Футбол роботов на кафедре СА и У

Одной из перспективных областей в искусственном интеллекте робототехнике является разработка и исследование роботов для игровых приложений. Отличной игровой средой для исследований в этой области является недавно появившийся и быстро распространяющийся среди университетов и исследовательских центров мира футбол роботов (ФР) в рамках инициативы RoboCup [RoboCup, 1997].

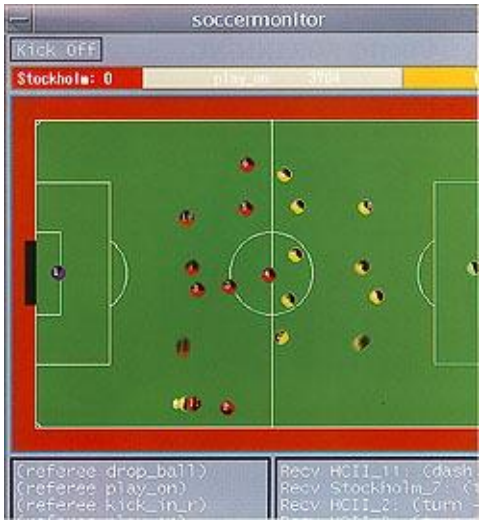
ФР появился в результате развития искусственного интеллекта и робототехники. Развитие первого направления в течение 40 лет в мае 1997г. привело к долгожданному успеху - ЭВМ фирмы IBM Deep Blue выиграла у чемпиона мира по шахматам. Можно считать, что с появлением RoboCup началось наступление роботов, оснащенных интеллектуальными системами управления, и в самой популярной игре мира - футболе. Развитие национальной лиги Японии по ФР, которая функционирует с 1992 года, привело к созданию международного проекта RobotWorldCup (RoboCup). В рамках этого проекта была организована федерация RoboCup, под эгидой которой, начиная с 1997, года проводятся мировые и локальные соревнования роботов RoboCup в нескольких лигах: симуляционной футбольной, малой и средней футбольной для колесных роботов, футбольной для четырехногих роботов фирмы Sony , симуляционной спасательной. Начиная с 2002 года, планируется проведение соревнований в футбольной лиге для гуманоидных роботов фирм Sony, Honda и др.

Заявлена и конечная цель развития RoboCup - создание команды гуманоидных роботов, способных в 2050 году одержать победу над командой чемпионов мира среди людей [Kitano et al., 1997].

Лиги футбола роботов

Чемпионаты мира по ФР - **RoboCup -** проводятся в нескольких лигах.

Simulation League. Симуляционный **ФР** является **виртуальным** и в стандартном исполнении реализован как распределенныя **клиент-серверная** система для UNIX- или Windows - платформ. **ФР-сервер** является координационной программой, владеющей всей позиционной информацией о симуляционной футбольной среде. Он сообщает программным **ФР-клиентам** текущее положение управляемых ими игроков (симулятор отображает **графически** их позиции на поле), позиции других игроков из ближайшего



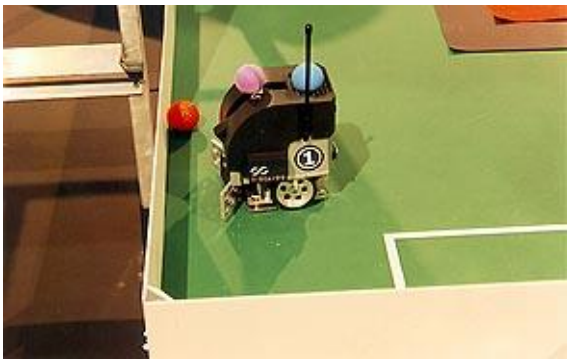
окружения и параметры мяча (позицию, скорость и направление). Он же принимает от ФР-клиентов информацию о выработанных ими решениях (действиях с мячом или без мяча: направление движения, сила и направление

удара по мячу, голосовые сообщения игрокам его команды), обрабатывает ее и отображает на графическом мониторе . В игре с ФР-сервером контактируют сеансами по 10 игроков все **22 РФ-клиента** (11 на 11 игроков команд соперников). Такая симуляция игры возможна как на отдельном

компьютере, так и на множестве компьютеров, объединенных сетью. Случайные факторы, введенные в параметры мяча, делают игру

разнообразной - нет двух одинаковых игр у одних и тех же команд - соперников.

