Министерство образования республики беларусь

Учреждение образования
«Брестский государственный технический университет»

Республиканский конкурс научных работ студентов
высших учебных заведений Республики Беларусь

Информатика и информационные технологии. Программное обеспечение вычислительной техники и автоматизированных систем. Методы искусственного интеллекта

АСПЕКТЫ МОДЕЛИРОВАНИЯ РОБОТОВ-ИГРОКОВ В РОБОФУТБОЛЬНОЙ КОМАНДЕ

Автор:

Дёмин Владимир Владимирович, IV курс

Научный руководитель:

Дунец Андрей Петрович,

доцент кафедры ИИТ УО БрГТУ

Брест, 2010

Реферат

Аспекты моделирования роботов игроков в робофутбольной команде. Научная работа. УО «БрГТУ», Дёмин В.В. студент; - Брест, 2010, - 29 стр, 0 табл., 4 илл., 5 библ..

СИМУЛЯЦИЯ, РОБОФУТБОЛ, ВИРТУАЛЬНЫЙ АГЕНТ, АЛГОРИТМЫ ОБУЧЕНИЯ, МУЛЬТИАГЕНТНЫЕ СИСТЕМЫ, АЛГОРИТМЫ КОЛЛЕКТИВНОГО ПОВЕДЕНИЯ

Объектом исследования являются средства моделирования игры в робофутбол.

Целью работы — анализ современных систем моделирования робофутбола, разработка на их основе экспериментальных логических модулей, решающих задачи мультиагентного взаимодействия и следования единой стратегии.

В процессе работы проводились экспериментальные исследования алгоритмов обучения агентов-игроков и поведения мультиагентной системы, на основе единой стратегии.

В результате получены алгоритмов высокоуровневой логики, взаимодействующие с моделью агента-футболиста и сервером симуляции Robocup Soccer Simulation Server, а так же исследованы сервера симуляции футбола.

Выявлены наиболее эффективные алгоритмы для обучения тактического модуля агента-игрока, такие как эволюционный метод и подкрепляющее обучение. А так же успешно проведен эксперимент по обучению конечного автомата.

Данные алгоритмы могут применятся для любых мультиагентных систем, эффективно решая поставленные задачи с наименьшими затратами производительности системы.

Содержание

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

ИНС — интеллектуальная нейронная сеть

RCSSS — RoboCup Soccer Simulation Server

RL — reinforcement learning

GA — genetic alghorithm

FIRA — Football International Robot Association

ВВЕДЕНИЕ

Возможность решения компьютером интеллектуальных задач интересовала ученых еще на заре развития компьютерной техники. Одной из таких задач на протяжении долгого времени была игра в шахматы. Но после поражения Каспарова от супер компьютера Deep Blue в 1997 году была необходима новая задача, которая смогла бы охватить уже не только программное решение, но и робототехнический сектор. Такой задачей был выбран робофутбол. Основной целью стала создание к 2050 году команды автономных роботов футболистов, идентичных по физическим параметрам к человеческим, которые одержат победу над командой людей-футболистов. В связи с этим было образовано несколько робофутбольных чемпионатов (FIRA, Robocup), которые ежегодно, начиная с 1997 г., проводят чемпионаты мира по робофутболу. Чемпионаты, в свою очередь, имеют деление на лиги — соревнования ходящих роботов, роботов небольших размеров, соревнование программ-агентов и другие. Особым интересом пользуется соревнования программ-агентов — лиги симуляций. Причиной их популярности среди разработчиков стала возможность поучаствовать в соревнованиях, написав свою программу агента-футболиста. Нет необходимости создавать механического робота или покупать дорогостоящую готовую реализацию такого робота. Не смотря на то что симуляция как правило дает лишь упрощенную модель окружающего мира, в которой не возможно учесть все проблемы, возникающие в реальной ситуации, она служит отличным полигоном для исследования мультиагнетных систем, алгоритмов обучения, алгоритмов интеллектуального поведения. Результаты таких исследований легко перенести на механического робота, т. к. они базируются на низкоуровневом поведении, т. е. используют уже обработанные данные с датчиков и подают лишь команды, а низкоуровневая логика уже выполняет все необходимые действия взаимодействия с окружающей средой.

Управление тактическим поведением групп объектов в реальном времени,

взаимодействующих с изменяющейся внешней средой, компьютерное зрение, распознавание образов, коммуникация объектов, выработка единой стратегии поведения — эти и многие другие задачи, решаемые в рамках проекта Robocup, находят применение в военных приложениях, спасательных операциях, охранных системах и поэтому являются перспективным направлением исследований на сегодняшний день.

Целью данного исследования ставится изучение возможности лиги симуляции Robocup, реализации на её основе интеллектуального агента-игрока, способного взаимодействовать с сервером симуляции, а так же оценка возможности переноса полученных результатов на механического робота.

Исследования, проводимые в рамках данной работы, имеют большую востребованность в военных, спасательных, мониторинговых приложениях, что подтверждает их актуальность и востребованность на сегодняшний день.

1. Соревнование Robocup

RoboCup — международные соревнования среди роботов, основанные в 1993 году. Целью является создание автономных роботов футболистов для содействия научным исследованиям в области искусственного интеллекта. Название RoboCup — сокращение от полного названия соревнования, англ. "Robot Soccer World Cup" (Чемпионат по футболу среди роботов), но, в рамках соревнования, существуют и другие виды состязаний, например, среди спасательных роботов, по танцам среди роботов.

Официальная цель проекта:

К середине 21-го века команда полностью автономных человекоподобных роботов футболистов должна выиграть футбольный матч, соблюдая правила FIFA, у победителя Чемпионата мира

1.1. Категории состязаний

Соревнование делится на четыре основных состязания, каждое с набором из нескольких лиг:

- Футбол (RoboCupSoccer)
    - Simulation League — состязание компьютерных программ.
    - Small Size League — состязание роботов футболистов малых размеров (не более 18 см. в диаметре).
    - Middle Size League — состязание роботов футболистов средних размеров (не более 50 см. в диаметре).
    - Standard Platform League (ранее Four Legged League), когда все команды состоят из одинаковых роботов. Роботы работают полностью автономно, т.е. без какого либо контроля со стороны человека или

компьютера. Изначально состязание проводились между роботами AIBO, позже к ним присоединились роботы Nao.

