Оглавление

[1. Расчетно-пояснительная записка 2](#_Toc358038944)

[1.1. Введение 2](#_Toc358038945)

[1.2. Техническое задание. 4](#_Toc358038946)

[1.3. Обзор существующих решений. 5](#_Toc358038947)

[1.4. Обзор существующих методов представления знаний. 9](#_Toc358038948)

[1.5. Метод представления знаний при проектировании модели. 23](#_Toc358038949)

[1.6. Предлагаемый метод моделирования 37](#_Toc358038950)

[1.7. Структура системы. 38](#_Toc358038951)

[2. Конструкторско-технологическая часть 38](#_Toc358038953)

[2.1. Технология программирования 38](#_Toc358038954)

[2.2. Выбор языка программирования 39](#_Toc358038955)

[3. Охрана труда 40](#_Toc358038956)

[3.1. Основные определения 40](#_Toc358038957)

[3.2. Требования к освещению на рабочих местах, оборудованных ПЭВМ 41](#_Toc358038958)

[3.3. Требования к качеству восприятия информации, отображаемой на дисплеях 43](#_Toc358038959)

[3.4. Эргономические требования к цветовым параметрам 45](#_Toc358038960)

[3.5. Требования безопасности к визуальным параметрам 47](#_Toc358038961)

[3.6. Искажения изображения по рабочему полю 48](#_Toc358038962)

[3.7. Требования безопасности к параметрам создаваемых полей 49](#_Toc358038963)

[3.8. Требования к конструкции 50](#_Toc358038965)

[3.9. Заключение 51](#_Toc358038966)

[4. Экологическая часть 51](#_Toc358038967)

[4.1. Термины и определения 52](#_Toc358038968)

[4.2. Методы измерений 53](#_Toc358038969)

[4.3. Критерии соответствия 54](#_Toc358038970)

[4.4. Заключение 54](#_Toc358038971)

[5. Решение задачи на ЭВМ 54](#_Toc358038972)

[5.1. Структура классов 54](#_Toc358038973)

[5.2. Пример функционирования 56](#_Toc358038974)

[6. Выводы 56](#_Toc358038975)

[7. Список литературы 56](#_Toc358038976)

# 1. Расчетно-пояснительная записка

## 1.1. Введение

Философы разделяют исследования в области искусственного интеллекта (далее – ИИ) на два направления. Первое направление ставит перед собой целью создание «программы, которая мыслит, а не моделирует мышление» [8]. Такая программа называется «сильным ИИ», в противовес второму направлению «слабого ИИ», являющийся программой, которая действует так, словно она разумна [8]. Другими словами, «слабый ИИ» - это программа, которая обладает способностями к решению задач в некоторых областях человеческой деятельности, равные или превосходящие способности человека решать такие задачи. Подход «слабого ИИ» является более практическим, и именно поэтому эта область настолько популярна на текущий день. В области «сильного ИИ» настолько заметных успехов сделано не было, хотя в ряде случаев эти два направления довольно сложно разграничить, и проекты находятся «между» этими двумя полюсами. Оба направления в общем случае используют один и тот же математический и алгоритмический аппарат.

Таким образом, исследования в области искусственного интеллекта привели к созданию огромного количества широко используемых на текущий день инструментов. Поисковые системы используют алгоритмы машинного обучения и обработки естественного языка для индексирования сайтов в сети Интернет. Алгоритмы обработки естественного языка также используются в автоматических переводчиках. Банки используют алгоритмы Data Mining для анализа кредитных данных. С помощью таких алгоритмов рекомендательные системы Интернет-магазинов предлагают пользователям купить товары, соответствующие их вкусу. Распознавание образов на фотографиях, видео- и звуковых данных осуществляется с помощью искусственных нейронных сетей. Экспертные системы, помогающие человеку принимать решения в различных областях жизнедеятельности, содержат огромное количество знаний – структурированной и обобщенной информации. Развитие робототехники привело к тому, что конструируются не только роботы-пылесосы и роботы-собаки, но и военные роботы, которые могут переносить в разы больше груза, чем человек, беспилотные самолеты, и даже роботы, которые могут охранять крупные мероприятия [17]. Компания Google Inc. в течение нескольких лет разрабатывает и тестирует проект Google Car – проект машины, которая имеет возможность ездить без прямого управления человеком [16]. Компания Mercedes-Benz, вслед за Google, также начала разрабатывать проект машины с «автопилотом» [15]. В индустрии компьютерных игр ИИ оппонента игрока является очень важным фактором интереса и, следовательно, популярности игры. Разработка игрового ИИ в первую очередь сосредоточена на проблемах поиска пути с учетом особенностей виртуального ландшафта, но в зависимости от жанра, важными направлениями могут быть моделирование жизненного цикла или распорядка дня игровых персонажей, применение ими командной или индивидуальной тактик против игрока, планирование будущих действий и коррекция своего поведения в зависимости от действий игрока. Такие модели поведения со временем становятся все сложнее, и игровой ИИ уже перестал быть «игрушкой», став сложным направлением разработок компьютерных игр, требующим специалистов своего профиля. Одним из перспективных направлений исследований ИИ является моделирование работы мозга на основе нейробиологических данных. Самым крупным из таких исследований является проект The Human Brain Project [18], конечной целью которого является наиболее приближенная к реальности модель мозга, способная помочь в понимании его механизмов и, в частности, болезней. Компания Numenta, Inc. разрабатывает систему под названием «Иерархическая Временная Память», которая представляет собой технологию машинного обучения, основанную на структурных и алгоритмических свойствах коры головного мозга, где происходят процессы мышления, планирования и других когнитивных функций [19]. Успешное развитие таких проектов в перспективе должно привести к возможности точного имитационного моделирования когнитивных функций человека, следовательно, и к созданию модели ИИ.

ИИ должен обладать возможностью хранить информацию внутри себя и оперировать ею. Такая возможность называется «представление знаний» [8]. Интеллект — способность приобретать, воспроизводить и использовать знания для понимания конкретных и абстрактных понятий и отношений между объектами и идеями и использовать знания осмысленным способом [3]. В дальнейшем под знаниями будем понимать некоторую структурированную информацию об окружающем мире. Исследование возможных способов представления знаний и операций над ними является областью пересечения фактически всех исследований ИИ. Целью данной работы является создание модели поведения искусственного животного, называемого «анимат», и имитационное моделирование этого поведения в виртуальной среде. Основной задачей будет являться разработка метода представления знаний анимата.

## 1.2. Техническое задание.

Анимат – это автономный агент, поведение которого должно следовать принципам поведения животных [9]. Анимат действует во внешней среде, взаимодействуя с объектами и принимая решение о своем поведении на основе данных, полученных в результате этих взаимодействий. По аналогии с живыми существами, он обладает следующими параметрами, указывающее на общее состояние его виртуального организма: «здоровье», «усталость» и «насыщенность». Анимат на базе разработанного инструментария должен научиться адаптироваться к любым изменениям во внешней среде, к примеру, смена места обитания или нападение хищника. Технически, он не должен допустить падение параметра «здоровье» до значения 0. Оценка «адаптивности» поведения будет происходить с помощью эмоциональных весов – оценок тех или иных действий по отношению к состоянию анимата. Как следствие, целевой функцией анимата будет являться оптимизация функции от эмоциональных оценок.

Для проведения экспериментов и исследований необходимо реализовать программу имитационного моделирования поведения аниматов, которое будет основано на предлагаемом методе представления знаний.

## 1.3. Обзор существующих решений.

Breve – программное обеспечение с открытым исходным кодом для имитационного моделирования агентов в виртуальной среде [21]. Поведения агентов задается либо с помощью встроенного языка программирования, либо с помощью языка программирования Python 2. Виртуальная среда представляет собой трехмерное поле, на котором можно размещать агентов. Присутствует также поддержка простейшей физики, например, столкновений между объектами. Встроенный язык Breve не является подходящим средством для решения текущей задачи в силу ограниченности своих возможностей, язык Python 2 не полностью поддерживается Breve, и разработка программы была прекращена несколько лет назад, поэтому Breve не является подходящим инструментом для реализации поставленной цели.

Darwinbots представляет собой программное обеспечение с открытым исходным кодом для имитационного моделирования искусственных организмов [22]. Такие организмы, называемые «ботами», живут в виртуальной двумерной среде, взаимодействуя друг с другом различными способами, эволюционируя и размножаясь, пытаясь выжить. Поведение ботов контролируется с помощью базовых программ поведения – «генов». Каждый ген содержит условия активации гена и программу поведения, описываемую на специальном языке. Задача бота – выжить, контролируя приход энергии от поедания пищи и расход энергии на совершение действий. Программа Darwinbots является очень наглядной и удобной программой для имитационного моделирования поведения и эволюции организмов вроде бактерий, но из-за простоты структуры ботов и невозможности настройки этих структур и строительства сложных моделей поведения, позволяющих обобщать и структурировать информацию из внешней среды, она не является программным обеспечением, подходящим текущим целям.

DigiHive – это программное обеспечение для имитационного моделирования поведения объектов, называемых «частицами» [23]. Частицы – это точки в абстрактном двумерном пространстве, обладающие способностями к передвижению по этому пространству и объединению в более сложные структуры с помощью простых программ, задаваемых на внутреннем языке. Такие сложные «организмы» за счет взаимодействия программ отдельных частиц могут влиять на другие частицы, манипулируя ими и изменяя их или свою собственную структуру. Программное обеспечение DigiHive предназначено в первую очередь для построения систем, свойства которой определяются за счет взаимодействия многих отдельных базовых единиц, и не может являться инструментом для имитационного моделирования когнитивных процессов сложных организмов.

EcoSim позволяет проводить имитационное моделирование экосистемы на основе отношений «хищник – жертва» между объектами-особями [24]. Внешняя среда для особей представляет собой плоскость. Модель поведения каждой особи задается в виде графа отношений между такими понятиями как действия, восприятие и решения. В процессе имитационного моделирования особи могут изменять этот граф, изменяя таким образом свое поведение адаптируясь к внешней среде. EcoSim является очень мощным средством для моделирования эволюции экосистем, но не позволяет гибко настраивать поведение каждой особи и организовывать информацию в знания, в связи с чем не является подходящим инструментом для решения текущей задачи.

