**Реферат**

Расчетно-пояснительная записка 73 с., 4ч., 32 рис., 2 табл., 16 источников.

НЕЙРО-НЕЧЕТКАЯ СЕТЬ, МАНИПУЛЯТОР, УПРАВЛЕНИЕ МАНИПУЛЯТОРОМ, ANFIS, НЕЧЕТКИЙ ВЫВОД

Объектом исследования является манипулятор с планарно-ангулярной кинематической схемой. Объект разработки – нейро-нечеткая сеть ANFIS, используемая для вычисления обобщённых координат манипулятора. Цель работы – разработка и реализация программного метода управления манипулятором с использованием сети ANFIS. Задачи, решаемые в работе:

* анализ задачи управления манипулятором;
* анализ архитектуры нейро-нечеткой сети ANFIS;
* подготовка входных данных;
* обучение и тестирование работоспособности нейросети.

Область применения – системы управления манипуляторами.

В первой части работы описываются существующие подходы к решению задачи. Во второй части описываются алгоритмы и структуры данных. В третьей части описываются технологии, примененные при реализации алгоритмов. В четвертой части проведены экспериментальные исследования точностных характеристик метода.

Предлагаемые направления развития:

* оптимизация процесса обучения;
* планирование движения на основе текущего положения.

Поставленная цель была достигнута: метод управления манипулятором разработан, реализован и проверен на практике. Были рассмотрена существующие недостатки метода и предложены пути дальнейшего развития.

**Содержание**

[Введение 5](#_Toc453594183)

[1 Аналитический раздел 7](#_Toc453594184)

[1.1 Выбор объекта исследования 7](#_Toc453594185)

[1.2 Кинематические задачи 9](#_Toc453594186)

[1.3 Существующие методы управления 15](#_Toc453594187)

[1.3.1 Ручное управление 15](#_Toc453594188)

[1.3.2 Программное управление 16](#_Toc453594189)

[1.3.3 Использование нечеткой логики 17](#_Toc453594190)

[1.3.4 Использование нейро-нечетких моделей 25](#_Toc453594191)

[1.4 Модель манипулятора 26](#_Toc453594192)

[1.5 Формализация постановки задачи 28](#_Toc453594193)

[Выводы 28](#_Toc453594194)

[2 Конструкторский раздел 30](#_Toc453594195)

[2.1 Архитектура сети ANFIS 30](#_Toc453594196)

[2.2 Описание этапов работы разрабатываемого алгоритма 33](#_Toc453594197)

[2.2.1 Генерация обучающей выборки 35](#_Toc453594198)

[2.2.2 Обучение сети 36](#_Toc453594199)

[2.2.3 Определение обобщённых координат 39](#_Toc453594200)

[2.3 Тестирование 40](#_Toc453594201)

[2.4 Структура программного обеспечения 41](#_Toc453594202)

[Выводы 42](#_Toc453594203)

[3 Технологический раздел 43](#_Toc453594204)

[3.1 Выбор средств программной реализации 43](#_Toc453594205)

[3.2 Описание основных моментов программной реализации 44](#_Toc453594206)

[3.2.1 Использование библиотек MATLAB 44](#_Toc453594207)

[3.2.2 Использование функций MATLAB в C++ 46](#_Toc453594208)

[3.2.3 Графическая спецификация OpenGL 47](#_Toc453594209)

[3.2.4 Использованные сторонние библиотеки 48](#_Toc453594210)

[3.3 Структура разработанного ПО 50](#_Toc453594211)

[3.4 Методика тестирования 52](#_Toc453594212)

[3.5 Информация, необходимая для сборки и запуска 53](#_Toc453594213)

[3.6 Форматы входных и выходных данных 53](#_Toc453594214)

[3.7 Интерфейс пользователя 56](#_Toc453594215)

[Выводы 57](#_Toc453594216)

[4 Экспериментальный раздел 59](#_Toc453594217)

[4.1 Критерии оценки 59](#_Toc453594218)

[4.2 Обучающие выборки 60](#_Toc453594219)

[4.3 Время обучения 62](#_Toc453594220)

[4.4 Оценка погрешностей 62](#_Toc453594221)

[Выводы 63](#_Toc453594222)

[Заключение 63](#_Toc453594223)

[Список использованных источников 63](#_Toc453594224)

# Введение

Появление в 1950-х механических манипуляторов, а затем систем числового программного управления (ЧПУ) привело к созданию промышленных роботов – программируемых манипуляторов, предназначенных для выполнения разнообразных технологических операций: непосредственного выполнения формообразования, изменения линейных размеров заготовки и др. К вспомогательным технологическим операциям относятся транспортные операции, в том числе операции по загрузке и выгрузке технологического оборудования [1, 2]

Хотя основной сферой применения манипуляторов остаётся промышленность, а основной их задачей – автоматизация производственных процессов, с развитием технологий манипуляторы получают всё большее распространение в самых различных областях. Примерами могут служить как простой погрузочный кран, так и манипулятор робота-сапёра или погрузочная система на космическом корабле.

Существуют различные подходы к управлению манипулятором, среди которых наиболее простым и распространенным является программное управление – робот действует согласно управляющей программе с фиксированным порядком выполнения параметризованных операций. Основным недостатком программного управления является то, что такая программа, как правило, жёстко привязана к конкретной задаче.

В связи с этим, актуальным является использование теории искусственного интеллекта и смежных научных дисциплин для реализации гибких механизмов планирования действий и управления движением роботов [3]. Так, применение теории нечетких множеств для управления манипулятором не предполагает знания моделей процессов, происходящих внутри объекта управления. В формализованных знаниях об управляемом объекте при постановке задачи управления содержится неопределённая или нечёткая информация, которая не может быть обработана традиционными количественными методами. В таких случаях можно использовать алгоритмы управления на основе математической теории нечётких множеств.

Одним из недостатков использования аппарата нечеткого вывода является необходимость формирования базы знаний, на основе которой происходит логический вывод. В частности, необходимо правильно определить параметры функций принадлежности лингвистических термов. Как правило, база знаний составляется на основе мнений экспертов в данной предметной области.

В данной работе описано решение задачи управления многозвенным механизмом робота-манипулятора на основе применения аппарата нечеткой логики и нейро-нечетких моделей. В качестве объекта управления был взят двухзвенный манипулятор с планарно-ангулярной кинематической схемой, как отражающий специфические особенности роботов-манипуляторов.

Целью работы является разработка и реализация программного метода управления манипулятором с использованием нейро-нечеткой сети. В рамках работы решаются следующие задачи:

* анализ задачи управления манипулятором;
* анализ архитектуры нейро-нечеткой сети ANFIS;
* подготовка входных данных;
* обучение и тестирование работоспособности сети.

Разрабатываемый метод позволяет совместить сильные стороны как систем нечеткого вывода, так и нейросетевого подхода. Важное свойство нейросетей – способность к обучению и обобщению данных. Благодаря этому становится возможной настройка параметров системы нечеткого вывода на основе обучающих данных.

# Аналитический раздел

В этом разделе производится анализ предметной области, описываются основы кинематики и рассматриваются различные подходы к решению задачи управления. Также в этом разделе описывается упрощённая модель манипулятора, используемого в данной работе, и производится формализация задачи управления.

## Выбор объекта исследования

Большинство роботов, применяемых в современной промышленности, представляют собой манипуляторы различной конструкции. Манипуляторы используются для воспроизведения двигательных функций человека в процессе трудовой деятельности – это многозвенные механизмы с управляемыми приводами на каждом звене. Многозвенная конструкция заканчивается захватом, с помощью которого можно перемещать объекты в пространстве, или каким-либо инструментом (например, сварочным аппаратом или сверлом).

Манипуляторы разделяются на биотехнические (управляются человеком вручную) и автоматические. Наибольший интерес представляют именно автоматические манипуляторы [2].

Манипулятор обладает несколькими степенями подвижности. Существует две характерных конструкции манипуляторов – в одной из них используются только шарнирные соединения, из-за чего звенья могут лишь поворачиваться относительно основных осей (рисунок 1.1 (а)), в другой также допускается поступательное движение звеньев (рисунок 1.1 (б)).

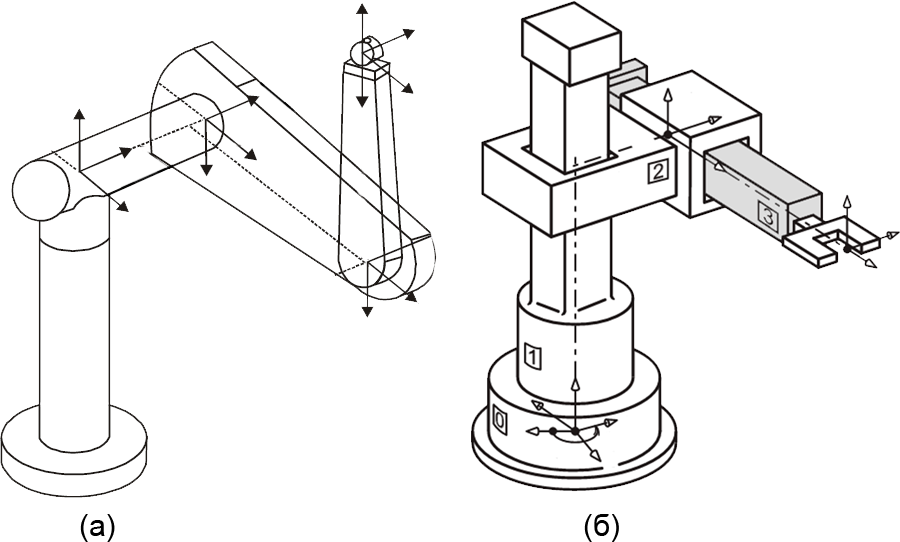


Рисунок . – а) манипулятор PUMA-560; б) манипулятор М20П

При использовании микроконтроллеров в системе управления манипулятором можно непосредственно программировать движение захвата или инструмента, а не движение каждого отдельного сустава. Кроме того, использование микропроцессорной системы управления позволяет отлаживать программу движения манипулятора с помощью графического образа на экране дисплея.

Задача управления манипулятором состоит из нескольких подзадач [2**Ошибка! Источник ссылки не найден.Ошибка! Источник ссылки не найден.**]:

1. Планирование положения – необходимо расположить звенья манипулятора таким образом, чтобы рабочий инструмент или захват мог быть доставлен в требуемое положение; возможно – с учетом требуемой ориентации инструмента.

2. Планирование движения – выбор траектории перемещения рабочего инструмента или объекта.

3. Планирование сил и моментов – согласование требований технологического процесса с возможностями робота развивать необходимые силы и моменты в различных точках рабочего пространства.

В рамках данной работы строится упрощённая модель манипулятора, в которой не рассматривается планирование сил (то есть, не моделируется работа приводов манипулятора), а планирование движения заключается в простом повороте звеньев манипулятора на требуемые углы. Таким образом, в рамках работы задача управления манипулятором может быть сведена только к планированию положения.

## Кинематические задачи

Планирование положения рабочей точки манипулятора разбивается на две основных составляющих: прямую и обратную кинематические задачи. Прямая задача (forward kinematics) – вычисление положения рабочей точки манипулятора по его кинематической схеме и заданной ориентации его звеньев ( – число степеней свободы манипулятора, A – углы поворота).

Обратная задача (inverse kinematics) – вычисление углов по заданному положению рабочей точки и известной кинематической схеме. При управлении манипулятором наиболее важной является именно обратная задача кинематики [1].

Основной проблемой при решении обратной кинематической задачи является то, что она, как правило, не имеет однозначного решения [2], а процесс его поиска значительно затрудняется с усложнением кинематической схемы манипулятора. Хотя заданной ориентации манипулятора соответствует ровно одно положение рабочей точки, одному положению рабочей точки может соответствовать несколько или бесконечность различных ориентаций манипулятора. С этим фактом также непосредственно связана сложность аналитического решения обратной задачи.