- Humanoid League — роботы, имеющие корпус, похожий на тело человека, конструкцию, дизайн и программное обеспечение роботов создаётся командами участниками самостоятельно.

- Спасение (RoboCupRescue)
  - RoboCupRescue Robot League — роботы исследуют специально построенные лабиринты, симулирующие районы бедствий, ищут пострадавших, идентифицируют их признаки жизни и формируют карту местности с отметками мест нахождения пострадавших.
  - Rescue Simulation League — состязание между программами-симуляторами роботов и различными программными алгоритмами, связанными с поиском, ориентаций на местности и работе в районе стихийных бедствий.
- Состязания для роботов молодых участников (RoboCupJunior).
  - Состязание по футболу
  - Состязание по танцам
  - Состязание по спасению
  - Общее состязание
- RoboCup@Home — дебютировавшее в 2006 году, и акцентирующее внимание на внедрении роботов в человеческое общество.

1.2. RoboCup – Simulation league

В большей степени настоящий футбол напоминает проект RoboCup, а вернее одна из его лиг - Simulation league. Он был основан в 1995–м году и

зарегистрирован в Швейцарии. Основные особенности:

1) реалистичная физика и модель игрока,
2) наличие почти всех правил футбола,
3) наличие помех,
4) усложненное взаимодействие игроков,
5) локальная визуальная информация,
6) более строгие ограничения по времени.

Вся информация приходит игроку с помехами и в системе координат с центром в самом игроке. Помехи зависят от расстояния и могут быть достаточно значительными (на расстоянии 100 метров они могут достигать 10-ти). Так как информацию о положение объектов на поле удобнее хранить в глобальных координатах, необходимо локализовать местоположение игрока на поле. Для этого используются метки с известными координатами, называемые флагами. По положению игрока относительно них можно определить свое глобальное местоположение. Для этого используются различные методы фильтрации случайных процессов.

Лига симуляции развивается в двух направлениях — 2D симуляция и 3D симуляция. Для 2D симуляции более характерно максимальное воссоздание качеств футболистов, таких как выносливость, возможный угол обзора, слух, которые возможны для 2D пространства. Для 3D мира более характерна реальная физика, механика суставов робота. В основе роботов 3D пространства лежат гуманоидные роботы. Управление каждой частью тела, распознование объектов в виртуальном пространстве полностью реализуется в программном агенте, которых пишет команда участница. По сравнению с 2D, где команды играют реальными футбольными составами — 11 на 11, для 3D мира матч происходит составом 2 на 2 и является своеобразной виртуальной копией humanoid league. Поэтому команды, которые обладают возможным техническим уровнем и необходимыми

компонентами, для создания гуманоидного робота, как правило так же участвуют в соответствующей лиге симуляции. Из-за схожести реального и виртуального мира, все высокоуровневые и среднеуровневые алгоритмы (то есть те которые не касаются управления электронной и механической частью робота) легко переносимы. Тем самым лига симуляции 3D является прекрасным дополнением для лиги с гуманоидными механическими роботами.

В данной работе исследования проводились на основе лиги симуляции 2D. Так как в дальнейшей перспективе исследования для реальных роботов в small league, то наилучшим выбором для наработки алгоритмов является указанная лига симуляции.

1.3. Структура Robocup Simulation league

1. Сервер, реализующий футбольное поле – физику происходящих событий – т.е. движение игроков и мяча, передачу информации игрокам – визуальную, звуковую и сенсорную, а также стандартный алгоритм судейства.

2. Мониторы, подключаемые к серверу, и осуществляющие визуализацию игры и её судейство оператором.

3. Игроки, подключаемые к серверу – общение между игроками в обход севера запрещено.

4. Тренеры, подключаемые к серверу – их основная роль заключается в отладке и настройке программ-игроков.

Simulation league - это часть проекта Robocup, которая направлена на моделирование игры в простейшем 2D - варианте, без реализаций реальных роботов-футболистов.

В лиге симуляции принимают участие две команды с одиннадцатью автономными программами (которых называют агентами), играющие в футбол на двухмерном виртуальном стадионе, который предоставляется центральным

сервером под названием Robocup Simulation Soccer Server. На сервере содержится вся необходимая информация об игре, такая как позиции игроков и мячика, физика, направления игроков и другие параметры и свойства виртуального мира. Игра основана на сообщениях между сервером и агентом. С одной стороны каждый игрок получает данные с его виртуальных сенсоров (визуальные, акустические и физические), а с другой может формировать некоторые базовые команды (ускорение, поворот, удар) в целях оказания воздействий на окружающую среду.

При построении своей реализации агента-игрока были рассмотрены программные модули различных команд, занимавших в последние года места в топ-10 чемпионата мира Robocup.

Как и любой другой программный продукт, робот-агент должен содержать ряд модулей. Специфика Robocup SoccerServer заключается в том, что агент должен уметь буквально всё, от возможности отправлять и принимать сообщения по средствам UDP сокетов, алгоритмов управления поворотом направления обзора относительно расположения мячика на поле до алгоритмов коллективного поведения [1]. Всё это приводит к явному разделению модуля агента на три уровня:

- Низкоуровневая логика — простейшие действия, такие как поворот агента на угол, отправка/принятие сообщения от сервера, служащие в первую очередь основой для более сложных действий. Этот уровень является абстракцией, драйвером скрывающим от программиста низкоуровневую реализацию и дающий возможность работать непосредственно с действиями робота-игрока, т. е. манипулированием им.
- Среднеуровневая логика — представляет собой наборы из действий низкоуровневой логики. Каждое действие среднеуровневой логики направлено на решение какой либо ситуации происходящей с роботом: оценка ситуации, исполнения какого либо действия (удар, отбор мяча, ловить мяч в воротах). Сами по себе эти действия уже составляют серьёзную базу, и на их основе возможно полноценное функционирование каждого агента в

отдельности.