Именно в этой лиге все новые идеи и подходы проходят первичную аппробацию, а все новые команды начинают с нее свое вхождение в **RoboCup**.

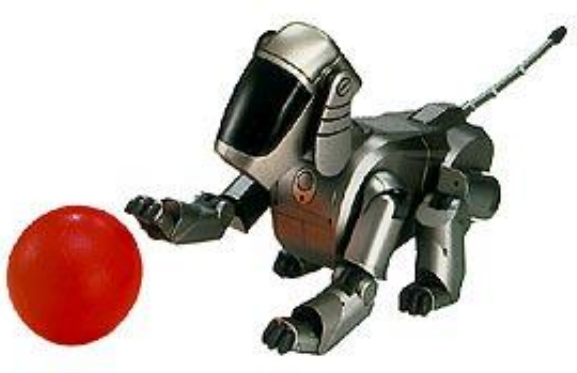


Small Size Robot League. В этой лиге соревуются **автономные мини-роботы** рамером не более **15*18 см** по периметру на колесном шасси. В качестве игрового поля используется стол размерами как для игры в пинг-понг. Положение роботов-игроков на поле фиксируется общей телекамерой, размещенной над центром поля. Координация и управление роботами осуществляется **клиент-серверной** системой сходной по идеологии с аналогичной системой симуляционной лиги. Аппаратное оснащение игры в этой лиге является относительно **дешевым**, что облегчает переход в нее из самой младшей симуляционной лиги.

Middle Size Robot League. В этой лиге играют **реальные роботы** размером **50*50 см**. в периметре на колесном шасси. Роботы могут иметь вес до 80-ти кг., что позволяет им иметь достаточно мощный бортовой компьютер и собственную систему технического зрения с подвижной телекамерой. **Полная автономность роботов** обеспечивается соответствующим программным обеспечением бортового компьютера, позволяющим управлять роботом, как отдельным игроком и взаимодействовать с другими участниками команды. Размер поля здесь значительно больше и определяется размерами 9-ти столов для пинг-понга. В команде играют не более 4-х роботов, один из которых является вратарем**.** Такие автономные роботы существенно более сложны, чем мини-роботы, и каждый из них стоит более 5-ти тысяч долларов. При их создании используются передовые технологии в области электромеханики и электроники, что часто оказывается под силу только серьезным робото-техническим лабораториям.

Sony Legged Robot League. В этой лиге играют только **четырехногие** роботы, производимые **корпорацией Sony**. Они полностью автономны и имеют цветную мини-телекамеру. Фактически - это игрушечные роботы-собачки, перепрограммированные для игры в футбол.

Каждый подобный робот стоит около 2-х тысяч долларов и поставляется **только с**



согласия Sony. В команде играют 3 робота, один из которых - вратарь**.** Игра роботов-собачек очень **зрелищна**, но движение и поведение их в игре пока еше далеки от совершенства, поэтому данная лига пока носит лишь **рекламный характер.**

Humanoid Robot League. Особенно трудно разработать и заставить играть **автономные двуногие шагающие роботы**, но и на этом пути исследовательские разработки уже достигли крупных успехов**.** На **2002** год в рамках **RoboCup02**, который будет проводиться во время чемпионата мира по футболу в Японии, запланировано первое соревнование двуногих **гуманоидных** роботов типа **HondaP3**.

RoboCupRescue. **RoboCup** федерация не ограничивается при аппробации новых идей и решений в искусственном интеллекте только игровыми футбольными приложениями. В **2000 году** в рамках чемпионата RoboCup, проводимого в Мельбурне, аннонсировано соревнование **роботов-спасателей** в новой **симуляционной лиге**. Разрабатывается сервер для моделирования операций спасения в условиях ситуаций **катастроф и стихийных бедствий**. В отличие от существующих приложений связанных с футболом, где количество **программ-агентов** ограничено, здесь число участников может достигать **нескольких тысяч.**

Наиболее интересные исследования связаны с симуляционной лигой RoboCup, в которой соревнуются команды виртуальных роботов в среде футбольного сервера [Noda, 1996]. Программные агенты, симулирующие роботов, имеют различные архитектуры. Типовой является архитектура, разработанная в университете Карнеги Меллона (США) [Veloso, 1996]. Соответствующая ей технология разработки агентов описана в монографии [Stone, 2000]. В настоящее время при разработке агентов начинают использоваться более сложные архитектуры когнитивных агентов с выделенным сценарным уровнем координации. Один из новых подходов в этом направлении развивается в Санкт-Петербургском политехническом университете с 1999 года [Станкевич и др., 1999].

