираясь на ранее упомянутые топологические законы.

Многоуровневое моделирование усилий на штоке штангового глубинного насоса в рамках формализма многоуровневого метода компонентных цепей

На *V*-слое модели (рис. 4.11) располагается интерфейс для взаимодействия с компьютерной моделью. В качестве варьируемых параметров выступают: 1) длина хода штока, 2) частота откачивания, 3) динамический уровень жидкости. Остальные параметры модели считаются неизменными, так как зависят от конфигурации оборудования на скважине и задаются непосредственно в свойствах соответствующих компонентов модели. В качестве измеряемых переменных выступают: 1) нагрузка на плунжер, 2) нагрузка на шток, 3) величина потери хода плунжера при сжатии и растяжении штанг, 4) скорость хода плунжера. Все компоненты этого слоя имеют одноимённые отображения на *L*-слое, что позволяет реализовать механизм обмены данными между слоями по замкнутой итерационной схеме *V-L-C-L-V*.

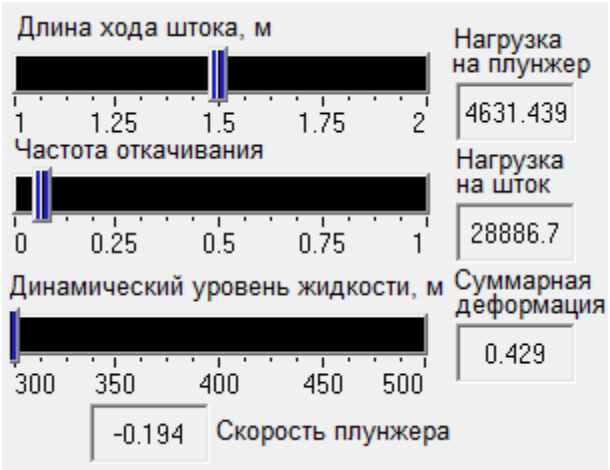


Рисунок 4.11 – Модель ШГН на V-слое

На *L*-слое модели (рис. 4.12) располагается:

1. Диаграмма состояний (КЦ «Начало» – «Вниз (шток)» – «Вверх (шток)»), характеризующая направление движения плунжера (вниз или вверх), что необходимо для расчёта величины деформации штанг и силы трения по отличающимся выражениям при движении вниз и вверх, а также для реализации эффекта «запаздывания» плунжера при движении «за» штангами. Диаграммы состояний представляют собой концептуальную модель объекта в виде конечного (гибридного) автомата, характеризующую поведение объекта в форме последовательности его состояний, сменяющих друг друга при выполнении определённых условий. В данной задаче диаграммы состояний дополняют непрерывное поведение системы, описываемое на *C*-слое, дискретным поведением, тем самым формируя гибридное поведение [Сениченков, 2004]. Схема работы диаграмм состояний в СМ МАРС следующая: при получении значения *true* на вход *Start* компонент начинает передавать значения, подаваемые на вход *in* через выход *out*, и продолжает до тех пор, пока не станет истинным вложенное в него условие (математическое выражение), после чего прекращает свою работу, передав значение *true* на выход *End*.

2. КЦ, осуществляющая параметрирование основных компонентов модели – «Плунжер», «Штанги», «Трение», «Деформация», «Давление смеси», которые имеют одноимённые отображения на *C*-слое. Значения переменных на этом слое передаются последовательно от компонента к компоненту (пройдя при необходимости заложенные в компоненты преобразования). После того как завершается инициализация всех переменных (передача значений на *C*-слой), начинается вычислительный эксперимент на *C*-слое (решение систем алгебро-дифференциальных уравнений вычислительным ядром), результаты которого на каждой итерации работы модели возвращаются обратно на *L*-слой и при необходимости отображаются пользователю на *V*-слое через компоненты-приёмники (графики и компоненты вида «Цифровое табло»).

Значения параметров модели, принятых за константу, (например, площадь сечения штанг, плотность материала штанг и пр.) задаются на данном слое через свойства соответствующих компонентов.

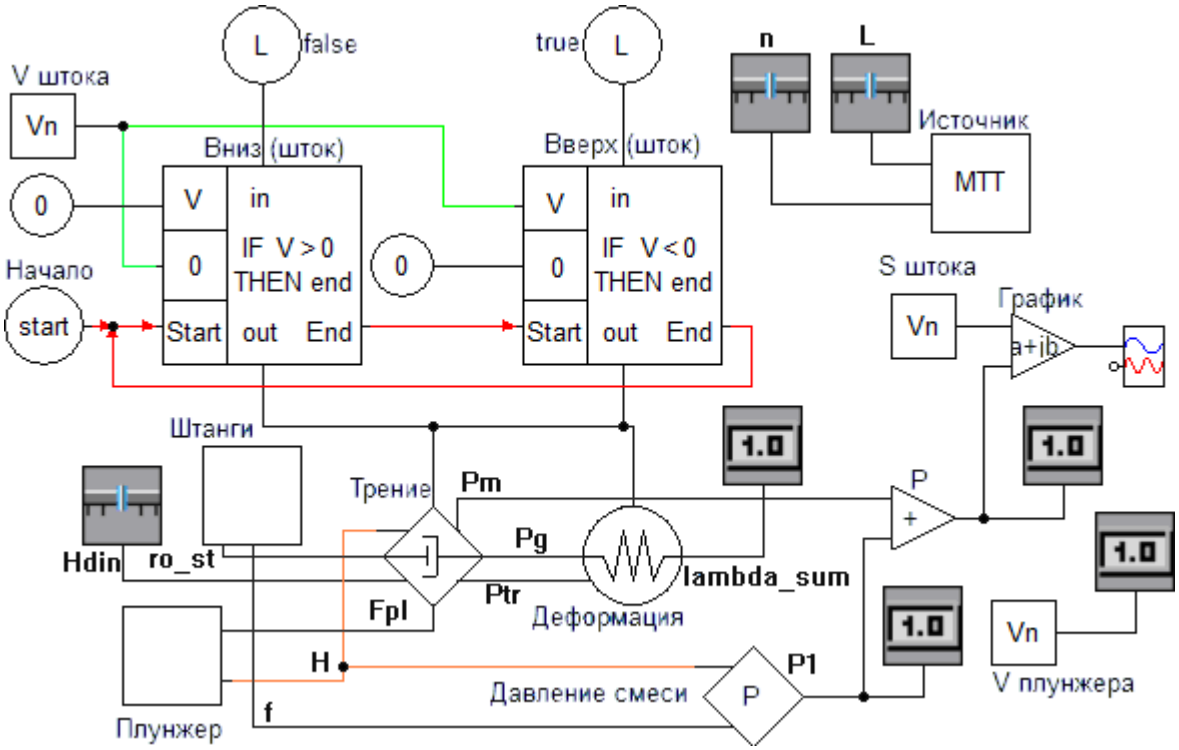


Рисунок 4.12 – Модель ШГН на L-слое

На *C*-слое (рис. Рисунок 4.13) располагается модель непрерывного поведения моделируемой системы (визуально соответствующая виду моделируемого объекта «Шток – Штанги – Плунжер») в виде КЦ, состоящей из следующих компонентов:

1) «Плунжер», «Штанги» – компоненты, представляющие собой модели инерционных звеньев (тел), осуществляющие возвратно-поступательное движение;

2) «Источник» – компонент, представляющий собой модель штока (источника скорости);

3) «Трение» – компонент, содержащий систему уравнений для расчёта величины силы трения плунжера при движении вниз и вверх;

4) «Деформация» (упругость) – компонент для расчёта деформации штанг при ходе вниз и вверх;

5) «Нагрузка» – компонент для расчёта нагрузки газожидкостной смеси на плунжер;

6) измерители потенциальных переменных *Vn*, передающие значения на одноимённые отображения на *L*-слое;

7) интеграторы *in*, используемые для расчёта положения штока и плунжера.

Каждый компонент (за исключением измерителей и интеграторов) содержит в себе систему алгебро-дифференциальных уравнений, которая решается на каждой итерации. Рассчитанные скорости плунжера и штока передаются на *L*-слой через компоненты-измерители, значения других переменных передаются на выводы соответствующих компонентов по ходу решения. Топологические связи компонентов определяют порядок их опроса и, как следствие, порядок решения системы уравнений (которая решается по завершению опроса).

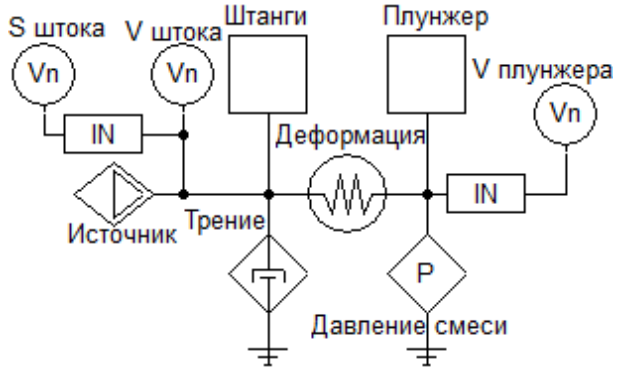


Рисунок 4.13 – Модель ШГН на С-слое

Использование компонентов-интеграторов на этом *C*–слое позволяет перейти от базисных переменных – силы *F* и скорости *v* – к работе силы и перемещению. Аналогичным образом могут использоваться компоненты-дифференциаторы. Для измерения значений потоковых переменных используются соответствующие измерители V_b. На этом слое также возможно моделирование движения многофазных потоков (например, нефть/газ) через компоненты типа «насос», «труба» и пр. за счёт использования неоднородных

векторных связей [Дмитриев, Ганджа, Важенин, 2014], содержащих несколько пар дуальных переменных (потенциальную и потоковую).

Результаты работы построенной модели представлены на рис. 4.14.

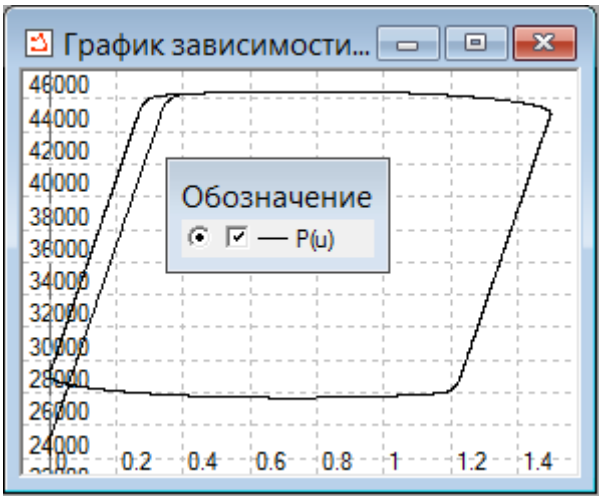


Рисунок 4.14 – График зависимости нагрузки (Н) в точке подвеса штанг от положения (м) этой точки

4.3 МОДЕЛИРОВАНИЕ ОТСКОКОВ УПРУГОГО ТЕЛА ОТ ПОВЕРХНОСТИ СО СЛОЖНЫМ ПРОФИЛЕМ

Данная задача является модификацией задачи, рассмотренной в Приложении А. Отличие заключаются в том, что твёрдое тело (мяч) совершает соударения не с прямой поверхностью (y=0), а со сложной поверхностью, задаваемой табличной функцией. Для осуществления такой модификации необходимо заменить компонент-источник константы (y=0) на КЦ «Фигура1 – Поверхность» (рис. 4.15).

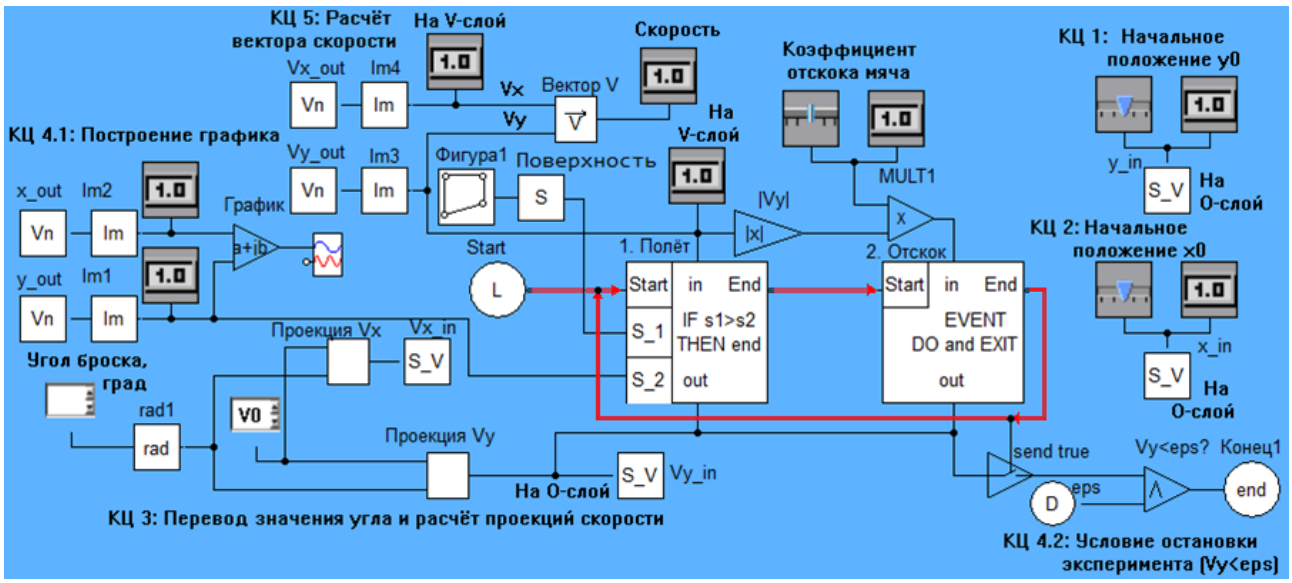


Рисунок 4.15 – КЦ изменённой модели на L-слое

Компонент «Фигура1» представляет собой геометрический компонент типа «Многоугольник по точкам» (см. описание компонента в п. 2.3.3) и задаёт массив точек (табличную функцию), а «Поверхность» представляет собой компонент типа «Кривая (по точкам)» (описание там же в п. 2.3.3) и интерполирует её значение в точке *x* – координате положения тела, получаемой посредством одноимённого отображения-измерителя компонента на C-слое (рис. 4.16). Рассчитанное компонентом «Поверхность» значение передаётся на вход S_1 компонента «1. Полёт», таким образом, граничное условие для срабатывания перехода (S_1 > S_2, где S_2 – положение тела по оси OY) пересчитывается на каждой итерации работы модели, тем самым реализуя моделирование отскакивания тела не от ровной поверхности, а от сложной – заданной табличной функцией.

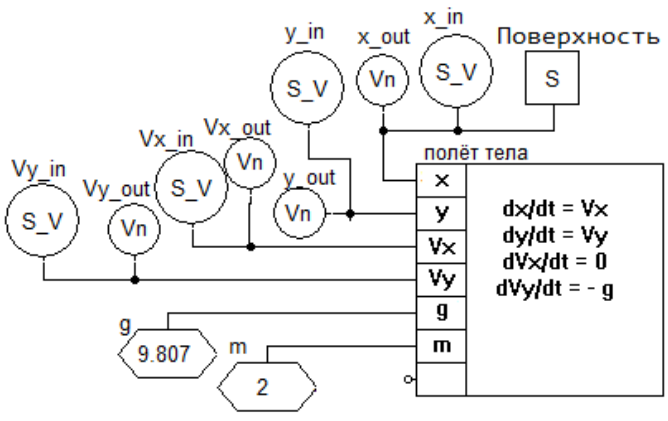


Рисунок 4.16 – КЦ изменённой модели на C-слое

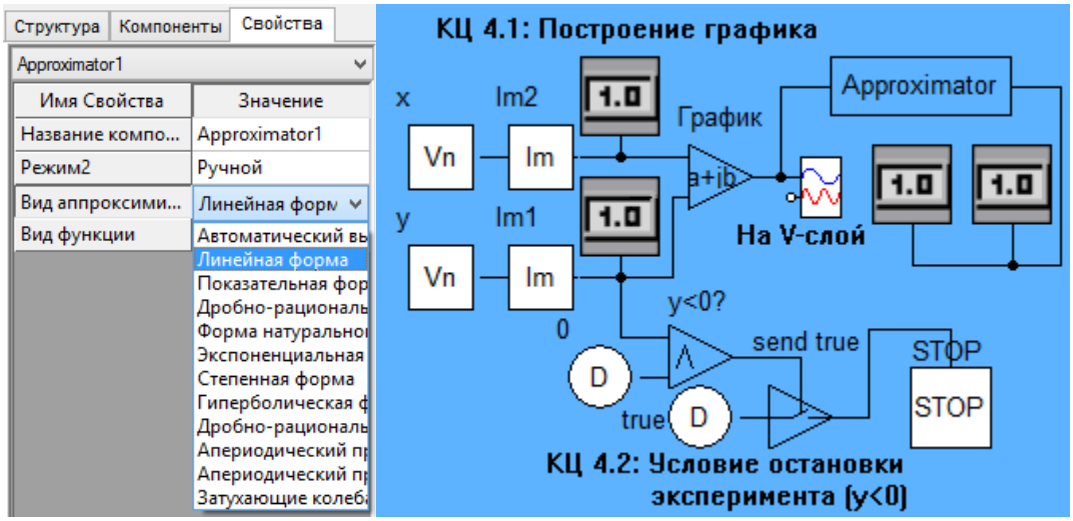
Изменения на С-слое заключаются во включении в КЦ отображения компонента «Поверхность» для измерения рассчитываемых значений координаты *x* движения тела. Результаты моделирования (траектория движения тела) представлены на рис. 4.17.

Рисунок 4.17 – Траектория отскока тела от ступеней

Отметим, что данная задача является (наряду с задачей моделирования усилий на полированном штоке) примером, иллюстрирующим эффект накопления амплитудно-временной погрешности (см. пункт 2.2.3) ввиду того, что предполагает достаточно большое количество переходов из одного дискретного состояния в другое.

4.4 АППРОКСИМАЦИЯ РЕЗУЛЬТАТОВ МОДЕЛИРОВАНИЯ

Рассмотрим модель полёта тела в атмосфере Земли, рассмотренную в параграфе 4.1 данной работы. Допустим, что перед исследователем стоит задачи аппроксимации траектории полёта тела, полученной в результате моделирования данной ФТЗ. В таком случае возможно использовать имеющийся в СМ МАРС компонент для аппроксимации табличных данных методом наименьших квадратов (МНК), однако набор приближающих функций при этом будет ограничен (см. рис. 4.18).



а б

Рисунок 4.18 – Аппроксимация в СМ МАРС по МНК

а – параметры компоненты, б – вид КЦ с компонентом

Такое ограничение не позволяет использовать в качестве приближающей функции те, которые на заложены в программном коде компонента. Для преодоления указанного ограничения в компоненте-аппроксиматоре была реализована возможность аппроксимации табличной функции произвольной (введённой пользователем в ИМАП – интерактивной математико-алгоритмической панели – на L-слое) приближающей функцией с применением численного метода, предложенного в п. 2.3.2 данной работы. На рис. 4.19 представлена изменённая КЦ задачи о полёте снаряда, реализующая аппроксимацию пользовательской функцией.

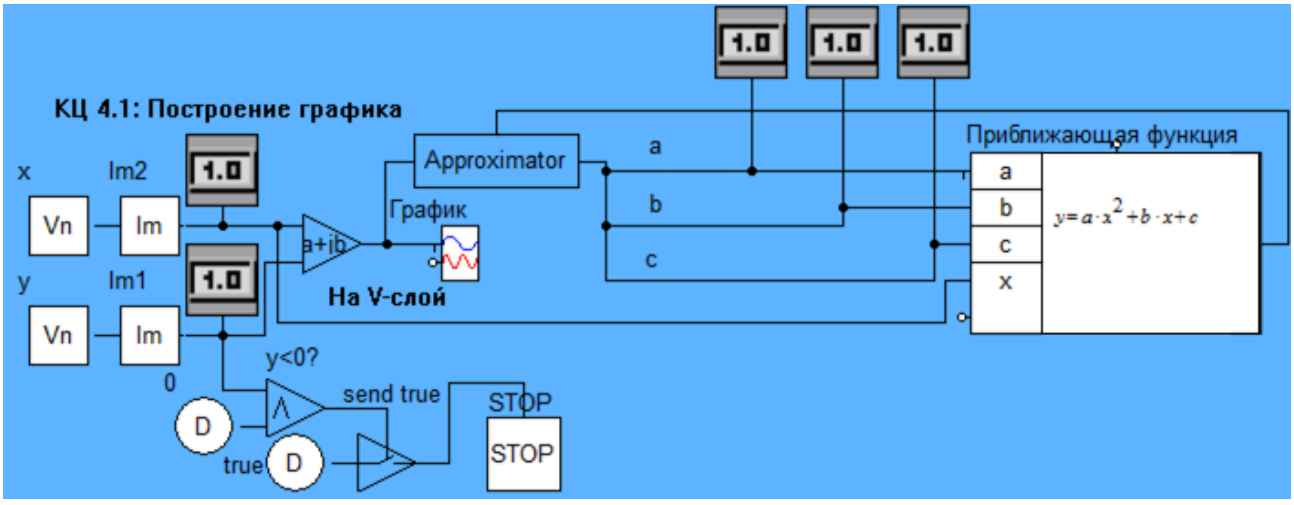


Рисунок 4.19 – Изменённая КЦ для аппроксимации данных

Содержимое ИМАП «Приближающая функция» представляет собой приближающую функцию в аналитическом (символьном) виде введённую в окно ИМАП, основанное на Макрокалькуляторе [Ганджа, 2005] и отображено на рис. 4.20.

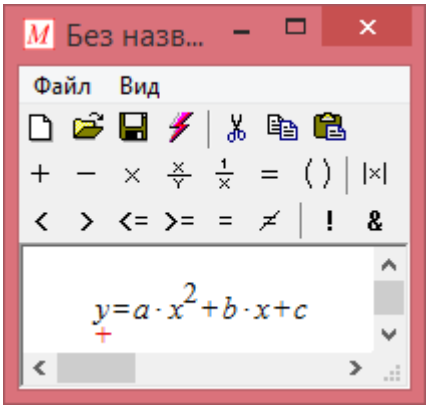


Рисунок 4.20 – Содержимое ИМАП

На V-слое добавлены 3 цифровых табло для каждого из параметров (коэффициентов) приближающей функции (рис. 4.21). Количество допустимых параметров функции не ограничено ввиду того, что ИМАП имеет динамическое число входных связей (узлов), а компонент-аппроксиматор принимает на свой верхний вход только итоговое рассчитанное значение функции, на основании которого осуществляет поиск коэффициентов. Необходимо ответить, что компонент-аппроксиматор имеет 2 режима работы: 1) по завершению расчёта модели, 2) на каждой итерации – что позволяет аппроксимировать табличные результаты моделирования непосредственно во время работы модели.

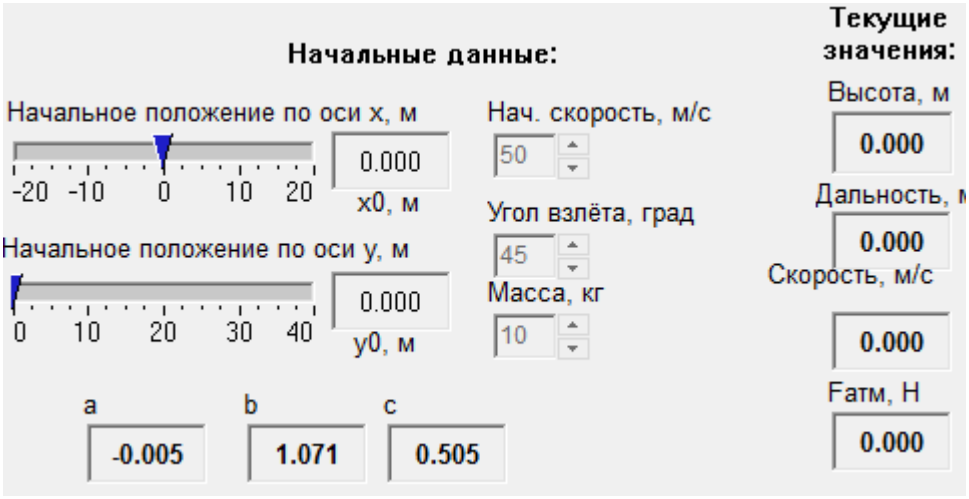


Рисунок 4.21 – Вид модели на V-слое

Как видно на рис. 4.22, произведённые изменения не затронули КЦ С-слоя, что подтверждает аддитивный характер вносимых в модель изменений.