- Высокоуровневая логика — это алгоритмы коллективного поведения, алгоритмы принятия решений, а так же список предписаний для агентов, учитывающих различные ситуации на поле, реализующие комплексный подход в решении данной задачи. Если вышеописанные уровни как правило совпадают или мало различаются у большинства команд, то высокоуровневая логика является концептуальной задачей для каждой команды, и реализуется с использованием последних исследований в данной области.

Для того, что бы лучше понять для какой системы будет вестись разработка следует дать более подробное описание программ, которые используются в проекте.

Robocup Soccer Simulator Server — это исследовательская и обучающая среда для мультиагентных систем и искусственного интеллекта. Среда позволяет двум командам с 11-ю программными автономными роботам играть в футбол. Сам сервер представляет собой программу-сервер и набор программ-утилит и имеет следующую структуру[2]:

1) Сервер — непосредственно сам сервер
2) Мониторы — программы, служащие для визуализации происходящего на поле в реальном времени. Так же служат для отладки агентов-игроков
3) Агенты-игроки — модули команд-соперников, предоставляющих по 11 виртуальных агентов с каждой стороны поля
4) Тренеры — виртуальные агенты, с возможностью обозревать всё поле одновременно. Их месторасположение — бровка поля, как и у тренеров в настоящем футболе. Тренер является важной частью каждой команды и выполняет функции управления стратегией команды в целом.
5) Журнал игрока — программа предоставляет возможность вести протокол всех действий на сервере, тем самым предоставляя возможность просмотреть ещё раз игру и более тщательно оттестировать модули агента-игрок.

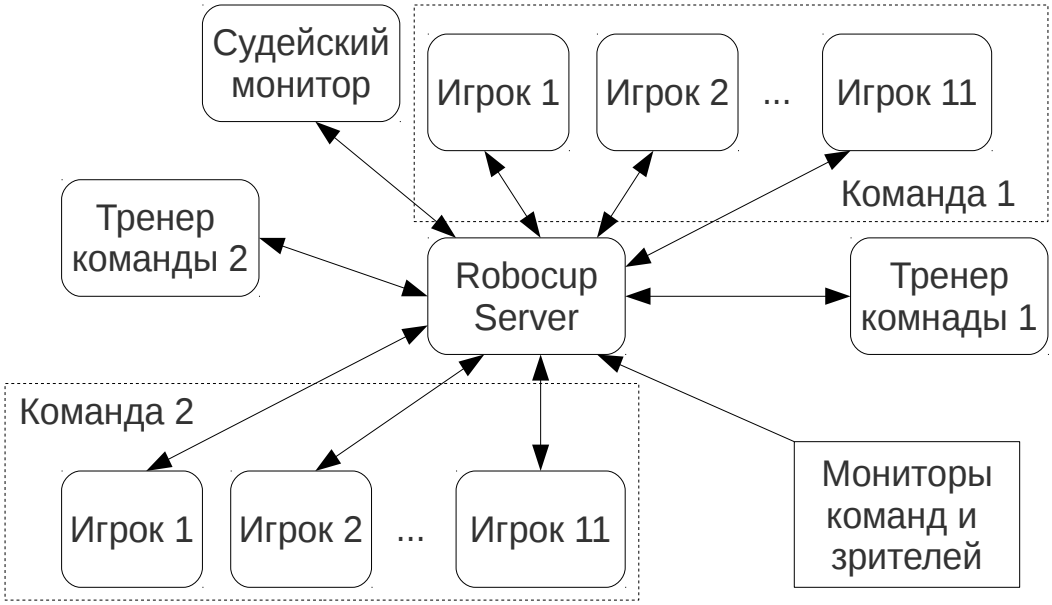

Рисунок 1 — схема связей программы-сервера, программ-мониторов и программ-игрвоков

Сервер это программа с помощью которой различные команды могут играть в футбол. На время игры он представляет собой клиент-серверную систему, где не имеет значения реализация программ-игроков. Передача клиент-серверной информации реализована на протоколе UDP/IP, через UDP сокеты. Каждый клиент представляет собой отдельный процесс и подсоединяется к серверу через специальный порт. После соединения с сервером, все сообщения между ними происходят только через этот порт. Каждая команда может иметь до 12 клиентов, 11 агентов-игроков (10 полевых и 1 вратарь) и тренера. Игроки посылают серверу сообщения с действиями, которые они хотят выполнить (ударить мяч, повернуться, бежать и т.д.). Сервер получая эти сообщения, выполняет запросы, и соответственно обновляет информацию о среде. Так же сервер обеспечивает всех игроков сенсорной информацией (визуальной — позиция объектов на поле или данные о физических свойствах игрока, таких как выносливость или скорость).

Сервер работает в режиме реального времени. Время делится на дискретные временные интервалы — циклы. Таким образом низкая производительность игроков, приводит к пропуску действий, что очень сильно влияет на результат команды в целом.

Последняя версия сервера является 14.0.3. Проект активно развивается, поэтому обновления выходят довольно часто — один или два раза в месяц. Основной проблемой в исследовании сервера и реализации основных модулей агента является недоработанная документация. В результате большинство команд начинает свои исследования с разбора исходных кодов уже готовых реализаций. И как правило их собственные реализации основываются на исходном коде низкоуровневой и среднеуровневой логики.