Развитие футбола роботов в СПбГПУ и России

В 1998г в Санкт-Петербургском государственном политехническом университете была организована группа преподавателей и студентов, которая начала подготовку команды для выступления в симуляционной лиге Robocup99. Эта команда состояла из преподавателей и студентов кафедры «Системный анализ и управление», Факультет технической кибернетики. Была разработана оригинальная программа агента-футболиста для участия в RoboCup99 и получила при рождении имя Polytech100 в связи со 100-летием Санкт-Петербургского государственного технического университета. Вторая команда PSI была организована в Институте Программных Систем (Переславль-Залесский). Обе команды успешно дебютировали в RoboCup99 в Стокгольме. В 2001 году на базе Polytech100 образовалась команда DrWeb, которая заняла 4-е место на Открытом Чемпионате Германии в Падеборне, где собрались все лучшие команды Европы. С 2002 г. по 2004 г. российская команда STEP (Soccer Team of Electro Pult) участвовала в ежегодных Чемпионатах Мира по ФР RoboCup и достигла максимального результата, став Чемпионом Мира в Симуляционной двухмерной Лиге ФР на RoboCup2004 в Лиссабоне

(Португалия). В настоящее время в Санкт-Петербургском государственном университете, кафедра СА и У, функционирует и готовится к выступлениям на RoboCup команда по ФР в Симуляционной трехмерной Лиге.

На верхнем снимке представлены члены команды на вручении приза за 1-е место в Чемпионате Мира по ФР в Лиссабоне: капитан – Алексей Кричун (выпускник кафедры СА и У) и программист Антон Иванов (студент 6-го курса), а на нижнем – вся команда

после приезда в Санкт-Петербург: организатор команды доцент Л.А. Станкевич и капитан А. Кричун (в центре), члены команды С. Серебряков (слева) и А. Иванов (Справа).

Общепризнано, что участие в RoboCup даже в симуляционной лиге является очень престижным и показывает текущий уровень разработок в области искусственного интеллекта, МАС и обучения машин.

ФР - это удачный полигон для комплексного исследования и разработки в области информационных технологий 21-го века. Разработка играющих интеллектуальных роботов - сложная научно-техническая задача, имеющая огромное значение для развития искусственного интеллекта и робототехники. Велика роль ФР и в образовательной сфере, поскольку он является чрезвычайно привлекательным для студентов и аспирантов, обучающихся многоагентным технологиям.

Многоагентная среда симуляционного футбола роботов

Наиболее интересным для исследований по искусственному интеллекту и робототехнике является симуляционная футбольная лига. В многоагентной виртуальной среде ФР очень удобно проводить исследования по-разному построенных агентов, команды которых являются соперниками.

Реально в ФР используется симулятор типа RoboCup soccer server, развиваемый с 1996 года коллективом разработчиков и участников RoboCup [Noda, 1996] . Симулятор, действующий как сервер, обеспечивает среду и поддержку пользователей, которые желают строить свои собственные агенты, называемые клиентами или игроками. Клиентские программы связываются с сервером через UDP сокеты. Сервер симулирует движение каждого из объектов мира (игроков и мяча): клиент действует как мозг игрока, посылающий в каждом цикле симулятора (simulation_step) команды движения на сервер, а сервер отрабатывает эти команды и асинхронно посылает информацию о новом положении игрока и мяча обратно клиенту. Симулятор имеет инструмент визуализации, отображающий на экране монитора поле, движущихся игроков и мяч. Он также имеет функции судьи, который следит за правилами игры.