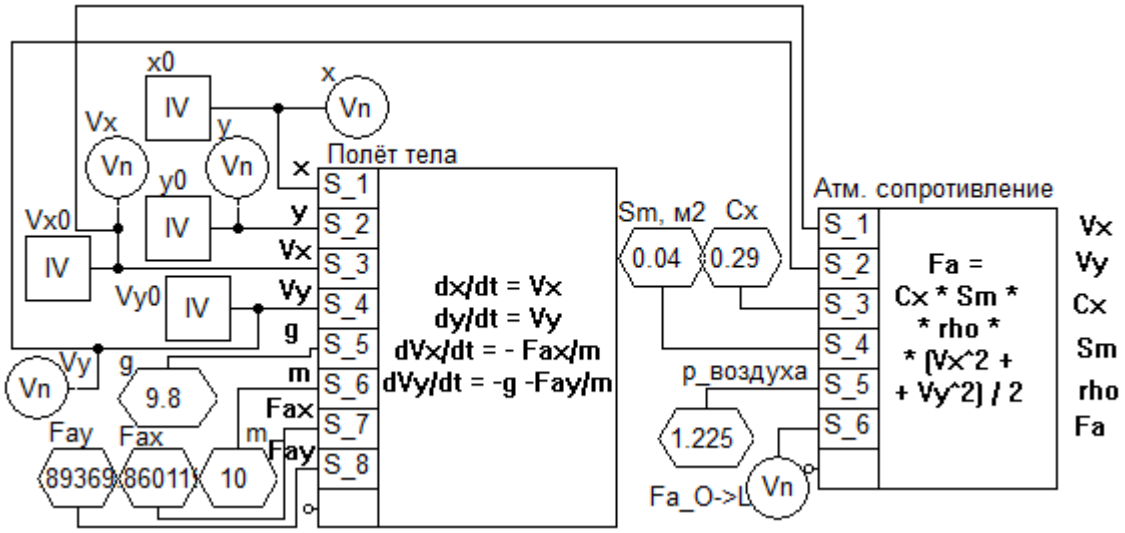


Рисунок 4.22 – Вид модели на С-слое

ВЫВОДЫ ПО ГЛАВЕ 4

1. Разработанные компоненты позволяют строить в СМ МАРС многоуровневые компьютерные модели ФТЗ для статических и динамических режимов, собирая модель из готовых блоков более высокой степени абстракции.

2. Построенные с применением разработанного комплекса программ учебно-иллюстративные модели использованы в образовательном процессе в ФГБОУ ВО «Томский государственный университет систем управления и радиоэлектроники» при преподавании дисциплин «Компьютерное моделирование физических задач», «Основы компьютерного моделирования физико-технических задач», «Моделирование систем» в качестве тренажёров на лабораторных и практических занятиях, так и в качестве иллюстративного материала на занятиях других видов, в том числе как при аудиторной, так и самостоятельной работе.

3. Проведённая интерпретация диаграмм состояний нашла отражение в компонентах «Начало», «Состояние», «Событие, «Конец», которые образуют КЦ на L-слое многоуровневой компьютерной модели, позволяющие управлять не только непрерывными моделями на С-слое, но и образовывать дискретно-событийные сценарии с другими конструкциями L- слоя.

4. Построенная модель штангового глубинного насоса была использована в АО «Энергонефтемаш» для определения оптимальных эксплуатационных характеристик нефтедобывающей установки и построения алгоритма работы системы управления штанговым глубинным насосом.

ЗАКЛЮЧЕНИЕ

Выполненная диссертационная работа направлена на решение научно-технической задачи создания методики, алгоритма, эффективных и адекватных инструментальных средств компьютерного моделирования ФТЗ, позволяющих строить их многоуровневые (с отделением модели непрерывного поведения объекта от алгоритма его дискретного поведения) модели задач из блоков высокого уровня абстракции с понижением погрешности моделей. Также в данной работе закладываются основы исследования возможностей автоматизированной формализации словесного портрета ФТЗ, конечной целью которого является автоматизированное построение (с частичным участием пользователя) компьютерных моделей процессов в задачах физики и техники.

В результате проведённого исследования получены следующие результаты:

1. Описан и классифицирован класс ФТЗ как объекта моделирования.

2. Разработан алгоритм многоуровневого компьютерного моделирования ФТЗ, включающий этапы формализации моделируемой задачи и её многоуровневой декомпозиции на объектный, логический (алгоритмический) и визуальный уровни с применением методики многоаспектного анализа.

3. Разработаны средства моделирования физических свойств объектов, геометрических свойств их связей в ФТЗ, а также инструменты для моделирования дискретно-непрерывного (гибридного) поведения объектов с реализацией механизма компенсации накапливаемой амплитудно-временной погрешности.

4. Разработанный численный метод аппроксимации на базе поисковых методов оптимизации позволяет аппроксимировать табличные результаты моделирования ФТЗ приближающими функциями произвольного вида, обеспечивая выбор глобального минимума отклонения значений приближающей функции от аппроксимируемой из ряда локальных минимумов.

5. Разработан комплекс программ, включающий в себя библиотеки моделей компонентов для многоуровневого моделирования ФТЗ на базе СМ

МАРС, программные модули для самостоятельного обучения студентов моделированию ФТЗ и организации взаимодействия студентов с виртуальной лабораторией моделирования ФТЗ во время проведения аудиторных лабораторных работ.

6. Разработанный алгоритм формализации словесного портрета ФТЗ, используемый в программном модуле обучения моделированию ФТЗ, позволяет проводить автоматический анализ текстовых условий задач с целью их перевода на формальный язык (в соответствии с формализмом ММКЦ) для иллюстрации пользователю процедуры анализа и многоуровневой декомпозиции задачи, что позволяет дополнять аудиторную работу студента по дисциплинам, связанным с моделированием, автономной самостоятельной работой, направленной на закрепление навыков анализа задач.

7. Построенная многоуровневая компьютерная модель штангового глубинного насоса была использована в АО «Энергонефтемаш» для определения оптимальных эксплуатационных характеристик нефтедобывающей установки и построения алгоритма работы системы управления штанговым глубинным насосом, другие модели использованы в образовательном процессе ФГБОУ ВО «Томский государственный университет систем управления и радиоэлектроники».

СПИСОК ЛИТЕРАТУРЫ

1. Аладьев В.З, Богдявичюс М.А., Хунт. Ю. Решение физико-технических и математических задач с Maple V. – Таллинн; Вильнюс: Salcombe, 1999. – 660 с.

2. Андерсон Д.А. Дискретная математика и комбинаторика.: Пер. с англ. – М.: Издательский дом "Вильяме", 2004. – 960 с.

3. Анфилатов В.С. и др. Системный анализ в управлении: Учеб. пособие / В.С. Анфилатов, А.А. Емельянов, А.А. Кукушкин; Под ред. А.А. Емельянова. – М.: Финансы и статистика, 2002. – 368 с.

4. Балханов В.К. Моделирование геометрических и электрических характеристик физико-технических сред фрактальным методом: дис. … канд. техн. н. – Иркустк, 2007. – 112 с.

5. Боголюбова И.А., Скроботова Т.В., Федоров О.Л. Развитие технического мышления студентов посредством решения технических задач [Электронный ресурс] // Известия Волгоградского государственного педагогического университета. – 2007. – URL: https://cyberleninka.ru/article/n/razvitie-tehnicheskogo-myshleniya-studentov-posredstvom-resheniya-tehnicheskih-zadach (дата обращения: 16.06.2019).

6. Бояршинова А.К., Фишер А.С. Теория инженерного эксперимента. – Челябинск: Изд-во ЮУрГУ, 2006. – 85 с.

7. Буш Г.Я. Рождение изобретательских идей. – Рига: Издательство «Лиесма», 1976. – 127 с.

8. Вакула Я.В. Нефтегазовые технологии. – Альметьевск: Альметьевский государственный нефтяной институт, 2006. – 168 с.

9. Ганджа Т.В. Комплекс программ автоматизации вычислительного эксперимента в расчетно-моделирующей среде МАРС: дис. … канд. техн. н. – Томск, 2005. – 178 с.

10. Ганджа Т.В. Развитие метода компонентных цепей для реализации комплекса программ моделирования химико-технологических систем: дис. … д-ра техн. н. – Томск: 2017. – 457 с.

11. Глушков В.М. Абстрактная теория автоматов. – УМН, 1961. – Т. 16. – Вып. 5(101). – С. 3-62.

12. Голдовский Б.И., Вайнерман М.И. Комплексный метод поиска решений технических проблем. – М.: Метод, 1990. – 112 с.

13. Голованчиков А.Б., Минь К.Д., Шибитова Н.В. Аппроксимация экспериментальных данных методом наименьших квадратов и методом наименьших относительных квадратов // Энерго- и ресурсосбережение: промышленность и транспорт. 2019. – № 1 (26). – С. 42-44.

14. Десятов А.Д., Сирота А.А. Имитационное моделирование сложных динамических систем в интегрированной инструментальной среде Matlab + Simulink+Stateflow // Вестник Воронежского государственного университета. Серия: Системный анализ и информационные технологии. – 2006. – № 2. – С. 62-69.

15. Дмитриев В. М., Ганджа Т. В., Панов С. А. Система виртуальных инструментов и приборов для автоматизации учебных и научных экспериментов // Программные продукты и системы: в 4 ч. – Тверь: «Центрпрограммсистем», 2016. – Ч. 3. – С. 154-162.

16. Дмитриев В.М. Методика применения учебно-иллюстративных модулей в интерактивном учебнике / В.М. Дмитриев, А.В. Шутенков, А.В. Сторчак // Современное образование: актуальные проблемы профессиональной подготовки и партнерства с работодателем: материалы междунар. науч.-метод. конф., 30-31 января 2014 г., Россия, Томск. – Томск : Изд-во Томск. гос. ун-та систем упр. и радиоэлектроники, 2014. – 310 с. С. 81-82.

17. Дмитриев В.М. Принципы построения моделей сложных технологических объектов с неоднородными векторными связями / В.М. Дмитриев, Т.В. Ганджа, С.К. Важенин // Современные технологии. Системный анализ. Моделирование. – 2014. – № 1 (41). – С. 104–111.

18. Дмитриев В.М., Ганджа Т.В. Метод и язык моделирования интеллектуальных систем управления сложными технологическими объектами // Объектные системы. – 2015. – № 10. – С. 44-50.

19. Дмитриев В.М., Ганджа Т.В. Построение и исследование активных компонентов в системах многоуровневого моделирования // Информатика и системы управления. – 2016. – № 3(49). – С. 25–35.

20. Дмитриев В.М., Ганджа Т.В. Расчетно-моделирующая среда для учебных и научных лабораторий // Вестник Московского городского педагогического университета. Серия: Информатика и информатизация образования. – 2004. – № 3. – С. 43-48.

21. Дмитриев В.М., Ганджа Т.В. Формирование учебно-иллюстративных модулей в среде многоуровневого компьютерного моделирования МАРС // Электронные средства и системы управления. – 2013. – № 2. – С. 96-100.

22. Дмитриев В.М., Ганджа Т.В., Кочергин М.И. Многоуровневое моделирование задач физики // Современное образование: практико-ориентированные технологии подготовки инженерных кадров Материалы международной научно-методической конференции. – 2015. – С. 47–49.

23. Дмитриев В.М., Ганджа Т.В., Панов С.А. Формирование системы автоматизированного документирования методом компонентных цепей // Информатика и системы управления. – 2014. – № 3 (41). – С. 12-22.

24. Дмитриев В.М., Ганджа Т.В., Шутенков А.В. Построение компьютерных моделей многофракционных физико-химических систем газопромысловых объектов в формате метода компонентных цепей // Доклады Томского государственного университета систем управления и радиоэлектроники. – 2012. – № 2 (26), ч. 1. – С. 145-150.

25. Дмитриев В.М., Кочергин М.И. Интерпретация гибридных моделей в многоуровневую компьютерную модель // Наука и практика: проектная деятельность – от идеи до внедрения: материалы региональной науч.-прак. конф., Томск, 2015. – Томск: Томск. гос. ун-т систем упр. и радиоэлектроники, 2015. – С. 331-334.

26. Дмитриев В.М., Филиппов А.Ю., Шарова О.Н. Метод многоаспектного анализа как алгоритм формализации задач по физике // Вестник

Московского городского педагогического университета. Серия: Информатика и информатизация образования. – 2005. – № 4. – С. 60-66.

27. Дмитриев В.М., Филиппов А.Ю., Шарова О.Н. Формализованное представление задач для компьютерного моделирования // Вестник Московского городского педагогического университета. Серия: Информатика и информатизация образования. – 2004. – № 3. – С. 53-59.

28. Дмитриев В.М., Шутенков А.В., Зайченко Т.Н., Ганджа Т.В. МАРС – среда моделирования технических устройств и систем. – Томск: В-Спектр, 2011. – 277 с.

29. Дьяконов В.П. VisSim+Mathcad+MATLAB. Визуальное математическое моделирование. – М.: Солон-Пресс, 2004. – 384 с.

30. Жуков Ю.А., Горбунов А.В., Лычагин Ю.В. Создание имитационной модели динамики гексапода в интеграции SolidWorks и Matlab // Автоматизированное проектирование в машиностроении. – 2017. – № 5. – С. 98-103.

31. Закревский А.Д., Поттосин Ю.В., Черемисинова Л.Д. Логические основы проектирования дискретных устройств. – М.: ФИЗМАТЛИТ, 2007. – 592 с.

32. Ивановский В. Н. Повышение интереса к штанговым насосным установкам – в чем причина? //Территория Нефтегаз. – 2013. – № 8. – С. 48–49.

33. Имашев Г.И. Решение задач с политехническим содержанием // Сб. материалов междунар. научно-практ. конф. «Наука и технологии: шаг в будущее – 2007» [Электронный ресурс]. – http://www.rusnauka.com/5._NTSB_2007/Pedagogica/19818.doc.htm

34. Инихов Д.Б., Колесов Ю.Б., Сениченков Ю.Б. Пакеты моделирования в образовании: современная ситуация и нерешенные проблемы // Компьютерные инструменты в образовании. – 2012. – № 6. – С. 44-55.

35. Клишкова Н.В. Подготовка студентов к решению физико-технических проблем в исследовательском обучении физике // Известия

Российского государственного университета им. А.И. Герцена. – 2011. – № 139. – С. 152-156.

36. Клишкова Н.В. Формирование у студентов умений решения физико-технических проблем в процессе обучения физике: на примере проблематики полупроводниковой оптической и квантовой электроники: дис. … канд. пед. наук. – Санкт-Петербург, 2011. – 185 с.

37. Ковшов В.Д. Моделирование динамограммы станка-качалки. Нормальная работа насоса / В.Д. Ковшов, М.Е. Сидоров, С.В. Светлакова // Нефтегазовое дело. – 2004. – Т. 2. – С. 75–81.

38. Колесов Ю.Б. Моделирование систем. Практикум по компьютерному моделированию / Ю. Б. Колесов, Ю. Б. Сениченков. – СПб.: БХВ-Петербург, 2007. – 352 с.

39. Колесов Ю.Б. Объемно-ориентированное моделирование сложных динамических систем. - СПб.; Изд-во СПбГПУ, 2004. – 239 с.

40. Колесов Ю.Б. Развитие метода объектно-ориентированного анализа для задач проектирования гибридных систем управления: дис. ... д-ра техн. н. – Санкт-Петербург, 2003. – 252 с.

41. Колесов Ю.Б., Сениченков Ю.Б. Моделирование систем. Динамические и гибридные системы. - СПб.: БХВ-Петербург, 2012. – 224 с.

42. Колесов Ю.Б., Сениченков Ю.Б. Объектно-ориентированное моделирование в среде Rand Model Designer 7: учебно-практическое пособие. – М.: Проспект, 2016. – 256 с.

43. Королев А.Л. Компьютерное моделирование в образовании // Problems of modern pedagogics in the context of international educational standards development. Materials digest of the XL International Research and Practice Conference and I stage of the Championship in Pedagogical sciences. (London, January 31 – February 05, 2013). – London: IASHE, 2013. – С. 126–128.

44. Кочергин М.И. Автоматический анализ текстов задач по физике для сопровождения процесса их решения в среде компьютерного моделирования задач // Материалы Международного молодежного научного форума

«ЛОМОНОСОВ–2014» / Отв. ред. А.И. Андреев, А.В. Андриянов, Е.А. Антипов, М.В. Чистякова. [Электронный ресурс] – М.: МАКС Пресс, 2014. – 1 электрон. опт. диск (DVD-ROM). – Режим доступа: http://lomonosov-msu.ru/archive/Lomonosov_2014/2609/2200_74450_c2ef12.pdf (дата обращения: 10.03.2015).

45. Кочергин М.И. Алгоритм решения задачи аппроксимации через задачу многомерной оптимизации // Сборник избранных статей научной сессии ТУСУР, Томск, 16–18 мая 2018 г.: в 3 частях. – Томск: В-Спектр, 2018 – Ч. 3. – С. 92-95.

46. Кочергин М.И. Компьютерное моделирование полёта тела в атмосфере для образовательных целей // Новые информационные технологии и системы: сб. науч. ст. XIV Междунар. науч.-техн. конф., посвящ. 70-летию кафедры «Вычислительная техника» и 30-летию кафедры «Системы автоматизированного проектирования» (г. Пенза, 22–24 ноября 2017 г.). – Пенза: Изд-во ПГУ, 2017. – С. 400-404.

47. Кочергин М.И. Моделирование логико-лингвистических конструкций для формализации текста задачи по физике // Современное образование: актуальные проблемы профессиональной подготовки и партнерства с работодателем: материалы междунар. науч.-метод. конф., 30–31 января 2014 г., Россия, Томск. – Томск: Изд-во Томск. гос. ун-та систем упр. и радиоэлектроники, 2014. – С. 82–83.

48. Кочергин М.И. Моделирование экземпляров предметной области посредством формализации их текстовых описаний // Перспективы развития фундаментальных наук: труды XIII Международной конференции студентов и молодых учёных. Россия, Томск, 26–29 апреля 2016 г. – Томск: Изд-во – Национальный Исследовательский Томский политехнический университет, 2016. – Том 7. IT-технологии и электроника. – С. 81–83.

49. Кочергин М.И. Обзор инструментов для компьютерного моделирования физических процессов // Научная сессия ТУСУР–2017: материалы Международной научно-технической конференции студентов,

аспирантов и молодых ученых, посвященной 55-летию ТУСУРа, Томск, 10–12 мая 2017 г.: в 8 частях. – Томск: В-Спектр, 2017 – Ч. 4. – С. 96-99.

50. Кочергин М.И. Обучение многоуровневому компьютерному моделированию физических задач: статические и динамические модели // Современное образование: повышение профессиональной компетентности преподавателей вуза – гарантия обеспечения качества образования: материалы междунар. науч.-метод. конф., 1–2 февраля 2018 г., Россия, Томск. – Томск: Изд-во Томск. гос. ун-та систем упр. и радиоэлектроники, 2018. – С. 167-169.

51. Кочергин М.И. Обучение решению задач методом моделирования // Современное образование: практико-ориентированные технологии подготовки инженерных кадров: материалы междунар. науч.-метод. конф., 29–30 января 2015 г., Россия, Томск. – Томск: Изд-во Томск. гос. ун-та систем упр. и радиоэлектроники, 2015. – С. 46–47.

52. Кочергин М.И. Обучение решению задач по физике в среде компьютерного моделирования задач при использовании системы автоматического анализа текстов // Научная сессия ТУСУР–2014: Материалы Всероссийской научно-технической конференции студентов, аспирантов и молодых учёных, Томск, 14–16 мая 2014 г. – Томск: В-Спектр, 2014: В 5 частях. – Ч. 2. – С. 247–249.

53. Кочергин М.И. Отображение геометрических свойств объектов в многоуровневых компьютерных моделях // Перспективы развития фундаментальных наук: сб. трудов XVI Междунар. конф. студентов, аспирантов и молодых ученых (Томск, 23-26 апреля 2019 г.), в 7 томах. Том 7. IT-технологии и электроника. – Томск: Изд-во Томского политехнического университета, 2019. – С. 89-91.

54. Кочергин М.И. Построение учебно-иллюстративной модели динамической системы (на примере физического маятника) // Современное образование: развитие технологий и содержания высшего профессионального образования как условие повышения качества подготовки выпускников:

материалы междунар. науч.-метод. конф., 26-27 января 2017 г., Россия, Томск. – Изд-во Томск. гос. ун-та систем упр. и радиоэлектроники, 2017. – С. 112-114.

55. Кочергин М.И. Применение интерактивных математических панелей для моделирования физических задач в рамках среды многоуровневого моделирования // Моделирование. Фундаментальные исследования, теория, методы и средства: материалы 17-ой Междунар. науч.-практ. конф., г. Новочеркасск, 26-27 сент. 2017г. / Южно-Российский государственный политехнический университет (НПИ) имени М.И. Платова. – Новочеркасск: Лик, 2017. – С. 54-59.

56. Кочергин М.И. Применение логической модели знаний для классификации физического поведения объектов // Перспективы развития фундаментальных наук: сборник трудов XIV Международной конференции студентов, аспирантов и молодых ученых. Россия, Томск, 25–28 апреля 2017 г. – Томск: Изд-во – Национальный Исследовательский Томский политехнический университет, 2017. – С. 63–65.

57. Кочергин М.И. Применение объектно-ориентированного подхода при компьютерном моделировании задач физики // Материалы 55-й Международной научной студенческой конференции МНСК-2017: Информационные технологии / Новосиб. гос. ун-т. – Новосибирск: ИПЦ НГУ, 2017. – С. 69.

58. Кочергин М.И. Система интеллектуальной поддержки процесса решения задач по физике методом моделирования // Материалы Всероссийской научно-технической конференции студентов, аспирантов и молодых учёных «Научная сессия ТУСУР–2015»: Томск, 13–15 мая 2015 г. – Томск: В-Спектр, 2015: В 5 частях. – Ч. 5. – С. 309–312.

59. Кочергин М.И. Система обучения компьютерному моделированию задач по физике // Научная сессия ТУСУР–2016: материалы Международной научно-технической конференции студентов, аспирантов и молодых учёных, Томск, 25–27 мая 2016 г. – Томск: В-Спектр, 2016: в 6 частях. – Ч. 3. – С. 108–111.

60. Кочергин М.И. Система обучения решению задач по физике методом моделирования // Перспективы развития фундаментальных наук: труды XII Международной конференции студентов и молодых учёных. (Томск, 21–24 апреля 2015 г.) / Томский политехнический университет. – Томск: Изд-во Томского политехнического университета, 2015. – С. 1500–1502.

61. Кочергин М.И. Учебно-иллюстративные модели физических задач в образовательном процессе // Электронные средства и системы управления: материалы докладов XIII Международной научно-практической конференции (29 ноября – 1 декабря 2017 г.): в 2 ч. – Ч. 2. – Томск: В-Спектр, 2017. – С. 114-117.

62. Кочергин М.И. Численная аппроксимация табличных функций на основе методов многомерной оптимизации при моделировании физико-технических задач // Перспективы развития фундаментальных наук: сборник трудов XV Международной конференции студентов, аспирантов и молодых ученых (Томск, 24–27 апреля 2018 г.): в 7 т. Т. 3: Математика. – Томск: Издательский Дом Томского государственного университета, 2018. – С. 58-60.

63. Кочергин М.И., Ганджа Т.В. Схема автоматизированной формализации физических задач на основе метода компонентных цепей // Перспективы развития фундаментальных наук: труды XI Международной конференции студентов и молодых учёных. Россия, Томск, 22–25 апреля 2014 г. / под ред. Е.А. Вайтулевич. – Национальный Исследовательский Томский политехнический университет, 2014. – С. 1044–1047.