Так, уже в случае манипулятора, состоящего из двух звеньев, большинству положений рабочей точки (кроме находящихся строго на окружности, соответствующей полностью выпрямленному манипулятору) будут соответствовать две различные ориентации (см. рисунок 1.2):

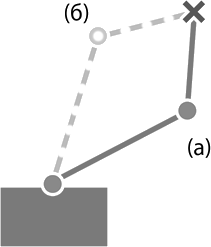


Рисунок . – Возможные ориентации манипулятора

Рассмотрим пример решения прямой и обратной кинематической задачи для случая (а). Пусть у нас есть манипулятор, способный работать только в одной плоскости и имеющий два сустава: первый сустав длины закреплен на основании и повернут на угол , второй сустав длины крепится к концу первого и повернут относительно него на угол (см. рисунок 1.3). Прямая задача заключается в нахождении координат рабочей точки по заданным .

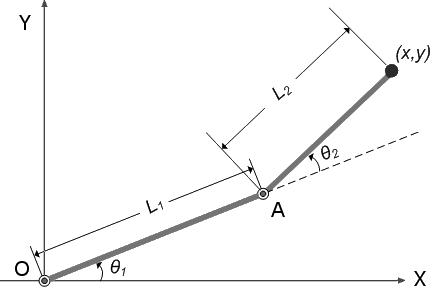


Рисунок . – Пример схемы планарно-ангулярного манипулятора

Имеем две системы отсчета: первая связана с точкой крепления звена , вторая – с началом координат в точке крепления звена . Найдем смещение второй системы относительно первой (координаты точки А в системе отсчета О):

Координаты в системе отсчета А:

Из рисунка 1.3 видно, что в системе О звено повернуто относительно плеча на угол , следовательно, в этой системе отсчета координаты :

Таким образом, решение прямой задачи имеет вид

Теперь найдем решение обратной кинематической задачи: найдем обобщённые координаты манипулятора (углы ) при заданных . Проведем прямую B (см. рисунок 1.4), соединяющую начало координат О с заданной точкой :

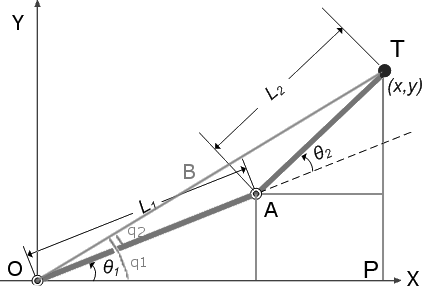


Рисунок . – Вычисление обобщённых координат манипулятора

Из рисунка видно, что

Здесь – угол между осью OX и прямой B, – угол между прямой B и звеном . Отсюда имеем:

Из треугольника можно получить

или

найдем по теореме косинусов:

Следовательно,

Таким образом, имеем

Аналогичным образом найдём угол . Исходя из рисунка 1.4, , следовательно, из рассмотрения треугольника следует:

Для данных координат возможно другое расположение манипулятора (см. рисунок 1.5).

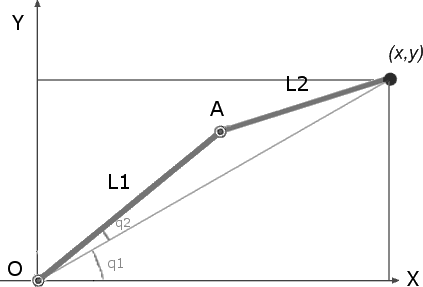


Рисунок . – Второе возможное расположение манипулятора

В этом случае , а нужно брать с противоположным знаком.

При работе в трёхмерном пространстве, в уравнениях необходимо учитывать уже не только углы наклона звеньев манипулятора , но и угол поворота всей конструкции вокруг своей оси.

При увеличении количества звеньев в манипуляторе или усложнении его кинематической схемы возникает ещё большее количество возможных комбинаций взаимного расположения звеньев. Кроме того, возможно возникновение так называемых точек сингулярности [4]:

* позиций в пространстве, которых манипулятор может достичь с помощью бесконечного числа возможных комбинаций;
* позиций, при которых манипулятор теряет одну или несколько степеней свободы;
* позиций, при которых возможна потеря управления механизмом.

Возможности манипулятора обычно ограничены – он функционирует только в определенном объеме рабочего пространства, называемом рабочей зоной, а его скорости и ускорения обусловлены характеристиками приводов. При этом дополнительные ограничения накладывает и сама кинематическая схема манипулятора: например, одно или несколько звеньев могут поворачиваться только в определенных пределах, а рабочая точка при этом не должна приближаться к звеньям манипулятора ближе определенного расстояния.

## Существующие методы управления

В данном подразделе выполнен обзор существующих подходов к управлению манипулятором.

### Ручное управление

Самым простым и одновременно наименее эффективным способом управления манипулятором является ручное управление, при котором специально обученный оператор контролирует движение каждого движущегося звена в отдельности. Простейшим примером является портовый погрузочный кран, при работе с которым оператор контролирует перемещение основания крана в горизонтальной плоскости и захвата в вертикальной плоскости.

Основным недостатком этого подхода является человеческий фактор. Оператор должен быть обучен работе с данным типом манипуляторов, при этом возможны различные ошибки, связанные с принятием оператором решений. Кроме того, данный подход практически неприменим при использовании манипуляторов со сложной кинематической схемой, поскольку от человека потребуется контролировать слишком большое число подвижных частей манипулятора.

В таких случаях ручное управление как правило совмещается с интеллектуальной системой управления, когда оператор задает позицию рабочей точки, а система самостоятельно переводит манипулятор в нужное положение. Примером может являться система управления манипулятором WAM-Titan II TS [10] (см. рисунок 1.6). Манипулятор состоит из двух частей – рабочей (непосредственно механической «руки», обладающей 4 степенями подвижности), и управляющей части (дополнительным манипулятором, обладающим 7 степенями подвижности). Оператор работает с управляющей частью – на основе её обобщённых координат вычисляется положение рабочей точки (прямая кинематика), эта точка проецируется в систему координат, связанную с рабочей частью, и на основе координат полученной точки вычисляются обобщённые координаты рабочей части манипулятора (обратная кинематика).

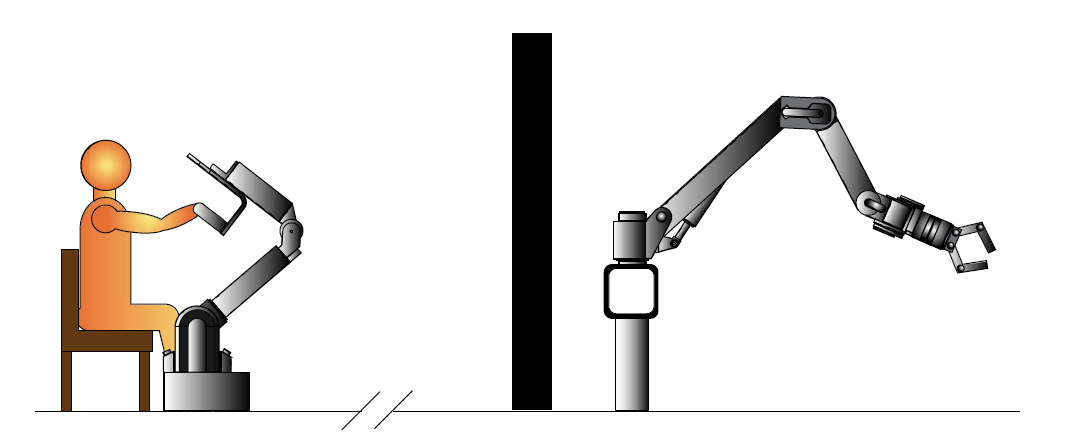


Рисунок 1.6 – Манипулятор WAM-Titan II Teleoperation System

### Программное управление

Наиболее распространенным является использование программного управления, при котором робот действует согласно некоторой управляющей программе с фиксированным порядком выполнения операций. Создание такой программы предполагает детальный анализ процессов, выполняемых манипулятором, составление и решение уравнений динамики, описывающих поведение системы. Классическая теория автоматического управления предполагает, что такие уравнения можно построить на основе законов физики.

Программное управление подразумевает строгую последовательность действий, которую должен выполнить манипулятор (развернуть звено; ожидать; передвинуть звено и т.д.). Данный подход наиболее эффективен в тех случаях, когда последовательность движений тривиальна или известна заранее. Хорошим примером могут являться сборочные конвейеры, где каждый манипулятор многократно повторяет одну и ту же последовательность действий.

Тем не менее, если последовательность действий не определена заранее или если предполагается совмещение рабочей точки манипулятора с произвольной точкой окружающего пространства (например, манипулятор на роботе-сапёре), возникает необходимость решения уравнений динамики манипулятора для определения обобщённых координат манипулятора. Пример аналитического решения задачи управления был приведен в разделе 1.2.

Кроме того, в случае использования сложных кинематических схем решение уравнений динамики манипулятора может требовать значительных вычислительных затрат, поскольку эти уравнения зачастую не решаются аналитически и требует использования численных методов. Наконец, итоговая программа оказывается ориентированной на конкретный процесс, а для её повторного использования в другом процессе могут потребоваться значительные изменения.

### Использование нечеткой логики

Успехи в развитии теории искусственного интеллекта и смежных научных дисциплин открывают перспективы для реализации гибких алгоритмов управления движением и планированием действий. Довольно популярно использование для управления манипулятором теории нечетких множеств и построенного на её основе аппарата нечеткой логики [1, 3].

Механизм нечёткого вывода используется в различного рода экспертных системах и управляющих системах; в своей основе он имеет базу знаний, формируемых специалистами предметной области в виде нечётких предикатных правил. Данные правила имеют вид

где

– входная лингвистическая переменная (имя для известных значений данных);

– выходная лингвистическая переменная (имя для значений данных, которое будет вычислено);

и – термы, определённые соответственно на и y [3].

Лингвистическая переменная – кортеж вида где:

– имя лингвистической переменной;

– множество её значений (термов);

– универсум нечетких переменных.

Простейший пример лингвистической переменной: «расстояние между точками» (β), которое субъективно может быть описано как «очень близко», «близко», «недалеко» или «далеко» (T), объективно может принимать значения (X), а процедуру задания на универсуме значений термов из множества T можно представить в виде функции принадлежности, представленной на рисунке 1.7:

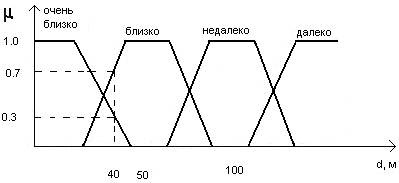


Рисунок . – Пример функции принадлежности нечеткой переменной «расстояние»

В качестве условий и заключений правил могут использоваться только простые и составные нечеткие высказывания, соединенные бинарными операциями «И», «ИЛИ», «НЕ». При этом в каждом из нечетких высказываний должны быть определены функции принадлежности значений множества термов для каждой лингвистической переменной. Как правило, функции принадлежности отдельных термов представляют треугольными или трапецеидальными функциями.

В случае с манипулятором, система вывода может содержать базу знаний из предикатов, например следующих:

*«если текущая рабочая точка 'далеко от целевой точки' то 'скорость поворота высокая'»,*

*«если 'рассогласование углов незначительное' то 'скорость поворота низкая'»*.

В общем виде, работа системы нечеткого вывода может быть представлена в виде схемы, приведённой на рисунке 1.8.

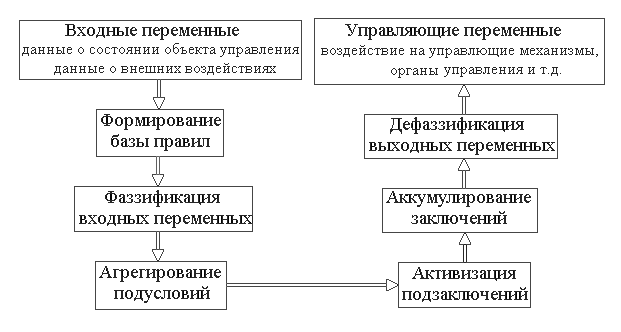


Рисунок . – Диаграмма процесса нечеткого вывода в нечетких САУ

*База правил* предназначена для формального представления эмпирических знаний в форме нечетких продукционных правил. Обычно формируется на основании мнений экспертов в той или иной предметной области.