Сервер реализует двух размерную симуляцию (у объектов отсутствует высота). Поле имеет размерность 68*105 метров, соответствующих 680*1050 дискретам на экране (field_lenght*field_wight). Ворота имеют ширину 14,2 метра или 142 дискреты (goal_wight). На поле кружками отображаются 22 игрока и один мяч. Имеются также видимые игроками маркеры, включающие флаги и линии поля. Игроки и мяч являются мобильными объектами, для которых в каждый момент времени t определены: позиция (p^t_x, p^t_y), скорость (v^t_x, v^t_y) и направление головы θ^t. В каждом симуляционном цикле движение каждого объекта рассчитывается следующим образом:

$(u^{t+1}_x, u^{t+1}_y) = (v^t_x, v^t_y) + (a^t_x, a^t_y)$: ускорение ;

$(p^{t+1}_x, p^{t+1}_y) = (p^t_x, p^t_y) + (u^{t+1}_x, u^{t+1}_y)$: движение;

$(v^{t+1}_x, v^{t+1}_y) = decay * (u^{t+1}_x, u^{t+1}_y)$: затухание скорости;

$(a^{t+1}_x, a^{t+1}_y) = (0, 0)$:сброс ускорения,

где *decay* является параметром затухания, определяемым через ball_decay или player_decay. Параметры (a^t_x, a^t_y) являются ускорениями объекта, которые определяются параметром *Power* в команде разгона dash для игрока или команде kick для мяча по формуле

$(a^t_x, a^t_y) = Power * (\cos(\theta^t) , \sin(\theta^t))$.

Для игрока его направление вычисляется как

$\theta^t = \theta^t = Angle$,

где *Angle* является параметром команды поворота turn. Направление движения мяча определяется как

$\theta^t_{ball} = \theta^t_{kicker} + Direction$,

где θ^t_{ball} θ^t_{kicker} являются направлениями мяча и ударяющего игрока соответственно, а *Direction* – второй параметр команды kick.

Для имитации неопределенностей сервер добавляет распределенный вероятностный шум в движения всех объектов. Так, ускорение с шумом вычисляется по формуле

$(u^{t+1}_x, u^{t+1}_y) = (v^t_x, v^t_y) + (a^t_x, a^t_y) + (r_x, r_y)$,

где r_x, r_y являются случайными переменными с униформным распределением в диапазоне [-max, max], причем параметр max определяется как

$max = rand * |(v^t_x, v^t_y)|$,

где *rand* – параметр, определяемый played_rand или ball_rand.

Шум добавляется также в параметр *Power* в команде turn, т.е.

$Angle = (1+ r_{angle}) * Angle$.

Сервер превентивно предохраняет игроков от перемещения с максимальной скоростью (player_sp_max) путем присвоения каждому игроку параметра усталости. Этот параметр имеет 3 части: stamina – является текущим ограничением параметра *Power* команды dash; effort – представляет эффективность, с которой игрок может двигаться; recovery – представляет скорость, с которой stamina восстанавливается.

Агент получает три разных типа сенсорной информации: звуковую, зрительную и физическую. Для передачи звуковой информации все 22 агента используют единый ненадежный канал. Когда один агент или судья говорит, ближайшие агенты обеих команд могут слышать сообщение незамедлительно без задержки в формате hear(Time Direction Message). Здесь: Time – текущий цикл симуляции; Direction – относительное направление откуда пришло сообщение; Message – содержание сообщения. Заметим, что в сообщении нет информации об агенте, пославшем сообщение. Другое ограничение – агенты слышат только сообщения от игроков, находящихся внутри расстояния, определяемого параметром слуха (audio_cut_off_dist). Емкость сообщения также ограничена. Сообщение судьи является привелигированным и может прерывать сообщения агентов.

Зрительная информация формируется сервером в следующем формате:

see(Time ObjInfo ObjInfo ...),

где Time – текущее время; ObjInfo – информация о видимом объекте, в которой указывается выборочно имя объекта (имя команды и номер игрока, сторона ворот, мяч, разные флаги поля); дистанция до объекта и направление; скорость изменения дистанции и направления. Частота, диапазон и качество зрительной информации, посылаемой от отдельного агента, управляется целочисленными параметрами, определяющими: шаг посылки, угол зрения, качество зрения (высоко, низкое), ширину зрения (узкая, нормальная, широкая). Агент может изменять частоту получения зрительной информации: при низком качестве или узком угле зрения частота может быть увеличена в 2 раза.