64. Кочергин М.И., Дмитриев В.М., Ганджа Т.В. Система управления лабораторией «Элементы и устройства роботизированных систем» // Сборник избранных статей научной сессии ТУСУРа (Томск, 22–24 мая 2019 г.): в 2 ч. – Томск: В-Спектр, 2019. – Ч. 2. – С. 23-25.

65. Кочергин М.И., Кочергина К.С. Формализация текстовых условий задач по физике // Доклады ТУСУР, том 19, №1, 2016. – С. 65-68.

66. Кочергин М.И., Спиридонова К.С. Особенности модуля автоматического анализа в системе логико-лингвистического анализа текстов

задач по физике // Научная сессия ТУСУР–2014: Материалы Всероссийской научно-технической конференции студентов, аспирантов и молодых учёных, Томск, 14–16 мая 2014 г. – Томск: В-Спектр, 2014: В 5 частях. – Ч. 2. – С. 257–260.

67. Кочергин М.И., Спиридонова К.С. Моделирование логико-лингвистических конструкций для формализации описания объекта моделирования // Электронные средства и системы управления: Материалы докладов IX Международной научно-практической конференции (30–31 октября 2013 г.): В 2 ч. – Ч. 2. – Томск: В-Спектр, 2013. – С. 110–114.

68. Кочергин М.И., Шутенков А.В. Иллюстрация баланса мощности в электрических цепях в учебно-иллюстративном модуле // Моделирование. Фундаментальные исследования, теория, методы и средства: материалы 17-ой Междунар. науч.-практ. конф., г. Новочеркасск, 26-27 сент. 2017г. / Южно-Российский государственный политехнический университет (НПИ) имени М.И. Платова. – Новочеркасск: Лик, 2017. – С. 50-53.

69. Леоненков А.В. Самоучитель UML. – СПб.: БХВ-Петербург, 2004. – 427 с.

70. Линник Ю.В. Метод наименьших квадратов и основы математико-статистической теории обработки наблюдений. – М.: Государственное изд-во физико-математической литературы. – 1958. – 336 с.

71. Ловецкий К.П. Математический синтез оптических наноструктур / К.П. Ловецкий, Л.А. Севастьянов, М. В. Паукшто, О.Н. Бикеев. – М.: Российский университет дружбы народов, 2008. – 123 с.

72. Маликов Р.Ф. Практикум по компьютерному моделированию физических явлений и объектов. – Уфа: Изд-во БашГПУ, 2005. – 291 с.

73. Машинная арифметика и идиомы численного программирования. – [Электронный ресурс] URL: https://www.sites.google.com/site/ltwood/projects/numpro/float

74. Мусабеков О. Задачи физики с техническим содержанием как средство профессиональной подготовки будущих технологов // Вестник Алматинского технологического университета. – 2017. – № 3. – С. 105-109.

75. Низамов И.М. Задачи по физике с техническим содержанием. – М.: Просвещение, 1980. – 96 с.

76. Новиков Е.А., Шорников Ю.В. Компьютерное моделирование жестких гибридных систем. - Новосибирск: Изд-во НГТУ, 2012. – 451 с.

77. Охорзин В.А. Компьютерное моделирование в системе Mathcad: учеб. пособие. – М.: Финансы и статистика, 2006. – 144 с.

78. Палагин А., Крывый С., Величко В., Петренко Н. К анализу естественно-языковых объектов // International Book Series. – Number 9 «Intelligent Processing». – ITHEA, Sofia, 2009. – С. 36-43.

79. Панов С.А., Григорьева Т.Е., Кочергин М.И. Разработка программных средств автоматической параметризации компьютерных моделей эколого-экономических систем предприятий нефтегазовой промышленности // Вестник Российского фонда фундаментальных исследований. – 2018. – № 4 (100). – С. 52-57.

80. Пархоменко С.С. , Леденёва Т. М. Обучение нейронных сетей методом левенберга-марквардта в условиях большого количества данных // Вестник ВГУ. Серия: Системный анализ и информационные технологии – 2014. – № 2. – С. 98-106.

81. Перегудов Ф.И., Тарасенко Ф.П. Введение в системный анализ. – М.: «Высшая школа», 1989. – 367 с.

82. Разработка школьных виртуальных лабораторий на базе среды программирования LabView [Электронный ресурс]. – URL: http://window.edu.ru/resource/401/47401/files/virt.pdf

83. Рассел С., Норвиг П. Искусственный интеллект: современный подход: Пер. с англ. К.А. Птицына – М.: Издательский дом «Вильямс», 2006. – 1408 с.

84. Редько В.Г. Эволюция, нейронные сети, интеллект. Модели и концепции эволюционной кибернетики. – М.: URSS, КомКнига, 2005. – 220 с.

85. Рожкин М.Е. Совершенствование контроля работы штанговых установок при эксплуатации скважин пермо-карбоновой залежи Усинского месторождения: дис. … канд. техн. наук: 25.00.17. – Ухта, 2010. – 163 с.

86. Романенко В.В. Использование новых стандартов электронного обучения xAPI и LTI при разработке адаптивных обучающих курсов // Современное образование: качество образования и актуальные проблемы современной высшей школы. – Томск: ТУСУР, 2019. – С. 123.

87. Саданова Б.М., Олейникова А.В., Альберти И.В., Одинцова Е.А., Плеханова Е.Н Применение возможностей виртуальных лабораторий в учебном процессе технического вуза // Молодой ученый. – 2016. – №4(108). – С. 71-74.

88. Светлакова С.В. Информационно-измерительная система динамометрирования скважин, оборудованных штанговыми глубинными насосами: автореф. дис. … канд. техн. наук: 05.11.16. – Уфа, 2008. – 18 с.

89. Свидетельство о государственной регистрации программы для ЭВМ №2019660693. Библиотека моделей компонентов для многоуровневого компьютерного моделирования физико-технических задач в среде моделирования МАРС / Кочергин М.И. – 12.08.2019. – М.: Роспатент, 2019.

90. Свидетельство о государственной регистрации программы для ЭВМ №2019660759. Программный модуль для обучения компьютерному моделированию физико-технических задач / Кочергин М.И. – 13.08.2019. – М.: Роспатент, 2019.

91. Сениченков Ю.Б. Инновационные возможности проекта Rand Model Designer // Компьютерные инструменты в образовании. – 2010. – № 5. – С. 29-34.

92. Сениченков Ю.Б. Основы теории и средства моделирования гибридных систем : автореферат дис. ... д-ра техн. н. – Санкт-Петербург, 2005. – 312 с.

93. Сениченков Ю.Б. Численное моделирование гибридных систем. – СПб.: Изд-во Политехн. ун-та, 2004. – 206 с.

94. Стукалин А.А. Имитационное моделирование в среде AnyLogic // Аллея науки. – 2018. – Т. 1. – № 1 (17). – С. 470-477.

95. Суранова Д.А. Алгоритмы и комплекс программ моделирования персонифицированного естественно-языкового взаимодействия оператора с ЭВМ: дисс. … канд. техн. наук: 05.13.18. – Томск, 2013. – 128 с.

96. Тихонов К.М., Тишков В.В. SimMechanics Matlab как средство моделирования динамики сложных авиационных робототехнических систем // Труды МАИ. – 2010. – № 41. – С. 13.

97. Турчак Л. И., Плотников П. В. Основы численных методов: Учебное пособие. – М.: ФИЗМАТЛИТ, 2003. – 304 с.

98. Физикам – преподавателям и студентам [Электронный ресурс]. – URL: teachmen.ru

99. Филиппов А.Ю. Алгоритмы формализации и автоматизации решения задач на основе среды компьютерного моделирования задач: дис. … канд. техн. наук. – Томск, 2007. – 232 с.

100. Филиппов А.Ю., Шарова О.Н., Кураколов А.Н. Компьютерное учебное пособие (КУП) по физике с элементами моделирования // Материалы докладов Всероссийской научно-технической конференции студентов, аспирантов и молодых ученых «Научная сессия ТУСУР – 2007»: в 5 частях. Томск, 2007. – С. 247-250.

101. Формалев В. Ф., Ревизников Д. Л. Численные методы. – М.: ФИЗМАТЛИТ, 2004. – 400 с.

102. Черемисина, Е.Н., Антипов О.Е., Белов М.А. Роль виртуальной компьютерной лаборатории на основе технологии облачных вычислений в современном компьютерном образовании // Дистанционное и виртуальное обучение.— 2012.— № 1.— с. 53–60.

103. Шиян А.А. Экспериментальное решение физико-технических задач в развивающем и личностно-ориентированном обучении студентов вузов: дис. … канд. пед. н. – Санкт-Петербург, 2000. – 165 с.

104. Шорников Ю.В. Прикладное математическое, алгоритмическое и программное обеспечение компьютерного анализа гибридных систем: дис. … д-ра техн. наук: 05.13.11. – Новосибирск, 2009. – 313 с.

105. Янченко В.С. Моделирование обобщенного термодинамического цикла тепловых двигателей // Транспорт на альтернативном топливе. – 2011. – № 4 (22). – С. 38–40.

106. Янченко В.С., Лукутцова Н.П., Дегтярёв Е.В., Королева Е.Л. Ширко С.В. Математическое моделирование процесса подбора состава песчаной смеси // Строительство и реконструкция. – 2012. – № 4(42). – С. 66–71.

107. Ясницкий Л.Н. Введение в искусственный интеллект: Учеб. пособие для студ. высш. учеб. заведений. – М.: Издательский центр «Академия», 2008. – 176 с.

108. Baum F.I., Kozlov O.S., Parshikov I.A., Petuhov V.N., Timofeev K.A., Shchekaturov A.M. Simintech software for programming control-system devices // Atomic Energy. – 2013. – Т. 113. – № 6. – P. 443-446.

109. Brill E. A simple rule-based part of speech tagger // Proceedings of the third conference on Applied natural language processing. – 1992.

110. Bruck D. Dyraola user’s manual / D. Bruck, H. Elmqvist, M. Otter. - Lund Switzerland, 1996. – 340 p.

111. Cellier F. Combined discrete continuous system simulation by use of digital computers: techniques and tools // PhD thesis, ETH Zurich. - Switzerland, 1979. – P. 144-156.

112. Daciuk J. Treatment of Unknown Words, proceedings of Workshop on Implementing Automata. – Potsdam, Germany, 2001. – Vol. 2214. – Pp. 71–80.

113. Filmor C. Frames and the semantics of understanding / C. Filmor // Quaderni di semantica. – Vol. 4/2. December. – 1985. – P. 222–254.

114. Gear C.W. Solving ordinary differential equations with discontinuities / C.W. Gear, O. Osterby // Technical report. - Dept. of Comput. Sci., University of Illinois, 1981. – P. 27-31.

115. Haas S.W., Metzler D.P. The flexibility of case grammar representations: a porting procedure for natural language interfaces // International Journal of Man-Machine Studies. – 1989. – Vol. 31. – № 5. – P. 535–556.

116. Harel D. Statecharts: A visual formalism for complex systems // Science of Computer Programming. – 1987/ – №8. – P. 231–274.

117. Huffman D.A. The synthesis of sequential switching circuits // Journ. Franklin Inst. 257. – 1954. – № 3. – P. 161—190.

118. Huffman S.B. Learning information extraction patterns from examples // Learning for Natural Language Processing. – 1995. – № 4 (6). – P. 246–260.

119. Jurafsky M., Martin J.H. Speech and Language Processing // Upper Saddle River. – NJ, USA: Prentice-Hall, Inc., 2009. – 1024 p.

120. Kochergin M.I. Automated Training System for Modeling Physics Problems // Электронные средства и системы управления: материалы докладов XII Международной научно-практической конференции (16–18 ноября 2016 г.): в 2 ч. – Ч. 2. – Томск: В-Спектр, 2016. – С. 211-213.

121. Kochergin M.I. Formalization of low structured physics and technics problems for computer modeling and simulation // Научная сессия ТУСУР–2017: материалы Международной научно-технической конференции студентов, аспирантов и молодых учёных, посвященной 55-летию ТУСУРа, Томск, 10–12 мая 2017 г. – Томск: В-Спектр, 2017: в 8 частях. – Ч. 8. – С. 185–187.

122. Kochergin M.I. Interpretation of the statechart diagram into a multilevel simulation language // Доклады ТУСУР. – 2017. – Т. 20, № 4. – С. 122–125. DOI: 10.21293/1818-0442-2017-20-4-122-125

123. Kochergin M.I. Numerical approximation of table functions based on multidimensional optimization methods // Сборник избранных статей научной сессии ТУСУР, Томск, 16–18 мая 2018 г.: в 3 частях. – Томск: В-Спектр, 2018 – Ч. 3. – С. 242-245.

124. Lafferty J.D., McCallum A., Fernando C.N. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data // Proceedings of

the Eighteenth International Conference on Machine Learning. – ICML'01, 2001. – Pp. 282–289.

125. Levenberg K. A method for the solution of certain problems in least squares// Quart. Appl. Math. – 1944. – Vol. 2. – p. 164–168.

126. Lui J. A hierarchical hybrid system model and its simulation / J. Lui [и др.] // Proceedings of the 38th Conference on Decision and Control. 1999. – P. 2407-2411.

127. Marquardt D. An algorithm for least-squares estimation of nonlinear parameters// SIAM J. Appl. Math. – 1963.– Vol. 11. – p. 431–441.

128. MFC Desktop Applications [Электронный ресурс]. – URL: https://docs.microsoft.com/en-us/cpp/mfc/mfc-desktop-applications?view=vs-2019

129. Mikheev A. An Automatic Rule Induction for Unknown Word Guessing, In Computational Linguistics. – Vol. 23 (3). – ACL, 1997. – Pp. 405–423.

130. Moore E.F. Gedanken-experiments on Sequential Machines // Automata Studies,Annals of Mathematical Studies. Princeton, N.J.: Princeton University Press. – 1956. – №34: – P. 129—153.

131. Mosterman P. An overview of hybrid simulation phenomena and their support by simulation packages // Hybrid Systems: Computation and Control, vol. 1569 of Lecture Notes in Computer Science. — Springer Verlag. 1999. – P. 165-177.

132. Mea1y G.H. A method for synthesizing sequential circuits // Bell System Tech. Journ. – 1955. – №34. – P. 1045-1079.

133. Park T. Stale event location in differential-algebraic models. ACM Transactions on Modeling and Computer Simulation / T. Park, P.I. Barton // TOMACS. 1996. №6(2). – P. 137-165.

134. Press W.H. Numerical Recipes in C. The Art of Scientific Computing / W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery. – Cambridge: Cambridge University Press, 1997. – 1018 p.

135. Simulation of a Bouncing Ball [Электронный ресурс] URL: https://www.mathworks.com/help/simulink/slref/simulation-of-a-bouncing-ball.html

136. Software Tools for Academics and Researchers Internet [Электронный ресурс]. – URL: http://www.virtulab.net/

137. The Virtual Lab for controlling real experiments via Internet [Электронный ресурс]. – URL: http://www.researchgate.net/publication/3826574_The_Virtual_Lab_for_controlling_real_experiments_via_Internet.

138. Thompson Joe F., Warsi Z. A., Mastin C. V. Numerical Grid Generation, Foundations and Applications. — Amsterdam: North-Holland, 1985. – 331 p.

139. VirtuLab – виртуальная образовательная лаборатория [Электронный ресурс]. – URL: http://www.virtulab.net/

140. Wolfram Demonstrations Project [Электронный ресурс]. – URL: https://demonstrations.wolfram.com/

ПРИЛОЖЕНИЕ А. ПРИМЕРЫ МОДЕЛЕЙ ЗАДАЧ

Модели сгруппированы следующим образом:

1. Непрерывные задачи (модели), характеризующиеся отсутствием дискретного поведения. Примеры таких задач зачастую рассматриваются в литературе (в том числе в учебной) и могут быть смоделированы как с использованием языков программирования, так и математических пакетов, например, Mathcad.

2. Дискретно-непрерывные задачи, характеризующиеся осложнением в виде гибридного поведения, сочетающего как дискретную модель, так и непрерывную – см. параграф 2.2). Моделирование таких задач в математических пакетах типа Mathcad затруднено, поэтому для таких целей используются среды моделирования – СМ МАРС, Rand Model Designer, ИСМА, Simulink и др. (см. параграф 1.2).

3. Физико-геометрические задачи, характеризующиеся осложнением в виде нестандартной структуры объектов и их связей. Сюда относятся такие задачи, как движение тел, связанных нежёсткой связью, движение тел по профилям поверхности и пр. (см. пункт 2.3.3). Специальные компоненты для моделирования таких задач в СМ как правило отсутствуют.

Модели ФТЗ, созданные с применением компонентов, отражающих физические свойства объектов, в данной главе не обособляются и присутствуют в каждом из выделенных классов. Для каждой задачи может быть представлено аналогичное представление – с применением ИМП вместо физических компонентов. Отдельно необходимо отметить задачу о моделировании усилий на полированном штоке штангового глубинного насоса (параграф 4.2.1), смоделированную в СМ МАРС с помощью модификации «классических» физических компонентов, таких как: источник силы/скорости, упругость (жёсткость), демпфер (трение).

4. Учебно-иллюстративные модели, созданные с применением разработанного подхода и инструментария по разным дисциплинам в образовательных целях.

5. Модели задач с управляющими конструкциями – модификации рассмотренных задач, иллюстрирующие возможности параметризации моделей, экспорта результатов моделирования, создания сценариев проведения вычислительных экспериментов.

1 Многоуровневое моделирование непрерывных задач

1.2 Физический маятник

Одной из простейших динамических систем, с которой студент знакомится в первую очередь является физический маятник – тело, подвешенное на нити, совершающее колебательные движения и имеющее горизонтальную ось вращения, которое не проходит через его центр тяжести [Кочергин 2017].

На V-слое вводятся исходные данные (начальный угол отклонения, длина нити, масса тела) и отображаются выходные (текущие линейная скорость, кинетическая и потенциальные энергии), также возможна визуализация результатов в виде графиков – на рис. А.1.

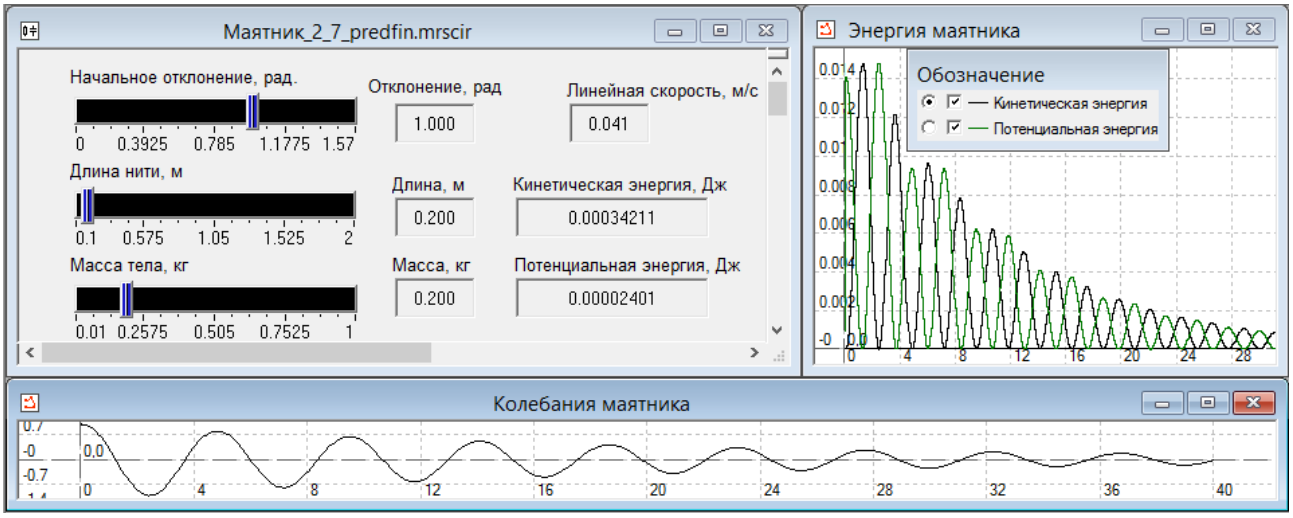


Рисунок А.1 – Визуализация результатов моделирования маятника

На объектном слое (С-слое) располагается непосредственно математическая модель маятника (см. рис. А.2). Основными компонентами в модели являются инерционное звено (источник массы) J1, источник скорости «Источник колебаний», источник трения «100000». Начальные значения переменных в модели задаются через компоненты «Начальное значение» (квадрат с надписью *IV* в центре), параметры – источниками (окружность с *C* в

центре), рассчитанные значения измеряются и передаются на L-слой компонентами-измерителями (обозначены *Vn*).

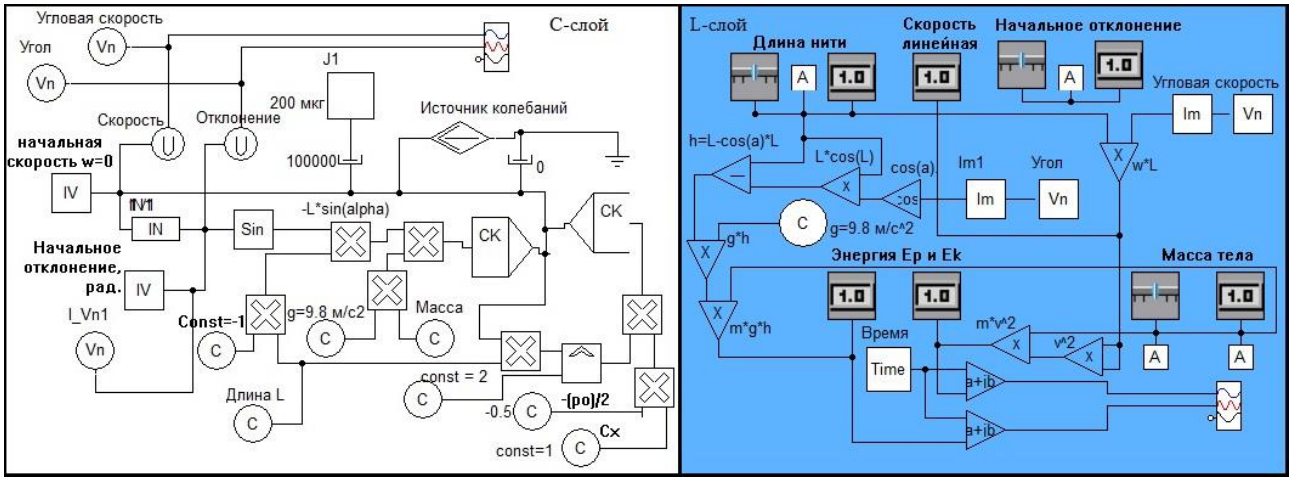


Рисунок А.2 – Многоуровневая модель маятника в СМ МАРС

L-слой является связующим между визуальным и объектным, а также используется для расчета значений параметров системы: линейной скорости маятника и его энергии.

Взаимодействуя с моделью можно варьировать входные параметры и наблюдать за изменением амплитуды колебаний маятника, его скорости, кинетической и потенциальной энергии.

1.3 Математический маятник в полярных координатах

Рассмотрим построенную модель математического маятника с затуханием в полярных координатах.

На рис. А.3 представлен интерфейс модели на V-слое. Входными параметрами модели являются длина нити маятника, коэффициент затухания и величина начального отклонения маятника. Выходными переменными – текущие значения угловой скорости, отклонения и время моделирования.

Рисунок А.3 – Модель маятника на V-слое

На рис. А.4 представлена модель маятника на С-слое.