*Фаззификация* – установка соответствия между численным значением входной переменной и значением функции принадлежности соответствующего ей терма лингвистической переменной. На этом этапе значениям всех входных переменных (полученных, например, при помощи датчиков) ставятся в соответствие конкретные значения функций принадлежности соответствующих лингвистических термов, которые используются в условиях нечетких правил.

*Агрегирование* – это процедура определения степени истинности *условий* каждого из правил системы нечеткого вывода. При этом используются полученные на этапе фаззификации значения функций принадлежности термов лингвистических переменных.

Если условие нечеткого продукционного правила является простым нечетким высказыванием, то степень его истинности соответствует значению функции принадлежности соответствующего терма лингвистической переменной.

Если условие представляет составное высказывание, то степень истинности определяется на основе известных значений истинности составляющих его элементарных высказываний при помощи введенных ранее нечетких логических операций.

*Активизация* – это процедура нахождения степени истинности каждого из элементарных логических высказываний, составляющих *заключения* нечетких правил. Поскольку заключения делаются относительно выходных лингвистических переменных, то степеням истинности элементарных подзаключений при активизации ставятся в соответствие элементарные функции принадлежности.

Если заключение является простым нечетким высказыванием, то степень его истинности равна алгебраическому произведению весового коэффициента и степени истинности предпосылки данного нечеткого продукционного правила.

Если заключение представляет составное высказывание, то степень истинности каждого из элементарных высказываний равна алгебраическому произведению весового коэффициента и степени истинности антецедента данного нечеткого продукционного правила.

Если весовые коэффициенты продукционных правил не указаны явно на этапе формирования базы правил, то их значения по умолчанию равны единице.

*Аккумуляция* (или *аккумулирование*) – это процесс нахождения функции принадлежности для каждой из выходных лингвистических переменных. Цель аккумуляции состоит в объединении всех степеней истинности подзаключений для получения функции принадлежности каждой из выходных переменных. Результат аккумуляции для каждой выходной лингвистической переменной определяется как объединение нечетких множеств всех подзаключений нечеткой базы правил относительно соответствующей лингвистической переменной. Объединение функций принадлежности всех подзаключений проводится как правило по принципу max-объединения:

*Дефаззификация* – это процесс перехода от функции принадлежности выходной лингвистической переменной к её четкому (числовому) значению. Цель дефаззификации состоит в том, чтобы, используя результаты аккумуляции всех выходных лингвистических переменных, получить количественные значения для каждой выходной переменной, которое используется внешними (по отношению к системе нечеткого вывода) устройствами.

Переход от полученной в результате аккумуляции функции принадлежности выходной лингвистической переменной к численному значению выходной переменной производится одним из следующих методов:

* метод центра тяжести заключается в расчете центроида площади: , где – носитель нечеткого множества выходной лингвистической переменной;
* метод центра площади заключается в вычисление абсциссы y, делящей площадь, ограниченную кривой функции принадлежности (так называемой биссектрисы площади):
* метод левого модального значения: ;
* метод правого модального значения: .

В то время как аналитическое решение обратной кинематической задачи предполагает максимально точное определение углов поворота составных частей манипулятора, использование нечеткого вывода предполагает постоянное уточнение параметров в процессе движения манипулятора. Характеристиками достижения целевого состояния могут служить как скорость достижения – количество шагов, за которое будет достигнуто нужное состояние с заданной точностью, так и время колебания манипулятора возле целевого положения – состояние, когда манипулятор приблизился к целевому состоянию, но не может его достичь, так как на каждом шаге "перескакивает" нужную позицию.

**Алгоритм Мамдани**

Является первым из созданных алгоритмов нечеткого вывода. Был предложен в 1975 году английским математиком Е. Мамдани [5] для управления паровым двигателем.

Базу правил составляют правила вида «если A то B», где предпосылки построены при помощи логических связок «И», а заключения – простые.

Фаззификация входных переменных осуществляется так же, как и в общем случае построения системы нечеткого вывода.

Агрегирование подусловий правил производится с помощью классической нечеткой операции «И» двух высказываний:

Активизация осуществляется методом min-активизации:

где – функции принадлежности и степени истинности нечетких высказываний, образующих соответствующие следствия правил.

Аккумуляция подзаключений правил проводится с помощью max-объединения функций принадлежности:

Дефаззификация проводится методом центра тяжести или центра площади.

**Алгоритм Цукамото (Tsukamoto) [3]**

Формирование базы правил, фаззификация и агрегирование проводятся аналогично алгоритму Мамдани.

Активизация проводится в два этапа: на первом этапе степени истинности заключений находятся аналогично алгоритму Мамдани (алгебраическое произведение весового коэффициента и степени истинности предпосылки). На втором этапе для каждого из правил вместо построения функций принадлежности решается уравнение и определяется четкое значение выходной лингвистической переменной.

Соответственно, аккумуляция заключений не требуется, поскольку на этапе активизации уже получены дискретные множества четких значений.

При дефаззификации для каждой лингвистической переменной осуществляется переход от дискретного множества значений к единственному четкому значению с помощью дискретного метода центра тяжести:

**Алгоритм Сугено (Sugeno) [6]**

База правил представляется списком правил вида «если А и В то », где предпосылки правил построены из двух простых нечетких высказываний с помощью связки «И», и – четкие значения входных переменных, соответствующие высказываниям А и В, и – весовые коэффициенты пропорциональности между четкими значениями входных переменных и выходной переменной, – четкое значение выходной переменной, определенное как действительное число.

Фаззификация и агрегирование проводятся аналогично алгоритму Мамдани.

Активизация проводится в два этапа. На первом этапе, степени истинности заключений правил, ставящих в соответствие выходной переменной действительные числа, находятся аналогично алгоритму Мамдани (алгебраическое произведение весового коэффициента и степени истинности предпосылки данного правила). На втором этапе, в отличие от алгоритма Мамдани, для каждого из правил находится четкое значение выходной переменной . Таким образом, каждому i-му правилу ставится в соответствие точка , где – степень истинности правила, – четкое значение выходной переменной, определенной в заключении правила.

Аккумуляция заключений правил нечеткой продукции не проводится, поскольку на этапе активизации уже получены дискретные множества четких значений для каждой из выходных лингвистических переменных.

Дефаззификация проводится как в алгоритме Цукамото.

Существует также упрощенный алгоритм нечеткого вывода, основанный на алгоритме Сугено [3]. В нем при задании четких значений в заключениях правил вместо соотношения используется явное задание непосредственного значения .

Алгоритм Цукамото применяют только для монотонных функций соответствия выходного параметра, поэтому этот алгоритм не универсален, зато прост. Алгоритм Сугено применяется, когда известна не форма функции соответствия выходного параметра, а весовые коэффициенты, через которые входные параметры вносят свой вклад [1].

### Использование нейро-нечетких моделей

Часто встает вопрос о наделении системы управления способностями приспосабливаться к условиям внешней среды и самостоятельно решать такие задачи, как планирование поведения. Одним из подходов, используемых для решения всей совокупности задач управления роботами, стал подход, базирующийся на применении нейросетевых технологий [3].

Обладая высоким быстродействием и относительной простотой реализации, эта технология послужила базой для развития алгоритмов управления роботами на всех уровнях иерархии процессов управления: от уровня управления приводами звеньев робота до уровня планирования поведения.

Одним из вариантов интеллектуализации является совмещение нейросетевого подхода и механизмов нечёткого вывода – использование так называемых нейро-нечетких моделей и гибридных нейронных сетей. Основным преимуществом использования нечеткой логики является её способность описывать желаемое поведение системы простыми правилами «если – то». При этом во многих случаях знания, описывающие поведение системы, содержатся в различных наборах данных, требующих больших вычислений. В этом случае хорошо подходит методология нейронных сетей, которые обучаются на этих данных.

Данный метод позволяет учитывать ошибки в ходе выполнения задачи и не требует больших вычислительных ресурсов. Также возможна перенастройка правил поведения манипулятора в случае изменения параметров системы. Основным недостатком метода является необходимость обучения нейронной сети на некоторой тестовой выборке – данный процесс может потребовать значительных временных затрат.

В данной выпускной квалификационной работе предполагается использование именно нейро-нечеткого подхода к управлению. Разрабатываемый метод является развитием метода управления с использованием нечеткой логики и позволяет системе нечеткого вывода обучаться, корректируя параметры системы.

## Модель манипулятора

В рамках данной выпускной квалификационной работы моделируется работа манипулятора, состоящего из двух вращающихся в вертикальной плоскости звеньев известной длины (планарно-ангулярная кинематическая схема) и вращающегося в горизонтальной плоскости основания. Таким образом, координаты рабочей точки манипулятора при известных длинах звеньев определяются тремя углами , как показано на рисунке 1.9:

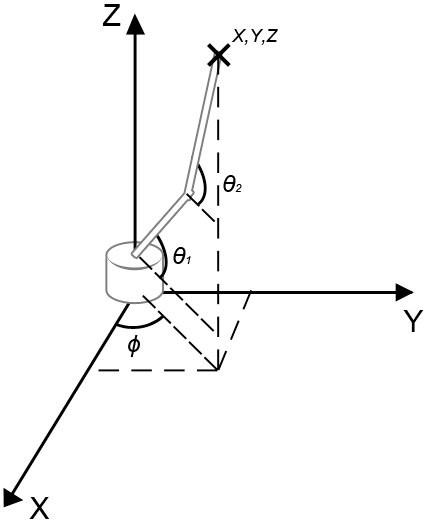


Рисунок . – Модель рассматриваемого манипулятора

На манипулятор наложены следующее ограничения: первое звено поворачивается в пределах , второе – в пределах Основание может вращаться как по часовой, так и против часовой стрелки в пределах .

Такая кинематическая схема была выбрана исходя из следующих соображений:

1. Два звена позволяют ограничить не только максимально допустимое, но минимально допустимое расстояние рабочей точки от основания манипулятора;
2. Ограничения поворота позволяют избежать дублирования допустимых точек в рабочей зоне – т.е. для каждого положения рабочей точки можно однозначно вывести углы поворота манипулятора;
3. Вращающееся основание позволяет работать в трёхмерном пространстве, при этом угол также может быть легко вычислен аналитически.

Таким образом, благодаря возможности аналитически определить требуемую ориентацию и положение рабочей точки манипулятора, можно провести анализ точности выбранного метода управления манипулятором.

## Формализация постановки задачи

Основываясь на проведённом анализе предметной области, постановка решаемой в данной работе задачи может быть сформулирована следующим образом.

Необходимо реализовать программный модуль, использующий нейро-нечеткую сеть для решения обратной кинематической задачи. Входными данными являются координаты целевой точки в декартовой системе координат в трёхмерном пространстве. Выходными данными является набор обобщённых координат манипулятора, необходимый для совмещения рабочей точки манипулятора с целевой точкой.

Для функционирования нейро-нечеткой сети необходимо провести её обучение с использованием обучающего набора данных. В качестве такого набора данных выступает выборка из множества точек рабочей зоны манипулятора.

Также необходимо разработать ПО, использующее данный программный модуль для моделирования движения манипулятора в простейшем варианте, и провести исследование точности работы реализованного метода.

## Выводы

В аналитическом разделе был проведен анализ предметной области, в ходе которого была сформулирована задача управления манипулятором, проанализированы существующие методы решения поставленной задачи и предложен вариант решения задачи управления с использованием нейро-нечеткого подхода. Были описаны принимаемые в рамках работы ограничения, накладываемые на модель манипулятора, а также требуемая функциональность разрабатываемого ПО.

# Конструкторский раздел

В данной работе используется нейро-нечеткая сеть ANFIS (adaptive neuro-fuzzy inference system), которая осуществляет настройку параметров функций принадлежности системы нечеткого вывода с помощью нейросетевого подхода. В этом разделе приводится описание архитектуры данной сети и описываются основные этапы метода определения обобщённых координат манипулятора с использованием сети ANFIS.