По запросам агента сервер может послать физическую информацию о самом агенте (sense_body). Эта информация включает: текущие значения параметров stamina, effort, recovery; текущую скорость агента; текущие качество и ширину зрения. Таким образом, агент может иметь три типа сенсорной информации: hear – без задержки от ближайших агентов или судьи, see – каждые 150 мсек. с ограничениями по ряду параметров, sense_body – по запросу.

Агент может посылать серверу для исполнения несколько типов команд. Команда say передает текст, sense_body – требует от сервера информацию о физических параметрах игрока, change_view – изменяет параметры зрения игрока Агент имеет 4 команды для физического манипулирования миром: поворот (turn), разгон (dash), удар (kick), поймать мяч (catch). Далее рассмотрим эти команды подробнее:

Turn(*Angle*): параметр *Angle* индицирует угол поворота игрока в диапазоне [-180, 180 градусов]. Для движущегося игрока текущий угол уменьшается пропорционально его скорости (*actual_angle* = *Angle*/(1.0 + *interia_moment* * *player_speed*))

Dash(*Power*): параметр *Power* индицирует энергию ускорения в диапазоне [-30, 100 единиц]. Агент движется в направлении головы или корпуса вперед или назад (в последней версии сервера голова может поворачиваться отдельно от корпуса). Ускорение может уменьшается при низком параметре усталости (stamina).

Kick(*Power Angle*): параметр *Power* индицирует энергию удара, а парметр *Angle* – угол по отношению головы (или корпуса), в направлении которого ускоряется мяч. При ударе регулируется его сила: наиболее сильный удар обеспечивается, когда мяч близок к игроку и прямо перед ним по фронту; энергия удара уменьшается при увеличении расстояния игрока от мяча и/или угла удара по отношению к направлению игрока.

Catch(*Angle*): параметр *Angle* индицирует угол захвата мяча голкипером. Только он может хватать мяч и только в штрафной площадке. Захват мяча эффективен, если мяч находится в прямоугольнике размером catchable_area_w * catchable_area_l.

Sense_body(): требует физическую информацию от сервера и может выдаваться 3 раза за цикл (в последней версии сервера эта команда выполняется сервером каждый цикл автоматически).

Change_view(*view quality, view widht*): определяет изменение качества зрения (низкое/высокое) и ширины зрения (узкое/нормальное/широкое). Чем выше качество и больше ширина, тем меньше частота зрения. Команда может подаваться с частотой 1 раз за цикл.

Say(*Message*): параметр *Message* содержит текстовое сообщение, длинной 512 байт. Сообщение могут слышать все другие агенты, в том числе и противники. Агент может говорить так часто, как захочет. Но часто говорить бесполезно, поскольку другие члены команды слушают его только один раз за 2 цикла симуляции.

Описанные свойства симулятора позволяют создавать изменяемую реалистичную футбольную среду для проведения исследований. Реалистичность среды обеспечивается многими источниками шума (параметры действий, движений и зрения). Сенсорная информация может быть потеряна, так как нет гарантии, что все посланные команды выполнятся сервером. Симуляция происходит в реальном времени: зрительная информация получается агентом с интервалом 150 мсек. (при высоком качестве и нормальной ширине); звуковая информация получается асинхронно; агенты могут асинхронно с получением сенсорной информации формировать команды действия каждые 100 мсек.

Разработка программных агентов

Чтобы создать пригодную для игры команду агентов, требуется использовать архитектуру агента, приспособленную для работы в команде с общей целью. Теория многоагентных систем предлагает много архитектур таких агентов с распределенным планированием, торговыми или контрактными протоколами обмена информацией для инициации планов команды. Однако, динамика, реальное время и ненадежный коммуникационный канал сервера ФР не позволяют организовать сложные протоколы обмена и распределенные планы. Один из приемлемых для ФР подходов предполагает использование при конструировании агентов концепции периодической синхронизации команды и соответствующей архитектуры [Stone, 2000]. При этом в течение ограниченных периодов агенты действуют самостоятельно, а синхронизация предполагается только периодически после получения по коммуникационному каналу необходимой информации.

Рассмотрим построенную в рамках такого подхода архитектуру агента (рис.1) соответствующую играющему агенту команды Polytech- DrWeb [Akhapkin et al., 2001]. Агент включает несколько уровней: тактический, поведения и навыков.