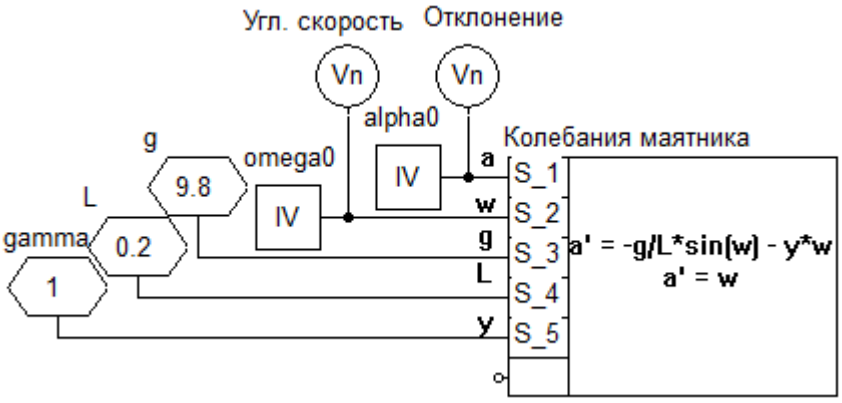


Рисунок А.4 – Модель маятника на С-слое

Непрерывная модель, как видно из рис. А.4 реализована с применением ИМП. Источники физической величины gamma, L, g и источники начального значения omega0 и alpha0 имеют одноимённые отображения на L-слое (рис. А.5), что позволяет параметрировать модель С-слоя на V-слое посредством передачи данных через L-слой. Измерители «Угл. скорость» и «Отклонение» на каждой итерации передают результаты решения системы уравнений в ИМП на L-слой через одноимённые отображения.

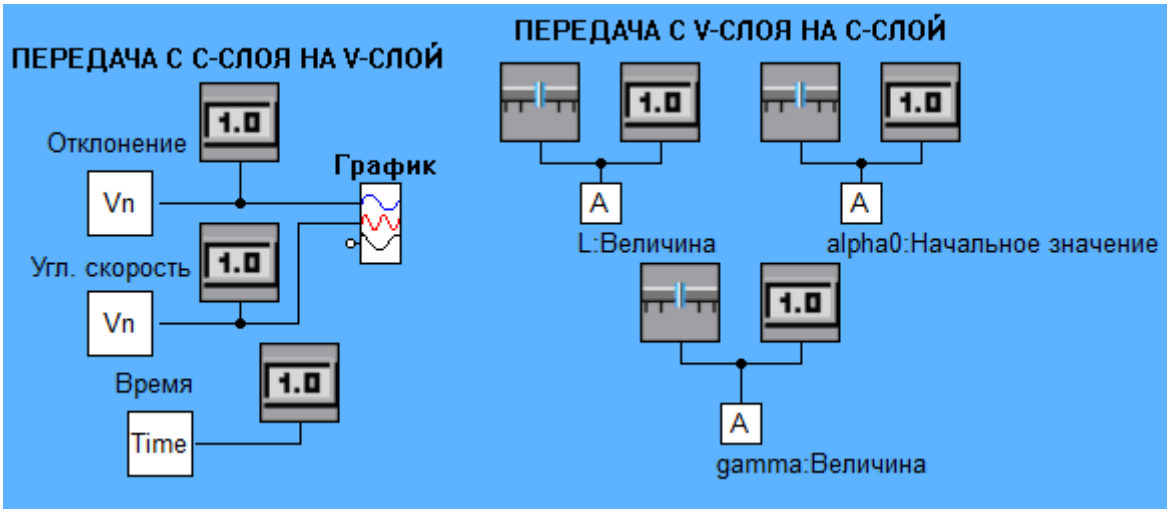


Рисунок А.5 – Модель маятника на L-слое

Подаваемые на компонент «График» значения визуализируются в отдельном окне (рис. А.6).

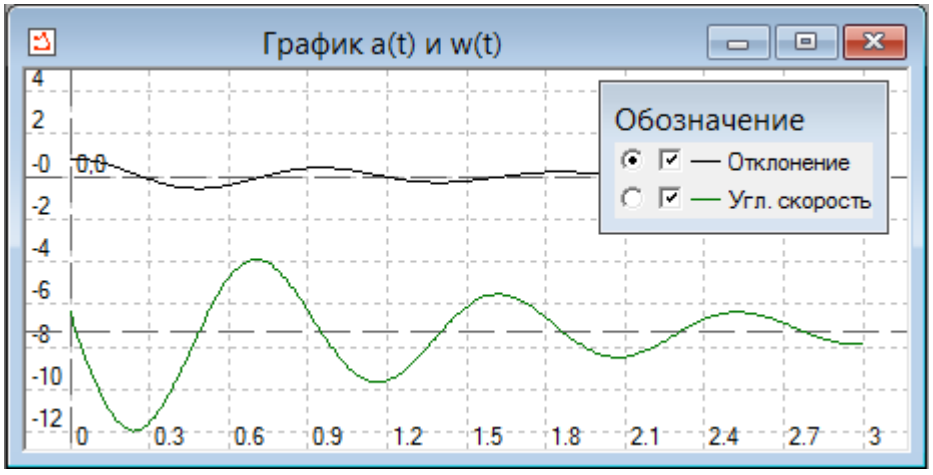


Рисунок А.6 – Визуализация результатов моделирования

Из графиков видна динамика изменения значения угловой скорости и отклонения маятника от вертикальной оси.

1.4 Расчёт тормозного пути автомобиля

Многоуровневая компьютерная модель динамики поведения автомобиля на дорожном покрытии может быть представлена с помощью следующих физических компонентов:

- источника силы, задающего силовое воздействие двигателя на колеса с целью их вращения при движении автомобиля;
- демпфера с функционально-параметрическим коэффициентом трения, описывающего процесс трения колес о дорогу;

- управляемого демпфера, реализующего воздействие силы трения при торможении;
- инерционного звена, выражающего массу автомобиля (ввиду одномерности движения в задаче заменяем модель твёрдого тела на инерционное звено).

Исследование характеристик тормозного режима автомобиля в многоуровневой компьютерной модели осуществляется с помощью панели на V-слое (рис. А.7).

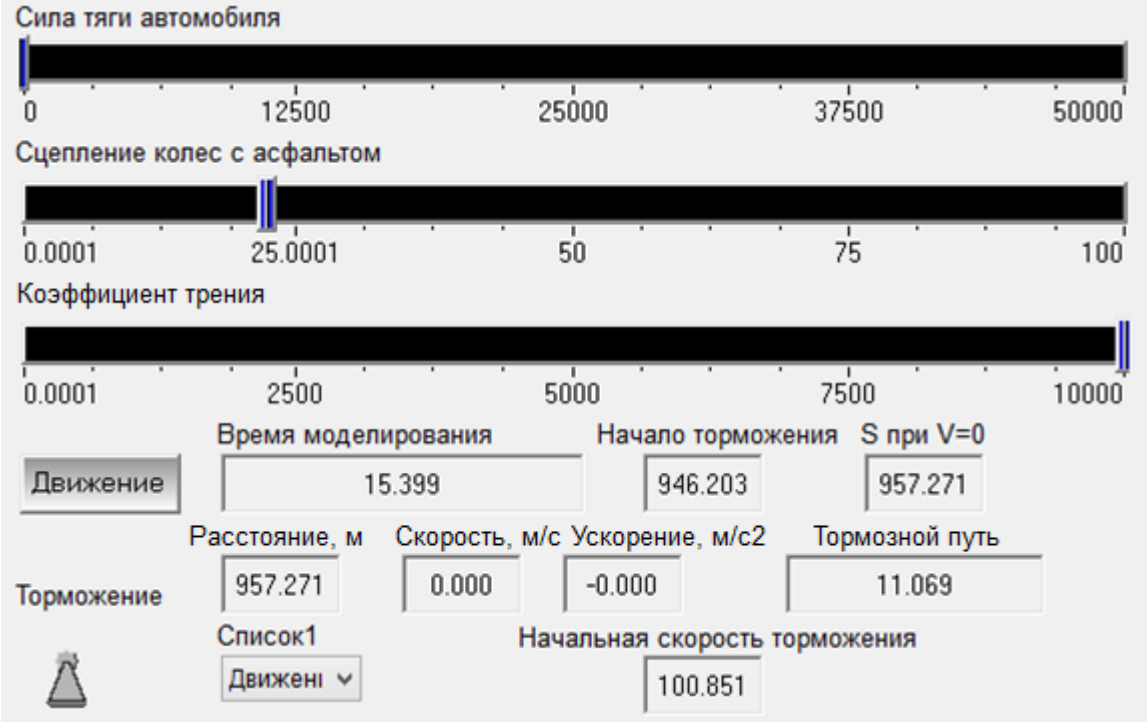


Рисунок А.7 – Интерфейс модели расчёта тормозного пути

Как видно из рис. А.7 при указанных начальной скорости движения 100 км/ч время торможения составило 15.399 с, при этом тормозной путь составил 11.069 м.

Управление моделью возможно во время её исполнения в СМ – в режиме, приближенному к реальному времени. Регулятор «Сила тяги автомобиля» функционально соответствует педали акселератора автомобиля. Регулятор «Сцепление колес с асфальтом» задаёт коэффициент трения колес о дорожную поверхность, который определяет величину трения в зависимости от дорожного покрытия (асфальт, грунтовая дорога, гравийная насыпь и т.п.) и от погодных

условий: дождь, гололед, снег и т.д. Регулятор «Коэффициент трения» определяет коэффициент трения между колесами и тормозными колодками автомобиля.

Переключение кнопки «Торможение» инициирует торможение автомобиля – сила тяги автомобиля будет установлена равной 0, а величина коэффициента трения между колесами и тормозными колодками автомобиля будет принята равной максимальному значению. Соответствующая этому механизму КЦ *L*-слоя представлена на рис. А.8.

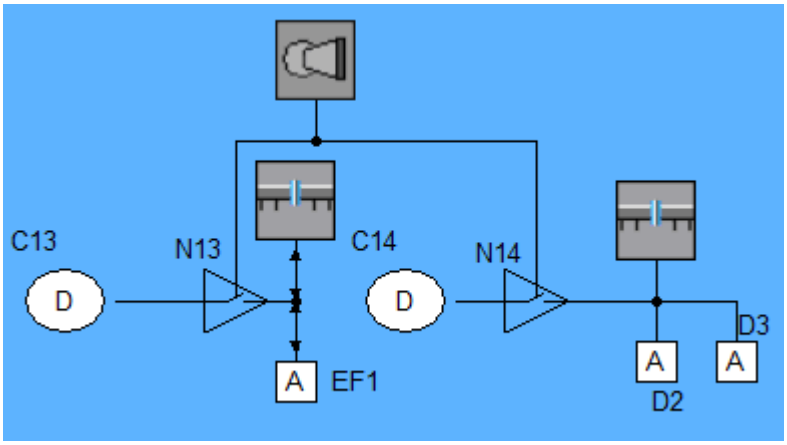


Рисунок А.8 – КЦ L-слоя для инициализации торможения

Измеряемыми данными модели являются: время моделирования, пройденное расстояние, текущая скорость, текущее ускорение, отображаемые на соответствующих табло V-слоя. Начальная скорость торможения, которая была зафиксирована при первом моменте наблюдения отрицательного ускорения, отображается в цифровом табло «Начальная скорость торможения». Текущее значение перемещения автомобиля (координаты по оси движения) в момент начала торможения визуализируется с цифровым табло «Начало торможения». Вычисленный на основе данных значений перемещений тормозной путь автомобиля визуализируется цифровым табло «Тормозной путь».

КЦ тормозной системы автомобиля, отражающая его поведение в режиме разгона, движения с постоянной скоростью и торможения, представлена на рис. А.9 и включает в себя следующие компоненты:

- *EF*1 – источник силы, выражающий тяговую мощность автомобиля при разгоне;

- *J*1 – масса автомобиля, оказывающая инерционное воздействие на него при разгоне, движении и торможении;
- *D*1 – компонент «Демпфер», выражающий трение колес о дорогу и другие виды трения в режиме движения автомобиля;
- *D*2 – компонент «Демпфер», отражающий в модели трение не вращающихся тормозных колодок о вращающихся колесных барабанах.

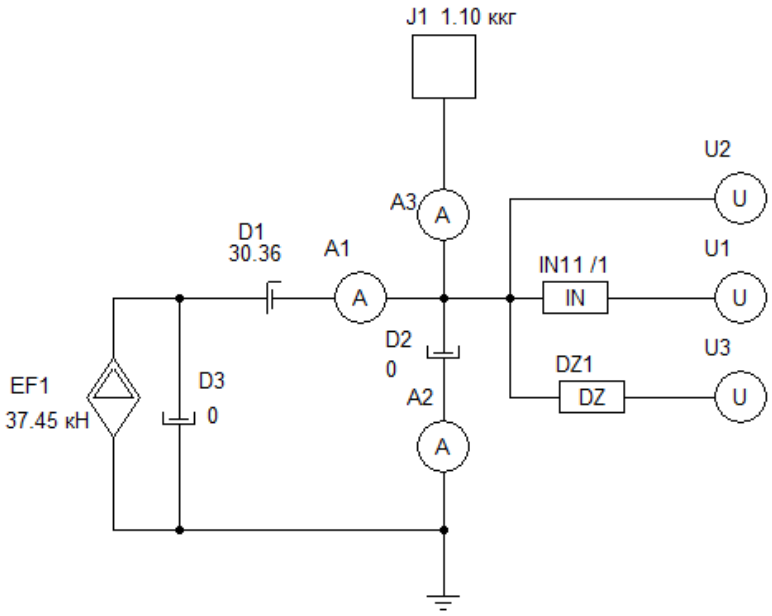


Рисунок А.9 – Компьютерная модель тормозной системы автомобиля на С-слое

2 Многоуровневое моделирование дискретно-непрерывных задач

2.2 Прыгающий мяч

Рассмотрим «классическую» задачу, используемую для иллюстрации явления гибридного поведения – модель скачущего мяча. Будем считать, что мяч массы m запущен вверх под некоторым углом α к горизонтальной плоскости с некоторой начальной скоростью V_0 $(\overrightarrow{V_0} \neq 0)$ в плоско-параллельном поле тяготения. После броска шарик летит вверх, затем вниз, ударяется о плоскость и отскакивает вверх. Траектория мяча при упруго-вязком ударе будет напоминать серию перевёрнутых парабол. В этой задаче хорошо видны фазы «полёт» и «отскок» сменяющие друг друга. В фазе полета поведение шарика описывается системой уравнений (1).

$$\frac{dx}{dt} = V_x, \frac{dy}{dt} = V_y, \frac{dV_y}{dt} = -g, \frac{dV_x}{dt} = 0 \quad (1),$$

где $[x, y]$ – дальность и высота полета; $[V_x, V_y]$ – горизонтальная и вертикальная составляющие скорости; g – ускорение силы тяжести. Будем рассматривать отскок как мгновенное дискретное действие, в результате которого происходит изменение знака вертикальной составляющей скорости на противоположный. Момент отскока считаем мгновенным и определяем по условию выполнения неравенств: $y \le 0,\ V_y < 0$ [Новиков, Шорников 2012].

Таким образом, алгоритм поведения тела в этой задаче представляет собой «склейку» непрерывных поведений в виде кусочно-непрерывной функции. Вследствие этого будем использовать диаграммы состояния для моделирования дискретного поведения мяча (смена фаз «Полёт» и «Отскок») и ИМП для моделирования непрерывного поведения.

На рис. А.10 представлен вид модели на V-слое.

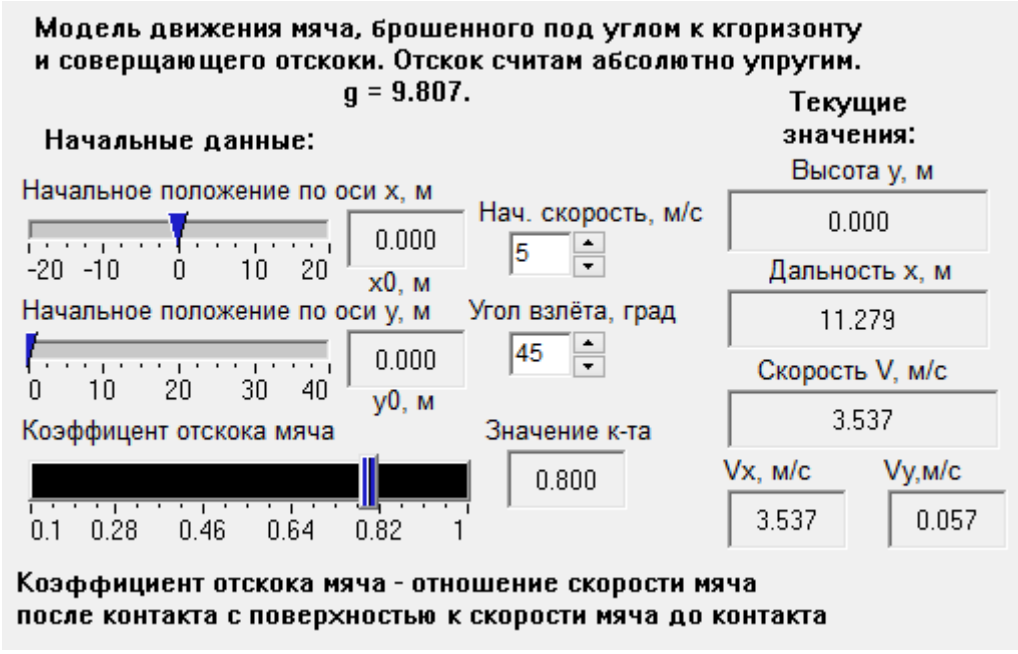


Рисунок А.10 – Модель прыгающего мяча (V-слой)

Как видно из рисунка в качестве входных параметров выбраны: начальная скорость, угол броска, коэффициент отскока мяча и начальное положение по осям OX и OY. Выходные переменные – координаты мяча, значение скорости

(векторное и в проекциях на оси OX и OY). График траектории мяча представлен на рис. А.11.

Рисунок А.11 – Траектория полёта мяча

Модель дискретного поведения представлена на рис. А.12, где также можно видеть процедуры преобразования значений переменных и параметров (перевод из градусов в радианы, расчёт текущего значения вектора скорости и проекций начального значения скорости – КЦ 3, 5), передачи значений на С-слой (КЦ 1, 2) и пр.

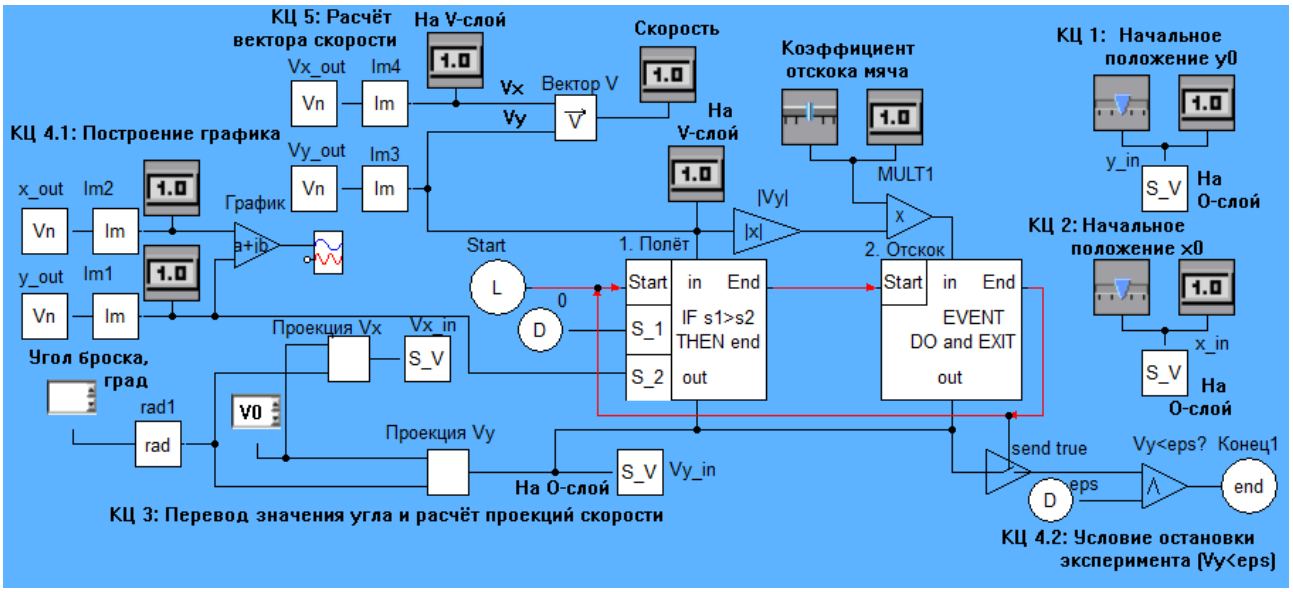


Рисунок А.12 – Модель прыгающего мяча (L-слой)

Диаграмма состояний, представляющая собой замкнутую цепь *Start* – «1. Полёт» – «2. Отскок» – «1. Полёт», является моделью дискретного поведения,

параметрирующей непрерывную модель поведения (рис. А.13), которая описывает модель непрерывного поведения согласно системе уравнений (1).

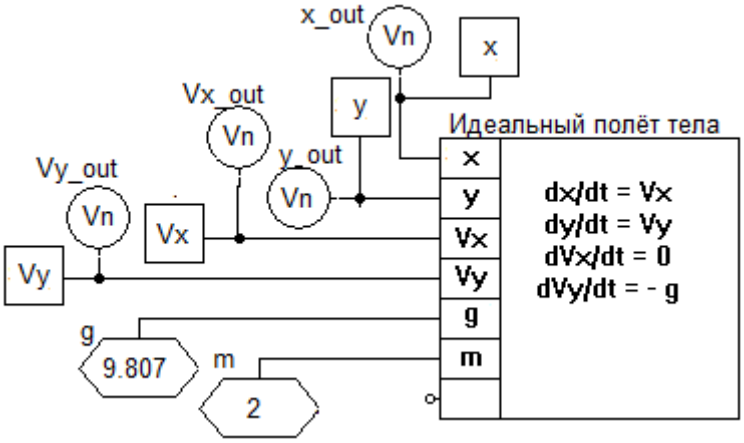


Рисунок А.13 – Модель прыгающего мяча (С-слой)

На *С*-слое представлены:

а) компоненты источники значений переменных *S_V* и параметров *g*, *m* (которые передают на *С*-слой значения, получаемые со своих отображений на *L*-слое);

б) компонент-преобразователь – ИМП «Идеальный полёт тела», содержащий уравнения кинематики и динамики движения тела;

в) компоненты-измерители *Vn*, передающие рассчитанные значения на *L*-слой.

3 Многоуровневое моделирование физико-геометрических задач

В данном параграфе представлены многоуровневые компьютерные модели ФТЗ с геометрическим содержанием – то есть обладающие ярко выраженным пространственно-геометрическим субаспектом (согласно методике многоаспектного анализа – см. пункт 2.1.1). Для моделирования таких задач использовались разработанные геометрические компоненты, описанные в пункте 2.3.3.

3.1 Скатывание тела по поверхности со сложным профилем

Рассмотрим движение твёрдого тела по поверхности, задаваемой некоторой аналитической функцией: «Тело массой m и радиусом R катится по

наклонной плоскости, составляющей угол α с горизонтом, без скольжения (на него действует сила трения качения)». На цилиндр действуют внешние силы: сила тяжести *mg*, сила трения $F_{тр}$, сила реакции со стороны плоскости *N*. Движение рассматриваем как поступательное со скоростью, равной скорости центра масс, и вращательное относительно оси, проходящей через центр масс. Уравнение для движения центра масс шара представим так (А.1):

$$m\vec{a}_C = m\vec{g} + \vec{N} + \vec{F}_{mp} \quad \text{(А.1)},$$

где a_c – ускорение поступательного движения центра масс *C*.

Представим уравнение (А.1) в скалярном виде (А.2) в проекциях на оси OX и OY:

$$\begin{cases} ma_{C_x} = m \cdot g \cdot \cos^2 \alpha - F_{mp_x} - N_x \\ ma_{C_y} = N_y - mg_y \cdot \cos\alpha \cdot \sin\alpha - F_{mp_y} \end{cases} \quad \text{(А.2)}.$$

Представим задачу схематично (рис. А.14). В пунктирном прямоугольнике отметим направления оси OZ, углового ускорения ε, момента силы трения $M_{тр}$.