Монитор представляет собой средство визуализации которое позволяет наблюдать, что же происходит на поле внутри сервера в течении игры. В настоящий момент есть две реализации монитора rcssmonitor — написанной с использованием библиотеки Qt4 и rcssmonitor_classic, использующий для вывода графической информации на экран средства фрэимбуфера (реальное или виртуальное электронное устройство, или область памяти для кратковременного хранения одного или нескольких кадров в цифровом виде перед его отправкой на устройство видеовывода. Буфер может быть использован для выполнения над кадром различных предварительных операций, организации стоп-кадра, устранения мерцания изображения и др. Обычно кадр хранится в виде последовательности цветовых значений каждого пиксела изображения. Размер памяти, необходимый для хранения одного кадра, зависит от разрешения и глубины цвета).

Информация, предоставляемая на экране, содержит счет, названия команд, позиции всех игроков и мяча. Так же монитор содержит простейшие команды управления сервером. Например, когда обе команды присоединены, копка «Kick-Off» (вбросить мяч) в интерфейс монитора позволяет человеку-судье начать игру.

Основные особенности монитора:

1) Возможно приближать различные участки поля. Это свойство обычно используют для отладки программ и поиска ошибок в реализации;
2) Положение каждого игрока на поле и мяча может выводится в консоль, в каждый цикл сервера;
3) Различного вида информация может быть отображена на мониторе, такая как угол обзора, выносливость, вид игрока;
4) Игроки и мяч могут быть перемещены с помощью мышки.

На самом деле функционирование сервера не зависит от наличия подсоединенного монитора. В том случае, если он необходим, определенное количество мониторов соединяется с сервером одновременно.

Программу журнал игрока можно представить как видео съёмку. Это утилита для проигрывания прошедших матчей. Когда запущен сервер, есть возможность задать ему опцию сохранения данных на жесткий диск. Когда журнал игрока используется вместе с монитором, появляется возможность просматривать матч столько раз, сколько это необходимо. Это очень полезная утилита, позволяющая наглядно изучать стратегии соперников, их минусы и плюсы. А так же оценивать свои возможности, поиск дырок в оборонной тактике или недостатков в нападающей. Так же журнал игрока представляет возможность каждой команде сохранять свою игру и далее легко вставлять в свои презентации.

На рис. 1 схематично изображена взаимосвязь вышеописанных компонентов сервера и программ-игроков обеих команд.

2. Структура и функционирование программных средств имитации игры в футбол

В имитационном моделировании существует несколько парадигм – постановок проблем и подходов к их решению, используемых в качестве каркаса при построении и анализе моделей. Можно выделить четыре парадигмы: динамические системы, системная динамика, дискретно-событийное моделирование, мультиагентные модели.

Эти парадигмы различаются не столько областями применения, сколько концепциями и взглядами на проблему и подходами к решению проблемы. Очень часто приверженцы одной парадигмы считают, что “правильные” постановка и решение проблем имитационного моделирования возможны только в рамках концепций и методик именно этой парадигмы. Например, апологеты моделирования и анализа динамических систем считают, что остальные подходы “не совсем” научны, либо они являются частным случаем представления и анализа систем в виде систем алгебро-дифференциальных уравнений. В действительности, каждая из парадигм имеет право на жизнь, их использование определяется только целью моделирования и связанным с этой целью выбранным уровнем абстракции при решении проблем.

Новое, недавно возникшее направление в имитационном моделировании – так называемое агентное (мультиагентное) моделирование (“agent-based modeling”), имеет свои особенности. Агентная модель представляет реальный мир в виде многих отдельно специфицируемых активных подсистем, называемых агентами. Каждый из агентов взаимодействует с другими агентами, которые образуют для него внешнюю среду, и в процессе функционирования может изменить как внешнюю среду, так и свое поведение. Обычно в таких системах не существует глобального централизованного управления, агенты функционируют по своим законам асинхронно.

Для построения модели робота-футболиста будет использоваться именно мультиагентный подход. Кончено, первоначально это обусловлено программой-сервером, предоставляющей среду моделирования. Но на самом деле мультиагентный подход помогает преодолевать множество проблем, возникающем при обычном моделировании, такие как взаимодействие объектов на основе выработанного поведения, в зависимости от внешних условий, коллективное решение одной задачи, реализация общей стратегии, предсказание и так далее.

Не смотря на то, что на сегодняшний день существует множество соревнований по симуляции, моделирование в которых осуществляется на основе собственной реализации сервера, существует общая модель агента-игрока. Для внешней среды каждый агент представляет собой черный ящик, который получает входную информацию (внешние воздействия), обрабатывает её, и производит воздействие на окружающую среду — какое-либо действие (рис. 2).

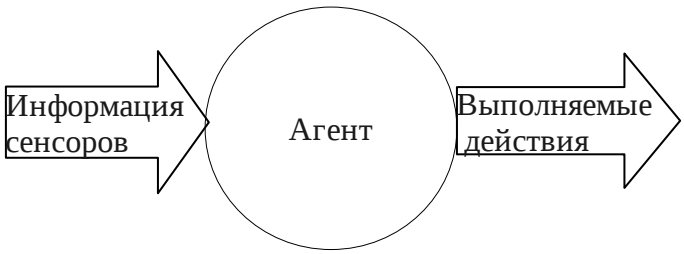

Рисунок 2 - Представление агента для среды моделирования

Для того что бы правильно формировать выполняемые действия, агенту необходимо сформировать представление об окружающей среде [3]. Для этого, необходимо классифицировать виды входной информации. Так как моделирование среды основано на приближенных к реальности условиях, то все входные данные агент получает со своих виртуальных сенсоров. Поэтому входную информацию можно разделить на следующие виды:

- машинное зрение (детектирование объектов на поле)
- слуховые сенсоры (услышать сообщение партнера по команде)
- различные сенсоры, для определения местоположения соперников, партнеров по команде и мяча. А так же разметки поля и ворот.

На основе вышеописанных входных данных игрок строит свою модель мира и предпринимает какие-либо действия в соответствии с поставленной перед ним задачей. Отсюда следует что входной набор данных неоднозначно определяет последующее действие. А только совокупность прошедших действий на поле в сумме с командной стратегией и специализацией игрока дают выходное воздействие.