Тактический уровень выбирает текущее поведение, основываясь на глобальной информации обо всех игроках, положениях маркеров поля по отношению к игроку и положении мяча, а также параметрах определяющих стратегию команды (роли игроков, формации, планы игры), которые инициализируются на старте игры. Возможно динамическое изменение стратегии команды в процессе игры.

Уровень поведения включает процедуры, определяющие поведение агента при игре: ведение мяча с препятствиями (dribbling) , перехват мяча (intercepting), пассование (passing), прессинг (pressing) и т.д. Различное поведение выбирается в соответствии с текущими состояниями среды и игрока. При реализации разного поведения используются процедуры уровня навыков игрока.

Уровень навыков игрока включает бегание с разгонами и поворотами (running), обход препятствий (avoiding), удар в данную позицию (kicking) и т.д. Результатом работы процедур этого уровня является последовательный набор команд, исполняемых сервером (типы таких команд были перечислены в предыдущем разделе).

Состояние игрока (Play state)- это набор его внутренних параметров: энергических (stamina, effort, recovery), кинетических и различных статистических параметров. Этот набор параметров необходим для выработки правильных решений с учетом усталости игрока и его текущих возможностей.

Синхронизация - "сердце" агента. Она использует моментальный снимок (snapshot) состояния окружающей среды и агента для управления его реакцией. Действие этой компоненты похоже на планирование работы ЭВМ операционной системой реального времени. Качественная синхронизация позволяет избежать возможных пропусков команд и оптимизирует взаимодействие клиента с сервером.

Модель окружающей среды (World model) реализует: моделирование поведения противника, слежение и предсказание движений мяча, моделирование и слежение за расходом энергии игрока. От качества модели среды зависят прогнозирующие возможности агента и эффективность принимаемых им решений.

Представление поля (Field representation) дополняет модель среды, сохраняя текущие параметры, привязанные к маркерам футбольного поля: позиции игроков, позиции маркеров по отношению к игроку, позицию и направление движения мяча и т.д.

Восприятие (Perception) отвечает за прием информации (зрительной, звуковой, физической) от сервера, ее распознавание и формирование условий, необходимых для выбора текущего поведения агента.

Агент должен работать, как клиент сервера, определяющего окружающую среду (Environment). Поэтому очень важна правильная синхронизация его действий в цикле симуляции. При нарушении синхронизации возможны случаи нарушения работы агента, например, пропуски летящего мяча, неправильные пассы и пр.

Программа-агент, как правило, реализуется на языке Си для операционной среды Unix,

Рис.1. Архитектура программного агента

поскольку сервер работает в этой среде. Существуют достаточно обширные библиотеки модулей и оболочки, которые целесообразно использовать при разработке программы (см. сайт www.robocup.org). Некоторое упрощение разработки достигается использованием готовых модулей нижнего уровня агента (навыков работы с мячом). При этом основные усилия сосредотачиваются на понимании работы заимствованных модулей и разработке верхних поведенческих уровней агентов. При таком подходе, однако, агент может потерять работоспособность при изменении настроек сервера перед соревнованиями. Наиболее сложный момент – отладка поведения агента в игровых ситуациях, которых может быть очень много. Лучшие программы 2000-2001 годов включали обязательный сценарный уровень координации команды в игре и дополнительную программу тренера, позволяющую менять уставших игроков и корректировать стратегию и тактику команды в процессе игры.

Когнитивный подход к организации агента

В настоящее время существенно возросли ресурсы вычислительных сетей, которые используются при проведении чемпионатов RoboCup. В принципе, стало возможным запускать в игре каждого агента на отдельном процессоре большой мощности. Это позволяет реализовать более мощные архитектуры агентов с усложненным поведением. В данном разделе приводится краткое описание архитектуры когнитивного агента (КА), разработанного с целью существенного повышения эффективности игры при достаточно больших вычислительных ресурсах [Stankevich, 1999].