Evolve 4.0 – это программное обеспечение с открытым исходным кодом, предназначенное для имитационного моделирования жизни одноклеточных и многоклеточных организмов в двумерном среде [25]. Такие организмы могут поедать друг друга, размножаться и объединяться в более сложные организмы. Поведение каждого такого организма задается на встроенном языке программирования, который основан на языке программирования FORTH. Как и DigiHive, Evolve 4.0 является реализацией клеточного автомата, и не предоставляет средств для накопления информации и ее обобщения в знания.

Framsticks – это свободное программное обеспечение для имитационного моделирования поведения искусственных организмов в трехмерной виртуальной среде [26]. Каждый организм состоит из «тела» и «мозга». Тело представляет собой совокупность базовых частей вроде статических частей, мускулов (которые способны двигать статические части), рецепторов (воспринимающие информацию из внешней среды) и сочленений между различными частями. Мозг такого организма является искусственной нейронной сетью, контролирующей поведение отдельных частей. Эволюция организмов производится с помощью функции приспособленности этого организма (например, показатель скорости передвижения), которую организм будет пытаться оптимизировать. С помощью Framsticks возможно построить сложные и интересные организмы, но из-за того, что «мозг» организма является искусственной нейронной сетью, неподходящей для предлагаемой модели представления знаний, данное программное обеспечение не подходит в качестве инструмента имитационного моделирования анимата.

NetLogo – это программное обеспечение с открытым исходным кодом для имитационного моделирования естественных и социальных явлений [27]. С помощью NetLogo возможно проводить имитационное моделирование поведения искусственных организмов в виртуальной среде. Обладает большим количеством инструментов для расчета статистик имитационного моделирования. Программы задаются на собственном языке программирования, и, как и в случае с другими похожими программами, такой язык в силу своей простоты не предоставляет возможности для реализации предлагаемой модели представления знаний.

Polyworld – это программное обеспечение с открытым исходным кодом для имитационного моделирования поведения аниматов в виртуальной среде [28]. Каждый анимат может заниматься поиском еды, размножаться, сражаться с другими агентами и охотиться на них. Поведение аниматов контролируется с помощью искусственных нейронных сетей. Как и в случае с Framsticks, Polyworld не подходит для текущей задачи из-за возможности контроля поведения анимата только с помощью искусственных нейронных сетей.

Для создания программного обеспечения имитационного моделирования модели поведения необходим язык программирования, позволяющий как реализовать задуманную концепцию, так и обеспечивающий легкую переносимость на другие языки программирования за счет своей наглядности. Встроенные языки программирования или описания рассмотренных программ имитационного моделирования не подходят для реализации предлагаемого метода в силу ограниченности своих возможностей. Кроме Breve, которая не подходит по вышеназванным причинам, не было найдено программы имитационного моделирования, которая позволяла бы реализовать свои собственные модели поведения на распространненом языке программирования. Поэтому было принято решение создать свою собственную программу имитационного моделирования и реализовать в ней модель поведения анимата на языке Python, т.к. он обладает как простым и ясным синтаксисом (что позволит легко реализовать предложенный метод на другом языке программирования), так и широкими возможностями.

## 1.4. Обзор существующих методов представления знаний

Самой простой и очевидной моделью представления знаний является продукционная модель. Она основана на правилах вида «ЕСЛИ (условие), ТО (действие)». Условием является проверка одной или нескольких независимых переменных [5]. Чаще всего такие модели используются в экспертных системах. Сравнивая факты, полученные из внешней среды, такая система может заносить в свою память новые правила. Применяя алгоритм вывода, система может строить гипотезы и заключения. Недостатки этой модели кроются в ее простоте: знания связаны неясным образом, практически невозможно оценить целостную картину знаний и выделить отдельные логически-связные части. Такая система, по сравнению с другими, обрабатывает знания гораздо менее эффективнее.

Логическая модель знаний основывается на концепции, что знания можно представить в виде совокупностей фактов и утверждений, которые рассматриваются как формулы в некоторой логике. Выведение новых знаний производится с помощью правил логического вывода. Логическая модель знаний строится на понятии «формальная система».

Формальная система **S** определяется как кортеж

Где

T – алфавит теории (конечное множество базовых символов);

F – перечислимое множество формул, построенных из элементов T с использованием некоторого набора синтаксических правил;

A – множество формул, называемых аксиомами;

R – конечное множество правил вывода.

Правило вывода – это отношение вида на множестве F. При этом формула g называется непосредственно выводимой из множества по правилу ; формулы называются посылками, g – следствием или заключением [10].

Наиболее широкое распространение при моделировании ИИ получила формальная система *исчисление предикатов первого порядка*. Символами исчисления предикатов являются:

1. Символы истинности true и false.
2. Символы констант – символьные выражения, начинающиеся с символа нижнего регистра.
3. Символы переменных – символьные выражения, начинающиеся с символа верхнего регистра.
4. Функциональные символы – это символьные выражения, начинающиеся с символа нижнего регистра. С функциями связывается арность, указывающая на число их аргументов.

Функциональное выражение состоит из константы арности n, за которой следует n термов , заключенных в круглые скобки и отделенных друг от друга запятыми.

Терм исчисления предикатов может быть константой, переменной или функциональным выражением.

Символы в исчислении предикатов также могут представлять предикаты. Они начинаются с буквы нижнего регистра и представляют собой n-арную формулу, возвращающую либо символ true, либо символ false. Если терм обозначает объекты и свойства («кошка», «синий»), то предикат – отношения между несколькими объектами («раскрашен(кошка, синий)») [6].

Недостатки исчисления предикатов заключаются в сложности описания ими непрерывных и неоднозначных явлений.

Для представления знаний одним из широко используемых инструментов являются графы. Рассматривается множество V, состоящее из соединенных некоторым образом точек. Назовем V множеством вершин и элементы – вершинами. Граф

с множеством вершин V есть некоторое семейство сочетаний или пар вида

указывающих, какие вершины считаются соединенными. Такие пары называется ребрами. Если ребра являются неупорядоченными парами, то такое ребро называется неориентированным ребром; иначе ребро называется ориентированным. Граф называется ориентированным, если каждое его ребро ориентировано, и неориентированным, если каждое его ребро является неориентированным [12].

В психологии графы используются для построения модели долговременной памяти человека. В общем виде в таких графах каждая вершина представляет собой некоторую сущность, а дуги между ними являются ассоциациями. В модели Куиллиана «Обучаемая система, понимающая язык» каждая такая сущность может принадлежать к одному из двух типов – элементу (структура, которая соответствует объекту, событию или идее) или свойству (сообщающую некоторую информацию об элементе), ассоциации между ними называются «стрелками» и связывают элементы друг с другом и с принадлежащими им свойствами [Клацки]. Андерсон и Боуэр разработали модель «Ассоциативная память человека», в которой ассоциации делятся на несколько типов, в отличие от модели Куиллиана [Клацки]. Сэлз использовал графы для представления иерархии понятий и наследования свойств и разработал теорию схематического упреждения, которая повлияла на работы по фреймам и схемам в ИИ. Такие теории, которые определяются значение объекта в терминах сети ассоциаций с другими объектами, называются ассоционистскими теориями. [6]

С точки зрения такой теории восприятие объекта происходит через некоторые понятия, которые являются частью всего нашего знания о мире и связываются соответствующими ассоциациями с другими понятиями. Эти связи представляют свойства и поведение объекта, к примеру, понятия «улитка», «раковина», «медленный» и «рога» ассоциативно связаны друг с другом на основании нашего предыдущего опыта (безусловно, исключая опыт людей, которые не видели улиток, но видели раковины и рогатый скот). Наше знание об улитках и истинность утверждений вида "улитка медленная" возникают из этой сети ассоциаций. Графы оказались идеальным средством для формализации ассоционистских теорий знаний за счет точного представления отношений посредством дуг и узлов.

Одной из первых моделей представления знаний являются семантические сети, которой, по сути, является модель Куиллиана. Семантическая сеть представляет знания в виде графа, узлы которого соответствуют фактам или понятиям, а дуги – отношениям или ассоциациям между понятиями. Как узлы, так и связи обычно имеют метки. [6]

Разделяют шесть видов семантических сетей, которые отличаются главным образом именами узлов, связей и выводами, которые можно делать в этих структурах [13]:

1. Definitional networks акцентированы на отношении «является чем-то» между некоторым типом и его подтипом. Такая сеть позволяет строить иерархии типов, поддерживая свойство наследия, т.е. копирования свойств супертипа его подтипами. Информация в таких сетях считается истинной по умолчанию.
2. Assertional networks – это сети для утверждения предположений. Информация в таких сетях считается правдивой, пока на ней не висит специальная метка «ложь».
3. Implicational networks используют отношение «следствие» как основное отношение между вершинами сети. Такие сети могут быть использованы для представления причин или выводов.
4. Executable networks включают в себя специальные механизмы, такие как передача меток или выполнение некоторых процедур, которые могут выполнять логический вывод или искать образы и ассоциации.
5. Learning networks – это сети с возможностью изменять свою структуру в зависимости от новой информации, поступившей в них.
6. Hybrid networks комбинируют в себе две или более семантические сети, которые взаимодополняют и взаимодействуют друг с другом.

Дальнейшее развитие семантические сети получили в виде теории фреймов и концептуальных графов.

Фрейм является способом представления знаний, который был разработан М. Минским. Основная идея фрейма заключается в следующем: когда агент попадает в новую ситуацию (или изменяет текущую так, что она становится новой), он загружает из памяти структуру, которая называется фреймом. Фрейм – это структура данных, представляющая собой некоторую обобщенную или стереотипную ситуацию, понятие, объект, категорию или иную сущность, которая обладает атрибутами и связана отношениями с другими фреймами. Каждому фрейму соответствует некоторая информация о том, как использовать этот фрейм, что ожидать на следующем шаге и что делать в случае, если ожидания не выполняются. [11] Фреймы – это схема представления знаний, ориентированная на включение в строго организованные структуры данных неявных (подразумеваемых) информационных связях, существующих в предметной области. Это представление поддерживает организацию знаний в более сложные единицы, которые отображают структуру объектов этой области. [6] Фактически, по своей структуре фрейм представляет собой набор ячеек, каждая из которых содержит специальную информацию. Ячейки бывают следующих видов в зависимости от хранимой информации:

1. Данные для идентификации фрейма.
2. Взаимосвязь этого фрейма с другими фреймами. Фрейм «улитка» может быть подтипом для фрейма «Gastropoda», который, в свою очередь, может служить подтипом фрейма «Mollusca».
3. Набор требований для фрейма, которые используются для определения соответствия новых объектов стереотипу данного фрейма. К примеру, улитка должна иметь раковину.
4. Набор процедур, используемых в заданных случаях.
5. Значения ячеек, определенные по умолчанию.
6. Информация для нового экземпляра. Некоторые ячейки могут оставаться незаполненными до того, пока не понадобиться определить информацию в них для отдельного экземпляра или решения некоторой задачи.