## Архитектура сети ANFIS

Сеть ANFIS представляет собой обобщённую нейросеть [7] (generalized neural network) состоящую из пяти слоёв (см. рисунок 2.1). Входы и выход сети в отдельные слои не выделяются.



Рисунок . – Структура сети ANFIS

Для объяснения назначения каждого из слоёв сети, введём следующие обозначения:

– количество входных переменных;

– входы сети, ;

– выход сети;

– количество правил;

– нечеткое правило с порядковым номером ;

– нечеткий терм, соответствующей -й входной переменной, с функцией принадлежности ; используется для лингвистической оценки переменной в правиле ;

– действительные числа в заключении правила , .

Опишем назначение слоёв рассматриваемой сети.

1. Узлы первого слоя служат для представления термов входных переменных – каждый узел соответствует одному терму. При конкретных значениях входов на выходе узла формируются значения колоколообразной функции принадлежности:

Здесь – множество параметров, которые изменяются в процессе обучения нейронной сети. Изменение данных параметров приводит к изменению вида функции принадлежности – сжатию или смещению. Параметры называются параметрами предпосылок (premise parameters) [7].

Количество узлов на этом слое равно сумме мощностей терм-множеств входных переменных и может быть рассчитано по следующей формуле:

где – количество термов -й лингвистической переменной.

2. Узлы второго слоя исполняют роль посылок нечетких правил – каждый узел соединен с теми узлами первого слоя, которые формируют посылки правила. Выходом каждого узла этого слоя является степень выполнения.

Количество узлов на этом слое равно количеству правил в базе знаний системы:

Выход -го узла этого слоя рассчитывается как произведение выходов тех узлов первого слоя, которые соответствуют посылкам -го правила:

3. Узлы третьего слоя служат для нормализации степеней выполнения правил: относительной степени выполнения конкретного нечеткого правила относительно всей их совокупности.

Каждый узел третьего слоя связан со всеми узлами второго слоя. Количество узлов на этом слое также равно количеству правил:

Выход каждого узла рассчитывается по следующей формуле:

4. Четвертый слой предназначен для вычисления заключений нечетких правил. -й узел данного слоя связан с -м узлом третьего слоя, а также со всеми входами сети (на рисунке 2.1 связи с входами не показаны). Каждый узел четвертого слоя рассчитывает вклад соответствующего правила в выход всей сети:

Параметры изменяются в процессе обучения сети и называются параметрами следствий (consequence parameters) [7].

Количество узлов на этом слое также равно количеству правил в базе знаний системы:

5. Наконец, пятый слой выполняет агрегирование результата, полученного по различным правилам. Единственный узел этого слоя вычисляет выходное значение сети, суммируя все выходы узлов 4-го слоя:

Таким образом, ANFIS можно рассматривать как систему нечеткого вывода, представленную в виде обобщённой сети, включающей в себя узлов, объединенных в пять слоёв. При обучении сети производится настройка параметров системы нечеткого вывода [8, 9].

## Описание этапов работы разрабатываемого алгоритма

Применительно к задаче определения обобщённых координат манипулятора, сеть ANFIS используется для сопоставления входного значения (координат целевой точки) выходному (обобщённым координатам).

Главным ограничением, которое возникает при использовании ANFIS, является то, что в ANFIS допускается только один выходной параметр, функция которого является константой или линейной. Данное ограничение связано с использованием в ANFIS алгоритма вывода Сугено [8]. С учетом этого, основным подходом к решению обратной кинематической задачи с использованием ANFIS является создание нескольких сетей, каждая из которых будет определять угол поворота одного из звеньев манипулятора.

В рамках данной работы, с учетом конструкции манипулятора, необходимо использовать три сети: для определения углов и .

Таким образом, процесс управления манипулятором с использованием ANFIS можно представить в виде следующей последовательности действий (см. рисунок 2.2):

1. Генерация обучающей выборки – набора точек из рабочей зоны манипулятора и соответствующих им обобщённых координат манипулятора.

2. Обучение трёх сетей ANFIS с помощью данной выборки. При этом возможны различные конфигурации сетей, ввиду использования различного количества функций принадлежности, количества эпох обучения, допустимой ошибки и т.п.

3. Непосредственно управление манипулятором с использованием сгенерированной системы нечеткого вывода – предсказание необходимой ориентации манипулятора на основе требуемой позиции рабочей точки.

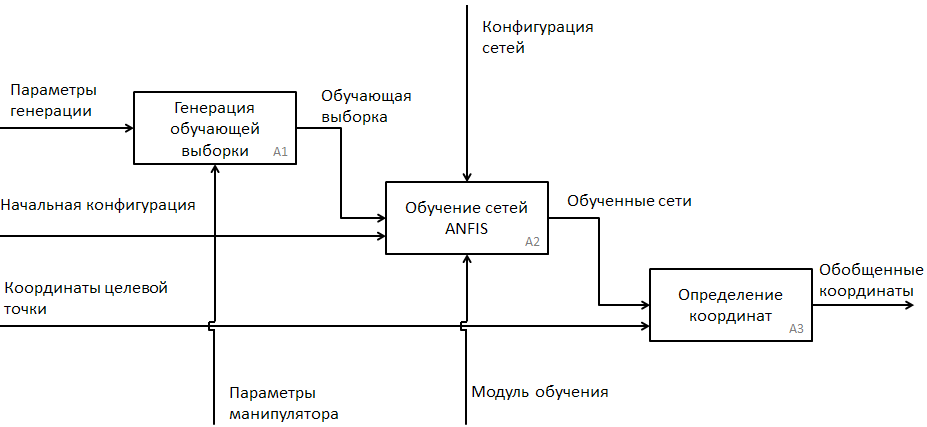


Рисунок . – IDEF0-диаграмма метода определения обобщённых координат

### Генерация обучающей выборки

Для обучения ANFIS необходима обучающая выборка, сгенерированная на основе допустимых точек из рабочей зоны манипулятора, и состоящая из пар *«вектор обобщённых координат + координаты рабочей точки»*. Основное требование, предъявляемое к выборке: векторы обобщённых координат не должны дублироваться. В противном случае возможно возникновение неточностей при обучении сети.

Обучающая выборка генерируется на основе параметров манипулятора – к примеру, допустимых углов поворота, определяющих рабочую зону. На рисунке 2.3 приведена схема алгоритма генерации обучающей выборки.

D:\Dropbox\Apps\drawio\выборка.png

Рисунок . – Схема алгоритма генерации обучающей выборки

### Обучение сети

Поскольку для определения каждого из углов используется отдельная сеть ANFIS, обучающая выборка разбивается на три матрицы следующего вида:

где – один из углов . Каждая из матриц используется для обучения соответствующей сети – первые три столбца являются входными данными, четвёртый – выходным значением.

Перед началом обучения сети необходимо задать начальные параметры системы вывода. Такими параметрами являются количество термов, (membership functions count, MFcount), вид функций принадлежности для термов входных переменных, и вид функций принадлежности для термов выходных переменных. При этом для каждой переменной допускается использовать различное количество термов, а для каждого терма – различные виды функций принадлежности. Наиболее распространёнными являются треугольная, трапецеидальная и колоколообразная функции.

Для настройки системы нечеткого вывода с использованием ANFIS могут быть применены типовые процедуры обучения нейронных сетей, т.к. в ней использует только дифференцируемые функции. Часто применяется комбинация градиентного спуска в виде алгоритма обратного распространения ошибки и метода наименьших квадратов. Алгоритм обратного распространения ошибки настраивает параметры функций принадлежности; методом наименьших квадратов оцениваются коэффициенты заключений правил [9].

Ниже приведен пример результатов настройки системы вывода с двумя входами в программе MATLAB (компонент Fuzzy Editor). На рисунке 2.4 представлены исходные функции принадлежности входов системы, на рисунке 2.5 – функции принадлежности после настройки системы с использованием ANFIS.

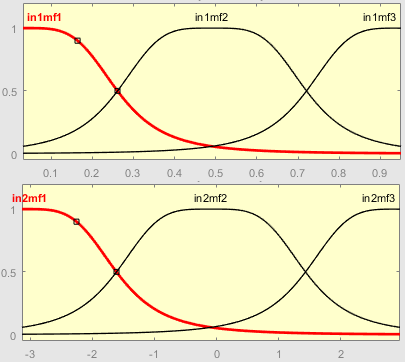


Рисунок . – Исходный вид функций принадлежности

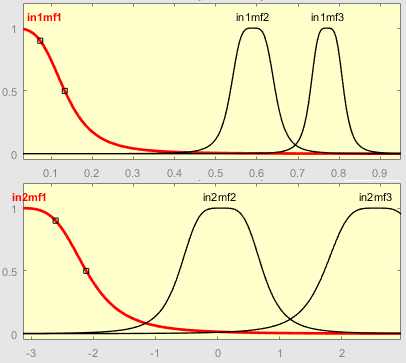


Рисунок . – Вид функций принадлежности после настройки с использованием ANFIS

### Определение обобщённых координат

После настройки параметров, обученная система нечеткого вывода используется для предсказания требуемых обобщённых координат манипулятора на основе декартовых координат целевой точки. Для этого используется база правил. Благодаря тому, что используется система вывода типа Sugeno, выходным значением системы вывода является не нечеткий терм, а конкретное значение угла. Таким образом, алгоритм определения обобщённых координат манипулятора может быть представлен схемой, изображенной на рисунке 2.6.

D:\Dropbox\Apps\drawio\Untitled Diagram.png

Рисунок . – Схема алгоритма определения обобщённых координат

Алгоритм определения одного угла с помощью конкретной сети (в соответствии с описанной ранее архитектурой ANFIS) представлен на рисунке 2.7.

D:\Dropbox\Apps\drawio\один угол.png

Рисунок . – Схема алгоритма определения угла по координатам точки

## Тестирование

Использование алгоритмов нечёткого вывода и нейросетей так или иначе ведёт к возникновению погрешностей при определении выходных результатов. Значения данных погрешностей зависят от множества факторов, связанных с начальными значениями параметров, конфигурацией сетей, объемом обучающей выборки, количеством эпох обучения и т.п.

Таким образом, необходимо модульное тестирование составляющих алгоритма и функциональное тестирование алгоритма в целом. Так, однозначно протестировать можно, к примеру, уникальность обобщённых координат при генерации обучающей выборки, успешность фаззификации каждого входа, принадлежность результирующих обобщённых координат интервалу .

При этом работоспособность и точность работы алгоритма в целом является скорее предметом исследования – к примеру, можно определить зависимость погрешности от использованных конфигураций сетей и проводить функциональное тестирование с учётом данных зависимостей.

## Структура программного обеспечения

Примерная схема ПО, реализующего описанные выше методы представлена на рисунке 2.8.

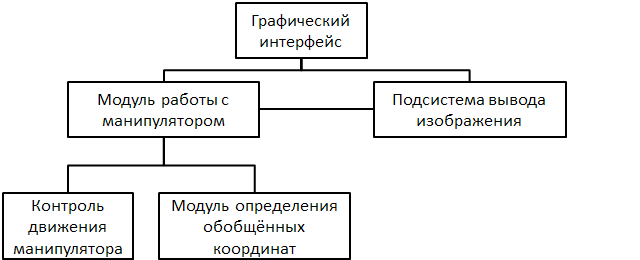


Рисунок . – Схема взаимосвязей модулей программного обеспечения

Данная схема представляет собой вариант реализации паттерна MVC (Model – View – Controller).

Компонент графического интерфейса должен обеспечивать взаимодействия пользователя с приложением.

Модуль работы с манипулятором должен отвечать за обработку логики программы, включая как перемещение манипулятора, так и непосредственно реализацию метода определения обобщённых координат.

Подсистема вывода изображения должна предоставлять возможности для отображение трёхмерной сцены и графического интерфейса пользователя.

## Выводы

Было дано подробное описание архитектуры обобщённой сети ANFIS и описаны основные этапы метода определения обобщённых координат манипулятора с использованием сети ANFIS. Также было дано краткое описание взаимодействия компонентов программного обеспечения, реализующего данный метод, друг с другом.