КА для ФР имеет многоуровневую поведенческую организацию, в какой-то мере, соответствующую структуре поведенческих функций футболиста-человека. Используя подход с позиций когнитивной психологии, можно выделить ряд поведенческих функций футболиста, связанных с мыслительными способностями (когнитивных функций). Такими функциями являются: оценка позиций партнеров и противников, принятие решений в атаке или защите в положении с мячом или без мяча; научение при тренинге или в игре путем пополнения знаний о решениях в различных ситуациях игры и др. Эти функции определяют командную стратегию и тактику при противоборстве с противником. КА строится на основе иерархического набора когнитивных функций, определяющих его

поведение, и аффективных функций, формирующих исполнительные команды для позиционирования игрока и работы с мячом.

Можно выделить такие особенности разрабатываемого КА:

1. Агент построен, как многоуровневая когнитивная система, т.е. обучаемая интеллектуальная система с нервно-системной организацией поведения, функций и структуры. Накопление знаний и формирование когнитивных функций такой системе производится путем обучения в процессе тренинга или игры. Знания и функции хранятся и используются в ассоциативной нейрологической форме. Когнитивные функции определяются требуемым поведением при достижении цели.
2. Введена специфическая когнитивная функция для принятия решений агентом при организации поведения на верхнем уровне управления в соответствии с деревом решения, которая может подстраиваться путем обучения для адаптации к изменениям окружения.
3. Введен специфический обучающийся модуль для автоматического связывания агента с другими агентами команды в процессе игры с целью улучшения координации при защите и атаке.
4. Использован изменяемый набор когнитивных поведенческих функций средних уровней, определяющих индивидуальную тактику агента, с оперативной перестройкой для адаптации к игровой ситуации.

Архитектура КА представлена на рис. 2.

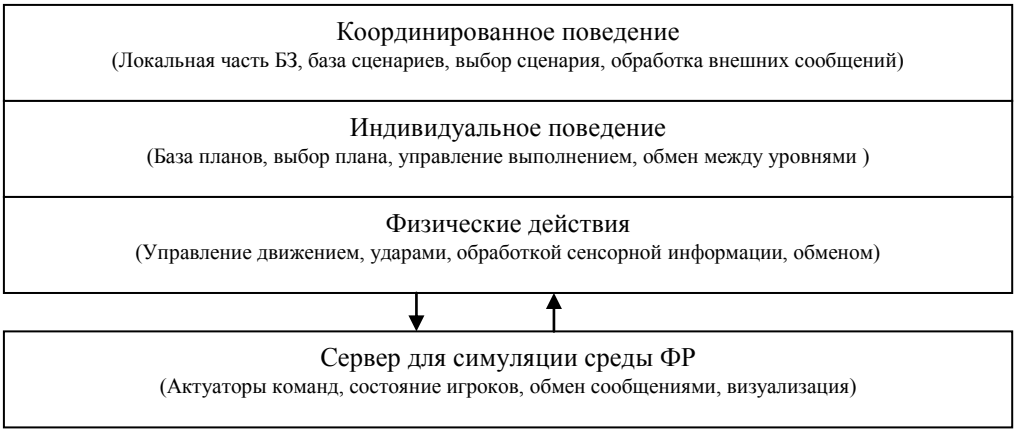


Рис. 2. Архитектура КА для ФР

КА включает три слоя: Физических действий (ФД), Индивидуального поведения (ИП), Координированного Поведения (КП). Так, построенный КА может управлять игроком, принимая сенсорную и обменную информацию и вырабатывая соответствующие сигналы управления на актуаторы игрока (модули сервера, реализующие команды агента).

ФД-слой КА реализует нижние уровни управления. В этом слое собраны модули, вырабатывающие команды управления игроком через актуаторы, расположенные в сервере:

- Управления движением – для координированного управления игроком с помощью выработки соответствующей последовательности команд движения: turn, dash, catch, реализующих его движение в пространстве и времени;
- Управление ударами и пассами - для координированного управления игроком при выполнении ударов по воротам и пассов, реализуемых с помощью команды удара kick;

- Обработка сенсорной информации - для управления получением информации от сервера о позиции игрока и его физических параметрах с помощью команд сенсорики: sense_body, change_view;
- Управление обменом – для координирующего управления приемом и передачей сообщений через канал связи с помощью команды коммуникации say;
- Движение, Удар, Сенсорика, Коммуникация – для сопряжения с соответствующими актуаторами сервера.