По аналогии с семантическими сетями, фреймы позволяют организовать иерархию знаний. При этом фреймы расширяют возможности семантических сетей, позволяя представлять сложные объекты не в виде большой семантической структуры, а в виде единой сущности: в такой сети каждое понятие представляется узлами и связями на одном и том же уровне детализации. Уровень детализации определяется текущей задачей: для одних целей необходимо рассматривать объект как единое целое, в то время как для других – как совокупность частей, учитывая детали его внутренней структуры. Такая сетевая модель позволяет естественным образом представить стереотипные сущности, классы, наследование и значения по умолчанию.

Хотя фреймы, подобно логическим и сетевым представлениям, являются мощным средством, многие проблемы организации сложных баз знаний все же должны решаться на основе опыта и интуиции программиста. [6] Фреймы являются очень обобщенным инструментом, и каждая задача требует от программиста своей реализации фреймов, учитывая специфику решаемой проблемы.

Концептуальные графы, разработанные Дж. Ф. Сова в качестве инструмента запросов на естественном языке к базам данных [14] — это конечный, связанный, двудольный граф. Узлы графа представляют понятия (конкретный или абстрактный объект в мире рассуждений), или концептуальные отношения (которые включают одно или несколько понятий). Метки дуг в концептуальных графах не используются. [6] Конкретные объекты – это фактические объекты из окружающего мира (домашняя кошка). В то же время, возможно сформировать некоторый обобщенный образ похожих объектов (абстрактная кошка). Концептуальные отношения описывают отношения, включающие одно или несколько понятий – за счет этого на языке концептуальных графов легко описывать отношения любой арности.

Каждый концептуальный граф является некоторым высказыванием, причем он может быть любой сложности, но должен быть конечным. База знаний, построенная на основе концептуальных графов, будет представлять собой набор таких высказываний.

Каждое понятие в концептуальном графе относится к какому-либо типу. Множество таких типов можно частично упорядочить с помощью введения операции «≤» - операции принадлежности к подтипу. Если s и t – типы и s ≤ t, то говорят, что t – это подтип s, s – это супертип t.

Над концептуальными графами определены правила, позволяющие выводить новые графы из имеющихся графов. Каждое правило оказывает один из следующих возможных эффектов на логические отношения между исходным графом *u* и результирующим графом *v* [14]:

1. Эквивалентность. Правила копирования и упрощения генерируют граф *v*, эквивалентный исходному: . Правило копирования позволяет сформировать граф, полностью идентичный исходному. Правило упрощения позволяет сокращать одинаковые отношения, удаляя полные копии отношений вместе с дугами. Эквивалентные графы являются истинными в полностью одинаковых моделях.
2. Специализация. Правила объединения и ограничения создают граф *v*, который включает в себя исходный: . Правило объединения позволяет интегрировать два графа в один, сливая идентичные узлы в один узел. Правило ограничения позволяет заменить узлы графа узлами, представляющими их специализацию.
3. Обобщение. Правила разделения и расширения формируют граф *v*, который включает в себя исходный граф: , разделяя граф на подграфы или заменяя узлы графы на их обобщение соответственно.

Концептуальные графы являются мощным инструментом для представления знаний, во многом потому, что они очень близки к естественному языку. С другой стороны, как в случае с фреймами и семантическими сетями, их чрезвычайная абстрактность и широта требует от программиста реализации собственной версии графа под конкретную задачу.

В повседневной жизни очень часто присутствуют явления и понятия, которые сложно формально описать в рамках двоичной логики из-за многозначного и неточного характера этих явлений. Данную проблему помогает разрешить инструмент нечетких множеств, разработанный Л. Заде. Концепция, на котором строится теория нечетких множеств, заключается в том, что принадлежность к множеству может определяться некоторой функцией, принимающей непрерывные значения, в отличие от дискретных отношений «принадлежит» (или «истина») и «не принадлежит» (или «ложь»).

Пусть **X** – некоторое (четкое) множество точек (или объектов). Нечетким множеством в некотором непустом множестве X, что обозначается как , называется множество пар:

где функция принадлежности нечеткого множества A:

Эта функция приписывает каждому элементу степень его принадлежности к нечеткому множеству A, при этом можно выделить три случая:

1. означает полную принадлежность элемента x к нечеткому множеству A ().
2. означает отсутствие принадлежность элемента x к нечеткому множеству A ().
3. означает частичную принадлежность элемента x к нечеткому множеству А.

Нечеткое множество А равно нечеткому множеству В, что записывается как А=В, тогда и только тогда, когда

Над нечеткими множествами определен набор операций, позволяющий выводить новые множества.

1. Пересечением нечетких множеств называется нечеткое множество с функцией принадлежности
2. Алгебраическим произведением нечетких множеств называется нечеткое множество , определенное как
3. Суммой нечетких множеств называется нечеткое множество с функцией принадлежности
4. Дополнением нечеткого множества называется нечеткое множество с функцией принадлежности

Неоднозначные понятия, например, «холодно», «тепло» и «горячо» очень удобно представлять в виде нечетких множеств. Каждое такое понятие будет являться нечетким множеством над множеством всех возможных температур. После того, как будет определена температура, это понятие можно использовать в дальнейших рассуждениях: «если холодно – спрячься». Такие рассуждения выглядят естественными по отношению к живым существам: важно понять, что оппонент близко и с ним нужно сражаться, а не какое точное расстояние до него.

Нечеткие множества в сочетании с другими моделями представления знаний создают очень интересный и практичный симбиоз, к примеру, нечеткие правила и нейро-нечеткие системы [7], которые оперируют нечеткими множествами и позволяют описывать многозначные процессы уже проработанными инструментами этих моделей в сочетании со спецификой аппарата нечетких множеств.

Биологические исследования часто служат толчком для создания новых инструментов и алгоритмов обработки информации. Одним из таких инструментов являются искусственные нейронные сети, представляющие собой упрощенную модель биологических нейронных сетей.

Мозг представляет собой чрезвычайно сложную, нелинейную, параллельную систему обработки информации. Он обладает способностью организовывать свои структурные компоненты, называемые нейронами, так, чтобы они смогли выполнять конкретные задачи (распознавание образов, обработку сигналов чувств, моторные функции) во много раз быстрее, нежели компьютеры. [4] Функционирование каждого нейрона заключается в пропускании нервных импульсов или возбуждений, сгенерированных некоторым внешним источником, вдоль своих отростков. Результатом такого процесса может стать, к примеру, сокращение мышцы или активация следующего нейрона. Тело нейрона называется сомой, из которого выходят два типа отростков: дендриты, по которым поступают входные импульсы, и аксон, по которым нейрон передает выходной импульс. Аксон на своем конце имеет некоторое число синапсов, которые соединяются с дендритами других нейронов, причем такое число связей могут быть огромное: каждый нейрон может принимать и передавать сигналы тысячам других нейронов. Синапсы играют роль репитеров информации, в результате функционирования которых импульс может усиливаться или ослабляться. Поступающие в нейрон сигналы, таким образом, могут оказывать возбуждающее или тормозящее воздействие. Нейрон суммирует все сигналы, и если их алгебраическая сумма превышает некоторое пороговое значение, то сигналы с выхода нейрона пересылается посредством аксона к другим нейронам. [7]

Рассмотрим простейшую модель нейрона. [7] Введем следующие обозначения: – количество входов нейрона; – входные сигналы; – синаптические веса, также называется порогом; – выходной сигнал нейрона; – пороговое значение; – функция активации.

Формула, описывающая функционирование нейрона, имеет следующий вид:

Функция активации может принимать различные формы в зависимости от конкретной модели нейрона. Алгоритм работы нейрона очень простой: каждый входной сигнал перемножается с соответствующим синаптическим весом. Полученные значения суммируются друг с другом и вместе с порогом. В результате возникает сигнал , который передается на вход функции активации. В зависимости от вида этой функции, нейрон посылает некоторое выходное значение другим нейронам. Как правило, это значения 1 и 0 (имеющие смысл «нейрон активирован» и «нейрон не активирован»), но бывают и другие варианты.

Нейроны объединяются в нейронную сеть, выступая в качестве элементарных ячеек обработки информации. Формальное определение нейронной сети следующее [4]: нейронная сеть – это громадный распределенный параллельный процессор, состоящий из элементарных единиц обработки информации, накапливающих экспериментальные знания и предоставляющих их для последующей обработки. Нейронная сеть сходна с мозгом с двух точек зрения:

1. Знания поступают в нейронную сеть из окружающей среды и используются в процессе обучения.
2. Для накопления знаний применяются связи между нейронами, называемые синаптическими весами.

Нейронная сеть состоит из нескольких слоев нейронов. Слой нейронов, куда поступают первичные входные сигналы (информация из внешней среды), называется входным слоем. Слой нейронов, сигнал с которых поступает на выход, называется выходным слоем. Между ними может быть некоторое количество слоев нейронов, называемых скрытыми слоями. Чаще всего сигналами обмениваются только нейроны, расположенные в разных слоях, т.е. нейроны одного слоя не могут передавать друг другу импульсы. Сигнал поступает от входных нейронов и распространяется далее по сети, активирует те или иные нейроны. В качестве выходного значения получают значения функций активации нейронов выходного слоя.



Рис. 1. Схема простой многослойной искусственной нейронной сети

Рисунок простейшей многослойной искусственной нейронной сети. Зеленым обозначен входной слой нейронов, голубым – скрытый слой нейронов, желтым – выходной слой нейронов.

Корректируя синаптические веса, нейронную сеть можно обучить распознавать различные вектора входных сигналов. Недостаток нейронных сетей заключается в том, что из-за своей структуры они в случае отсутствия примеров для обучения с учителем и потенциального многообразия ситуаций.