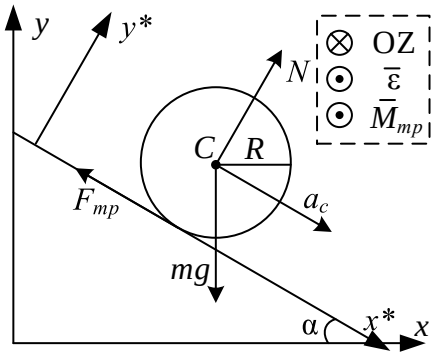


Рисунок А.14 – Иллюстрация к задаче скатывания тела

Для построения модели будем использовать геометрический компонент «Поверхность (аналитическая)», который задаёт поверхность посредством ввода в него символьного уравнения вида *y=f*(*x*) и физические компоненты «Источник силы реакции опоры» «Источник силы трения», «Источник силы тяжести», «Твёрдое тело в поступательном движении», описанные в пункте 2.3.4. Интерфейс модели представлен на рис. А.15.

Рисунок А.15 – Вид модели скатывания тела на V-слое

Работа модели начинается с инициализации компонента «Твёрдое тело в поступательном движении» на L-слое (рис. А.16) начальными данными (поступают слева).

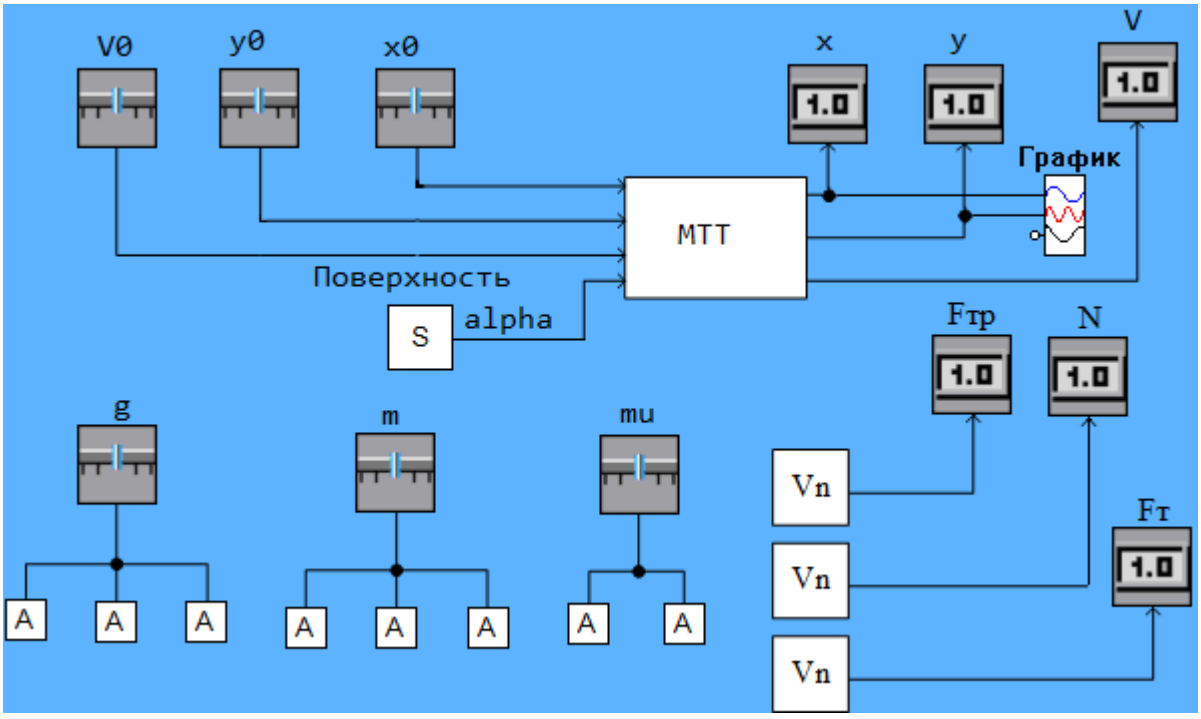


Рисунок А.16 – Модель скатывания тела на L-слое

Также инициализируются остальные компоненты-источники сил через компоненты-атрибуты, после чего вычислительное ядро СМ МАРС решает систему уравнений, сформированную из моделей компонентов на С-слое (А.17) – сумма потоковых переменных *F* в узле равна 0.

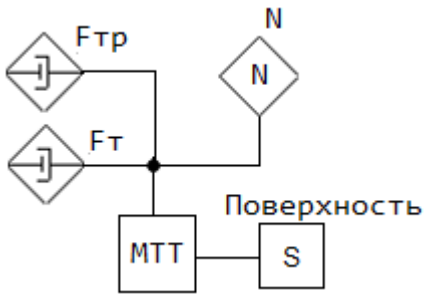


Рисунок А.17 – КЦ модели на С-слое

Результаты расчетов (координаты и скорости) передаются *L*-слой через отображение компонента «Твёрдое тело в поступательном движении», а также рассчитанное значение положение тела по оси OX передаётся на вход компонента «Поверхность», который рассчитывает значение угла наклона поверхности для следующей итерации работы модели и передаёт его на С-слой для расчёта проекций сил F на оси OX и OY – тем самым реализуется моделирование движения тела по наклонной плоскости (рис. А.18).

Рисунок А.18 – Траектория скатывания тела по кривой поверхности

4 Учебно-иллюстративные модели

Учебно-иллюстративным модулем (УИМ) [Дмитриев, Шутенков, Сторчак 2014] называется программно-инструментальное приложение к интерактивному учебнику, построенное на основе многоуровневых компьютерных моделей [Кочергин, Шутенков 2018]. В задачу УИМ входит иллюстрация с максимальной

наглядностью студентам смысла и действия закона, метода, принципа или физического эффекта, касающегося изучаемой дисциплины. Данный подход предполагает, что в теоретическом материале электронного курса (например, в *Moodle*) гиперссылки, по которым вызываются определенные учебно-иллюстративные модули.

Работа с такими многоуровневыми моделями может преследовать как цель углубления знаний в области физики, так и закрепления навыков моделирования.

В первом случае модель выступает в качестве инструмента обучения, позволяющего проводить вычислительный эксперимент и визуализировать результаты моделирования. Для этих целей СМ МАРС позволяет преобразовать любую созданную в ней модель в отдельное исполняемое приложение – учебно-иллюстративный модуль (УИМ) [Дмитриев, Ганджа 2013], который может использоваться в качестве дополнения к электронному учебному пособию или являться частью электронного курса, созданного, например, в системе дистанционного образования *Moodle* (в качестве иллюстративного материала к лекции или элемента домашнего, лабораторного занятия). Используя УИМ, студенты могут исследовать поведение моделируемого объекта, выявлять зависимость между различными величинами и оценивать то, какое влияние на поведение объекта оказывает изменение того или иного параметра модели. В качестве отдельных образовательных задач в работе студентов с УИМ можно отметить обучение интерпретации графиков зависимостей одной величины от другой. Также студентам может быть предложено сравнение и сопоставление результатов виртуальных экспериментов, полученных на моделях различной степени детализации.

Во втором случае компьютерная модель выступает объектом изучения: студенту необходимо взаимодействовать с самой средой моделирования – он может решать различные задачи, связанные с детализацией модели или введением каких-либо ограничений или дополнительных условий. В качестве критериев оценивания решения поставленных задач, выполненного пользователем на компьютерной модели, можно использовать следующие:

адекватность составленной модели, корректность составленного алгоритма поведения объекта, правильность составления модели, обоснованность выбора компонентов (достаточность и неизбыточность состава КЦ).

4.1 Иллюстрация баланса мощности в электрических цепях

Задачей данного УИМ является наглядная иллюстрация студентам явления баланса мощности в электрической цепи. Рассмотрим электрическую схему *RLC*-цепи, компонентная цепь которой расположена в объектном слое среды МАРС (рис. А.19).

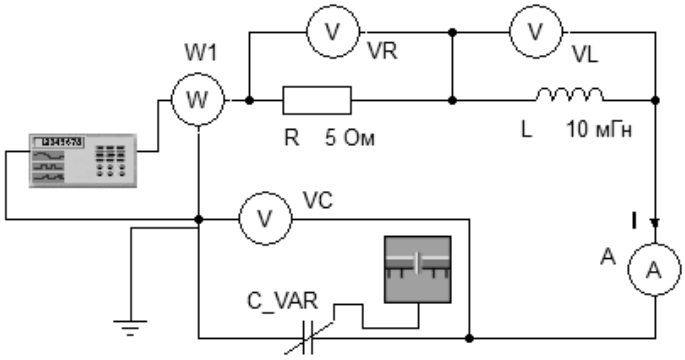


Рисунок А.19 – КЦ УИМ на С-слое

В качестве источника энергии используется функциональный генератор (компонент ), лицевая панель которого представлена на рис. А.20. С помощью лицевой панели осуществляется управление параметрами функционального генератора: возможно изменение амплитуды и частоты выходного сигнала, форма сигнала установлена синусоидальная.

Рисунок А.20 – Лицевая панель виртуального прибора

В схеме используется переменная ёмкость (элемент *C_VAR*). Регулятор изменения ёмкости (компонент) отображается в визуальном слое редактора. Измерительные приборы (на схеме компонентной цепи вольтметры *VR*, *VL*, *VC*, амперметр A, ваттметр *W*1) передают измеряемые значения на соответствующие компоненты L-слоя.

На L-слое производятся вычисления значений величин, необходимых для передачи в визуальный слой. L-слой используется непосредственным разработчиком УИМ. В среде моделирования МАРС реализован широкий набор компонентов, реализующих различные математические операции (сложение вычитание, умножение и т.д.) и позволяющих производить предварительную обработку результатов расчета моделируемой схемы с последующей передачей их на визуальный слой.

Для данной цепи на L-слое реализуются передача значений мощностей источника энергии, алгоритмы определения параметров отдельных элементов цепи, а также любые другие необходимые вычисления, результаты которых затем передаются в визуальный слой. Вид модели на L-слое для данной цепи представлен на рис. А.21.

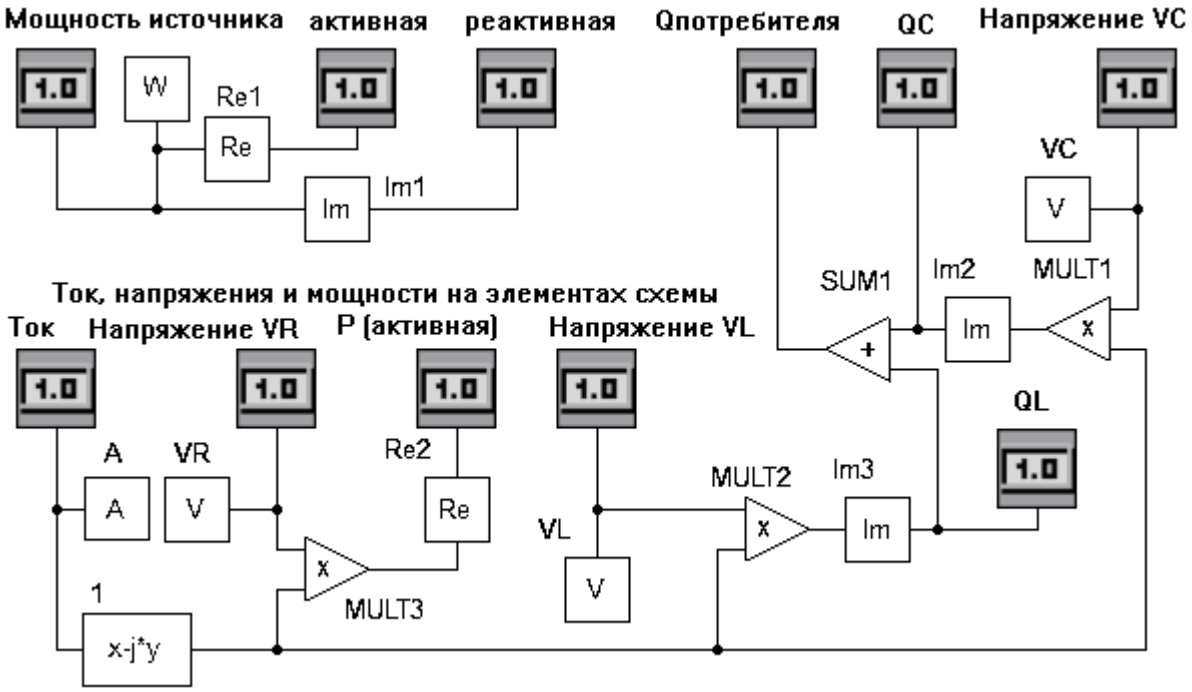


Рисунок А.21 – КЦ УИМ на L-слое

Все величины, обозначенные на L-слое редактора элементом «Цифровое табло» (компонент 1.0), выводятся в соответствующее окно визуального слоя.

В визуальном слое отображаются показания приборов, получаемые вычисления, а также располагается лицевая панель функционального генератора. Обучаемый студент наблюдает за процессом посредством визуального слоя, вид которого представлен на рис. 5. Там же присутствует элемент «Индикатор с прямоугольным бегунком» изменения ёмкости, обозначенный в объектном слое системы как ⊤⊤ (рис. А.22).

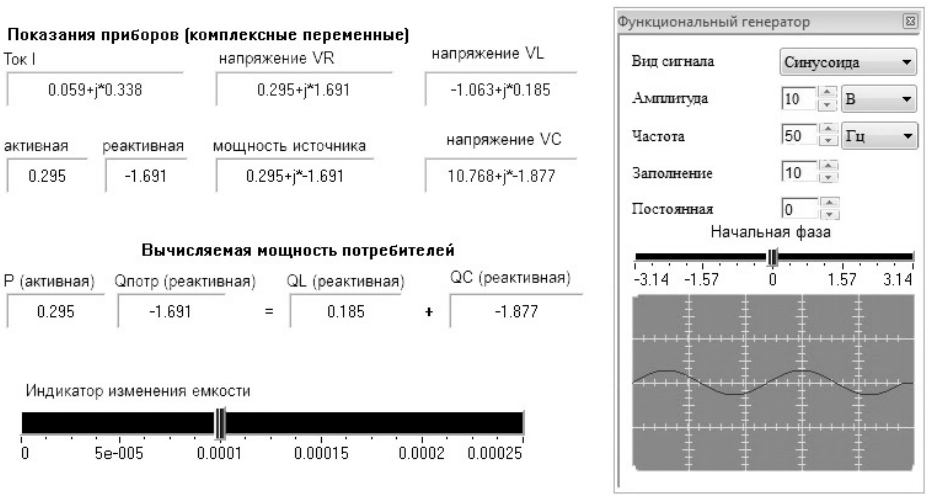


Рисунок А.22 – Визуальный слой УИМ

При изучении теоретических материалов студент может запустить данный УИМ. Познакомившись с исходной схемой анализируемой цепи, студент переключается в визуальный слой, где имеет возможность наблюдать за ходом эксперимента, т.е. видеть результаты математического расчета мощностей (активной и реактивной) как для источника электродвижущей силы, так и для потребителей.

В окне визуального слоя (см. выше) выводятся вычисляемые мощности потребителей («*P* (активная)», выделяемая на резисторе *R*, «$Q_{\text{потр}}$ (реактивная)», выделяемая на реактивных элементах цепи), а также показания приборов в виде комплексные переменных: «Ток *I*», «напряжение *VR*», «напряжение *VL*», «напряжение *VC*», «мощность источника» (последнее выводится как в комплексном виде, так и отдельно в виде активной и реактивной составляющих).

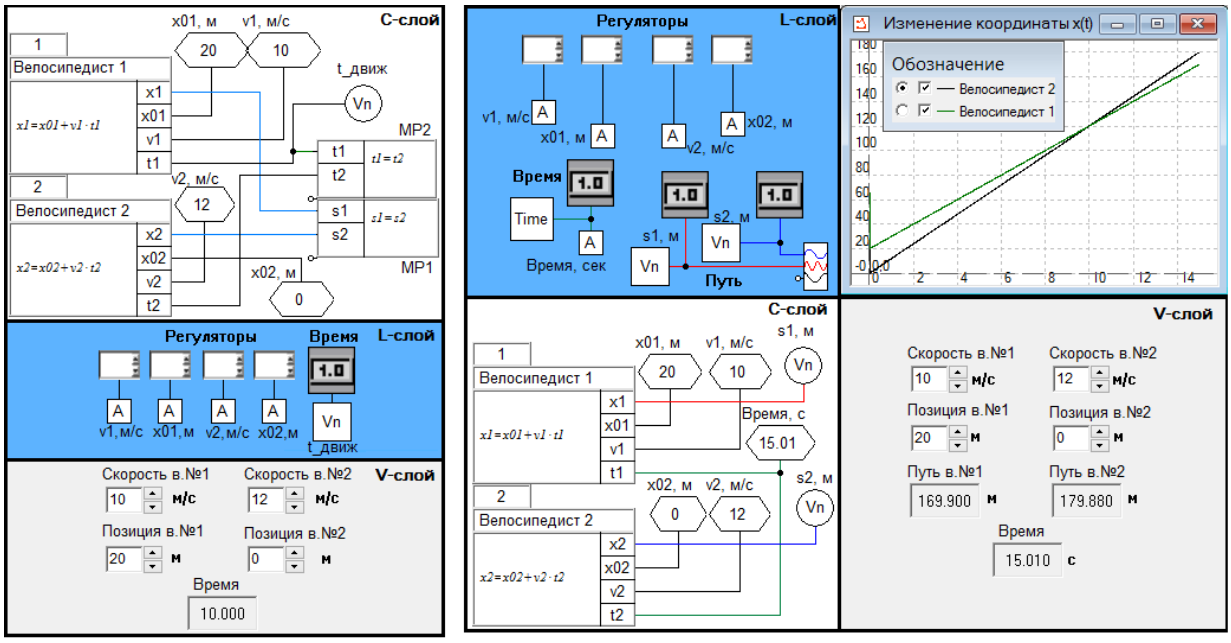
В ходе эксперимента студент может изменять значение элемента ёмкость (элемент *C_VAR*) перемещением движка индикатора изменения емкости, а также значения амплитуды и частоты переменного тока с помощью лицевой панели функционального генератора. Изменяя любые значения, студент непосредственно наблюдает баланс между мощностями источника электродвижущей силы и потребителями, а также зависимости активной и реактивной мощностей от этих изменений.

4.2 Статические иллюстративные модели

В статических моделях ФТЗ и ФЗ параметр «время» либо является локальным параметром объектной модели, либо вовсе отсутствует. Динамические модели ФТЗ представляют собой системы, к которым применимо понятие «состояние» как совокупность параметров в данный момент времени. На статических моделях учащиеся могут исследовать зависимости одной величины от другой, на динамических же появляется возможность гибкой настройки проведения эксперимента (а также его модификации в том числе для решения оптимизационных задач) и исследования изменения величин в динамике. Для иллюстрации понятия динамической системы и различий в

процедуре создания статических и динамических моделей студентам в рамках дисциплин «Компьютерное моделирование физических задач» и «Основы компьютерного моделирования физико-технических задач» предлагается производить преобразование созданной ими статической модели в динамическую.

Рассмотрим следующую задачу: «В велосипедной гонке между двумя велосипедистами расстояние в момент начала наблюдения равно 20 м. Первый имеет скорость 36 км/ч, второй – 12 м/с. Определите, через какое время второй велосипедист догонит первого?» [Филиппов 2007]. На рис. А.23 представлена статическая (а) и динамическая (б) модели задачи в СМ МАРС. Динамическая модель позволяет реализовать интерактивный эксперимент в реальном времени, результаты которого визуализируются на графике, соответственно в модели возникают новые переменные: пройденное расстояние 1 и 2.



а) б)

Рисунок А.23 – Иллюстрация перехода от статической задачи к динамической
а – статический вариант, б – динамический

В рамках названных выше дисциплин студенты строят статические модели задач, затем преобразуют полученную статическую модель в динамическую, после чего изменяют имеющиеся выражения в объектных моделях на другие, иллюстрирующие приращение величин параметров в каждый промежуток времени (например, в виде системы дифференциальных уравнений). По

окончании работы студенты производят сравнение результатов моделирования. Помимо основного задания студентам даются дополнительные, связанные с обработкой результатов моделирования (например, определение скорости на некотором участке по графику $s(t)$) или с модификацией самой модели (например, изменение характера движения с равномерного на равноускоренное).

Предложенный подход направлен, с одной стороны на развитие навыков моделирования (анализ и декомпозиция условий, синтез модели), с другой стороны – на закрепление знаний в области физики и приобретение общематематических и общенаучных знаний.

Рассмотрим ещё одну статическую задачу [Кочергин 2017] из раздела физики, представленную в текстовой форме: «Человек массой m_1 кг бежит со скоростью v_1 км/ч, догоняет тележку массой m_2 кг, движущуюся со скоростью v_2 км/ч, и вскакивает на неё. Найти, с какой скоростью будет двигаться тележка». На рис. 2 представлено отображение многоуровневой компьютерной модели этой задачи на V-слое.

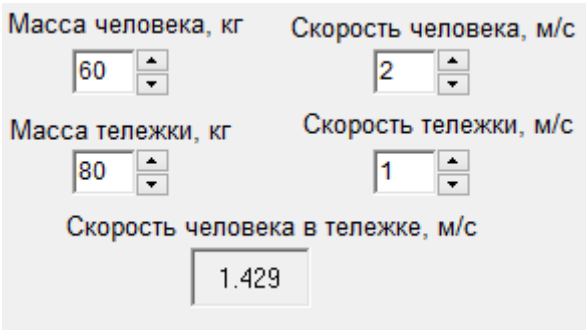


Рисунок А.24 – Модель на V-слое

На этом слое пользователь вводит и варьирует исходные данные: m_1, v_1, m_2, v_2 – в ответ ему отображается рассчитанная скорость v_3 человека в тележке после запрыгивания. На рис. А.25 изображено отображение модели на L-слое.

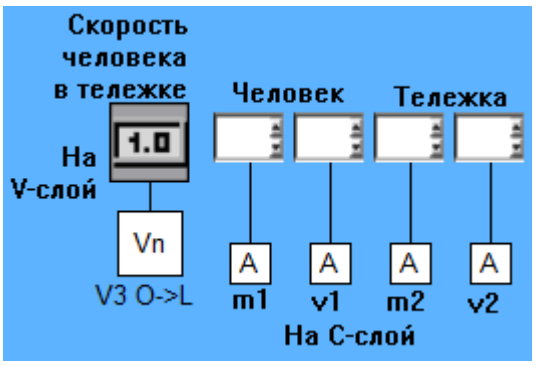


Рисунок А.25 – Вид статической модели на L-слое

В данной задаче этот слой используется только для передачи исходных данных с V-слоя на C-слой и рассчитанного значения v_3 обратно с C-слоя на V-слой. На рис. А.26 изображён вид модели на C-слое.

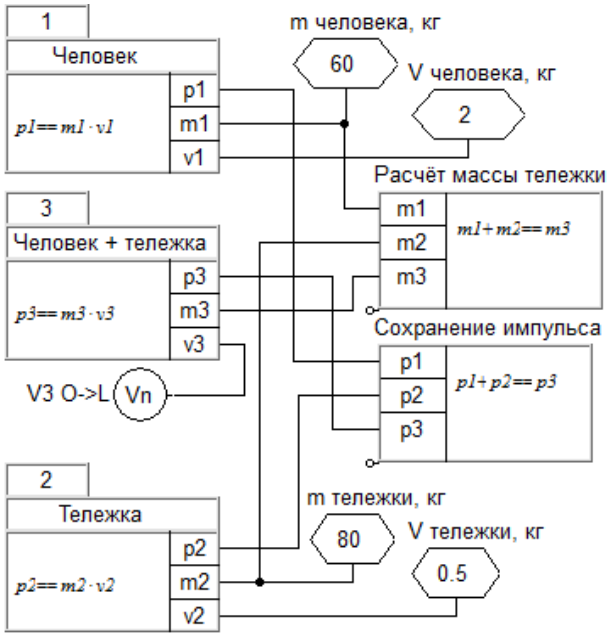


Рисунок А.26 – Модель статической задачи на C-слое

На C-слое можно увидеть 3 объектные модели, соответствующие объектам «человек», «тележка» и системе объектов «человек в тележке», а также 2 модели отношений для описания закона сохранения импульса и расчёта массы системы объектов. При поступлении значений переменных с L-слоя производится расчёт модели и результат передаётся обратно на L-слой через измеритель «V3 O->L».