На основе входных воздействий робот-футболист выполняет различные действия (если агент не произвел никакого действия в данный момент времени это так же считается действием). Действия игрока можно классифицировать следующим образом:

1) физическое действие (бег, поворот, удар по мячу и т.д.)
2) передать какую-либо информацию другому игроку или всей команде (как правило текстовое сообщение

Не смотря на кажущуюся простоту воздействий на смоделированный окружающий мир, множество вариантов очень велико. Такое количество вариантов, в свою очередь, определяет уровень сложности поставленной перед игроком задачи — его тактики. Поэтому каждый виртуальный агент должен обладать системой принятия решений, как на основе общей цели, так и на основе собственных прогнозов.

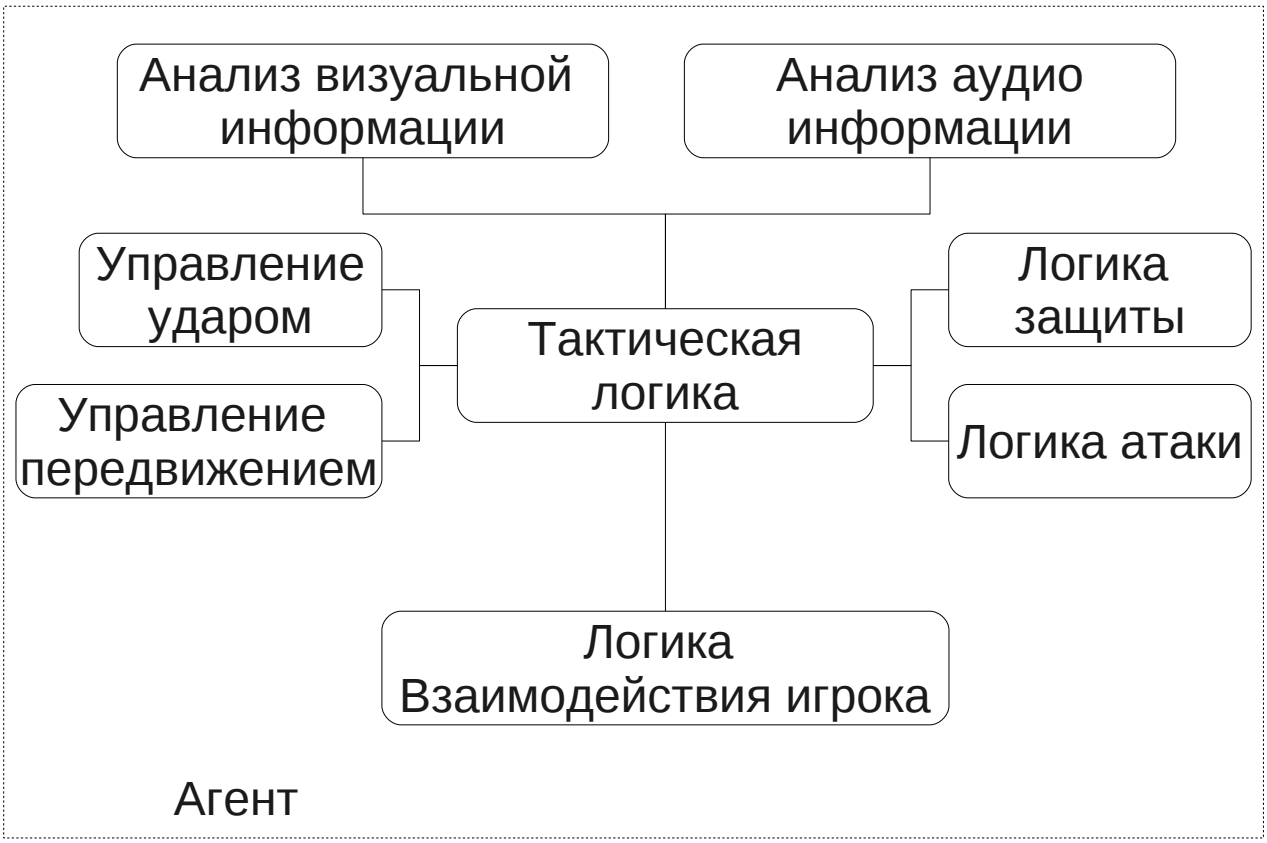

Рисунок 3 - Структура модуля агента-игрока

Не смотря на то, что сам по себе модуль будет являться агентом, необходимо определить его составные части. Основываясь на распространенных средах моделирования робофутбола можно выделить структуру, отображенную на рис. 3, описывающую модель робота-игрока. Управление ударом, анализ визуальной информации, расшифровка звуковой информации и управление бегом являются низкоуровневой функциональностью агента. Не смотря на кажущуюся простоту структурных единиц, существует множество подводных камней, встречающихся в некоторых из симуляционных сред. Одна из самых серьезных проблем — вносимые помехи. Помехи, сами по себе вносятся сервером, а так же как результат работы алгоритмов считывания и обработки данных, присылаемых с сервера. В связи с этим, для более точной обработки данных, одним из распространенных способов является применение искусственных нейронных сетей (ИНС), с помощью которых можно быстро и достаточно точно получать «предсказания» будущего поведения

окружающей среды. Ещё одним важным преимуществом ИНС является их работоспособность при минимальном количестве входных данных. Так же достаточно часто используются фильтры Калмана, обеспечивающие хорошее шумоподавление.

Функциональная часть модуля «тактика игрока» является самой важной и наиболее сложной, а значит и самой неопределенной, в плане реализации. При реализации мультиагентного подхода можно встретить множество различных архитектур агентов-игроков, многие из которых не имеют родственных корней. Но проанализировав многие из модулей, можно выделить некоторые подходы, наиболее популярные среди разработчиков:

1) ИНС;
2) нечеткая логика;
3) обучение с подкреплением.