В качестве отдельного подхода к моделированию ИИ следует выделить агентно-ориентированный подход. Агент – это некоторая сущность, которая воспринимает окружающую среду посредством сенсоров и выполняет в это среде некоторые действия посредством эффекторов. Агент-робот может иметь камеры и различные моторы для взаимодействия со средой. Агент-программа получает данные из сети, файлы и другую информацию, и действует в среде, обрабатывая данные, отсылая пакеты по сети и изменяя файлы. [8] Агент может обрабатывать информацию любым способом, и в наиболее простом виде состоит их трех блоков: сенсорный блок (воспринимающий информацию), аналитический блок (анализирующий информацию из сенсоров и принимающий решение, какое действие необходимо выполнить) и эффекторный блок (реализующий принятое действие во внешней среде).



Рис. 2. Схема простого агента.

Агенты чаще всего объединены в единую систему – мультиагентную систему. Мультиагентная система — это вычислительная программа, блоки решения задач (решатели) которой расположены в некоторой среде, и каждый из них способен к гибким, автономным и социально организованным действиям, направленным (либо нет) на предопределенные реалии или цели. Таким образом, четырьмя свойствами интеллектуальной агентной системы, включающей программные решатели задач,

являются: ситуативность, автономность, гибкость и социальность. [6]

Ситуативность интеллектуального агента означает, что он воспринимает окружение, в котором действует, и может изменять это окружение. Примеры окружения для ситуативных агентов — это Internet, среда ведения игры или "обитания" робота.

Автономная система может взаимодействовать со своим окружением без прямого вмешательства других агентов. Для этого она должна контролировать свои действия и внутренние состояния. Некоторые автономные агенты могут также обучаться на своем опыте, чтобы улучшить свое поведение в дальнейшем.

Гибкий агент должен демонстрировать качества отзывчивости или предусмотрительности (в зависимости от ситуации). Отзывчивый агент получает стимулы от своего окружения и вовремя отвечает на них соответствующим образом. Предусмотрительный агент не просто реагирует на ситуацию в своем окружении, но и адаптируется, целенаправленно действует и выбирает альтернативы в различных ситуациях.

Агент является социальным, если он может соответствующим образом взаимодействовать с другими программными или человеческими агентами. Таким образом, агент является частью системы, который выполняет некоторую подзадачу глобальной задачи системы. Взаимодействия социального агента с другими агентами ориентированы на достижение этой задачи и происходят не с помощью вызовов конкретных методов друг друга, а с помощью запросов на выполнение необходимых действий.

Модели представления знаний, как правило, разрабатываются для некоторого класса задач, и вне этого класса не являются настолько эффективными, насколько являются специально разработанные алгоритмы для других классов задач. Для представления знаний при моделировании поведения анимата наиболее подходящими являлись бы методы концептуальных графов и фреймов, но их применение потребовало бы собственной реализации данных моделей, подходящих к текущей задачи. Предложенный метод является методом как наследующим некоторые черты означенных моделей, так и обладающий своими собственными уникальными чертами.

## 1.5. Метод представления знаний при проектировании модели.

Предлагаемая модель ИИ основывается на когнитивных картах – некотором базовом знании о мире. Ключевые идеи, положенные в основу этой концепции, сходны с ключевыми идеями фреймовой модели:

1. Когда анимат попадает в новую ситуацию или изменяет текущую так, что она становится новой, он загружает из памяти набор структур, которые соответствуют объектам данной ситуации.
2. Один и тот же объект, в зависимости от текущей задачи, можно рассматривать и как некоторую целую, неделимую сущность, и как набор отдельных частей. Это зависит от степени детализации этого объекта, необходимой для решения текущей задачи.

Когнитивные карты являются понятием из когнитивной психологии и впервые были введены Толменом. Когнитивная карта – это активная, направленная на поиск информации структура [1]. Такая структура относится к некоторой области внешней среды. Когнитивные карты, которые вводятся ниже, являются, с одной стороны, обобщением психологических когнитивных карт (и в этом они близки к фреймам, семантическим сетям и концептуальным графам), а с другой стороны – некоторым упрощением, поскольку оригинальные когнитивные карты включают в себя гораздо более широкий биологический и психологический аспекты (к примеру, как и каким образом направляется внимание на отдельные объекты на карте), непрактичный при применении в реализации текущих целей.

Неформальное определение когнитивной карты следующее: это отражение ограниченной части окружающей среды в памяти анимата, содержащее воспринятые схемы этой части и отношения между схемами относительно некоторого уровня детализации, на котором следует рассматривать эти схемы. В каждый момент времени у анимата активна только одна когнитивная карта, соответствующая текущей местности. Такая карта называется активной когнитивной картой.

Таким образом, мышление анимата происходит в границах когнитивной карты, которая соответствует воспринятой им некоторой части внешней среды. Информация вне этих границ не будет использована в, т.к. этого не требует текущая когнитивная карта. Информация внутри этих границ будет представлять собой «карту» воспринятых объектов и различных связей между ними. Такие границы могут определяться различными способами, к примеру, это могут быть границы между разными территориями («лес» и «поле») или разными ситуациями («я ищу еду» и «я убегаю от хищника» - разные ситуации), но общее правило – такая карта должна отображать некоторую территорию или стереотипную ситуацию. Для оценки ситуации (и последующего принятия решения) используются специальный вектор оценок.

**Опр.** Вектор оценок – это упорядоченный набор значений .

В данной работе используется вектор из двух оценок , такой вектор называется общей эмоциональной оценкой или просто эмоциональной оценкой. называется позитивной эмоциональной оценкой, - негативной. Множество всех возможных пар обозначается как **W**.

Необходимость введения двух шкал оценок вместо одной возникает из-за того, что при взаимодействии с миром возможны неоднозначные ситуации, когда получается как позитивный результат (к примеру, «много еды»), так и негативный одновременно («много еды, но пришлось сильно постараться»). Несколько шкал позволяют создать систему, более гибко адаптирующуюся под окружающую среду.

При анализе какого-либо события или объекта в окружающей среде, анимат в первую очередь ориентируется на эмоциональные оценки окружающей среды. Таким образом, анимату нет необходимости постоянно перебирать все возможные варианты своих и чужих действий, чтобы принять решение – достаточно ориентироваться на имеющиеся оценки и корректировать их при получении нового опыта.

Когда объект на когнитивной карте воспринимается как единое целое, то он называется схемой. Неформальное определение схемы следующее: это отражение опыта взаимодействия анимата с объектом или объектами внешнего мира, и может обозначать конкретное или абстрактное понятие. Множество всех схем обозначается **S**.

Схемы на каждой когнитивной карте связаны между собой отношениями.

**Опр.** Отношение **r** некоторого типа – упорядоченный набор , где , – множество доступных меток для отношения данного типа. Схема **s1** называется схемой-источником взаимодействия, схемы – целями**.**

Множество всех отношений обозначается **R**.

Разделяют три типа отношений по логике взаимодействия между схемами в отношении.

**Опр.** Отношение *p*-типа – это отношение существования пути некоторого вида между схемами **s1** и **s2**. Множество меток Tp формируется следующим образом: все возможные способы перемещения (например, «пешком», «вплавь») имеют либо истинное значение (означающее, что можно переместиться от одной схемы к другой таким типом перемещения), либо ложное (означающее, что невозможно переместиться таким образом от одной схемы к другой), а также могу являться либо отношением к объекту, либо к точке перехода (об этих двух случаях будет сказано ниже). Мощность множества возможных способов перемещения, таким образом, увеличивается в четыре раза по сравнению с множеством возможных типов перемещений, задаваемых исходно программистом. В множество исходных способов перемещений возможно также добавлять метки вида «далеко идти пешком» или «близко идти пешком», которые показывают одинаковые способы перемещения, но имеют разное логическое значение.

**Опр.** Отношение *a*-типа – это отношение действия для двух схем, означающее, что **s1** может совершить указанное действие по отношению к **s2** и получить какой-либо результат. Множество меток содержит все возможные действия и формируется аналогично множеству : для каждого исходно заданного программистом действия в множество добавляется метка, соответствующая невозможности совершения такого действия.

**Опр.** Отношение *c*-типа – это отношение композиции для двух схем, означающее, что схема **s1** является частью схемы **s2** (к примеру, лапы у животного). Множество меток Tc состоит из одной схемы «является частью».

Когнитивная карта - это мысленный образ анимата некоторой ограниченной части внешнего мира. Это может быть абстрактная карта местности или карта взаимодействий между объектами, которые определяются отношениями.

**Опр.** Когнитивная карта – упорядоченная пара **c** = <**Sc**, **Rc**>:

Множество всех сформированных когнитивных карт обозначается как **C**.

**Опр.** Слой *p*-типа (или *p*-слой) когнитивной карты *c* – это когнитивная карта, содержащая только отношения *p*-типа и схемы, связанные этими отношениями. Этот слой показывает всевозможные пути между объектами на исходной карте.

**Опр.** Слой *a*-типа (или *a*-слой) когнитивной карты *c* – это когнитивная карта, содержащая только отношения *a*-типа и схемы, связанные этими отношениями. Такой слой представляет собой карту действий между различными схемами, то, как схемы могут взаимодействовать.

**Опр.** Слой *c*-типа (или *c*-слой) когнитивной карты *c* – это когнитивная карта, содержащая только отношения *c*-типа, схемы, связанные этими отношениями и схему «ядро», которое будет введено ниже. Этот слой показывает, из каких частей состоит исходная когнитивная карта, другими словами – структуру исходной когнитивной карты.

**Опр.** Схема **s** – это кортеж , где

Когнитивная карта – это когнитивная карта, обязательно содержащая *с*-слой и опционально содержащая *p*-слой, показывающая структурные свойства схемы и возможные пути между объектами на этой схеме. Множество – это множество кортежей, описывающих возможное поведение данной схемы как цельного объекта внешней среды. Каждый кортеж является тройкой «когнитивная карта – когнитивная карта – оценка». Первая когнитивная карта с помощью отношений *a*-типа показывает возможное поведение такой схемы и обязательно содержит эту схему. Вторая когнитивная карта представляет собой результат такого действия. Все схемы и отношения на этой карте в случае совершения действия заменяют похожие схемы на когнитивной карте, где было совершено это действие, или добавляются к ней. Оценка является эмоциональной оценкой данного действия по отношению к анимату. На основании таких оценок действий других схем и самого себя, анимат принимает решения.

Также условимся, что выражения вида «схема содержит схему » или «схема содержит *p*-слой» будут эквивалентны выражениям «когнитивная карта схемы содержит схему » и «когнитивная карта схемы содержит *p*-слой» соответственно.