# Технологический раздел

Данный раздел содержит обоснование выбора средств программной реализации разработанного метода и описание программной реализации разрабатываемого метода. Также приводится описание форматов данных, используемых в ПО, и приводятся скриншоты интерфейса приложения.

## Выбор средств программной реализации

Реализуемое программное обеспечение состоит из двух основных частей – блока управлением манипулятором и блока графического отображения. Блок управления основывается на использовании алгоритмов нечеткого вывода и нейро-нечеткой сети ANFIS, реализация которых присутствует в программном пакете MATLAB [9]. Было решено использовать MATLAB, поскольку он предоставляет удобный функционал для работы с математическими алгоритмами, а также содержит реализацию необходимых функций для обучения и использования сетей ANFIS.

В рамках данной работы предполагается, что результат работы метода управления можно увидеть не только в виде графиков, но и на примере движения манипулятора. В этом случае изображение должно строиться и обновляться в реальном времени с высокой частотой, чтобы создать иллюзию плавного движения манипулятора, без рывков. В связи с этим, было решено использовать возможности графического ускорения с использованием функционала OpenGL.

В качестве языка программирования был выбран язык С++ [11], в качестве среды разработки – Microsoft Visual Studio 2015.

Выбор С++ был обусловлен тем, что спецификация OpenGL придерживается структурного подхода и в Windows реализовывается в соответствии со стандартами языка С. В свою очередь, код на С совместим с С++, что позволяет использовать как объектно-ориентированный подход при разработке пользовательского приложения, так и структурную реализацию стандартной библиотеки OpenGL в Windows [12]. MATLAB также поддерживает использование функций в программах, написанных на языке C++.

Кроме того, язык C++ обеспечивает высокую производительность приложений, а также поддерживает большое количество библиотек, облегчающих работу с трёхмерной графикой.

Выбор Visual Studio 2015 в качестве среды разработки был обусловлен следующими факторами [13]:

* поддержка современных стандартов языка С++11 и С++14;
* интеграция с системой контроля версия Git;
* наличие удобных инструментов для отладки программы и рефакторинга кода;
* наличие встроенного компонента для модульного тестирования.

## Описание основных моментов программной реализации

### Использование библиотек MATLAB

Использовать функции, написанные на языке MATLAB, в проекте на C++ можно с помощью нескольких подходов.

Первый – использование утилиты MATLAB Coder, которая позволяет экспортировать написанные в MATLAB функции в код на C/C++/Fortran. Данный способ является наиболее простым и прямолинейным, но его главный недостаток – невозможность экспорта некоторых встроенных функций, в число которых входит модуль нечеткой логики Fuzzy Toolbox, включающий функции для работыANFIS.

Второй подход – интеграция в приложение фреймворка MATLAB Engine. Данный способ позволяет использовать всю функциональность MATLAB в программе на C++, в том числе работать с большинством инструментов, входящих в состав программного пакета, такими, как графопостроитель, и использовать все установленные модули MATLAB.

Несмотря на преимущества данного способа, его использование предполагает наличие на компьютере установленного пакета MATLAB. Этот способ наиболее целесообразен в тех случаях, когда в приложении на C++ не только используется большое количество различных функций MATLAB, но и предполагается взаимодействие с компонентами этого программного пакета.

Наконец, третий подход – использование библиотек MATLAB Runtime и компилятора MATLAB Compiler (MCR) (см. рисунок 3.1). В этом случае пользовательские функции компилируются в библиотеку .dll, которую можно подключить к проекту на C++. В итоге, для работы программы всё равно потребуется установка дополнительного ПО. Однако, в отличие от самого пакета MATLAB, runtime-библиотеки распространяются бесплатно и занимают в несколько раз меньше дискового пространства. Также MCR позволяет экспортировать библиотеки в различных форматах, к примеру в виде сборок .NET или модулей Python.

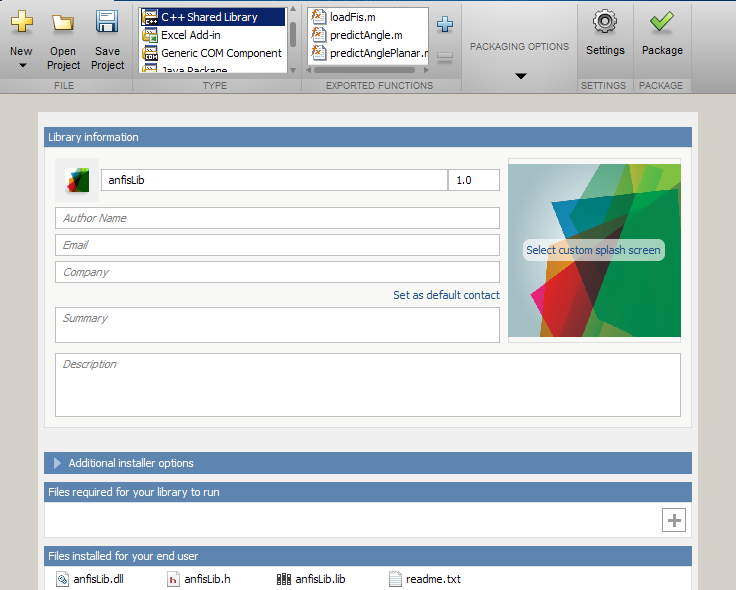


Рисунок . – Интерфейс MATLAB Compiler

В рамках данной работы, были написаны следующие функции с использованием язык MATLAB:

trainFis(pointsFile, anglesFile, termCount, epochCount) – обучает три сети ANFIS с использованием тренировочной выборки точек и углов (pointsFile, anglesFile); при тренировке указывается количество термов для FIS (termCount) и количество эпох обучения (epochCount). Возвращает массив из трёх обученных систем вывода, которые затем могут быть сохранены в файл с помощью стандартной функции MATLAB writefis.

loadFis(filename) – загружает обученную FIS из файла.

predictAngle(XYZ, fis) – получает на вход позицию рабочей точки и использует базу правил обученной системы нечеткого вывода для определения соответствующего .

В результате обучения сети генерируется готовая система нечеткого вывода (FIS) класса Sugeno, которая сразу же может быть использована в расчётах.

### Использование функций MATLAB в C++

Функции loadFis, predictAngle, predictAnglePlanar с помощью MCR были скомпилированы в динамическую библиотеку anfisLib и вызываются из пользовательского приложения в процессе работы.

Язык программирования MATLAB является высокоуровневым языком с динамической типизацией, основной особенностью которого являются широкие возможности для работы с матрицами: фактически, любая переменная в MATLAB представляется в матричном виде, включая скаляры (числа), которые обрабатываются как матрицы размерности 1х1. Это позволяет писать алгоритмы и функции, которые могут работать как со скалярными, так и с векторными и матричными данными.

В связи с этим, при использовании функций MATLAB в приложении на C++ необходимо использовать специальные типы данных, такие как class mwArray. Все экспортированные в библиотеку anfisLib функции принимают и возвращают значения в виде объектов класса mwArray; тип элементов массива определяется с помощью перечисления enum mxClassID.

Ниже приведён пример обращения к функции predictAngle из скомпилированной библиотеки:

*mwArray* **xyz**(3, 1, *mxDOUBLE\_CLASS*);

**xyz**(1) = **targetPoint**.*x*;

**xyz**(2) = **targetPoint**.*y*;

**xyz**(3) = **targetPoint**.*z*;

*mwArray* **yaw**, **th1**, **th2**;

*predictAngle*(1, **yaw**, **xyz**, fisYaw);

float **y** = **yaw**(1, 1);

### Графическая спецификация OpenGL

OpenGL (Open Graphics Library – открытая графическая библиотека) – спецификация, определяющая независимый от языка программирования платформонезависимый программный интерфейс для написания приложений, использующих двухмерную и трёхмерную компьютерную графику.

OpenGL ориентирован на две задачи:

* предоставить разработчикам единый API, скрывающий сложности адаптации и работы с графическими ускорителями;
* скрыть различия в возможностях аппаратных платформ.

Основным принципом работы OpenGL является использование наборов векторных графических примитивов в виде точек, линий и многоугольников, с последующей математической обработкой данных и построением растровой картинки на экране и/или в оперативной памяти [12].

OpenGL не включает средства для работы с окнами или получения пользовательского ввода – эта задача решается путём выбора и использования той системы управления, которая способна работать с конкретными аппаратными средствами. Кроме того, спецификация не подразумевает наличия команд высокого уровня для описания и отображения сложных трехмерных объектов – программист вынужден создавать нужную модель из ограниченного набора графических примитивов. Вместо этого, OpenGL предоставляет программисту единый интерфейс для создания и отображения примитивов независимо от используемых аппаратных средств.

Так, при использовании библиотеки OpenGL, рендеринг изображения происходит с использованием видеокарты компьютера. Поскольку в современных графических ускорителях количество процессорных ядер может достигать нескольких тысяч, использование средств отображения OpenGL позволяет получить многократный прирост производительности по сравнению с алгоритмами, использующими вычисления на ЦП.

### Использованные сторонние библиотеки

В связи с тем, что OpenGL предоставляет программисту лишь функции низкого уровня и не обладает средствами для управления оконной системой, при разработке приложения использовались несколько сторонних библиотек, предоставленных для свободного использования.

***SDL*** – данная библиотека реализует набор функций для управления оконными приложениями обеспечивает их кроссплатформеную реализацию [14]. SDL предоставляет простой интерфейс для создания оконных приложений в соответствии с используемой операционной системой, обеспечивает инициализацию стандартных параметров и организует удобную систему обработки событий ввода пользователя (движение мыши, нажатие клавиш, изменение размеров окна и т.д.).

Данная библиотека используется только в компоненте графического интерфейса пользователя.

***GLM*** – математическая библиотека (OpenGL Mathematics), предназначенная для использования совместно с OpenGL и основанная на спецификации OpenGL Shading Language [15]. Среди прочего GLM предоставляет классы, описывающие векторы, матрицы и кватернионы, а также предоставляет функции для работы с ними – операции векторного и скалярного произведения, поворота, масштабирования, умножения матриц и т.д.

Данная библиотека широко используется во всех компонентах разработанного ПО, поскольку предоставляет удобные механизмы для работы с трёхмерными векторами, с помощью которых можно представлять положение манипулятора в пространстве.

***IMGUI*** – используется для реализации графического интерфейса пользователя в так называемом «немедленном режиме» [16]. Поскольку ни OpenGL, ни SDL не предоставляют каких либо средств для создания и использования даже базовых управляющих компонентов графического интерфейса (кнопки, текстовые поля, списки данных и т.п.), было решено использовать данную библиотеку. Основным её преимуществом является «немедленный режим» – интерфейс не выносится на отдельный уровень, а встраивается напрямую в код программы. Это даёт возможность, например, «на лету» отображать и изменять значения переменных, без необходимости абстрагировать передачу параметров в отдельный уровень управления. Другим преимуществом является реализация библиотеки с использованием функций OpenGL.

Данная библиотека используется в компоненте графического интерфейса пользователя, а также в отладочных целях в компоненте управления движением манипулятора.

## Структура разработанного ПО

С учётом особенностей реализации, изложенных в разделе 3.2, диаграмма схема взаимосвязей модулей ПО, представленная ранее (см. рисунок 2.8), была дополнена компонентами, представленными на рисунке 3.2.