Задача управления агентом может быть решена с помощью ИНС. В настоящий момент новым веянием в науке является применение модели спайковых нейронных сетей. Их особенностью является максимальное приближение к реакции нейрона человека, используемой в функции активации, которая зависит так же от времени. Тем самым, на основе нейронных систем данного типа можно строить сложные системы взаимодействия с окружающей средой. Не смотря на хорошую адаптивность систем, построенных на ИНС, их обучение до сих пор остается довольно трудоемкой задачей.

Применение методов нечеткой логики предоставляет мощный аппарат для манипулирования агентом в исследуемой среде. Поведение объектов в окружающей среде описывается настраиваемым набором правил. Адаптация правил происходит в процессе обучения агента с учетом поставленных целей.

Одним из самых эффективных методов для обучения виртуальных агентов является подкрепляющее обучение. Его основы, построенные на методе поощрения

и наказания, формируют поведение агента в зависимости от входных данных (расположения игроков и мяча, их направления движений, режима нападения или защиты команды), и произошедших событий на поле (забитый гол, оффсайд, нарушение правил). При правильном задании ограничений для агента и назначении за них наказаний, агент самостоятельно обучается поведению в окружающей среде и, как показывает практика, находит оптимальное решение поставленных задач.

Так же очень популярным является применение разнообразных алгоритмов на графах, геометрических методов (Делонова триангуляция), Марковских процессов, конечных автоматов в подходе к решению задачи.

Но не смотря на множество методов и подходов, одним из важных аспектов при проектировании виртуального агента-футболиста, является синтез нескольких различных решений. Тем самым слабые стороны одних методов компенсируются сильными сторонами других.

Для достижения быстрого результата и более глубокого ознакомления с аспектами программирования для сервера симуляции было решено взять открытый код чужой команды. На сегодняшний день есть несколько реализаций агентов-игроков, имеющих популярность у команд-новичков. Многие из них были проанализированы и был выбран вариант, наиболее подходящий составленной логической схеме работы модуля.

За основы низкоуровневых и среднеуровневых алгоритмов были взята реализация агента-игрока команды UvA Trilearn университета Амстердама. Она имеет ту же концепцию разноуровневой структуры организации модуля игрока, которая была принята в реализации данной работы.

3. Структура агента-игрока

3.1. Низкоуровневые методы

Детальное описание всех классов и методов, реализованных в ходе разработки и перенятых с других исходных кодов большое количество, поэтому ниже следует только краткое описать основных из функций.

За соединение с сервером отвечает класс Connection, в котором реализованы функции подсоединения, отсоединения, посылки информации серверу. Все данные передаются и принимаются посредству UDP сокетов. За обработку пакетов, полученных по UDP сокетам создан класс Parse, содержащий всевозможные методы для разделения сообщения на части.

Реализацей основных функций игрока и основных функций тренера являются классы BasicPlayer и BasicCoach соответственно. Здесь реализован весь функционал, который используется в более высокоуровневых методах.

Абстрактным классом для представления объектов на поле является класс Object. Он является базовым для всех объектов и содержит методы получения информации и задания данных объектам. Классы наследники Object это DynamicObject и FixedObject, где первый описывает объекты, которые могут перемещаться за момент времени по игровому полю, второй — статические объекты, такие как флаги на углах поля, лини разметки, ворота. Несмотря на то, что эти два класса можно так же отнести и к среднеуровневой логике, их методы, в основном, представляют простейшие действия, используемые для построения более сложных комбинаций.

3.2. Среднеуровневые методы

На основе вышеописанных классов строятся более сложные.

Среднеуровневые методы — это комбинации низкоуровневых методов, или более сложные действия.

Одним из таких классов является VecPosition. Он содержит x и y позиции, с помощью которых может рассчитывать расстояния до других объектов их направления и т. д. Все что связано с навигацией агента-игрока реализуется в этом классе.

Класс Stamina отвечает за своевременный отдых игрока, и не позволяет ему экономить силы во время игры.

Класс Time реализует методы мониторинга времени сервера для своевременной реакции на задержки в работе модуля. Это производится так же за счёт класса Timing, реализующего внутренний таймер агента. Основной необходимостью данных классов и их взаимодействия является ограничение по времени одного цикла симуляции. Его длительность составляет 300мс. И если за это время от агентов не пришло сообщения (команды), то в следующей итерации ничего не произойдет и агент останется на своем месте. Что бы не допустить такой ситуации очень важно вовремя её отследить и начать использовать более простые алгоритмы, требующие меньше процессорного ресурса.

BallObject и PlayerObject представляют классы наследники DynamicObject и описывают объект мяча и объект игрока соответственно.

3.3. Высокоуровневые методы

Высокоуровневые методы сами по себе очень интересны, так как именно в их реализации разработчик может использовать то, что хочет именно он. Единственное ограничение — это сам разработчик.

В рамках данной работы были реализованы алгоритмы высокоуровневой логики футболиста, которые будут описаны ниже. Все они реализованы в классе Player. Модули представляют собой стандартные алгоритмы поведения, в зависимости от окружающей ситуации и позиции на поле.

4. Алгоритмы обучения, используемые в высокоуровневых методах

При создании высокоуровневой логики закладывались следующие концепции [4]:

- Эволюционное обучение (генетические алгоритмы для обучения конечных автоматов);
- Обучение методом проб и ошибок, с помощью поощрений и наказаний (подкрепляющее обучение);
- Взаимодействие между агентами-футболистами на основе выбранной стратегии;
- Выбор тактики, на основе происходящих событий на игровом поле и счета, агентом тренером.