Для ограничения способности к детализации схем, необходимо ввести специальный класс схем. Атомарная схема – это схема, которая не может состоять из частей по некоторым причинам, например, обусловленным входными данными и способами их обработки (такими схемами могут являться схемы «цвет» или «форма»). Атомарная схема может иметь некоторое конкретное значение (цвет = «зеленый», здоровье = «изранен»), значение «неизвестно» или значение «любое».

Для анимата такие атомарные схемы задаются программистом на этапе создания виртуальной окружающей среды для анимата. Когнитивная карта такой схемы пуста. Неатомарные схемы будем называть сложными.

Введем специальные схемы – это пустая схема, схема «ядро» и схема «я» .

Пустая схема (также обозначающаяся как ) – это схема, которая описывает «нулевую» схему, «пустоту», отсутствие схемы. Когнитивные карта, описывающая части этой схемы, и карты поведения таких схем пустые.

Схема «ядро» (также обозначающаяся как *K*) через отношения *c*-типа с другими схемами обозначает, что когнитивная карта, на которой содержится такая схема, имеет какие-либо свойства или состоит из каких-либо частей. «Ядро» обозначает некоторый «центр» или «смысл» схемы, которой соответствует эта когнитивная карта. На каждой карте, соответствующей схеме и не являющейся атомарной схемой, присутствует «ядро», но при этом такая схема только одна и несет исключительно функциональную нагрузку, не имеет карт поведения и имеет пустую когнитивную карту.

Схема «я» (также обозначающаяся как *I*) - это схема, указывающая на самого анимата, показывающая через отношения его связи с другими схемами, своего рода «самовосприятие» анимата – как он понимает, куда можно дойти и с какой схемой и как можно взаимодействовать, что именно он получит от этого взаимодействия, какой результат. Такая схема аналогична упоминанию о себе в третьем лице.

Отношение p-типа показывают принципиальную возможность найти путь определенным способом передвижения между двумя схемами на некоторой когнитивной карте. В зависимости от схемы, необходимо рассмотреть три варианта возможного перемещения между этими схемами. Будем рассматривать некоторое отношение .

В первом случае схема является некоторым объектом на когнитивной карте, до которого можно переместиться от схемы . В таком случае говорится, что анимат находится на определенной схеме в данной когнитивной карте. Логически это означает, что анимат находится рядом с этой схемой. Конкретная реализация понятия «рядом», как и алгоритмы нахождения пути между схемами, ложится на плечи программиста. Схема в этом случае не должна иметь *p*-слоя своей когнитивной карты. Для такой схемы метка равна «некоторое действие – истина – объект».

Во втором случае схема является некоторой местностью, «внутри» которой можно находиться, но с которой нельзя находиться «рядом». Как правило, это соседние большие территории. Для перехода на когнитивную карту, соответствующую такой схеме, анимату необходимо пройти по соответствующему *p*-отношению. Для такой схемы метка равна «некоторое действие – истина – точка перехода».

Третий случай представляет собой объединение предыдущих двух случаев. Другими словами, схема содержит как *p*-слой, так и сама является объектом во внешнем мире, с которым можно находиться «рядом». Для такой схемы существует два отношения. Первое, обозначающая первый случай, имеет метку равную «некоторое действие – истина – объект». Второе отношение имеет метку равную «некоторое действие – истина – точка перехода».

На рисунке показаны *p*-слои схем, иллюстрирующие все три случая. Схемы «Роща» и «Водоем» являются вторым случаем схемы для «Водоема» и «Рощи» соответственно. Схема «Ручей» является схемой первого случая для «Водоема», «Улей» является схемой первого случая для схемы «Дерево #1», схемы «Дерево #2» и «Дерево #3» являются схемами первого случая для «Рощи» и «Водоема». Схема «Дерево #1» является схемой третьего случая для «Рощи» - она содержит *p*-слой, перейти на которой можно, если пройти по *p*-связи между этой схемой и ею самой с меткой «взобраться – истина – точка перехода».



Рис. 3. *P*-слои когнитивных карт схем «дерево #1», «роща» и «водоем».

Анимату необходимо не только понимать, что происходит на текущем «рабочем» участке внешней среды, но и ориентироваться на более глобальном плане. Для представления карты всей исследованной аниматом территории окружающей среды введем карту путей.

**Опр.** Карта путей

.

Карта мира представляет собой карту всех тех схем, которые являются картами территории, т.е. содержат *p*-слой. Связи между этими схемами являются *p*-отношениями, по которым возможно перейти из одной схемы в другую. Для каждой схемы **а** на карте путей любая связанная с ней схема **b** является схемой, находящейся на *p*-слое схемы **a**.

Карта путей для описанных выше схем выглядит следующим образом:



Рис. 4. Карта путей.

Карту путей необходима анимату для выбора активной когнитивной карты.

Получая из внешней среды информацию о некотором цельном объекте, анимат должен уметь интерпретировать полученные данные. Информация об объекте поступает в виде когнитивной карты, другими словами, в виде набора некоторых схем и связей между ними. Если какая-либо схема текущего распознанного объекта не является атомарной схемой, то такую схему необходимо предварительно распознать отдельно. Процесс повторяется до тех пор, пока схема, полученная изначально, не будет полностью распознана – или как старая схема, или как новая. Для выполнения такого процесса введем семантическую карту.

**Опр.** Семантическая карта .

Семантическая карта – это граф, в котором каждая вершина представляет собой некоторую схему, а дуги между ними обозначают, что одна схема является структурной частью другой. Изначально на семантической карте есть только пустая схема (для удобства будет считать, что она считается связанной *c*-отношением с каждой атомарной схемой), схема «я» и набор базовых атомарных схем, задаваемых программистом заранее (в данной модели это параметры «здоровье», «усталость» и «насыщенность»).

Алгоритм распознавания схем следующий.

1. Инициализировать , входящая схема .
2. Если схема – неизвестная атомарная схема, то добавить к .
3. Если схема – известная атомарная схема, то перейти на шаг 8.
4. Выбрать схему из присутствующих на *c*-слое когнитивной карты входящей схемы , которая не содержится в множестве .
5. Принять за входящую схему и применить алгоритм к ней.
6. Поместить схему в .
7. Если количество выбранных схем не равно количеству схем на *с*-слое когнитивной карте входной схемы, то перейти на шаг 4.
8. Если распознана как уже известная схема, то присвоить , перейти на шаг 13.
9. Добавить в множество схем из , которые не содержатся в .
10. Добавить в карту
11. Добавить в множество
12. Добавить в .
13. Вернуть .

Разберем на примере, как будет распознана схема, состоящая не только из атомарных схем, но и из более сложных схем.

Пусть схемы «ствол» и «дерево» имеют следующие когнитивные карты:



Рис. 5. *C*-слои когнитивных карт схем «ствол» и «дерево с ульем».

Предположим, что все эти схемы и несколько иных схем уже были распознаны ранее. Тогда семантическая карта анимата выглядит следующим образом:



Рис. 6. Семантическая карта

Пусть анимат пытается распознать объект во внешней среде, который является схемой «Дерево с ульем». В качестве входной информации ему поступает следующая когнитивная карта:



Рис. 7. Входная когнитивная карта неизвестной схемы.

На этом рисунке «схема #2» является неизвестной сложной схемой, которую необходимо распознать отдельно.

В качестве первой схемы выберем схему «Листья». Она является известной атомарной схемой, поэтому распознается как известная и помещается в множество . В качестве второй схемы выберем схему «Улей». Так же, как и схема «Листья», она распознается однозначно и помещается в множество . Третья выбранная схема – это схема «Схема #2», которую необходимо рекурсивно распознать. Ее карта выглядит следующим образом:



Рис. 8. Когнитивная карта схемы «Схема #2»

Эта карта содержит два отношения: («К», «Ветка») и («К», «Улей»). Следуя алгоритму, необходимо рекурсивно распознать схемы «Ветка» и «Улей», которые являются известными атомарными схемами. Эти схемы добавляются в множество . Поскольку количество схем в множестве равно количеству схем на с-слое когнитивной карты «Схема #2», то на шаге 8 такая схема распознается однозначно как схема «Ствол» , что будет результатом работы алгоритма по распознаванию схемы «Схема #2». Возвращаясь из рекурсии на уровень выше, добавим схему «Ствол» в множество исходной входной схемы. Это множество теперь содержит все три схемы, поэтому алгоритм переходит на шаг 8. На этом шаге исходная схема однозначно распознается как схема «Дерево с ульем», и результатом работы алгоритма будет являться эта схема. В случае, если в процессе распознавания были встречены неизвестные ранее схемы, то они будут добавлены в общее множество схем, и, следовательно, в семантическую карты, и могут быть распознаны в будущем.

Над множеством схем определен оператора равенства:

Две атомарные схемы называются равными, если выполняются следующие условия:

1. Обе схемы распознаны как одна и та же схема на семантической карте.
2. Значения атомарных схем равны или либо хотя бы одно из значений равно значению «любое»;
3. Значение каждой схемы не равно значению «неизвестно».

Две сложные схемы называются равными, если их когнитивные карты равны.

Над множеством отношений определен оператор равенства:

Два отношения называются равными, если тип обоих отношений одинаков, равны все схемы в этих отношения и равны метки этих отношений.

Операция включения одной когнитивной карты в другую принимает значение «истина», тогда и только тогда, когда когнитивная карта a является подкартой когнитивной карты b, т.е. все схемы и отношения карты a содержатся в b.

Операция равенства двух когнитивных карт следует из операции включения:

Операция слияния (также называемая сложением) двух когнитивных карт объединяет две исходные когнитивные карты. Получившаяся карта включат в себя исходные карты.

Результатом операции умножения (или пересечения) двух когнитивных карт является когнитивная карта, которая содержит схемы и связи между ними, присутствующие на обеих иcходных когнитивных картах. Получившаяся карта представляет собой подкарту, которая содержится в исходных картах, позволяя выделять общие схемы и отношения между разными когнитивными картами.

Операция вычитания одной когнитивной карты из другой в качестве результата имеет левую исходную когнитивную карту без схем и связей, содержащихся в правой исходной когнитивной карте. Такая операция позволяет определить уникальные для левой исходной когнитивной карты схемы и отношения.