4.3 Иллюстративная модель задачи гидрополива

Моделирование гидравлических систем является актуальным не только для целей функционального проектирования (проверки работоспособности проектируемой системы на её модели до начала физического конструирования самой системы), но и для образовательных целей – обучения методам компьютерного моделирования и углубления знаний в предметной области (гидравлике).

Рассмотрим многоуровневое представление следующей общей задачи: «Пусть имеется система подачи воды потребителю, включающая в себя бак заданного объёма, по одной трубе в который под заданным постоянным давлением поступает вода, по другой она подаётся потребителю. Подача и поступление воды регулируются клапанами. Необходимо реализовать некоторый заданный алгоритм управления наполнением бака или входными/выходными потоками воды». КЦ описанного механизма подачи воды, выстраиваемая на объектном слое, представлена на рис. А.27.

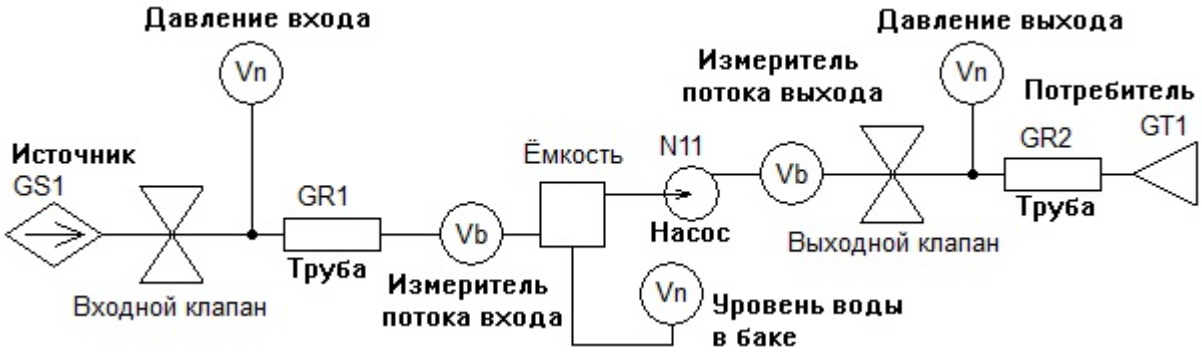


Рисунок А.27 – КЦ системы гидрополива на С-слое

Компоненты данной КЦ соответствуют элементам, составляющим «реальную» (физическую) систему подачи воды: ёмкость (бак), трубы подачи воды (гидрорезисторы), клапаны, источник и потребитель воды (гидротерминатор) [Ганджа 2017]. Также на объектном слое присутствуют измерители давления и объёмного расхода воды, которые будут передавать данные (значения) на L-слой. На рис. А.28 представлен вид модели на визуальном слое.

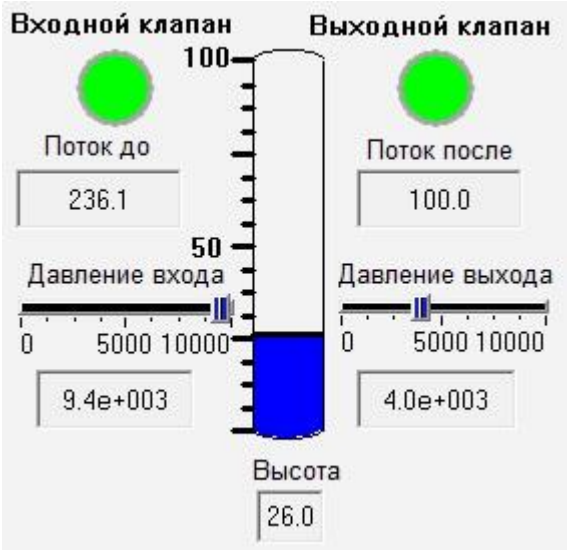


Рисунок А.28 – Вид УИМ задачи гидрополива на V-слое

На визуальном слое отображаются регуляторы для задания варьируемых данных (давление поступающей в бак воды – давление входа, давление воды отпускаемой потребителю – выходное давление), а также на цифровых табло отображаются значения выходных данных: высоты воды в баке, объёмного расхода передаваемой воды. Кроме того, на этом слое расположены переключатели состояний клапанов, которые также отображают их текущее состояние (зелёный цвет индикатора означает, что клапан открыт, красный – закрыт). На рис. А.29 представлена алгоритмическая КЦ L-слоя с реализованным алгоритмом: «При переполнении бака закрыть входной клапан, а при опустошении – выходной, открыв при этом входной».

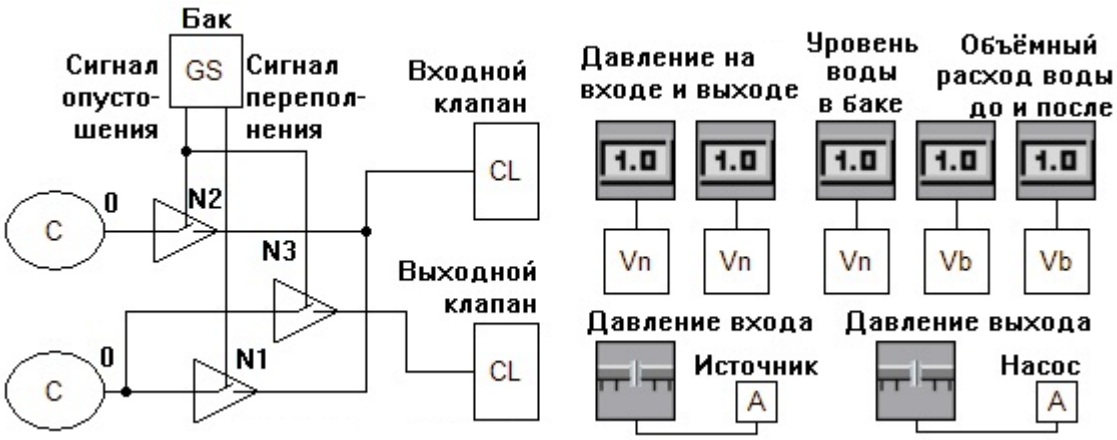


Рисунок А.29 – КЦ задачи гидрополива на L-слое (вариант 1)

Для реализации указанного алгоритма используются специальные алгоритмические компоненты – накопители N_1, N_2, N_3, которые передают

хранимое в них значение только в случае получения ненулевого управляющего сигнала. На рассматриваемой схеме накопители хранят получаемые на вход от компонентов-источников вещественные значения, в данном случае – константы, равные 1 или 0, а в качестве управляющих сигналов используются сигналы опустошения и переполнения бака.

Вышеописанная модель системы гидрополива может использоваться как для обучения студентов, владеющих методом и инструментарием моделирования, законам и принципам гидравлики, так и для совершенствования навыков моделирования студентов, уже знакомых с разделом гидравлики. В первом случае обучение должно сопровождаться работой с преподавателем или самостоятельной работой с компьютерным учебным пособием [Филиппов, Шарова, Кураколов 2007] по гидравлике, которое может быть представлено в курсе *Moodle*, при этом компьютерная модель выступает в качестве вспомогательного учебно-иллюстративного материала, позволяющего продемонстрировать изученные студентом законы гидравлики и провести вычислительный эксперимент.

Во втором случае компьютерная модель становится основным объектом взаимодействия со студентами, работа с которым сопровождается изучением компьютерного учебного пособия по многоуровневому моделированию. Учащийся совершенствует навыки моделирования за счёт решения поставленных перед ним задач. Приведём примеры таких задач с указанием требуемых действий пользователя:

1) провести декомпозицию (формализацию) условий задачи. Для решения поставленной задачи от учащегося требуется провести первичный анализ системы, её декомпозицию на составляющие элементы с целью дальнейшего формирования компьютерной модели;

2) сформировать сценарий (алгоритм) управления моделируемой системой, например, разработать механизм автоматической регуляции входного давления для: а) поддержания заданного уровня воды в баке, б) поддержания

заданной величины выходного давления. Для решения задач такого рода учащемуся необходимо модифицировать схему управления объектом на L-слое;

3) произвести существенное усложнение модели, например, добавление дополнительного потребителя воды и индивидуальных измерителей расхода воды. Решение задач такого типа требует от учащегося изменения модели на всех трёх слоях: объектном, логическом (алгоритмическом) и визуальном.

Приведём краткий пример задачи одного из перечисленных выше типов для рассматриваемой компьютерной модели системы гидрополива и пример её решения: «Смоделировать алгоритм управления системой гидрополива, предотвращающий ситуации переполнения или опустошения бака за счёт автоматической регуляции давления входного потока при преодолении заданных (регулируемых) критических значений уровня воды в баке. Реализовать работу системы в условиях ограничения входного потока». Целью решения данной задачи является развитие навыков алгоритмизации и компьютерного моделирования, при этом её прикладной характер способствует вовлечению студента в изучение предметной области – гидравлики. В качестве одного из вариантов решения описанной задачи приведём на рис. А.30 изменённую алгоритмическую КЦ исходной компьютерной модели системы гидрополива, описанной ранее.

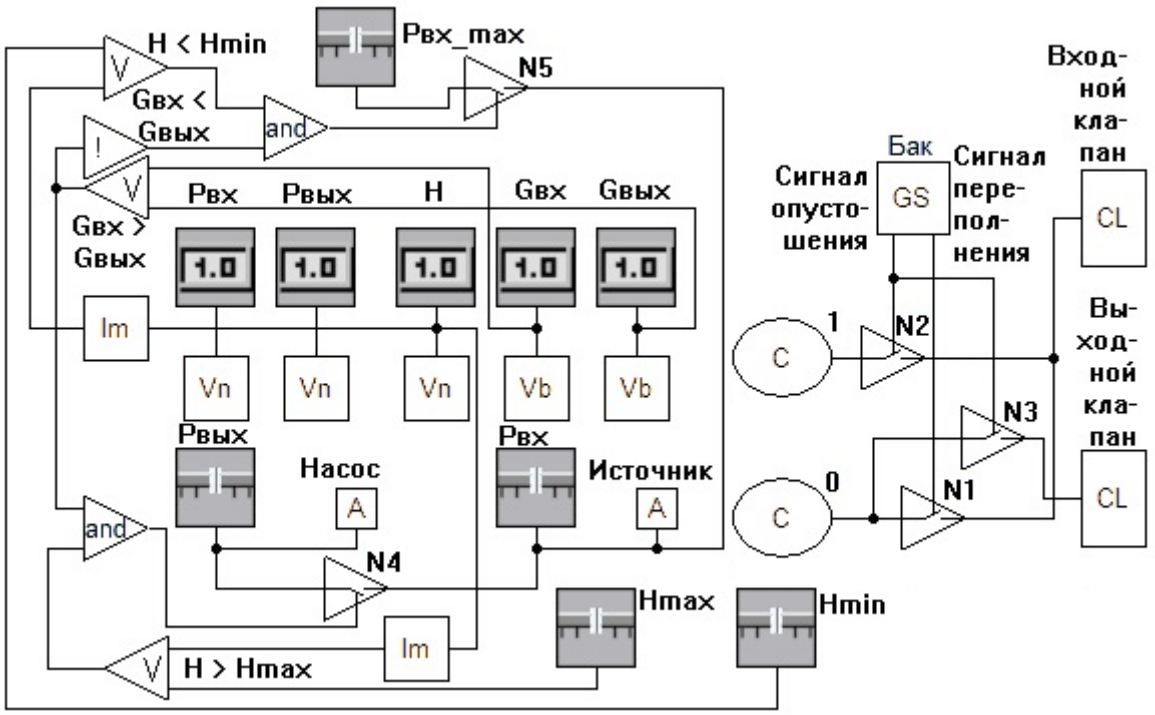


Рисунок А.30 – КЦ задачи гидрополива на L-слое (вариант 2)

Для решения подобной задачи учащемуся необходимо провести формализацию её постановки с целью получения чёткого алгоритма её решения, в том числе определить новые вводимые переменные и их связь с уже существующими. Для рассматриваемого случая новыми переменными являются H_{max}, H_{min} – максимально и минимально допустимые значения высоты наполнения бака, $P_{вх\ max}$ – максимально допустимое давление входного потока воды. Введение новых переменных также влечёт за собой необходимость добавления соответствующих им компонентов-регуляторов или измерителей на логическом слое, которые автоматически будут отображены на визуальном. Требуемый алгоритм для управления системой гидрополива может быть таким: «Если объём поступающей в бак воды $G_{вх}$ меньше отпускаемого потребителю $G_{вых}$ (т.е. бак наполняется) и уровень H воды в баке больше максимально допустимого значения H_{max}, то понизить давление $P_{вх}$ поступающей в бак воды до значения $P_{вых}$ – давления отпускаемой потребителю воды. Если первое условие предыдущего правила не выполняется и уровень H воды в баке меньше минимально допустимого значения H_{min}, то повысить давление $P_{вх}$ поступающей

в бак воды до значения $P_{вх\ max}$ – максимально допустимого давления поступающей в бак воды (технического ограничения системы)».

После составления алгоритма работы управляющего устройства учащийся осуществляет выбор компонентов L-слоя для реализации этого алгоритма и производит формирование алгоритмической КЦ в выбранном базисе. Рассматриваемая КЦ реализована за счёт компонентов групп «Логические операторы» («Конъюнкция» и «Инверсия»), «Операторы сравнения» («Больше», «Меньше») и компонентов «Накопитель».

По окончании формирования КЦ учащийся должен произвести проверку составленной модели на адекватность.

В качестве критериев оценивания решения на компьютерной модели поставленных задач можно использовать следующие:

1) корректность анализа описания поставленной перед учащимся задачи или описания системы (процесса), предлагаемой для моделирования,

2) адекватность составленной учащимся модели процесса или системы (насколько корректно составленная модель описывает объективную действительность),

3) корректность составленного алгоритма решения (насколько предложенный пользователем алгоритм решения способствует достижению поставленной цели),

4) корректность сформированной КЦ (правильность сбора КЦ, обеспечивающая её корректную работу и функциональность),

5) обоснованность выбора компонентов (достаточность и неизбыточность компонентного состава модели).

5 Модели задач с управляющими сценариями

В данном параграфе представлены примеры моделей, иллюстрирующие применение разработанных компонентов (таких как, например, диаграммы состояний) для создания сценариев вычислительного эксперимента, экспорта результатов моделирования в файлы для дальнейшей обработки другими

программными средствами, параметризации моделей данными из файлов на локальном компьютере, решения оптимизационных задач и обработки табличных результатов моделирования.

5.1 Экспорт результатов моделирования

Одним из важных критериев универсальности СМ является возможность взаимодействовать с внешним программным обеспечением: структурированными файлами данных, другими СМ, программами визуализации данных и пр. СМ, имеющая такую возможность, становится открытой для взаимодействия с «внешней средой», что расширяет возможности среды. СМ, не имеющая такой возможности, является закрытой, и ценность такой среды, несомненно, ниже. Ещё большей ценностью обладают СМ способные взаимодействовать с моделями (блоками), созданными в других средах – обмениваться с ними данными. Реализация такой возможности является перспективой данного исследования. На данный момент реализована возможность импорта исходных данных модели (параметров модели, начальных значений переменных) и экспорта результатов моделирования. Для этих целей были разработаны компоненты «Запись в csv-файл» и «Чтение из csv-файла», осуществляющие чтение/запись в файлы формата csv – файлы с разделителями, поддерживаемые различными программами, в том числе Excel [Excel]. Использование таких компонентов позволяет, в том числе, оценивать корректность моделей в СМ МАРС с аналогичными моделями в других СМ.

На рис. А.31представлен фрагмент КЦ, иллюстрирующей запись результатов моделирования полёта тела.

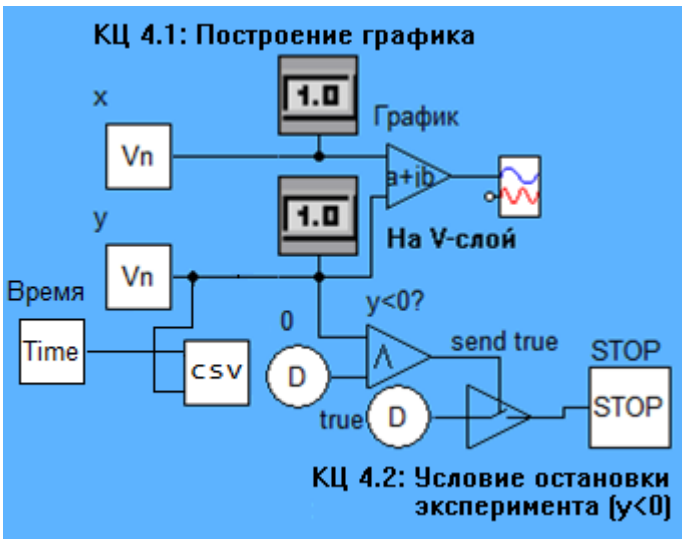
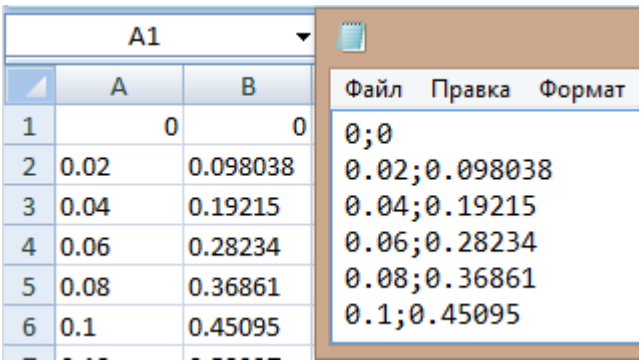


Рисунок А.31 – КЦ для записи данных в файл

На рис. А.32 представлено содержимое файла, данные в который записаны разработанным компонентом.



A1

	A	B
1	0	0
2	0.02	0.098038
3	0.04	0.19215
4	0.06	0.28234
5	0.08	0.36861
6	0.1	0.45095

Файл Правка Формат

```
0;0
0.02;0.098038
0.04;0.19215
0.06;0.28234
0.08;0.36861
0.1;0.45095
```

Рисунок А.32 – Содержимое файла при открытии в Excel и Notepad

5.2 Параметризация компьютерных моделей

Для параметризации компьютерных моделей, наряду с компонентами для взаимодействия с базами данных [Панов, Григорьева, Кочергин 2018], могут использоваться компоненты «Чтение из *csv*-файла». В таком случае пользователю не требуется знания языка *MySQL* для создания запросов к базе данных – данные, считываемые компонентом из *csv*-файла передаются на компонент-мультиплексор, который передаёт в КЦ многоуровневой модели значение, соответствующее пункту из выпадающего списка. Таким образом может осуществляться параметризация многоуровневых компьютерных моделей, например, параметризация модели насоса нефтедобывающей скважины, в которой такие параметры как глубина спуска насоса, площадь

сечения плунжера и пр. могут передаваться в модель из внешнего файла (на рис. А.33 представлен фрагмент КЦ из задачи моделирования усилий на полированном штоке, рассмотренной в п. 4.2.1).

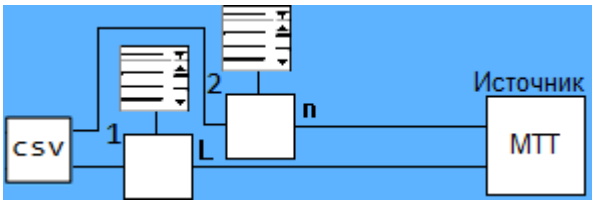


Рисунок А.33 – Параметрирование частоты откачивания и длины штока из csv-файла

Массивы значений, полученные из *csv*-файла (путь к которому задаётся в параметрах компонента) передаются на мультимплексоры «1» и «2», которые передают в компонент «Источник» те, значения которые выбраны пользователем на V-слое в графических отображениях компонентов с выпадающим списком.

ПРИЛОЖЕНИЕ Б. ДИАГРАММЫ КЛАССОВ КОМПЛЕКСА ПРОГРАММ

1 Библиотека моделей компонентов для моделирования физико-технических задач

1.1 Компоненты для моделирования гибридного поведения

Охарактеризуем классы компонентов для моделирования гибридного поведения:

1) компонент «Состояние» реализован в классе *CStateChart* (рис. Б.1), наследующем методы от классов *IRedactComponent*, *ISerialize*, *IMessengerComponent*, *IBaseImitationComponent*, *IInteractivePanel*, описанных в [Ганджа 2017]. Класс содержит следующие поля:

- *S* – вектор значений переменных, подаваемых на входы компонента слева и участвующих в проверке логического условия, выполнение которого инициирует переход в новое состояние;
- *Start* – маркёр начала работы компонента; при истинности данной переменной компонент начинает работу;
- *End* – маркёр завершения работы компонента; при истинности данной переменной компонент завершает работу и присваивает переменной Start значение false;
- *Param* – вектор значений переменных, передаваемых на выход Out компонента снизу, во время работы компонента;
- *WeHaveGotIt* – массив маркёров состояния переменных S; принимает истинное значение в случае присваивания соответствующей переменной S некоторого значения;
- *WeHaveGotItAll* – маркёр состояния готовности компонента к проверке логического условия, инициирующего переход; становится истинным при получении значения всех переменных вектора *S*;
- *GotParameter* – маркёр состояния соответствующей переменной из вектора *Param*; становится истинным при получении переменной из вектора

Param некоторого значения и свидетельствует о готовности передачи значений на выход out компонента.

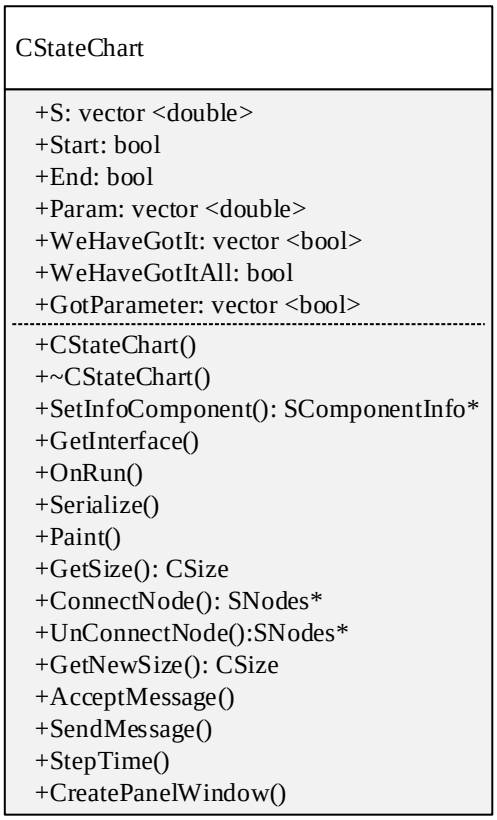


Рисунок Б.1 – Диаграмма класса компонентов CStateChart

На рис. Б.1и далее диаграммы классов представлены в нотациях UML: в верхней части расположено имя класса, ниже – поля (атрибуты) класса, в самом низу – методы класса (в терминах объектно-ориентированного программирования). Слева от полей и методов расположено условное обозначение видимости членов класса: публичный – *Public* (+), приватный – *Private* (-), защищённый – *Protected* (#), производный – *Derived* (/), пакет – *Package* (~).