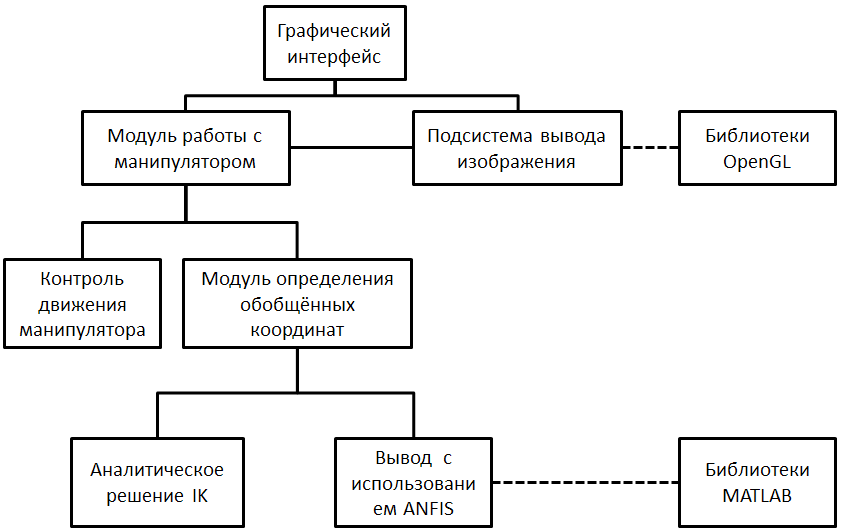


Рисунок . – Дополненная схема взаимосвязей модулей программного обеспечения

Компонент графического интерфейса использует библиотеки SDL и IMGUI для создания окна и работы с пользовательским вводом;

Модуль работы с манипулятором отвечает за обработку логики программы, включая как перемещение манипулятора, так и непосредственно реализацию метода определения обобщённых координат (как с помощью аналитического решения обратной задачи кинематики, так и с использованием нейро-нечеткой сети). В данном модуле также содержатся функции для генерации обучающей выборки на основе параметров манипулятора.

На рисунке 3.3 представлена диаграмма классов данного модуля, сгенерированая средствами Visual Studio.

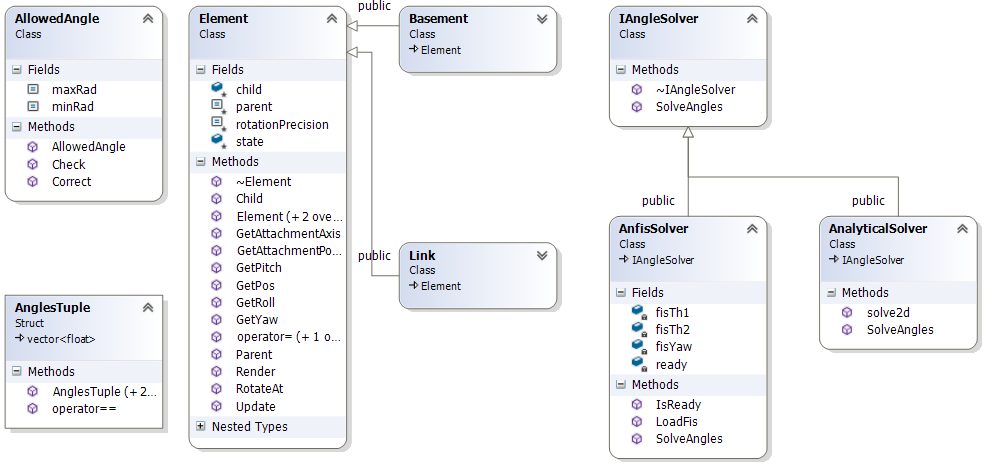


Рисунок . – Диаграмма классов модуля работы с манипулятором

Класс *AllowedAngle* используется для представления допустимых углов поворота звеньев манипулятора.

Класс *AnglesTuple* представляет собой кортеж обобщённых координат.

*Абстрактный* класс *Element* используется для представления составных частей манипулятора. Элементы могут образовывать структуру типа двунаправленный список, обеспечивая представление звеньев манипулятора, закреплённых одно на другом. Предоставлен общий интерфейс для работы с элементами манипулятора – получение углов поворота, позиции, места крепления и т.п.

Классы *Basement* и *Link* представляют, соответственно, вращающееся основание манипулятора и подвижны звенья. Данные классы реализуют интерфейс класса Element в соответствие с характерными особенностями конкретного элемента.

Интерфейс *IAngleSolver* представляет абстракцию для использования одного из нескольких возможных методов определения. Наследующие IAnleSolver классы должны реализовывать метод SolveAngles данного интерфейса. Этот метод должен принимать координаты целевой точки и ссылку (reference) на базовый элемент манипулятора (например, основание), который будет использоваться для расчёта обобщённых координат. Метод возвращает результат вычислений в виде экземпляра AnlesTuple.

Класс *AnalyticalSoler* реализует интерфейс IAngleSolver и при вычислении обобщённых координат использует аналитическое решение обратной кинематической задачи. Данный класс используется как демонстрационный пример для проверки работоспособности системы планирования движения манипулятора, а также для проверки корректности результатов работы метода, использующего нейро-нечеткую сеть.

Класс *AnfisSolver* используется для непосредственно демонстрации работоспособности разработанного метода. Экземпляр этого класса необходимо предварительно инициализировать, загрузив необходимые файлы сетей ANFIS с помощью метода LoadFis.

Подсистема вывода изображения отвечает за отображение трёхмерной сцены и графического интерфейса пользователя с использованием средств OpenGL. Данный модуль содержит функции отображения на экране координатной сетки, целевой точки, элементов манипулятора и т.п.

## Методика тестирования

Для проведения модульного тестирования программы использовался компонент Test Explorer среды разработки Visual Studio. При работе с данным компонентом необходимо создать отдельный проект, который будет использоваться для прогонки тестов. Кроме того, тестируемые компоненты ПО должны быть оформлены в виде статических библиотек, которые включаются в тестовый проект при сборке.

Были протестированы следующие части программы:

* планирование движения манипулятора;
* загрузка данных из файлов .fis;
* генерация тестовой выборки и уникальность обобщённых координат в ней
* вычисление обобщённых координат с использованием аналитического метода;

Вычисление обобщённых координат с использованием ANFIS предполагает наличие погрешностей в результатах вычислений, что исключает возможность модульного тестирования метода.

## Информация, необходимая для сборки и запуска

Поскольку в приложении используются дополнительные библиотеки, при сборке проекта необходимо указать соответствующие пути для:

* заголовочных файлов GLM;
* заголовочных файлов и статических библиотек SDL;
* заголовочного файла и статической библиотеки anfisLib;
* заголовочных файлов и динамических библиотек MATLAB Runtime.

Для запуска необходимо поместить динамические библиотеки SDL и anfisLib в одну папку с запускаемым приложением.

При решении обратной кинематической задачи с использованием ANFIS необходимо наличие сгенерированных в MATLAB файлов \*.fis, в которых содержится информация о настроенной с помощью ANFIS системы нечеткого вывода.

## Форматы входных и выходных данных

Сгенерированные и обученные в MATLAB системы вывода хранятся в файлах с расширением .fis [9].

Формат этих файлов схож с форматом файлов настроек .ini и имеет следующий вид:

[System]

Name='MF6 Theta1 demo'

Type='sugeno'

Version=2.0

NumInputs=3

NumOutputs=1

NumRules=216

...

[Input1]

Name='X'

Range=[-17 17]

NumMFs=6

MF1='XbehindFar':'gbellmf',[3.13820965639203 1.73814624069413 -17.3938988156664]

...

[Input2]

Name='Y'

Range=[-17 17]

NumMFs=6

MF1='YbehindFar':'gbellmf',[3.20313300177768 1.79976131704845 -17.3173657428312]

...

[Input3]

Name='Z'

Range=[0 13.0876]

NumMFs=6

MF1='ZbehindFar':'gbellmf',[1.81018296008799 1.79899760020991 0.351287399745908]

...

[Output1]

Name='Theta1'

Range=[0 0.654498]

NumMFs=216

MF1='out1mf1':'linear',[6.77821411945033 -6.13563703951326 2.69862586953868 -0.27184930011587]

...

В файле хранится общая информация о системе вывода, а также конкретные данные об используемых функциях принадлежности термов и правилах вывода.

В разработанном методе используются три сети ANFIS, каждая из которых отвечает за определение одного из трёх углов, составляющих обобщённые кординаты: и . Таким образом, для работы модуля нечеткого вывода необходимо загрузить информацию из трёх файлов .fis.

Данные файлы генерируются в процессе обучения сетей, а имя файла имеет следующий формат:

*Fisname = mf<MFC>\_epoch<EC>\_anglestep<AS>\_\_anfis\_<A>,*

где

– количество используемых термов входных переменных;

– число эпох обучения сети; – использованная обучающая выборка (П24 или П48 соответственно);

– угол, который определяет данная система вывода.

Также разработанное ПО используется для генерации обучающей выборки точек из рабочей зоны манипулятора в соответствии с алгоритмом, описанным в конструкторском разделе.

На основе допустимых углов поворота звеньев манипулятора генерируются списки точек рабочей зоны и соответствующих углов поворота. Данные списки затем сохраняются в файлы с указанием использованного углового шага.

Формат данных файлов представляет собой простое перечисление наборов координат для списка точек и наборов углов для списка обобщённых координат:

angles.pi24

-3.14159; 0; 0

...

points.pi24

-17; 1.48619e-06; 0

...

## Интерфейс пользователя

Графический интерфейс, реализуемый с помощью средств IMGUI и OpenGL, показан на рисунке 3.4.

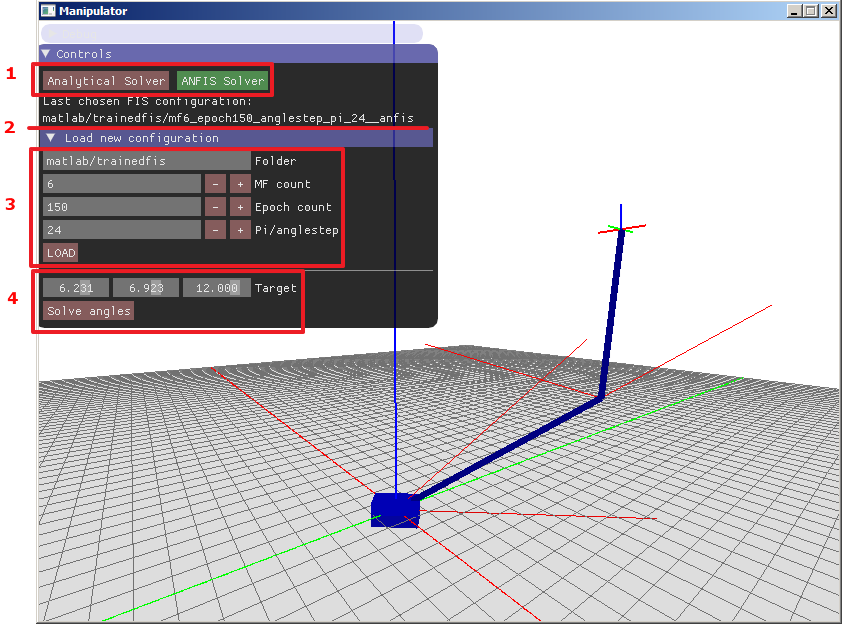


Рисунок . – Графический интерфейс приложения

При работе с приложением пользователь может выбрать способ определения обобщённых координат манипулятора: аналитическое вычисление или использование ANFIS (позиция 1 на рисунке 3.4). Кнопка, соответствующая выбранному методу, будет подсвечена зелёным.

При использовании ANFIS пользователь может выбрать параметры конфигурации сети (позиция 3 на рисунке 3.4), которая будет загружена: количество термов (MF Count), количество эпох обучения (Epoch count), угловой шаг выборки (Pi/anglestep). Если файлы, описывающие сети соответствующей конфигурации, существуют, то данная конфигурация сохраняется (позиция 2 на рисунке 3.4) и происходит инициализация системы нечеткого вывода на основе данных из файлов .fis.

После выбора метода управления и (при необходимости) настройки ANFIS пользователь может указать координаты целевой точки (позиция 4 на рисунке 3.4). При нажатии на кнопку Solve будут определены обобщённые координаты в соответствие с выбранным методом, а манипулятор придёт в движение. На рисунке 3.5 представлен результат определения обобщённых координат, соответствующих целевой точке с координатами .