Операция наложения – это операция совмещения в одной карте нескольких карт, которая позволяет формировать полную информацию об ограниченной части внешней среды как на основании информации о текущей активной карте, так и на основании информации из памяти. Когнитивные карты, которые могут быть релевантными активной когнитивной карте, загружаются из памяти и накладываются на текущую карту. Результатом этой операции является либо обновленная правая исходная когнитивная карта с новой информацией, взятой из левой исходной карты, либо правая исходная когнитивная карта (такой исход означает, что левая исходная когнитивная карта не имеет информации, полезной для правой карты).

## 1.6. Предлагаемый метод моделирования

В каждый момент времени анимат находится в некоторой области, которой соответствует некоторая когнитивная карта. Такая карта называется активной когнитивной картой. Взаимодействие с окружающей средой заключается в построении когнитивных карт на основе данных из среды и решений к действиям, принимаемых на основе анализа этих карт. Новые когнитивные карты схем и карты, отражающие поведение этих схем, загружаются в память анимата, и будут использовать в будущем для принятия решений и действий. Эмоциональная оценка различных действий высчитывается как разность предыдущим и текущим значениями параметров анимата.

В зависимости от собственного состояния и окружающей среды, анимат может либо целенаправленно выполнять некоторую задачу, либо находиться в состоянии исследовательского поведения.

Целенаправленное выполнение некоторой задачи происходит тогда, когда возникает некоторая опасная ситуация, угрожающая жизни. В таком случае анимат старается избегать присутствовать на когнитивных карт с высокой негативной эмоциональной оценкой.

Когда вокруг анимата «спокойная» обстановка и ему не нужно удовлетворять свои потребности, он занимается исследовательским поведением. «Исследовательское поведение — это поведение, направленное на познание реальности путем практического взаимодействия с ней» [2]. Экспериментируя с окружающей средой, анимат получает новые знания, который будет использовать в дальнейшем и которые помогут ему лучше адаптироваться к среде.

## 1.7. Структура системы.

В качестве входных данных программа имитационного моделирования принимает два текстовых файла с описанием начальных параметров внешней среды. Первый текстовый файл является описанием схем и когнитивных карт, которые будут участвовать в имитационном моделировании. Второй текстовый файл задает карту внешней среды и представляет собой массив символов, каждый из которых соответствует некоторой схеме, а также набор пар «имя схемы – символ».

В качестве выходных данных программа выдает графическое поле, на котором в зависимости от месторасположения объектов нарисованы графические представления этих объектов.

# 2. Конструкторско-технологическая часть

## 2.1. Технология программирования

Для реализации поставленной задачи наиболее удобной парадигмой программирования будет являться объектно-ориентированная парадигма. Использование этой парадигмы позволит разбить программу на отдельные модули, между которыми будет происходить обмен информацией.

## 2.2. Выбор языка программирования

Выбранный язык программирования Python является интерпретируемым языком программирования высокого уровня, широко применяющийся для создания программ различной сложности и направленности. Основными плюсами языка Python являются его мультипарадигменность, простой синтаксис и наличие большого количества подключаемых библиотек классов и функций. Мультипарадигменность позволяет реализовывать широкое поле задач разнообразными методами, включая объектно-ориентированную парадигму. Подключаемые библиотеки классов и функций позволяющих сэкономить общее время реализации программного обеспечения и сосредоточиться на основных аспектах поставленной задачи. Простой и прозрачный синтаксис позволяет программисту, даже не знакомому с языком Python, перенести программу на другой язык программирования. Другими словами, язык Python является очень подходящим инструментом для прототипирования программ. Это важно для текущей задачи, поскольку одной из основных задачи является разработка метода представления знаний, который не может быть привязан к тому или иному языку программирования. Одной из используемых в данном проекте библиотек является pygame [29], предоставляющий программисту удобный набор функций и классов для программирования вывода графической информации на экран.

В языке Python также присутствует сборщик мусора (часть интерпретатора языка Python, которая следит за памятью, используемой работающей программой, и удаляет неиспользуемые объекты, высвобождая память из-под них), в отличие от, к примеру, языка C++, что позволяет сосредоточиться на решении возникающих задач и ускорить процесс разработки.

# 3. Охрана труда

## 3.1. Основные определения

Охрана труда - система сохранения жизни и здоровья работников в процессе трудовой деятельности, включающая в себя правовые, социально-экономические, организационно-технические, санитарно-гигиенические, лечебно-профилактические, реабилитационные и иные мероприятия.

Условия труда - совокупность факторов производственной среды и трудового процесса, оказывающих влияние на работоспособность и здоровье работника.

Вредный производственный фактор - производственный фактор, воздействие которого на работника может привести к его заболеванию.

Опасный производственный фактор - производственный фактор, воздействие которого на работника может привести к его травме.

Следует очень четко соблюдать требования по санитарным нормативам освещенности в административных, учебных и иных учреждения. Наше зрение напрямую зависит от количества света в помещении и на рабочем месте. От освещенности также зависит здоровье всего организма, сопротивляемость стрессам, усталости, физическим и умственным нагрузкам.

Вопрос освещенности рабочих мест, оборудованных персональными компьютерами (ПЭВМ) коротко, но очень четко изложен в СанПиН 2.2.2/2.4.1340-03 "Гигиенические требования к персональным электронно-вычислительным машинам и организации работы".

Данный документ содержит много интересных разделов, которые должен знать каждый офисный работник, поскольку санитарно-гигиеническое состояние рабочего места касается каждого из нас.

## 3.2. Требования к освещению на рабочих местах, оборудованных ПЭВМ

1. Рабочие столы следует  размещать  таким   образом, чтобы видеодисплейные терминалы были ориентированы боковой стороной к световым проемам, чтобы естественный свет падал преимущественно слева.

2. Искусственное освещение в помещениях для эксплуатации ПЭВМ должно осуществляться системой  общего  равномерного   освещения. В производственных и административно-общественных помещениях, в случаях преимущественной работы с документами,  следует  применять  системы комбинированного  освещения  (к  общему  освещению  дополнительно устанавливаются светильники местного освещения,  предназначенные для освещения зоны расположения документов).

3. Освещенность на поверхности стола в зоне размещения рабочего документа должна быть 300 - 500 лк. Освещение не должно создавать бликов на поверхности экрана. Освещенность поверхности экрана не должна быть более 300 лк.

4. Следует ограничивать прямую блесткость от источников освещения, при этом яркость светящихся поверхностей (окна, светильники и др.), находящихся в поле зрения, должна быть не более 200 кд/м2.

5. Следует ограничивать отраженную блесткость на рабочих поверхностях (экран, стол, клавиатура и др.) за счет правильного выбора типов светильников и расположения рабочих мест по отношению к источникам естественного и искусственного освещения, при этом яркость  бликов на экране ПЭВМ не должна превышать 40 кд/м2 и яркость потолка не должна превышать 200 кд/м2.

6. Показатель ослепленности для источников общего искусственного освещения в производственных помещениях должен быть не  более 20.

Показатель дискомфорта в административно-общественных помещениях не более 40, в дошкольных и учебных помещениях не более 15.

7. Яркость светильников общего освещения в зоне углов излучения от 50 до 90 градусов с вертикалью в продольной и поперечной плоскостях должна составлять не более 200 кд/м2, защитный угол светильников должен быть не менее 40 градусов.

8. Светильники местного освещения должны иметь не просвечивающий отражатель с защитным углом не менее 40 градусов.

9. Следует ограничивать неравномерность распределения  яркости в поле зрения пользователя ПЭВМ, при этом соотношение  яркости между рабочими поверхностями не должно превышать 3:1 - 5:1, а между рабочими поверхностями и поверхностями стен и оборудования 10:1.

10. В качестве источников света при искусственном освещении следует применять преимущественно люминесцентные лампы   типа ЛБ и компактные люминесцентные лампы (КЛЛ). При  устройстве  отраженного освещения в производственных и административно-общественных помещениях допускается применение металлогалогенных ламп. В светильниках местного освещения допускается применение ламп  накаливания,  в   том числе галогенные.

11. Для освещения помещений с ПЭВМ следует применять светильники с зеркальными параболическими решетками, укомплектованными электронными пуско-регулирующими  аппаратами  (ЭПРА).  Допускается  использование многоламповых светильников  с  электромагнитными  пуско-регулирующими аппаратами (ЭПРА), состоящими из равного числа опережающих и отстающих ветвей.

Применение светильников без рассеивателей и экранирующих решеток не допускается.

При отсутствии светильников с ЭПРА лампы многоламповых светильников или рядом расположенные светильники общего освещения следует включать на разные фазы трехфазной сети.

12. Общее освещение при использовании люминесцентных светильников следует выполнять в виде сплошных или прерывистых линий светильников, расположенных сбоку от  рабочих  мест,  параллельно   линии зрения пользователя при рядном расположении видеодисплейных  терминалов. При периметральном расположении компьютеров  линии  светильников  должны располагаться локализовано над рабочим столом ближе к его переднему краю, обращенному к оператору.

13. Коэффициент запаса (Кз) для осветительных установок общего освещения должен приниматься равным 1,4.

14. Коэффициент пульсации не должен превышать 5%.

15. Для обеспечения нормируемых значений освещенности в помещениях для использования ПЭВМ следует проводить чистку стекол оконных  рам и светильников не реже двух раз в год и проводить своевременную замену перегоревших ламп.

Освещенность измеряется в Лк – люксах; 1 люмен/м. кв. (люмен Lm – единица величины светового потока). В Европейских нормах освещенности для ряда помещений введен еще один нормируемый параметр: для рабочих мест, оснащенных мониторами, устанавливаются требования к максимальной яркости тех поверхностей светильников, которые могут отражаться в экранах.

## 3.3. Требования к качеству восприятия информации, отображаемой на дисплеях

Согласно ГОСТ Р 50948-2001, для точного считывания информации и обеспечения комфортных условий ее восприятия работа с дисплеями должна проводиться при таких сочетаниях значений яркости и контраста изображения, внешней освещенности экрана, углового размера знака и угла наблюдения экрана, которые входят в оптимальные или предельно допустимые (при кратковременной работе) диапазоны.

Неравномерность яркости рабочего поля экрана должна быть не более 20 %.

Расчет неравномерности яркости рабочего поля экрана:

*Lцентр = 598 кд/м2*

*Lлево-верх = 599 кд/м2*

*Lлево-низ = 597 кд/м2*

*Lправо-верх = 600 кд/м2*

*Lправо-низ = 598* *кд/м2*

*= 598,4 кд/м2;*

Неравномерность яркости рабочего поля экрана -

Неравномерность яркости элементов знака должна быть не более 20 %.