Структура тактического модуля агента представлена на рис. 4. Анализ информации и составления карты мира представляет собой алгоритм, который собирает все данные с сенсоров (зрение, слух) и преобразует в понятную для агента-игрока схему, где отображены видимые на данный момент защитники, его положение на поле, а самое важное положение мяча на поле. После этого анализируется состояние матча и результат подается для выбора дальнейшей стратегии. Немаловажно, что стратегия всех агентов игроков должна совпадать, поэтому агенты-игроки обмениваются сообщениями на поле, для выбора единой тактики для всей команды. Далее, на основании существующих стратений, выбирается стратегия. Если текущая стратегия совпадает с нынешней, и приносит отрицательные результаты, то на объект стратегии поведения команды подаётся воздействие «наказать», и, либо тактика видоизменяется, либо выбирается новая стратегия, в зависимости от настроек файла конфигураций. Если же тактика приносит успех, подается воздействие «поощрить» и коэффициент эффективности

тактики увеличивается. Если же тактика приносит нейтральный результат, производится минимальное воздействие наказания.

Далее на основе полученной стратегии и карты мира выполняется расчет следующего действия. Передаются команды среднеуровневой логике выполнить необходимые действия, и, если необходимо, передать сообщения другим агентам-игрокам на поле.

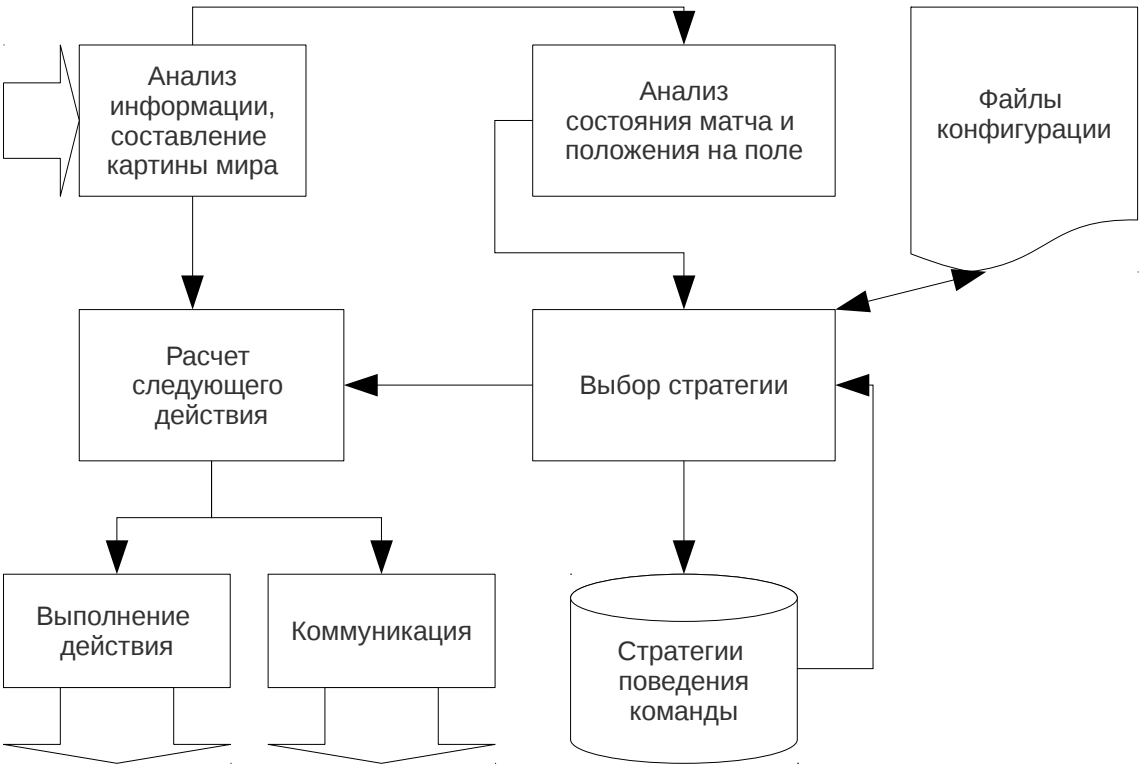

Рисунок 4 — Структура тактического модуля агента-игрока

Для полученной структуры агента необходимо адаптировать алгоритмы обучения [5]. Основной проблемой при обучении агента является задание ему минимальных исходных параметров. Так как в каждый момент времени на поле происходит множество изменений, и необходимо оценивать ситуацию и выполнять необходимые действия, то задание исходных параметров в ручную не представляется возможным. Для того что бы обучить агента-игрока начальным навыкам используется эволюционный метод обучения. Так как сама по себе

стратегия представляет ряд параметров-состояний, а тактика - конечный автомат, то возможно без особых проблем применить вышеуказанный способ обучения для агента-игрока.

Для обучения был создан ряд стандартных ситуаций (симуляций) на поле, и введена функция оценки отклонения от поставленной цели, что бы определить эффективность особи. Состояния конечного автомата, переходные состояния, входные и выходные параметры представляют собой хромосому, где каждая является её геном. Обучение происходит по стандартному алгоритму эволюционного метода, основные этапы цикла которого:

1. Оценка особей
2. Отбор лучшей половины особей
3. Скрещивание лучших особей, получение нового поколения, количеством равным старому.
4. Мутация нового поколения
5. Замещение старого поколения новым

Так как количество действий на поле ограничено и не столь велико, а количество входной информации наоборот имеет малую вероятность на повторения имеет смысл представить входные данные в виде десятичных дробных чисел, а выходные в виде бинарного числа. Тем самым, увеличивая количество генов к хромосоме и ускоряя процесс обучения. С начала модуль обучался для каждой отдельной ситуации, что бы более подробно проанализировать ситуации. После получения необходимых статистических данных агент обучался на всех ситуациях.

5. Результаты и перспективы развития

В ходе экспериментов было установлено что модуль справляется с поставленными перед ним задачами успешно. После обучения агент справлялся с поставленными задачами, и при возникновении новых ситуаций, адаптировался к ним, тем самым создавая новые, не заложенные в него с самого начала, стратегии. Единственным недостатком такого подхода, как оказалось, — время обработки информации, т. к. большое количество входной информации, которая поступает с сервера, требует много времени на обработку соответственно.