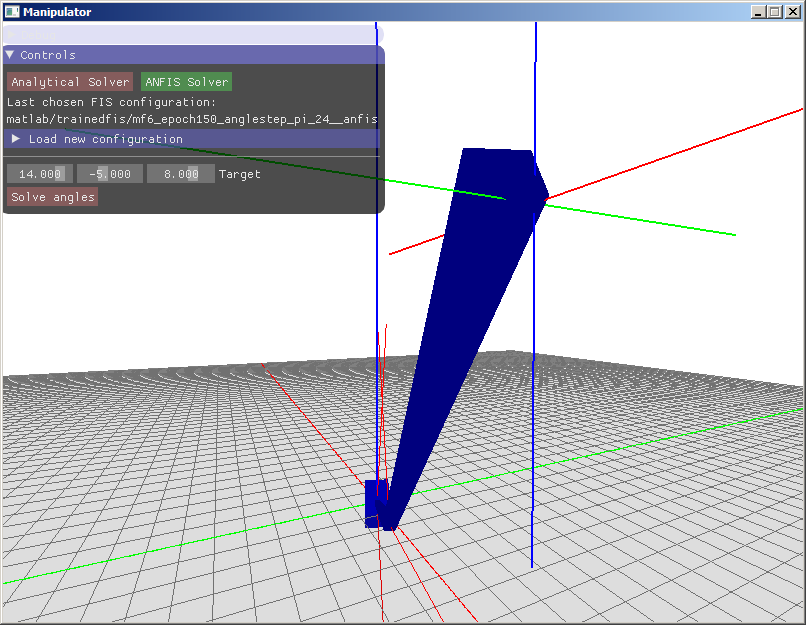


Рисунок . – Пример работы программы

## Выводы

В соответствии с описанными в конструкторском разделе алгоритмами, на языке С++ в среде разработки Microsoft Visual Studio под управлением ОС Windows был реализован программный метод управления манипулятором. Было произведено тестирование и отладка программного продукта. Описаны форматы используемых при работе ПО файлов и приведен пример интерфейса программы.

# Экспериментальный раздел

Целью данной работы является управление манипулятором, то есть приведение положения манипулятора из заданного в целевое. Как уже упоминалось в аналитическом разделе, данная задача состоит из трёх частей: планирования положения, планирования движения и планирования сил и моментов. В рамках данной работы строится упрощённая модель манипулятора, в которой не рассматривается планирование сил (то есть не моделируется работа приводов манипулятора), а планирование движения сводится к простому повороту звеньев манипулятора на требуемые углы. Основным объектом исследования, таким образом, является оценка планирование положения.

В данном разделе проводится экспериментальное исследование эффективности разработанного метода с использованием реализующего его программного обеспечения. В рамках экспериментального раздела также проводится сравнение результатов работы метода в зависимости от различных конфигураций сети и оценка времени обучения сети ANFIS.

## Критерии оценки

Использование ANFIS так или иначе приводит к возникновению погрешностей при предсказании требуемой ориентации манипулятора – в результате поворота манипулятора, результирующая рабочая точка будет смещена относительно положения, которое можно было бы вычислить аналитически. В качестве критериев оценки точности планирования положения были выделены следующие:

– погрешность позиции – расстояние между целевой точкой и точкой, соответствующей предсказанной ANFIS ориентации;

– относительная погрешность сети – отношение погрешности позиции i-й сети к погрешности j-й сети к радиус-вектору целевой точки (относительная величина);

– погрешность -го угла – разница между предсказанным ANFIS значением угла и значением, вычисленным аналитически.

Эти характеристики могут зависеть от следующих параметров:

* Модели манипулятора – количества и размера звеньев;
* Алгоритмов логического вывода;
* Количества термов в каждой лингвистической переменной;
* Функций принадлежностей каждого терма;
* Количества эпох обучения нейронной сети;
* Количества точек в обучающей выборке.

Параметры манипулятора в данной работе зафиксированы, также как и алгоритмы логического вывода – реализация ANFIS в MATLAB использует алгоритм Сугено. В качестве функций принадлежности во всех случаев принимается функция . Таким образом, при исследовании системы управления манипулятором возможно изменение размера обучающей выборки, количества эпох обучения и числа термов для каждой входной переменной.

Кроме того, интерес представляет время обучения ANFIS при заданном наборе параметров. Хотя данный процесс проводится только один раз и не влияет на быстродействие системы управления манипулятором, обучение нейронной сети на большой обучающей выборке или при использование большого (больше 3-х) количества термов может занимать значительное время.

## Обучающие выборки

В рамках работы было сгенерировано две обучающих выборки с угловым шагом (*П24*, 5 292 точки) и с угловым шагом (*П48*, 41 472 точки), представленные на рисунках 4.1 и 4.2.

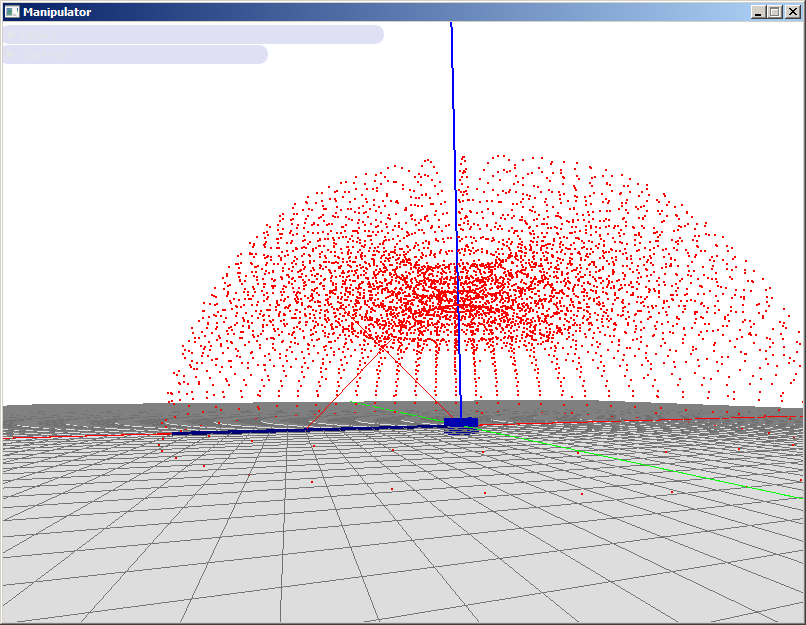


Рисунок . – Обучающая выборка П24

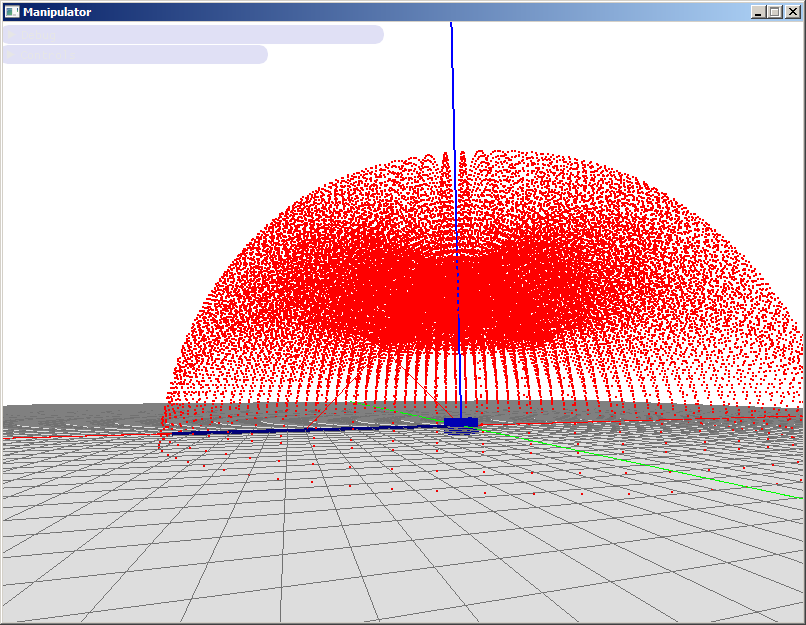


Рисунок . – Обучающая выборка П48

## Время обучения

При обучении наборов ANFIS была собрана следующая статистика зависимости времени обучения (в секундах) от числа термов:

*Таблица 4.1 – Время обучения на выборке П24*

|  |  |  |
| --- | --- | --- |
| **n** | **50 эпох** | **150 эпох** |
| *2* | 4 | 11 |
| *3* | 30 | 96 |
| *4* | 206 | 576 |
| *5* | 912 | 2958 |

*Таблица 4.2 – Время обучения на выборке П48*

|  |  |  |
| --- | --- | --- |
| **n** | **50 эпох** | **150 эпох** |
| *2* | 40 | 104 |
| *3* | 270 | 756 |
| *4* | 3634 | 5505 |
| *5* | 27603 | 41202 |

Обучение проводилось на компьютере, оснащённом процессором Intel Core i7 3.0GHz; все три сети в каждом наборе обучались параллельно. Как видно из таблиц, использование выборки П48 приводит к значительному увеличению времени обучения.

Анализ данных, приведенных в таблицах 4.1 и 4.2 показывает, что зависимость времени обучения сети от числа термов является экспоненциальной.

## Оценка погрешностей

C помощью обученных систем вывода были вычислены значения параметра на тестовой выборке точек. Выборка включала 100 точек из участка . На приведённых ниже рисунках представлены графики значений вычисленных погрешностей в различных точках выборки. Точки были пронумерованы от 1 до 100 в порядке увеличения вначале координаты Y, затем координаты Х.

На рисунках 4.3 – 4.6 представлены графики погрешности при использовании ANFIS с количеством термов равным 2, 3, 4 и 5, при 50 и 150 эпохах обучения, на выборках П24 и П48:

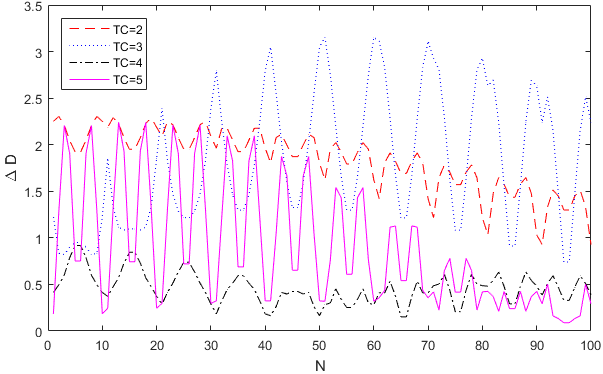


Рисунок . – Погрешность при 50 эпохах обучения на выборке П24

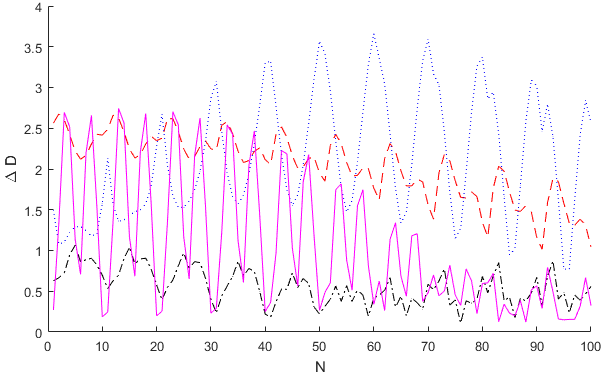


Рисунок . – Погрешность при 50 эпохах обучения на выборке П48

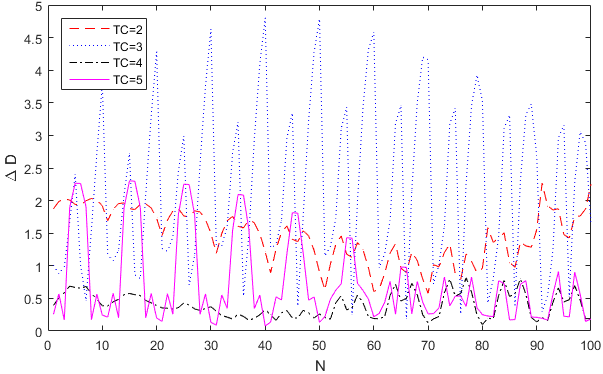


Рисунок . – Погрешность при 150 эпохах обучения на выборке П24

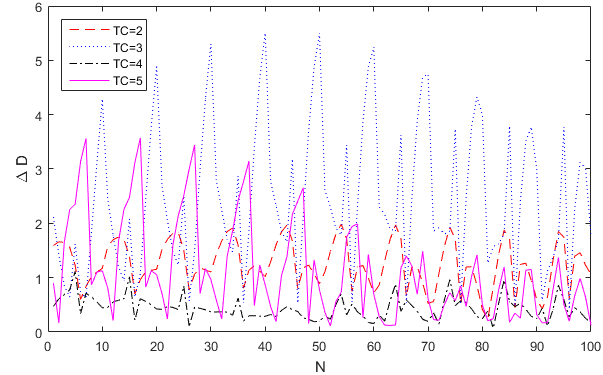


Рисунок . – Погрешность при 150 эпохах обучения на выборке П48

Как видно из графиков, в среднем наименьшую погрешность положения рабочей точки даёт ANFIS с 4-мя термами (сеть *A4*, штрих-пунктирный график) для каждой лингвистической переменной. При этом можно отметить, что ANFIS с нечетным количеством термов N даёт значительно бо́льшую погрешность по сравнению с четным количеством N-1.