Расчет неравномерности яркости элементов знаков плоских дискретных экранов:

*L’1 = 600 кд/м2*

*L’2 = 597 кд/м2*

*L’3 = 599 кд/м2*

*L’4 = 588 кд/м2*

*L’5 = 598* *кд/м2*

*L’6 = 599* *кд/м2*

*L’7 = 597* *кд/м2*

*= 598,285 кд/м2;*

Неравномерность яркости элементов знаков плоских дискретных экранов -

Угол наблюдения экрана оператором относительно горизонтальной линии взгляда не должен превышать 60˚.

Внешняя освещенность экрана должена быть не менее 100 и не более 250 лк.

Угловой размер знака должен быть не менее 16 и не более 60 угл.мин.

## 3.4. Эргономические требования к цветовым параметрам

Согласно ГОСТ Р 50948-2001, выделяют следующие требования к цветовым параметрам:

1. При необходимости распознавания или идентификации цветовых параметров прикладная программа должна предлагать устанавливаемый по умолчанию набор цветов, который соответствует требованиям настоящего стандарта. Если цвет может быть изменен пользователем, то должна быть предусмотрена возможность восстановления назначенного по умолчанию набора цветов.

2. При необходимости точной идентификации цвета в рядах буквенно-цифровых знаков и в полях ввода данных высота символа должна быть не менее 20¢ при проектном расстоянии наблюдения.

3. При необходимости точной идентификации цвета обособленного изображения (например знака или символа) угловой размер изображения должен быть не менее 30¢ при проектном расстоянии наблюдения (предпочтительно - 40¢).

4. Следует избегать применения насыщенного синего цвета для изображений, имеющих угловой размер менее 2°.

5. Для чтения текстов, буквенно-цифровых знаков и символов при отрицательной полярности изображения не следует применять синий и красный цвета спектра на темном фоне и красный цвет спектра на синем фоне.

6. Для чтения текстов, буквенно-цифровых знаков и символов при положительной полярности изображения не следует применять синий цвет спектра на красном фоне.

7. Насыщенные крайние цвета видимого спектра приводят к нежелательным эффектам глубины изображаемого пространства и не должны применяться для изображений, которые требуют непрерывного просмотра или чтения.

8. Для точного распознавания и идентификации цветов должны применяться цветное изображение переднего плана на ахроматическом фоне или ахроматическое изображение переднего плана на цветном фоне.

9. Число цветов, одновременно отображаемых на экране дисплея, должно быть минимальным. Для точной идентификации цвета каждый заданный по умолчанию набор цветов должен включать не более 11 цветов.

10. При необходимости проведения быстрого поиска, основанного на распознавании цветов, следует применять не более 6 различных цветов.

11. При необходимости вызова параметров цвета из памяти ЭВМ следует применять не более 6 различных цветов.

## 3.5. Требования безопасности к визуальным параметрам

Согласно ГОСТ Р 50948-2001, выделяют следующие требования безопасности к визуальным параметрам:

1. Яркость знака должна быть не менее 35 кд/м2 для дисплеев на ЭЛТ и не менее 20 кд/м2 для плоских дискретных экранов.

2. Яркостный контраст изображения должен быть не менее 3:1 (для плоских дискретных экранов при угле наблюдения от минус 40° до плюс 40°). Яркостный контраст внутри знака и между знаками должен быть не менее 3:1.

3. Ширина контура знака должна быть от 0,25 до 0,5 мм.

4. Степень несведения цветов в любом месте многоцветного экрана для дисплеев на ЭЛТ должна быть не более 3,4¢ при проектном расстоянии наблюдения.

Примечание - Если в документации на дисплей не оговорено проектное расстояние наблюдения, то его принимают равным 50 см для дисплеев с размером экрана по диагонали 14" - 17" и 75 см - для экранов 19" - 21".

5. Временная нестабильность изображения (мелькания) для дисплеев на ЭЛТ и на плоских дискретных экранах не должна быть зафиксирована. Для дисплеев на ЭЛТ частота обновления изображения должна быть не менее 75 Гц при всех режимах разложения, гарантируемых нормативной документацией на конкретный тип дисплея и не менее 60 Гц для дисплеев на плоских дискретных экранах.

6. Амплитуда смещения изображения (пространственная нестабильность изображения - дрожание) должна быть не более 2×10-4*l*, где *l* - проектное расстояние наблюдения, мм.

## 3.6. Искажения изображения по рабочему полю

Согласно ГОСТ Р 50949-2001, выделяют следующие требования и методы измерений и оценки пространственных искажений изображений по рабочему полю:

1. Изменение размеров однотипных знаков по рабочему полю должно быть в пределах ±5 % высоты знака.

2. Максимальная разность длин строк текста на рабочем поле должна быть не более 2 % средней длины строки.

3. Максимальная разность длин столбцов текста на рабочем поле должна быть не более 2 % средней длины столбца.

4. Отклонение формы рабочего поля от прямоугольника определяют по следующим формулам:

по вертикали

http://www.docload.ru/Basesdoc/40/40770/x012.gif

по горизонтали

http://www.docload.ru/Basesdoc/40/40770/x014.gif

по диагонали

http://www.docload.ru/Basesdoc/40/40770/x016.gif

где *Н*1,*H*2*, B*1,*B*2,*D*1и *D*2*-*значения длин крайнего левого и крайнего правого столбца, верхней, нижней строки и диагоналей на рабочем поле соответственно, мм.

Для текущего рабочего места и монитора получены следующие параметры:

Крайний левый столбец (H1) – 1080 мм;

Крайний правый столбец (H2) – 1080 мм;

Верхняя строка (B1) – 620 мм;

Нижняя строка (B2) – 620 мм;

Диагонали (D1, D2) – 1240 мм;

По вертикали:

По горизонтали:

По диагонали:

Полученные данные соответствуют требованиям ГОСТ Р 50949-2001.

## 3.7. Требования безопасности к параметрам создаваемых полей

# Согласно ГОСТ Р 50948-2001, выделяют следующие требования безопасности к параметрам создаваемых полей:

1. Электростатический потенциал экрана дисплеев на ЭЛТ должен быть не более ±500 В.

2. Напряженность электрической составляющей переменного электромагнитного поля дисплея должна быть не более:

25 В/м - в диапазоне частот от 5 Гц до 2 кГц (для дисплеев на ЭЛТ - в точке, расположенной по нормали к центру экрана на расстоянии 0,5 м от экрана дисплея, а для дисплеев портативных компьютеров - в точке, расположенной по нормали к центру экрана на расстоянии 0,4 м от центра клавиатуры портативного компьютера);

2,5 В/м - в диапазоне частот от 2 до 400 кГц (для дисплеев на ЭЛТ - в точках, имеющих координаты 0°, 90°, 180°, 270° на расстоянии *r = a*/2*+*0,5 м, где *а -*габаритный размер дисплея, измеряемый по нормали к центру экрана. Для дисплеев портативных компьютеров - в точках, имеющих те же координаты, но на расстоянии 0,4 м от центра клавиатуры портативного компьютера).

3. Плотность магнитного потока должна быть не более:

250 нТл - в диапазоне частот от 5 Гц до 2 кГц;

25 нТл - в диапазоне частот от 2 до 400 кГц.

Плотность магнитного потока переменного электромагнитного поля дисплея на ЭЛТ устанавливают для обоих диапазонов частот в 48 точках (в горизонтальной плоскости, проходящей через центр экрана, а также в горизонтальных плоскостях, расположенных на 0,3 м выше и ниже указанной плоскости) через 22°30¢ от нормали к центру экрана, на расстоянии *r* = *a*/2 + 0,5 м, где *а*- габаритный размер дисплея по нормали к центру экрана. Плотность магнитного потока переменного электромагнитного поля дисплея портативного компьютера устанавливают для обоих диапазонов частот в тех же 48 точках, но на расстоянии *r = a*/2 + 0,4 м от центра клавиатуры портативного компьютера.

## 3.8. Требования к конструкции

Согласно ГОСТ Р 50948-2001, выделяют следующие требования к конструкции дисплея:

1. Конструкция дисплея должна обеспечивать возможность фронтального наблюдения изображения на экране путем поворота корпуса дисплея вокруг вертикальной оси на ±30° и вокруг горизонтальной оси в пределах от плюс 30° до минус 15° с фиксированием дисплея в заданном положении.

2. Корпус дисплея должен быть окрашен в мягкие тона. Корпус дисплея, клавиатура и другие устройства персонального компьютера должны иметь матовую (диффузно отражающую) поверхность одного цвета и не иметь блестящих деталей, способных создавать блики.

3. На лицевой стороне корпуса дисплея не рекомендуется располагать органы управления. При необходимости расположения органов управления на лицевой панели они должны быть закрыты крышкой или утоплены в корпусе.

4. Конструкция дисплея должна предусматривать наличие органов регулирования яркости и контраста.

5. Конструкция дисплея должна обеспечивать максимально возможное снижение уровней электростатического и электромагнитного полей.

6. Если в комплект поставки дисплея входит защитный фильтр-экран, то этот фильтр должен быть сертифицирован на соответствие требованиям ГОСТ Р 51658.

## 3.9. Заключение

Расчет параметров рабочего места и дисплея на рабочем месте, проведенный согласно ГОСТ Р 50948-2001 и ГОСТ Р 50949-2001, показал, что конструкция дисплея и организация рабочего места соответствует СанПиН 2.2.2/2.4.1340-03 и требованиям, заявленным в ГОСТ Р 50948-2001, таким образом, меры охраны труда соответствуют безопасной и комфортной работе на данном рабочем месте.

# 4. Экологическая часть

Cтандарт ГОСТ Р 54148-2010 – «Воздействие на человека электромагнитных полей от бытовых аналогичных электрических приборов. Методы оценки и измерений» применяется при оценке воздействия на человека электромагнитных полей и устанавливает методы определения параметров электрических и магнитных полей, создаваемых бытовыми и аналогичными электрическими приборами в пространстве вокруг указанных приборов на частотах до 300 ГГц.