Методами класса *CStateChart* являются следующие унаследованные им методы:

- *CStateChart*() – конструктор компонента, предназначенный для его инициализации при добавлении на схему в редакторе СМ МАРС;
- ~CStateChart() – деструктор компонента, предназначенный для уничтожения компонента при его удалении из схемы в редакторе СМ МАРС;

- *SetInfoComponent*() – метод для предоставления информации о компоненте в его программной реализации; возвращает уникальный идентификатор компонента, его отображаемое в дереве компонентов имя, имя каталога, наименование модуля библиотеки моделей компонентов;
- *GetInterface*() – метод для запроса интерфейса компонента;
- *OnRun*() – метод, выполняемый при запуске МКЦ задачи на выполнение;
- *Serialize*() – функция сериализации, требуемая для компонентов, выполняющих чтение или запись данных при сохранении или открытии схемы модели;
- *Paint*() – метод для реализации динамического изображения компонента (при редактировании схемы или при анализе модели) в многослойном редакторе СМ МАРС;
- *GetSize*() – метод, возвращающий значение размера компонента (в пикселях) на выбранном слое редактора;
- *ConnectNode*() – метод, осуществляющий обработку события присоединения связи к узлу компонента;
- *UnConnectNode*() – метод, осуществляющий обработку события отсоединения связи от узла компонента;
- *GetNewSize*() – метод, устанавливающий размер компонента после добавления к нему узла;
- *AcceptMessage*() – метод приёма и обработки сообщений, декларирующий порядок формирования сообщения из данных, хранящихся в его параметрах и (или) полученных из ранее принятых сообщений, для передачи [Ганджа 2017];
- *SendMessage*() – метод для передачи сообщения из данных, доступных компоненту (полученных им ранее из сообщений или хранящихся в его параметрах);

- *StepTime*() – метод алгоритмического компонента (т.е. класса *IMessengerComponent*), выполняемый перед очередной итерацией анализа МКЦ модели;
- *CreatePanelWindow*() – метод, выполняющий открытие окна редактирования условия, инициирующего переход в другое состояние;

Алгоритм взаимодействия компонентов с вычислительным (C-слой) и имитационным (L-слой) ядрами подробно описан в [Ганджа 2017: с. 268-278], поэтому кратко опишем порядок выполнения методов во время автоматического анализа модели СМ МАРС: 1) *OnRun*() – при запуске, 2) *OnRunExperiment*() – в начале итерации, 3) методы взаимодействия компонента с вычислительным ядром (очередная итерация анализа модели), 4) *OnStopExperiment*() – в конце очередного анализа, 5) *StepTime*() – перед следующей итерацией, 6) далее с п. 2. На этапе 5 посредством исполнения метода *SendMessage*() одного компонента (отправителя) инициируется исполнение метода *AcceptMessage*() другим компонентом (получателем). В связи с этим основной алгоритм функционирования компонента класса *CStateChart* заложен в методе *AcceptMessage*() – (рис. Б.2).

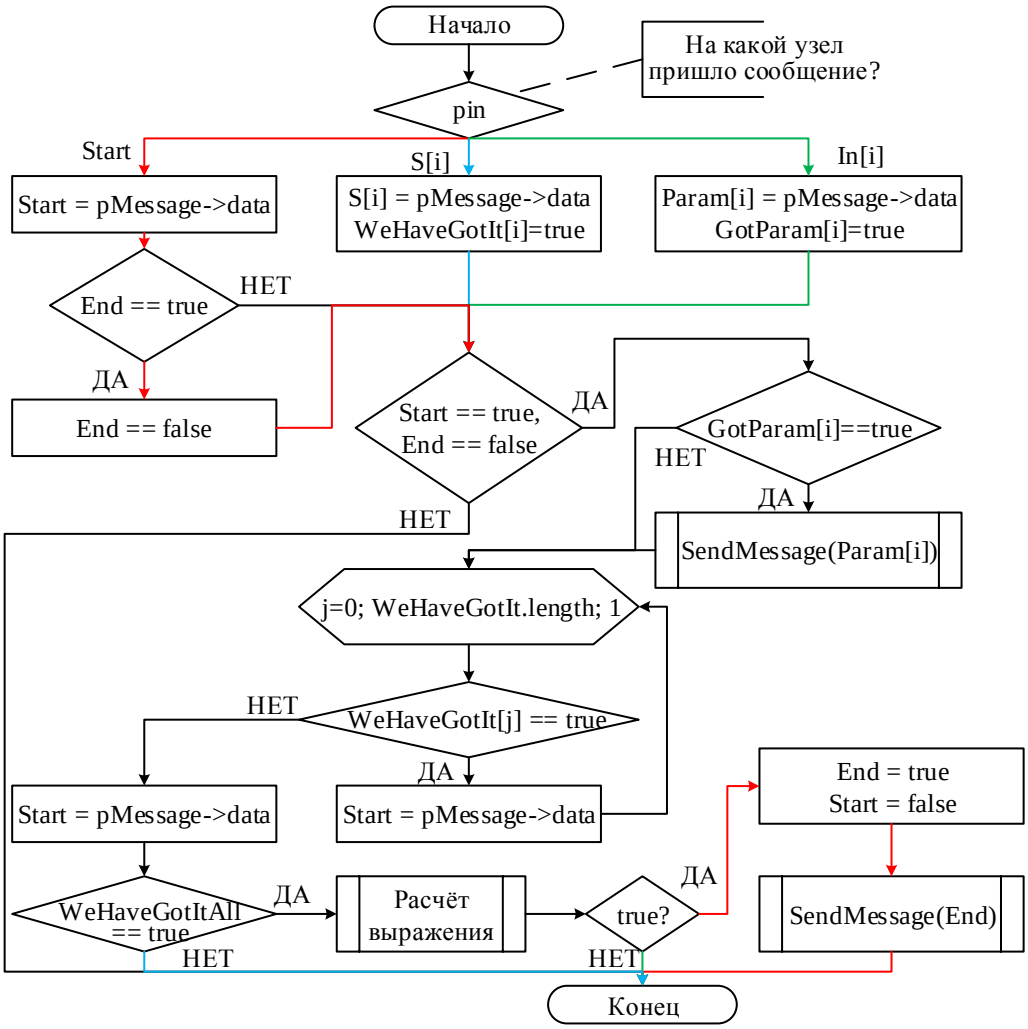


Рисунок Б.2 – Алгоритм функционирования компонента «Состояние»

Подпрограмма «Расчёт выражения» выполняет расчёт выражения, вложенного в данный компонент согласно алгоритму, предложенному в работе [Ганджа 2005], для проведения расчётов в интерактивных математических панелях, описанных в [Ганджа 2017: с. 286-289].

2) компонент «Начальное состояние» реализован в классе *CStateStart* (рис. Б.3), наследующем методы от классов *IRedactComponent, IMessengerComponent*. Класс содержит следующие поля:

- *S* – параметр компонента, хранящий значение *true* для передачи в сообщении;
- *bRun* – маркёр выполнения анализа схемы.

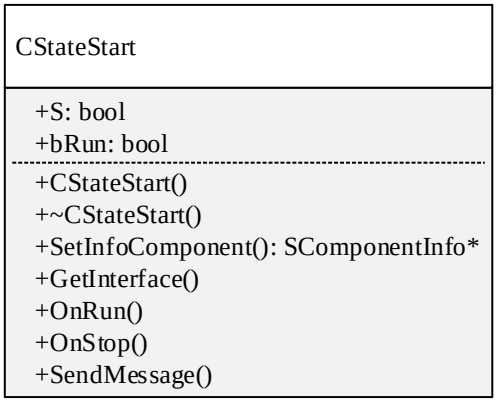


Рисунок Б.3 – Диаграмма класса компонентов CStateStart

Данный компонент осуществляет единственное действие – в начале анализа схемы модели отправляет сообщение на свой выход, тем самым запуская соединённый с ним компонент «Состояние».

Методы класса *CStateStart* являются наследуемыми, поэтому опишем только неописанные ранее:

- *OnStop*() – метод, вызываемый при окончании вычислительного эксперимента. При его вызове маркёру *bRun* присваивается значение *false*, которое изменится только при следующем запуске модели.

3) компонент «Конечное состояние» реализован в классе *CStateEnd* (рис. Б.4), наследующем методы от классов *IRedactComponent*, *IMessengerComponent*, *ICalculateControlsInterface*. Класс содержит поле *St* – маркёр состояния компонента; при получении значения *true* на свой единственный вход, компонент останавливает вычислительный эксперимент.

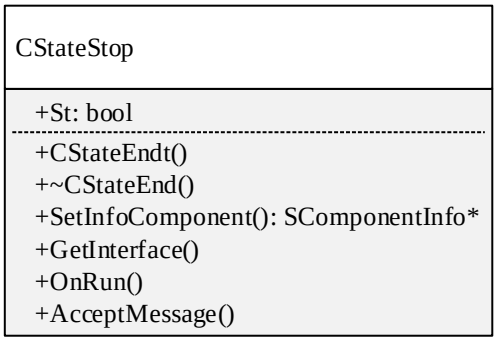


Рисунок Б.4 – Диаграмма класса компонентов CStateEnd

Наследуемые методы класса *CStateEnd* являются типовыми и были описаны выше. Компоненты данного класса предназначены для остановки

вычислительного эксперимента, тем самым реализуют конечное состояние гибридной системы.

4) компонент «Событие» реализован в классе CStateStart (рис. Б.5), наследующем методы от классов *IRedactComponent*, *ISerialize*, *IMessengerComponent*, *IBaseImitationComponent*, *IInteractivePanel*. Класс содержит следующие поля:

- *Start* – маркёр начала работы компонента; при истинности данной переменной компонент начинает работу;
- *End* – маркёр завершения работы компоненты; при истинности данной переменной компонент завершает работу и присваивает переменной Start значение false;
- *ready4End* – маркёр готовности компонента к завершению работы; ему присваивается значение *true* после однократной передачи значения на выход *Out*;
- *Param* – вектор значений переменных, передаваемых на выход Out компонента снизу, во время работы компонента;
- *WeHaveGotIt* – массив маркёров состояния переменных S; принимает истинное значение в случае присваивания соответствующей переменной S некоторого значения;
- *WeHaveGotItAll* – маркёр состояния готовности компонента к проверки логического условия, инициирующего переход; становится истинными при получении значения всех переменных вектора *S*;
- *GotParameter* – маркёр состояния соответствующей переменной из вектора *Param*; становится истинным при получении переменной из вектора *Param* некоторого значения и свидетельствует о готовности передачи значений на выход out компонента.

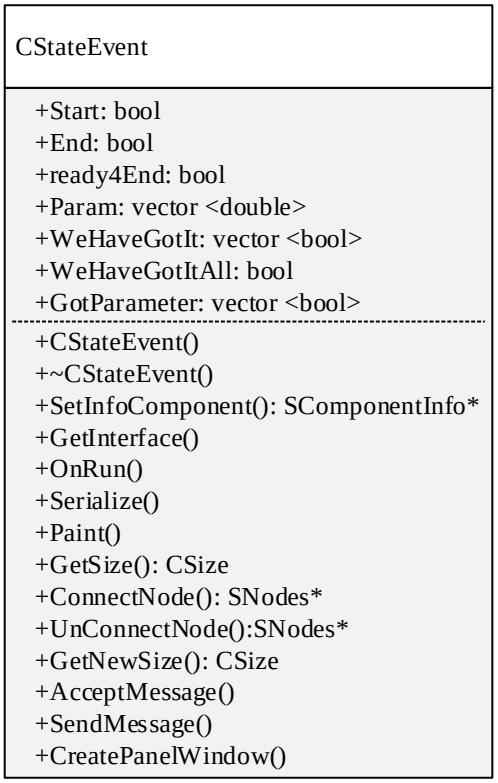


Рисунок Б.5 – Диаграмма класса компонентов CStateEvent

Как видно из диаграммы классов компоненты класса *CStateEvent* имеют функционал схожий с компонентами класса *CStateChart* – отличие заключается в том, что алгоритм компонентов *CStateEvent* выполняется однократно (т.е. имеем «мгновенное состояние» – которое после выполнения сразу же завершается; по аналогии с *exit*-активностью в диаграммах состояний *UML*), в то время как компоненты класса *CStateChart* продолжают работу до тех пор, пока вложенное в них условие не станет истинным.

Основной алгоритм функционирования компонента класса *CStateEvent* заложен в методе *AcceptMessage*, поэтому приведём блок-схему только этого метода (рис. Б.6).

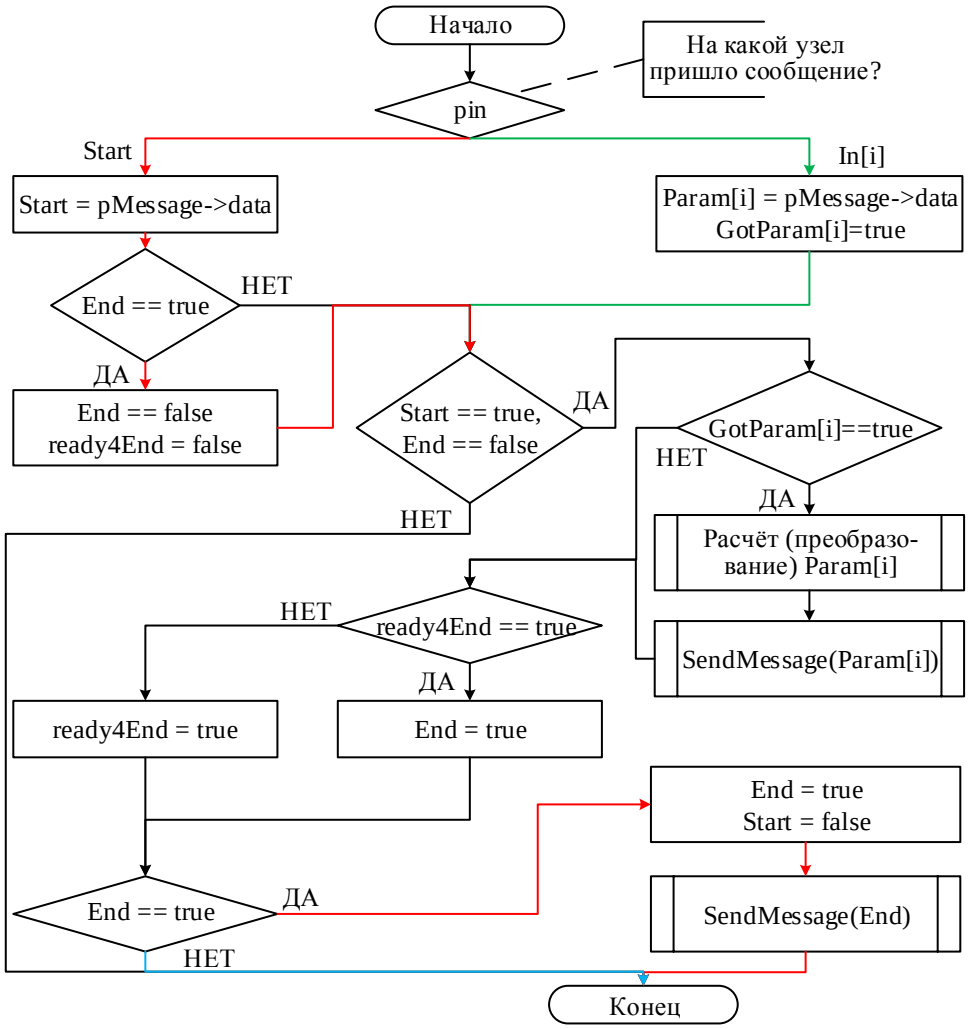


Рисунок Б.6 – Алгоритм работы компонента класса CStateEvent

1.2 Компоненты для отображения геометрических свойств в моделях

В соответствии с предлагаемым подходом отображения геометрических свойств в моделях (см. пункт 2.3.3 работы) разработаны следующие группы геометрических компонентов:

1) *геометрические примитивы* – наборы базовых геометрических фигур,

2) *геометрические преобразователи* – компоненты для выполнения геометрических преобразований,

3) *геометрические решатели* – компоненты для решения базовых геометрических задач,

4) *кривые* – компоненты, задающие профили поверхностей.

На рис. Б.7 представлены некоторые примеры геометрических компонентов.

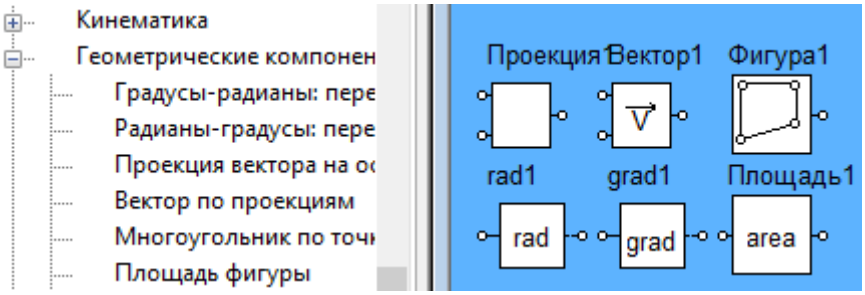


Рисунок Б.7 – Пример геометрических компонентов

Охарактеризуем классы компонентов первой группы – геометрические примитивы.

1) компонент «Многоугольник по точкам» реализован в классе *CPointPolygon* (рис. Б.8), наследующем методы от классов *IRedactComponent*, *IMessengerComponent*. Класс содержит следующие поля:

- *isRun* – маркёр выполнения анализа схемы;
- *polygonData* – массив точек многоугольника;
- *typeConnection* – тип соединения точек, определяемый степенью соединяющего их полинома;
- *xString*, *yString* – строковые переменные для ввода последовательности точек пользователем в параметры компонента;
- *buf* – вспомогательная переменная для парсинга строковых переменных *xString* и *yString* для перевода строковых данных в вещественные.

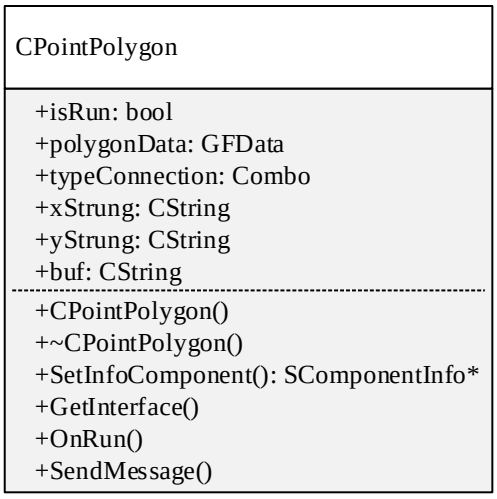


Рисунок Б.8 – Диаграмма класса компонентов CPointPolygon

Тип данных *GFData* представляет собой структуру из двух массивов *x*, *y* и значения *type* – тип соединения, что позволят оперировать объектом типа

«фигура» в механизме обмена сообщениями СМ МАРС. Таким образом другие компоненты, получающие сообщение от компонента «Многоугольник по точкам» получают информацию о виде геометрической фигуры, что даёт возможность автоматизации выбора соответствующей формулы (например, при вычислении площади фигуры.

Методы данного класса являются наследуемыми, их типовое описание было представлено ранее. Компоненты класса *CPointPolygon* представляют собой источники геометрических фигур, которые передают вначале процедуры анализа модели (метод *OnRun*()) объект типа «фигура» для дальнейших расчётов (например, для поиска точки пересечения фигур).

2) компонент «Площадь фигуры» реализован в классе *CFigureArea* (рис. Б.9), наследующем методы от классов *IRedactComponent, IMessengerComponent*. Компонент содержит следующие поля:

- *area* – переменная для хранения рассчитанного значения площади фигуры;
- *polygonData* – переменная, хранящая полученные на вход данные о фигуре.

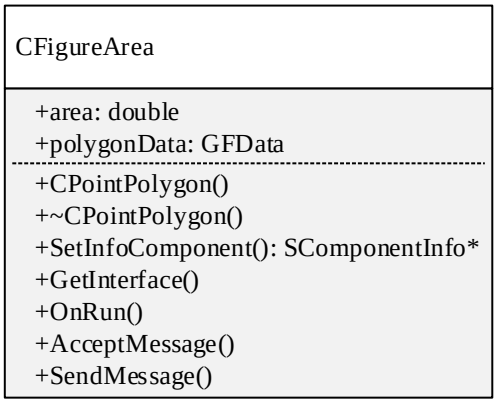


Рисунок Б.9 – Диаграмма класса компонентов CFigureArea

Данный компонент рассчитывает площадь фигуры, принимая на вход (слева) данные типа *GFData*, полученные от компонента-источника геометрической фигуры, затем передаёт их через свой выход (справа). Остальные компоненты этой группы (периметр фигуры, длина n-й стороны и др.) являются аналогичными рассмотренным, поэтому ограничимся описанием их

математической модели, представленным в пункт 2.3.1 и перейдём к описанию следующего блока компонентов.

Компоненты *геометрические преобразователи*:

1) Компоненты «Перевод из градусов в радианы» и «Перевод из радиан в градусы» реализованы в классах *CGradRad* и *CRadGrad* (рис. Б.10) соответственно, наследующих методы от классов *IRedactComponent*, *IMessengerComponent*. Компоненты содержит следующие поля:

- *grad* – переменная для хранения значения угла в градусах;
- *rad* – переменная для хранения значения угла в радианах.

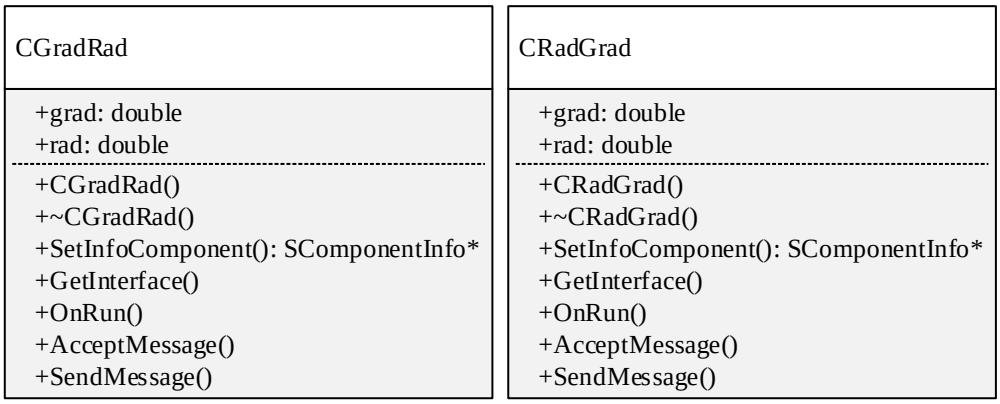


Рисунок Б.10 – Диаграммы классов CGradRad и CRadGrad

2) Компоненты «Расчёт проекции вектора на ось» и «Расчёт вектора по его проекциям» реализованы в классах *CVectorProjection* и *CVectorCalculate* (рис. Б.11), наследующих методы от классов *IRedactComponent*, *IMessengerComponent*.

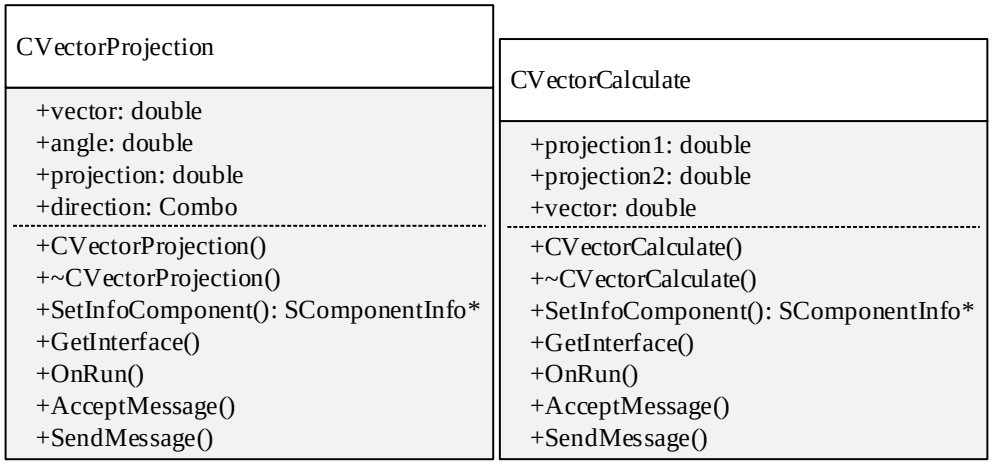


Рисунок Б.11 – Диаграммы классов CVectorProjection и CVectorCalculate

Компоненты «Расчёт проекции вектора на ось» содержит следующие поля:

- *vector* – значение длины вектора, подаваемое на вход компонента (слева) для расчёта его проекции;
- *angle* – угол между вектором и осью OX (рад);
- *projection* – рассчитанное значение проекции вектора на ось;
- *direction* – параметр, определяющий ось для расчёта проекции; значения выбирается пользователем из выпадающего списка в свойствах компонента.