Сравним результаты работы сети A4 из четырёх представленных выше конфигураций. Примем за «эталон» погрешность сети А4, обученной при 150 эпохах на выборке П24, и найдём отношение каждой из погрешностей к ней:. Результаты сравнения представлены на рисунке 4.7.

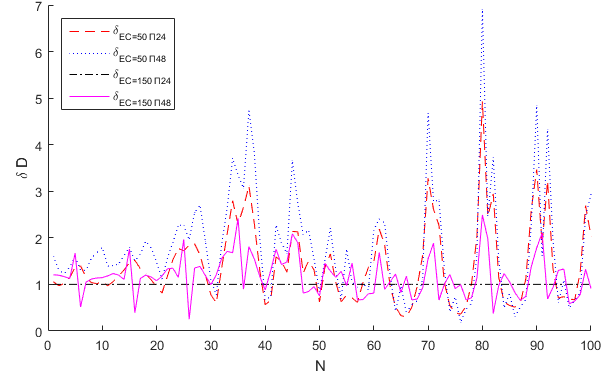


Рисунок . – Графики относительных погрешностей

Как видно из рисунка, сети A4, обученные на выборке П24 практически всегда дают меньшую погрешность по сравнению с сетями, обученной на выборке П48. При этом 150 эпох обучения дают лучший результат, чем 50. Кроме того, можно отметить, что разброс погрешностей увеличивается на второй половине тестовой выборки, т.е. при .

На основании полученных результатов было сделано предположение, что использование 6 термов может дать более точные результаты. Поскольку использование выборки П48 не дало прироста точности, было решено сгенерировать только две ANFIS, использующие 6 лингвистических термов и обученные на выборке П24 за 50 и 150 эпох. Обучение сетей заняло 10 206 секунд и 23 408 секунд соответственно.

Результаты сравнения погрешности с учетом новых сетей приведены на рисунках 4.8 и 4.9.

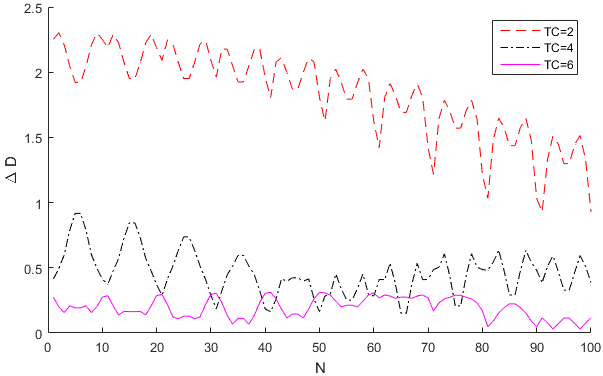


Рисунок . – Погрешность при 50 эпохах

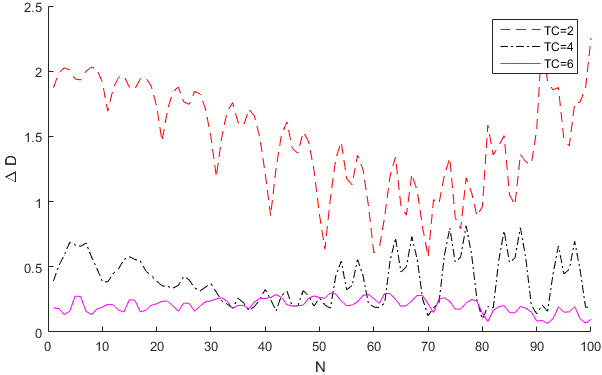


Рисунок . – Погрешность при 150 эпохах

Предположение подтвердилось, использование большего количества термов дало прирост точности и стабилизацию погрешности.

Погрешность не всегда является оптимальной оценкой точности, поскольку для манипуляторов с разной длиной звеньев значения расстояния между рабочей и целевой точками может отличаться. Предпочтительным является сравнение определенных ANFIS обобщённых координат (углов) с требуемыми углами, которые вычисляются аналитически.

Определив наилучшую из конфигураций ANFIS (6 термов, 150 эпох обучения, выборка П24), были проведены расчеты погрешностей для углов . Результаты расчётов приведены на рисунке 4.10.

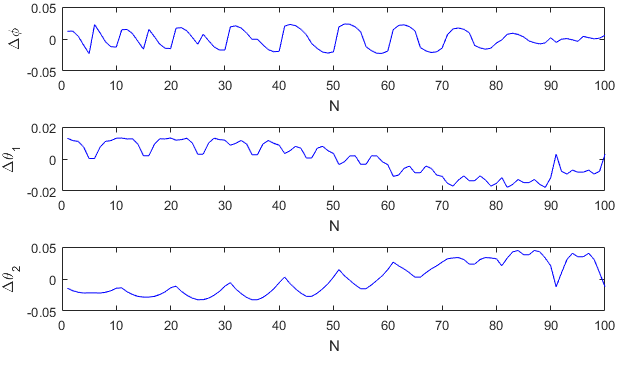


Рисунок . – Графики погрешностей углов

Погрешность определения каждого из углов не превышает 0.05 радиан, что составляет приблизительно 2.86°. Хотя при использовании большего числа термов можно было бы получить ещё меньшую погрешность, данная конфигурация сети обеспечивает довольно точное определение угла (см рисунок 3.5).

## Выводы

По результатам проведённого исследования можно сделать следующие выводы:

1. Использование большего количества точек в обучающей выборке нецелесообразно, поскольку это значительно увеличивает время обучения ANFIS, но при этом не даёт прироста точности.
2. Точность работы сети напрямую зависит от количества использованных термов. При этом время обучения сети возрастает экспоненциально.
3. Целесообразно использовать четное количество термов, чтобы минимизировать как величину погрешностей, так и разброс их значений.
4. Использование большего количества эпох обучения увеличивает точность работы сети, а в случае большего числа термов также уменьшает разброс значений.
5. Среди протестированных конфигураций ANFIS наилучший результат показывает система вывода, использующая 6 лингвистических термов для входных переменных, обученная на 150 эпохах с использованием выборки П24. Время обучения данной сети составило примерно 3 часа.

# Заключение

В результате выполнения выпускной квалификационной работы был спроектирован, реализован, протестирован и отлажен нейро-нечеткий модуль управления манипулятором с планарно-ангулярной кинематической схемой.

В процессе создания программной системы были достигнуты следующие результаты:

1. Разработан метод управления манипулятором с использованием нейро-нечеткого подхода.
2. Спроектирована структура программного обеспечения для моделирования работы планарно-ангулярного манипулятора.
3. Создано программное обеспечение, реализующее разработанный метод.

Проведено модульное тестирование и исследование работоспособности и точностных характеристик разработанного программного обеспечения. ПО полностью соответствует требованиям технического задания.

Достоинства разработанного программного обеспечения по сравнению с существующими методами:

* совмещение системы управления на основе нечеткой логики с однозначным определением обобщённых координат без необходимости запускать итеративный процесс;
* автоматическая настройка параметров системы нечеткого вывода;
* возможность применять используемый метод при использовании более сложных кинематических схем манипулятора.

Основными недостатками разработанной системы является зависимость от библиотек программного пакета MATLAB и значительное время обучения нейро-нечетких сетей.

Можно выделить следующие пути дальнейшего развития разработанной системы:

* реализация сети ANFIS на C/C++, что позволит избавиться от зависимости от библиотек MATLAB;
* оптимизация процесса обучения сети, что позволит уменьшить время генерации новых систем вывода;
* использование ANFIS также и для планирования движения манипулятора на основе его текущего положения.

# Список использованных источников

1. Интеллектуальные роботы: пособие для вузов / под общей ред. Е.И. Юревича / И.А. Каляев, В.М. Лохин, И.М. Макаров и др. - М.: Машиносторение, 2007.-360 с.: ил. - ISBN 5-217-03339-8.
2. Зенкевич С.Л., Ющенко А.С. Основы управления манипуляционными роботами : Учебник для вузов. – 2-е изд., исправ. и доп. М. : Изд-во МГТУ им. Н.Э. Баумана, 2004. – 480 с.: ил.(Робототехника / Под. ред. С.Л. Зенкевича, А.С. Ющенко)
3. Рутковская Д., Пилиньский М., Рутковский Л. Нейронные сети, генетические алгоритмы и нечеткие системы : Пер. с польск. И.Д. Рудинского. – 2-е изд., стереотип. – М.: Горячая линия – Телеком, 2013. – 384 с.: ил. ISBN 978-5-9912-0320-3
4. П.А. Ларюшкин, С.В. Палочкин - Рабочая зона манипулятора параллельной структуры с тремя степенями свободы. [Электронный ресурс]. Режим доступа: <http://ttp.ivgpu.com/wp-content/uploads/2015/10/339_23.pdf> (Дата обращения: 26.05.2016)
5. Mamdani, E.H. and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," International Journal of Man-Machine Studies, Vol. 7, No. 1, pp. 1-13, 1975.
6. Sugeno, M., Industrial applications of fuzzy control, Elsevier Science Pub. Co., 1985.
7. Jang, Jyh-Shing R (1991). Fuzzy Modeling Using Generalized Neural Networks and Kalman Filter Algorithm (PDF). Proceedings of the 9th National Conference on Artificial Intelligence, Anaheim, CA, USA, July 14–19. pp. 762–767.
8. Jang, J.-S.R. (1993). "ANFIS: adaptive-network-based fuzzy inference system". IEEE Transactions on Systems, Man and Cybernetics 23 (3)
9. С. Д. Штовба «Введение в теорию нечетких множеств и нечеткую логик»у [Электронный ресурс] : [интерактив учеб.]. Режим доступа: <http://matlab.exponenta.ru/fuzzylogic/book1/index.php> (Дата обращения: 03.06.2016)
10. Park, Joong-kyoo, "Inverse Kinematics Based on Fuzzy Logic and Neural Networks for the WAM-Titan II Teleoperation System." / Joong-kyoo Park // Master's Thesis, University of Tennessee, 2007.
11. The С++ Resource Network [Электронный ресурс] – Режим доступа: <http://www.cplusplus.com/> (Дата обращения: 04.06.2016)
12. OpenGL – The Industry Standard for High Performance Graphics [Электронный ресурс] – Режим доступа: <https://www.opengl.org/> (Дата обращения: 04.06.2016)
13. Visual Studio – Microsoft Developer Tools [Электронный ресурс] – Режим доступа: <https://www.visualstudio.com/> (Дата обращения: 04.06.2016).
14. Simple DirectMedia Layer [Электронный ресурс] – Режим доступа: <https://www.libsdl.org/index.php> (Дата обращения: 10.06.2016)
15. OpenGL Mathematics [Электронный ресурс] – Режим доступа: <http://glm.g-truc.net/0.9.7/index.html> (Дата обращения: 10.06.2016)
16. GitHub – ocornut/imgui: Bloat-free Immediate Mode Graphical User interface for C++ with minimal dependencies [Электронный ресурс] – Режим доступа: <https://github.com/ocornut/imgui> (Дата обращения: 10.06.2016)