Указанные методы применяют также при оценке воздействия на человека электромагнитных полей от электрических приборов, не предназначенных для эксплуатации в бытовых условиях, но которые, тем не менее, могут быть доступны широкому кругу лиц, например, приборов, предназначенных для использования неспециалистами в магазинах, на предприятиях легкой промышленности и в сельском хозяйстве

## 4.1. Термины и определения

В стандарте ГОСТ Р 54148-2010 применены следующие определения:

1. Основное ограничение: физическая величина, принятая для установления допустимого воздействия на человека изменяющихся во времени электрических и магнитных полей, непосредственно определяемая их влиянием на здоровье человека.
2. Опорный уровень: среднеквадратическое значение напряженности однородного магнитного поля, полученные исходя из основного ограничения, воздействию которого человек, может подвергаться без неблагоприятных последствия для здоровья.
3. Измерительное расстояние: расстояние между поверхностью прибора и ближайшей точной поверхности датчика.
4. Расстояние до оператора: расстояние между поверхностью прибора и ближайшей точкой головы или туловища оператора.
5. Участок повышенной магнитной индукции: локальная область сильного магнитного поля, обусловленная неоднородностью его распределения.
6. Коэффициент связи: коэффициент, учитывающий неоднородности магнитного поля вокруг приборов и размеры частей человеческого тела.
7. Магнитная индукция: векторная величина, являющаяся характеристикой магнитного поля, действующего на заряженную частицу.
8. Коэффициент удельного поглощения энергии SAR Вт/кг: производная по времени электромагнитной энергии, рассеиваемой элементом некоторой массы, содержащимся в одном элементе некоторого объема при данной плотности.
9. Суммарная стандартная неопределенность: стандартная неопределенность результата измерений, полученного через значения других величин, равная положительному квадратному корню суммы членов, причем члены являются дисперсиями или ковариациями этих других величин, взвешенными в соответствии с тем, как результат измерений изменяется при изменении этих величин.

## 4.2. Методы измерений

Согласно ГОСТ Р 54148-2010, используют следующие методы измерений для различных явлений:

1. Параметры электрического поля вблизи бытовых и аналогичных электрических приборов в общем случае не определяют. Для большинства приборов считают, что напряженность электрического поля не превышает опорный уровень, без выполнения соответствующих измерений.
2. Для магнитных полей рассматриваются полосы частот от 10 Гц до 400 кГц. На частотах выше 400 кГц бытовые и аналогичные электрические приборы считают удовлетворяющими требованиям данному стандарта. В ином случае необходимо включить все частоты магнитных полей в полосу частот при оценки магнитного поля, в том числе достаточное число гармоник.

Максимально допустимый уровень шума для измерительного прибора должен составлять 5% предельного значения. Измерения значения ниже максимально допустимого уровня шума, не учитывают. Уровень внешних магнитных полей должен составлять менее 5% предельного значения.

## 4.3. Критерии соответствия

1. Согласно ГОСТ Р 54148-2010, бытовые и аналогичные электрические приборы считают удовлетворяющими базовому ограничению, если отсутствует превышение опорных уровней.
2. Если значение превышает опорный уровень, то для демонстрации соответствия базовому ограничению допускается учитывать коэффициент связи.
3. Коэффициент связи установлен так, чтобы учесть наихудший случай создания поля приборами конкретного вида.
4. Если значение по-прежнему превышает опорный уровень, это не обязательно означает, что превышено базовое ограничение. В этом случае допускается применение расчетных методов, позволяющих проверить, соблюдено ли это ограничение.

## 4.4. Заключение

Бытовые и аналогичные электрические приборы необходимо проверять на соответствие нормам воздействия электромагнитных полей на человека, установленным согласно ГОСТ Р 54148-2010. Организация текущего рабочего места соответствует данным нормам согласно критерию 1.

# 5. Решение задачи на ЭВМ

## 5.1. Структура классов

Класс **Animat** – это класс, представляющий анимата во внешней среде. На каждой итерации имитационного моделирования в качестве входных данных он получает когнитивные карты схем и координаты во внешнем мире этих схем, и принимает решение о некотором действии. Каждый анимат содержит когнитивную карту с путевыми или структурными отношениями, активную когнитивную карту и память о распознанных схемах и поведении этих схем.

Класс **BehaviourMemory** реализует память некоторого анимата о поведении схем во внешнем мире. Содержит известные когнитивные карты, описывающие поведение распознанных схем. Позволяет запрашивать когнитивные карты по индексу схемы, добавлять и изменять когнитивные карты.

Класс **Cache** реализует память быстрого доступа, которая содержит когнитивные карты, которые предположительно будут нужны анимату. Позволяет добавлять, запрашивать и удалять когнитивные карты.

Класс **CognitiveMap** реализует когнитивную карту. Позволяет выделять слои, находить, добавлять и удалять схемы и отношения. Также поддерживает операции над когнитивными картами.

Класс **ConfigParser** обрабатывает входные данные о внешней среде, генерируя нужные схемы, отношения и когнитивные карты, и загружает их в класс **WorldMap**.

Класс **EmotionCost** реализует эмоциональные оценки.

Класс **PathMap** реализует карту путей некоторого анимата. Позволяет возвращать *p*-слои когнитивных карт схем по заданному индексу, когнитивные карты, близкие к активной когнитивной карте.

Класс **Relation** реализует отношение между некоторыми схемами.

Класс **RenderApp** на каждой итерации работы программы реализует вывод графической информации. В качестве входной информации принимает массив объектов класса **WorldObject**.

Класс **SemanticMap** реализует семантическую карту. Позволяет добавлять схемы, распознавать новые и старые схемы, возвращать *c*-слои когнитивных карт схем по заданному индексу.

Класс **Scheme** реализует некоторую схему. Каждой схеме сопоставляется уникальный индекс, по которому можно будет получить информацию о ней из объектов классов **PathMap**, **SemanticMap** и **BehaviourMemory**.

Класс **WorldMap** – класс для хранения информации о внешней среде и обработки действий, в которой будут действовать аниматы. Содержит массив объектов класса **WorldObject**.

Класс **WorldObject** содержит некоторую схему и ее графическое представление, используемое для графического вывода классов **RenderApp**.

## 5.2. Пример функционирования

Запуск программы производится с помощью терминала. В качестве параметров программы необходимо указать входные файлы с информацией о внешней среде. После запуска программа загружает данные из переданных файлов, выводя общую информацию о считанных данных в терминал, и запускает графический интерфейс, представляющий собой поле, на которое выводятся графические представления считанных из входных файлов схем. Управление процессом имитационного моделирования осуществляется с помощью команд, вводимых в терминал. С помощью команд также возможно вывести информацию о любых схемах, присутствующих во внешней среде, а также остановить или перезапустить процесс имитационного моделирования.

# 6. Выводы

Была разработана модель представления знаний на основе когнитивных карт, и реализовано программное обеспечение для имитационного моделирования поведения искусственных животных – аниматов – на основе предложенного метода. Данная модель обладает рядом недостатков – не рассмотрены способы обобщения информация и объединения похожих объектов в единые классы объектов и не рассмотрены методы представления знаний, изменяющихся во времени. Данные две проблемы будут рассмотрены в будущих исследованиях.

# 7. Список литературы

[1] У. Найссер. Познание и реальность. М.: Прогресс, 1981.

[2] Когнитивная психология. Учебник для вузов. / Под редакцией В.Н. Дружинина, Д.В. Ушакова. – М.: ПЕРСЭ, 2002.

[3] Когнитивная психология / Р. Солсо. — 6-е изд. — СПб.: Питер, 2011. — 589 с.:ил. — (Серия «Мастера психологии»).

[4] С. Хайкин. Нейронные сети: полный курс, 2-е изд., испр.: Пер. с англ. — М.: ООО "И.Д. Вильямс", 2006. — 1104 с.: ил. — Парал. тит. англ.

[5] Анализ данных и процессов: учеб. пособие / А.А. Барсегян, М.С. Куприянов, И.И. Холод, М.Д. Тесс, С.И. Елизаров, 3-е изд., перераб. и. доп. — СПб.:БХВ-Петербург, 2009, 512 с., ил. + CD-ROM – (Учебная литература для вузов), 2009.

[6] Дж.Ф. Люгер. Искусственный интеллект: стратегии и методы решения сложных проблем, 4-е издание. / Пер. с англ. – М.: Издательский дом «Вильямс», 2003.

[7] Л. Рутковский. Методы и технологии искусственного интеллекта. / Пер. с польск. И.Д. Рудинского. – М.: Горячая линия–Телеком, 2010.

[8] S.J. Russel, P. Norvig P. Artificial Intelligence – A Modern Approach. 3rd edition, Prentice Hall. 2010.

[9] http://www.keldysh.ru/pages/BioCyber/RT/Nepomn.htm

[10] http://rk6.bmstu.ru/electronic\_book/iisapr/AI.html

[11] M. Minsky, MIT-AI Laboratory Memo 306, June, 1974. http://web.media.mit.edu/~minsky/papers/Frames/frames.html

[12] О. Оре. Теория графов. – 2е изд. – М.: Наука Главная редакция физико-математической литературы, 336 с., 1980.

[13] John F. Sowa http://www.jfsowa.com/pubs/semnet.htm

[14] John F. Sowa - http://www.jfsowa.com/cg/cg\_hbook.pdf

[15] http://news.cnet.com/8301-17938\_105-20005170-1.html

[16] http://spectrum.ieee.org/automaton/robotics/artificial-intelligence/how-google-self-driving-car-works

[17] http://news.discovery.com/tech/robotics/military-robots-to-patrol-world-cup-130516.htm

[18] http://www.humanbrainproject.eu/

[19] https://www.groksolutions.com/htm-overview/education/HTM\_CorticalLearningAlgorithms.pdf

[20] http://www.forbes.com/sites/bruceupbin/2013/02/08/ibms-watson-gets-its-first-piece-of-business-in-healthcare/

[21] http://www.spiderland.org/

[22] http://digilander.libero.it/darwinbots/

[23] http://digihive.pl/

[24] https://sites.google.com/site/ecosimgroup/research/ecosystem-simulation

[25] http://stauffercom.com/evolve4/

[26] http://www.framsticks.com/

[27] http://ccl.northwestern.edu/netlogo/

[28] http://shinyverse.org/larryy/Polyworld.html

[29] http://www.pygame.org/

[30] ГОСТ Р 54148-2010

[31] ГОСТ 12.1.002-84

[32] ГОСТ 12.1.006-84

[33] ГОСТ Р 50948-96

[34] СНиП 11-4-79

[35] СанПиН 2.2.2/2.4.1340-03