Компоненты «Расчёт вектора по его проекциям» содержит следующие поля:

- *projection1* – проекция вектора на одну из осей;
- *projection2* – проекция вектора на другую ось;
- vector – рассчитанная длина вектора;

Остальные классы компонентов этой группы (например, компонента «Векторное произведение») аналогичны рассмотренным, поэтому ограничимся описанием их математических моделей, представленных в пункте 2.3.3 и перейдём к описанию следующего блока компонентов.

1.3 Компоненты для отображения физических свойств объектов в моделях

Физические компоненты позволяют учитывать физические свойства моделей за счёт использования типовых компонентов: 1) источники физических величин; 2) модели физических тел; 3) физические эффекты, – и представлять компьютерные модели непрерывного (физического) поведения объектов в виде цепи связных компонентов, непосредственно соответствующих взаимодействующим в задаче объектам, например, «твёрдое тело – атмосфера» или «твёрдое тело – поверхность».

На рис. Б.12представлены некоторые примеры физических компонентов.

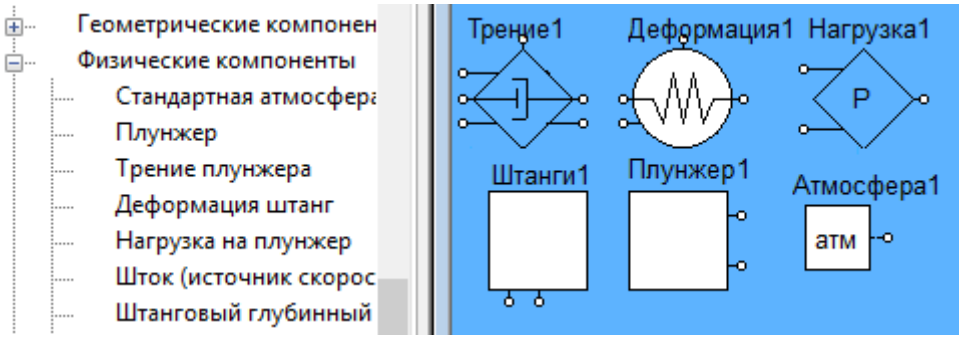


Рисунок Б.12 – Пример физических компонентов

1) компонент «Стандартная атмосфера» реализован в классе *CAtmosphereLevel* (рис. Б.13), наследующем методы от классов *IRedactComponent*, *IMessengerComponent*, *IBaseMarsComponent*. Компонент содержит следующие поля:

- *T0*, *rho0*, *G0* – значения параметров класса (температуры, плотности, ускорения свободного падения для стандартной атмосферы соответственно);
- *h* – входное значение текущей координаты тела по высоте над уровнем моря;
- константы *H1* = 11000.0, *H2* = 25000.0, *H3* = 46000.0 – представляющие собой границы уровней атмосферы;
- *rho* – возвращаемая плотность воздуха для высоты h;

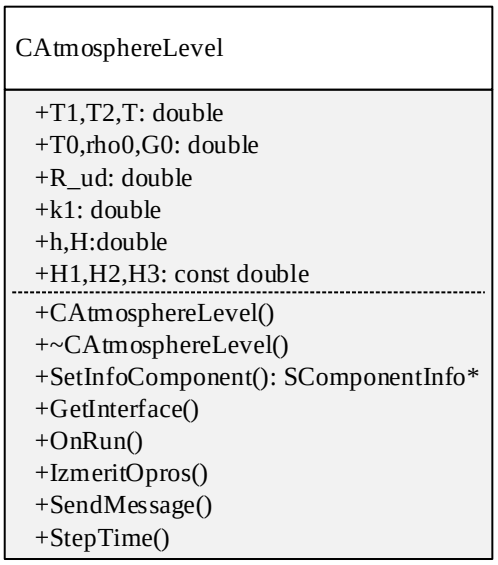


Рисунок Б.13 – Диаграмма класса CAtmosphereLevel

Компоненты класса *CAtmosphereLevel* представляют собой многоуровневые компоненты: компоненты-приёмники на С-слое, принимающие

значение при выполнении метода *IzmeritOpros*(), и компоненты-источники, передающие рассчитанное значение плотности воздуха на L-слое при выполнении метода *StepTime*().

2) компонент «Штанговая установка (плунжер)» реализован в классе *CPlunger* (рис. Б.14), наследующем методы от классов *IRedactComponent, IMessengerComponent, IBaseMarsComponent, IBaseImitationComponent*. Компонент содержит следующие поля:

- *Fpl* – площадь сечения плунжера;
- *H* – глубина спуска насоса;
- *Vf* – потоковая переменная (сумма сил *F*, воздействующих на тело);
- *Vp* – потенциальная переменная (скорость тела *V*);
- *m* – масса плунжера.

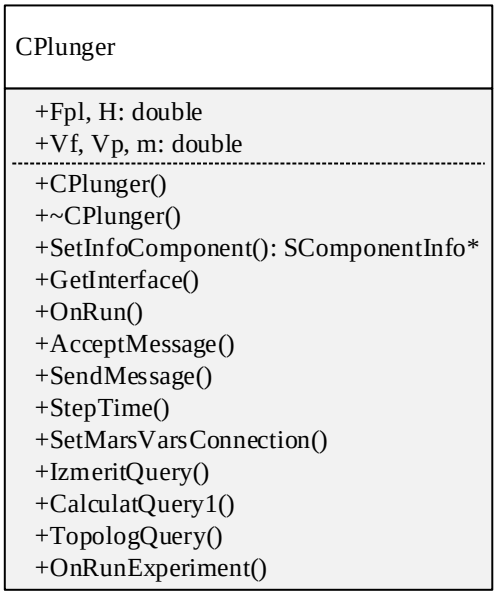


Рисунок Б.14 – Диаграмма класса CPlunger

3) компонент «Насосные штанги» реализован в классе *CSuckerRods* (рис. Б.15), наследующем методы от классов *IRedactComponent, IMessengerComponent, IBaseMarsComponent, IBaseImitationComponent*. Компонент содержит следующие поля:

- *ro_st* – плотность материала штанг;
- *f* – площадь сечения штанг;
- *Vf* – потоковая переменная (сумма сил *F*, воздействующих на тело);

- *Vp* – потенциальная переменная (скорость тела *V*);
- *m* – масса.

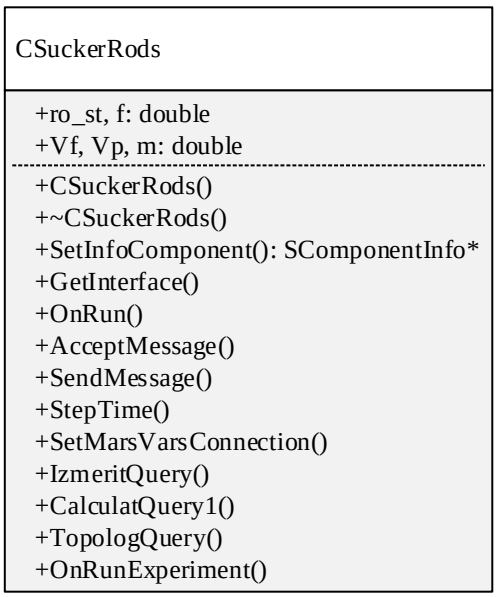


Рисунок Б.15 – Диаграмма класса CSuckerRods

4) компонент «Трение плунжера» реализован в классе *CFrictionPlunger* (рис. Б.16), наследующем методы от классов *IRedactComponent*, *IMessengerComponent*, *IBaseMarsComponent*, *IInteractivePanel*, *ISerialize*. Компонент содержит следующие поля:

- *H* – глубина спуска насоса;
- *Hd* – динамический уровень жидкости;
- *ro_g* – плотность жидкости;
- *ro_g_z* – плотность жидкости в затрубном пространстве;
- *Pg* – нагрузка от веса жидкости;
- Ptr – сила трения при ходе штанг вверх и вниз;
- *ro_st* – плотность материала штанг;
- *f* – площадь сечения штанг;
- *Mode* – маркёр направления движения плунжера, определяющий формулу для расчёта силы трения.

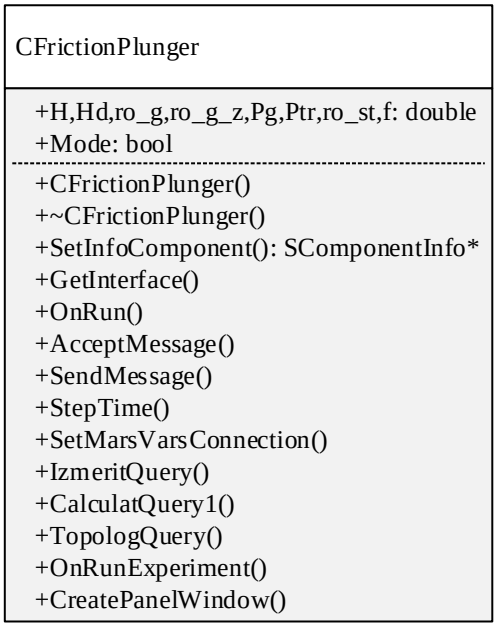


Рисунок Б.16 – Диаграмма класса CFrictionPlunger

5) компонент «Деформация штанг» реализован в классе *CDeformationRod* (рис. Б.17), наследующем методы от классов *IRedactComponent*, *IMessengerComponent*, *IBaseMarsComponent*, *IInteractivePanel*, *ISerialize*. Компонент содержит следующие поля:

- *v*1 – значение скорости в верхней точке штанг;
- *v*2 – значение скорости в нижней точке штанг;
- *F* – потоковая переменная (сила).
- *k* – коэффициент упругости;
- *Pg* – нагрузка от веса жидкости;
- *Ptr* – сила трения при ходе вверх или вниз;
- *lambda* – расчётная величина деформации штанг при движении;
- *Mode* – маркёр направления движения плунжера.

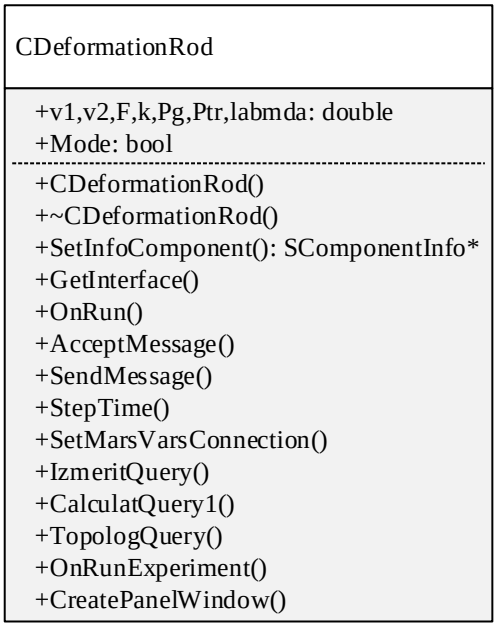


Рисунок Б.17 – Диаграмма класса CDeformationRod

Компоненты класса *CDeformationRod* предназначены для моделирования растяжения штанг при движении. Компонент содержит в себе закон Гука, а коэффициент *k* является функциональным параметром, рассчитываемым по выражению, записанному во встроенной в компонент ИМП, вызов которой по двойному клику на компонент осуществляется посредством метода *CreatePanelWindow*().

6) компонент «Нагрузка на плунжер» реализован в классе *CPressurePlunger* (рис. Б.18), наследующем методы от классов *IRedactComponent, IMessengerComponent, IBaseMarsComponent, IInteractivePanel, ISerialize*. Компонент содержит следующие поля:

- B_2 – некоторый коэффициент (константа или функциональный параметр),
- V_f – потоковая переменная (в нашем случае – сила),
- C – переменная-результат расчёта функциональной зависимости, заложенной в компонент посредством встроенной ИМП;
- H – глубина спуска насоса;
- f – площадь сечения штанг;
- $P1$ – нагрузка на плунжер;
- pu – давление газа на устье скважины;

- *pz* – давление газа в затрубном пространстве.

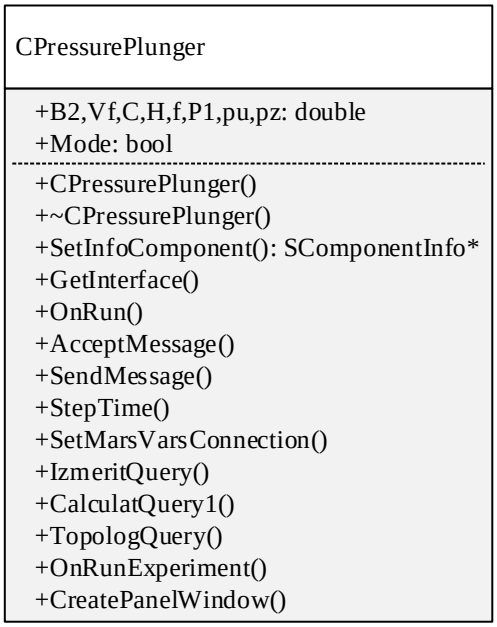


Рисунок Б.18 – Диаграмма класса CPressurePlunger

7) компонент «Шток (источник скорости)» реализован в классе *CVelocitySourceRod* (рис. Б.19), наследующем методы от классов *IRedactComponent*, *IMessengerComponent*, *IBaseMarsComponent*, *IInteractivePanel*, *ISerialize*. Компонент содержит следующие поля:

- B_1 – некоторый коэффициент (константа или функциональный параметр);
- V_p – потенциальная переменная (в нашем случае – скорость);
- $S(t)$ – некоторая функциональная зависимость.

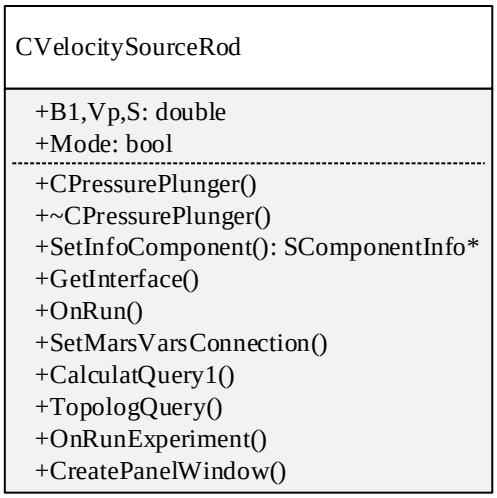


Рисунок Б.19 – Диаграмма класса CVelocitySourceRod

Данный компонент, в отличие от предыдущих, не является многоуровневым и располагается только в С-слое схемы, однако может параметрироваться с *L*-слоя за счёт использования компонентов типа «Атрибут».

Вышеописанные компоненты предназначены для моделирования класса ФТЗ и, в частности, конкретной модели – штангового глубинного насоса, рассматриваемого в пункте 4.2.1 Главы 4. Использование представленных компонентов предоставляет возможность быстрой разработки компьютерных моделей с высокой степенью визуализации и проведения виртуальных экспериментов, не требующих написания программного кода на языках программирования. Отмечается, что применение таких инструментальных средств как среды визуального моделирования в образовательной деятельности на аудиторных занятиях различного вида и во время самостоятельной работы студентов позволяет повысить степень их вовлеченности в работу, интерактивности и наглядности, а как следствие, качество преподавания и результаты учебной деятельности [Королёв 2013].

3 Информационная система управления виртуальной лабораторией моделирования физико-технических задач

ИСУЛ разработан на языке C# библиотеки NetFramework 4.5.2 базируется на фреймворке Chromium Embedded Framework – браузерном движке Chromium. ИСУЛ предназначена для исполнения на персональных компьютерах под управлением операционной системы Windows. Для нормального функционирования ИСУЛ не требуется дополнительного программного обеспечения, за исключением СМ МАРС для запуска схем компьютерных моделей и Microsoft Word (или аналогичных текстовых редакторов) для открытия файла отчёта.

ИСУЛ реализован в классе *ChromiumWebBrowser* (рис. Б.20). Опишем его поля:

- *startURL* – адрес стартовой страницы браузера (адрес электронного курса в Moodle);

- *domainSDO* – доменный адрес, на котором располагается электронный курс (sdo.tusur.ru);
- *currentURL* – переменная для хранения адреса открытой страницы.

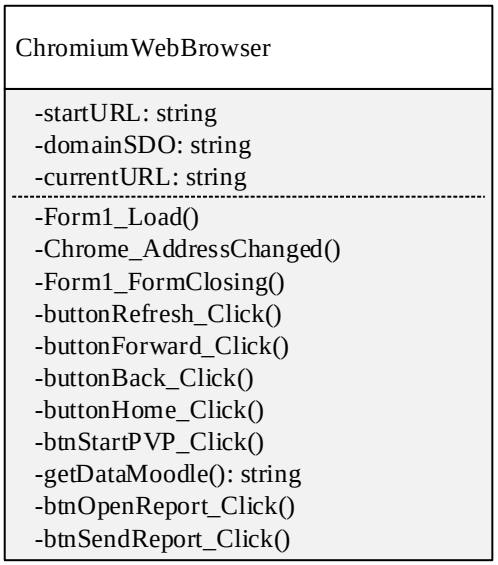


Рисунок Б.20 – Диаграмма класса ChromiumWebBrowser

Опишем методы класса *ChromiumWebBrowser*:

- Form1_Load() – действия при запуске приложения;
- Chrome_AddressChanged() – действия при переходе на новую веб-страницу.
- Form1_FormClosing() – действия при закрытии приложения;
- buttonRefresh_Click() – действия при нажатии кнопки «Обновить»;
- buttonForward_Click() – действия при нажатии кнопки «Вперёд»;
- buttonBack_Click() – действия при нажатии кнопки «Назад»;
- buttonHome_Click() – действия при нажатии кнопки «Домой»
- btnStartPVP_Click() – действия при нажатии кнопки «Запустить ПВП»;
- string[] getDataMoodle() – метод обмена данными с Moodle;
- btnOpenReport_Click() – действия при нажатии кнопки «Открыть отчёт»
- btnSendReport_Click() – действия при нажатии кнопки «Отправить отчёт».

ПРИЛОЖЕНИЕ В. АКТЫ ВНЕДРЕНИЯ РЕЗУЛЬТАТОВ ИССЛЕДОВАНИЯ

УТВЕРЖДАЮ:
Генеральный директор
АО «Энергонефтемаш»
В.А. Сеногноев
«22» 09 2019 г.

Акт о внедрении
результатов диссертационной работы М.И. Кочергина
«Развитие метода многоуровневых компонентных цепей для компьютерного моделирования физико-технических задач»

Комиссия в составе главного конструктора А.Г. Старинова, зам. главного конструктора А.М. Сергиенко, ведущего инженера В.И. Бесовой составила настоящий акт, подтверждающий факт использования в АО «Энергонефтемаш» результатов диссертационной работы М.И. Кочергина, а именно:

1. Многоуровневая компьютерная модель штангового глубинного насоса нефтедобывающей скважины, позволяющая производить расчёт усилий на полированном штоке в динамическом режиме.
2. Библиотека моделей компонентов для многоуровневого компьютерного моделирования физико-технических задач в среде моделирования МАРС, позволяющая отражать физические и геометрические свойства объектов в компьютерных моделях нефтедобывающих установок, а также моделировать гибридное (дискретно-непрерывное) поведение их подсистем с компенсацией амплитудно-фазовой погрешности, накапливаемой при смене дискретных состояний.

Указанные результаты позволили провести математическое моделирование нефтедобывающей установки, определить её оптимальные эксплуатационные параметры, а также реализовать алгоритм работы интеллектуальной системы управления штанговым глубинным насосом.

Главный конструктор / А.Г. Старинов

Зам. главного конструктора / А.М. Сергиенко

Ведущий инженер / В.И. Бесова

УТВЕРЖДАЮ:
Проректор по научной работе и инновациям
_____ В.М. Рулевский
«04» 09 2019 г.

Акт о внедрении
результатов диссертационной работы М.И. Кочергина
«Развитие метода многоуровневых компонентных цепей для компьютерного моделирования физико-технических задач»

Настоящий акт подтверждает, что результаты диссертационной работы М.И. Кочергина внедрены при выполнении научно-исследовательских работ по следующим темам:

1. «Исследование и разработка интеллектуальной системы управления штанговым глубинным насосом для поддержания оптимального динамического уровня жидкости в нефтяной скважине» (соглашение № 14.574.21.0157 от 26 сентября 2017 г.) – по федеральной целевой программы «Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2014 – 2020 гг.».
2. «Разработка программных средств автоматической параметризации компьютерных моделей эколого-экономических систем предприятий нефтегазовой промышленности», грант РФФИ №16-37-00027, 2016–2017 г.

В вышеуказанных научно-исследовательских работах использованы следующие полученные результаты:

1. Многоуровневая компьютерная модель штангового глубинного насоса нефтедобывающей скважины, позволяющая производить расчёт усилий на полированном штоке.
2. Компоненты среды многоуровневого моделирования МАРС для моделирования усилий на полированном штоке нефтедобывающего штангового насоса.
3. Библиотека моделей компонентов для многоуровневого компьютерного моделирования физико-технических задач в среде моделирования МАРС, позволяющая отражать физические и геометрические свойства объектов в компьютерных моделях, а также моделировать гибридное (дискретно-непрерывное) поведение систем с компенсацией амплитудно-фазовой погрешности, накапливаемой при смене дискретных состояний.

Результаты диссертационного исследования также использованы в учебном процессе ФГБОУ ВО «Томский государственный университет систем управления и радиоэлектроники» на кафедре компьютерных систем в управлении и проектировании (КСУП):

1. Разработанный подход к многоуровневому компьютерному моделированию физико-технических задач нашёл отражение в электронном курсе по дисциплинам: «Компьютерное моделирование физических задач», «Основы компьютерного моделирования физико-технических задач» для бакалавров направления 27.03.03 «Системный анализ и управление».
2. Разработанные компоненты среды моделирования МАРС и компьютерные модели физико-технических задач были использованы при создании учебно-методического пособия для проведения лабораторных работ по дисциплине «Моделирования систем» для бакалавров направления 27.03.03 «Системный анализ и управление».
3. Разработанный интерфейс для виртуальных и реально-виртуальных лабораторий, а также компоненты среды многоуровневого моделирования МАРС для обработки результатов моделирования, в том числе реализующие предложенный метод численной аппроксимации, были использованы при создании панелей виртуальных

приборов (ПВП) при организации работы учебной лаборатории «Элементы и устройства робототехнических систем».

На результаты интеллектуальной деятельности М.И. Кочергина получены:

1. Свидетельство о государственной регистрации программы для ЭВМ №2019660693. Библиотека моделей компонентов для многоуровневого компьютерного моделирования физико-технических задач в среде моделирования МАРС / Кочергин М.И. – 12.08.2019. – М.: Роспатент, 2019.
2. Свидетельство о государственной регистрации программы для ЭВМ №2019660759. Программный модуль для обучения компьютерному моделированию физико-технических задач / Кочергин М.И. – 13.08.2019. – М.: Роспатент, 2019.

Директор НИИ космических технологий,
д.т.н. _____ Ю.А. Шиняков

ПРИЛОЖЕНИЕ Г. СВИДЕТЕЛЬСТВА О РЕГИСТРАЦИИ ПРОГРАММ ДЛЯ ЭВМ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2019660693

Библиотека моделей компонентов для многоуровневого компьютерного моделирования физико-технических задач в среде моделирования МАРС

Правообладатель: ***Кочергин Максим Игоревич (RU)***

Автор: ***Кочергин Максим Игоревич (RU)***

Заявка № **2019619418**

Дата поступления **29 июля 2019 г.**

Дата государственной регистрации в Реестре программ для ЭВМ ***12 августа 2019 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Г.П. Ивлиев

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2019660759

Программный модуль для обучения компьютерному моделированию физико-технических задач

Правообладатель: ***Кочергин Максим Игоревич (RU)***

Автор: ***Кочергин Максим Игоревич (RU)***

Заявка № **2019619395**
Дата поступления **29 июля 2019 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***13 августа 2019 г.***

Руководитель Федеральной службы
по интеллектуальной собственности

Г.П. Ивлиев