При реализации модуля агента-игрока были использованы высокоуровневые алгоритмы, для исследования возможностей его функциональности. Была рассмотрена концепция создания автономного ротоба-футболиста для виртуального мира. В перспективах развития команды по виртуальному футболу предстоит произвести следующие модернизации:

- оптимизация кода низкоуровневой логики;
- добавление дополнительного специфического функционала;
- модернизация высокоуровневой логики:
  - замена конечных автоматов марковскими процессами;
  - внедрение нейронных сетей, для ускорения обработки поступающей информации;
  - разработка комбинаций для нападения и защиты;
  - применение аппарата нечеткой логики.

ЗАКЛЮЧЕНИЕ

Полученные результаты позволяют сделать вывод, что на сегодняшний день симуляция игры в футбол является отличным полигоном для полноценных исследований в направлении искусственного интеллекта. Возможность отрабатывать существующие алгоритмы и ставить эксперименты в своих исследованиях является неоспоримым достоинством Robocup simulation league.

Для разработки высокоуровневой логики были успешно применены алгоритмы машинного обучения. Полученные результаты их применения показывают, что решение данной задачи возможно без разработки статического математического аппарата, который не только отличается своей сложностью, но и требует глубоких познаний в области математики. Используемые же алгоритмы дали возможность обучить и обеспечить, что не мало важно, адаптивную работоспособность системы, основываясь лишь на структуре используемых данных и их функционировании.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Дёмин В.В. Реализация модели робота игрока для Robocup Soccer Simulation Server / В.В. Дёмин, А.П. Дунец // Сборник конкурсных работ студентов БрГТУ. – Брест: БрГТУ, 2010. - с. 255-259
2. Noda, I., Matsubara H. Soccer server and reseaches on multi-agent systems. In Kitano, H., ed.: Proceedings of IROS-96 Workshop on Robocup. (Nov. 1996) 1-7
3. Дёмин В.В. Модель агента игрока робофутбольной команды / В.В. Дёмин, А.П. Дунец // Сборник конкурсных работ студентов БрГТУ. – Брест: БрГТУ, 2010. - с. 242-245
4. Peter S. Layered Learning in Multi-agent System [D]. Pittsburgh: school of computer science, Camegie Mellon University, 1998
5. M. Riedmiller and Artur Merke. Using machine learning techniques in complex multiagent domain / In I. Stamatescu, W. Menzel, M. Richter and U. Ratsch, editors, Perspectiveson Adaptivity and Learning, LNCS, Springer, 2002.

ОТЗЫВ НАУЧНОГО РУКОВОДИТЕЛЯ

на конкурсную работу Демина Владимира Владимировича по теме:

"Аспекты моделирования роботов игроков в робофутбольной команде"

студента 4-го курса группы АС-24 факультета электронно-информационных систем

Конкурсная работа студента Демина В.В. посвящена решению актуальной исследовательской задачи – созданию программного обеспечения, которое моделирует роботов-игроков в робофутбольной команде.

Задачей конкурсной работы было исследование структуры моделирующий системы, анализ правил проведения соревнований, подбор и исследование алгоритмов интеллектуального поведения.

Поставленная задача была успешно решена. При проведении исследований были изучены различные варианты реализации алгоритма моделирования поведения робофутбольной команды; на основании достоинств и недостатков аналогов была предложена своя структура моделирующей системы с учетом требований, которые предъявляет поставленная задача. Были рассмотрены различные интеллектуальные подходы, которые могут добавить в создаваемую систему гибкости и адаптивности.

В процессе исследований, проводимых в рамках конкурсной работы, студент Демин В.В. проявил себя как самостоятельный, грамотный и целенаправленный специалист, умеющий самостоятельно делать постановку задачи и доводить до конца ее решение. Показал отличное умение пользоваться специальной литературой.

Задание на конкурсную работу студент Демин В.В. выполнил в срок и в полном объеме.

Научный руководитель

Доцент кафедры «ИИТ» Дунец А.П.

Список научных работ автора

1. Дёмин В.В. Реализация модели робота игрока для Robocup Soccer Simulation Server / В.В. Дёмин, А.П. Дунец // Сборник конкурсных работ студентов и магистрантов. – Брест: БрГТУ, 2010. - С. 255-259

2. Дёмин В.В. Модель агента игрока робофутбольной команды / В.В. Дёмин, А.П. Дунец // Сборник конкурсных работ студентов и магистрантов. – Брест: БрГТУ, 2010. - С. 242-245

СВЕДЕНИЯ
об авторе научной работы, представленной на конкурс 2010 года,
и научном руководителе

УО «Брестский государственный технический университет»
(полное наименование вуза)

Автор научной работы	**Научный руководитель**
Фамилия Дёмин	Фамилия Дунец
Имя Владимир	Имя Андрей
Отчество Владимирович	Отчество Петрович
Курс 4 (в год проведения конкурса)	Занимаемая должность доцент
Факультет электронно-инфромацион-ных систем	Ученая степень
Специальность автоматизированные системы обработки информации	Ученое звание
Результаты опубликованы (получены патенты, лицензии и прочее)	Основное место работы (полное наименование организации)
Сборник конкурсных работ студентов и магистрантов, 2010 (печатный источник, год)	УО «Брестский государственный технический университет»
Домашний адрес, тел. г. Брест, Партизанский пр-т, д. 14, кв. 6 тел +375333226111	Служебный телефон Домашний адрес, тел. 80162428898 г. Брест, ул. МОПРа д. 6/2, к. 38 +375297949734
E-mail: spas.work@gmail.com	E-mail dunets@gmail.com

Рекомендован для участия в Республиканском конкурсе научных работ студентов высших учебных заведений Республики Беларусь в 2010 году (протокол №_____ от "_"_____г.).

Ректор_____ **П.С. Пойта**
(подпись) (И.О. Фамилия)