ИП-слой КА реализует средний уровень управления игроком, вырабатывающий и исполняющий план поведения (последовательности действий) игрока, действующего в индивидуальном режиме. Здесь собраны модули:

- База знаний (БЗ) планов агента – для накопления и хранения планов индивидуального поведения игрока в различных игровых ситуациях;
- Выбор текущего плана – для определения текущей ситуации и генерации плана действий при игре в индивидуальном режиме или выполнении сценария координации при выполнении задания в группе игроков;
- Управления выполнением текущего плана - для выработки координирующих управлений для модулей нижнего уровня (ФД-слоя) в соответствии с выбранным планом и сценарием;
- Обмен между уровнями – для связи с верхним или нижними уровнями управления.

КП-слой КА реализует высший уровень управления игроком, вырабатывающий сценарий координации при выполнении некоторой игровой «заготовки» группой игроков, решающей общую задачу. Сюда включены модули:

- Локальная часть распределенной БЗ – для накопления и хранения определенной для данного игрока, как члена группы, части БЗ, необходимой для выполнения своего задания при реализации игровой «заготовки»;
- База сценариев – для накопления и хранения набора сценариев координированного поведения игрока в группе при выполнении своего задания;
- Сценарий координации - для выработки сценария индивидуального поведения игрока с учетом общей цели группы, внешних сообщений от других игроков и текущей обстановки при выполнении сценария;
- Внешнее сообщение – для передачи и приема сообщений от других игроков своей команды и команды противника через общий канал связи.

Построение КА как многослойной когнитивной системы с перечисленными модулями в соответствующих слоях позволяет реализовать многоуровневое управление игроком при выполнении индивидуальной игры или группового задания. Сценарии координации, планы действий и координированные управления актуаторами при реализации игровых действий могут быть сформированы за счет обучения соответствующих модулей системы по примерам, генерируемым специальным агентом-тренером с учетом опыта выполнения игровых «заготовок» и типовых действий игроков. БЗ сценариев и планов могут быть уточнены в процессе накопления опыта реальных игр за счет оперативного дообучения системы. Возможна организация оперативного самообучения агентов в процессе игры.

Рассмотрим для примера важную когнитивную функцию ИП-слоя агента, определяющую выбор поведения игрока, владеющего мячом (Action with ball). Пусть номер этого игрока i. Тогда выбор поведения с учетом положения всех остальных игроков может быть сделан следующим образом.

Для всех номеров игроков нашей команды j, информация о положении которых известна, вычисляем оценку:

$$K(j) = F\{C(i, j)W(i, j)\sum_{t=0}^{n}[A(t)P(j, M(t))]\};$$

$$\forall j \subset [0, \ldots, 11] \vee j \neq i ,$$

где:

C(i,j) – коэффициент, выражающий «желательности» взаимодействия с j-ым игроком, постоянен в рамках стратегии.

W(i,j) – коэффициент адаптации, перед началом игры устанавливается в некоторое значение, меняется в ходе игры и не зависит от стратегии.

A(t) – коэффициент выражающий степень учета прогноза игровой ситуации на t-ый момент времени, постоянен в рамках стратегии.

P(j,M(t)) – функция нейрологики, вначале по игровой ситуации M(t) для j-ого объекта, вычисляющая входные параметры для нейрологического модуля, а потом производящая его запуск.

F() – нормирующая функция.

Выбираем S - множество номеров соответствующих оценкам со значениями большими некоторого порога.

В множестве S случайным образом выбираем элемент, если его номер равен:

0 – то выбираем в качестве поведения удар по воротам;

i – то выбираем в качестве поведения ведение мяча;

иначе пас игроку с соответствующим номером.

Внутренняя оценочная функция позиции P(j, M(t)) реализуется с помощью нейрологического модуля, который может обучаться по примерам. Он представляет собой ассоциативную сеть с нечетко-логическим вычислительным базисом в узлах и слоистой структурой. Такая сеть способно обучаться в реальном времени, поскольку, в отличие от обычной нейронной сети, при запоминании примера подстраивает только один весовой коэффициент. Примеры правильного поведения могут формироваться либо вручную, либо с помощью специальной программы-тренера.

Другие когнитивные функции, реализуемые в ИП-слое агента: Действия без мяча (Action without ball), Выбор текущего плана (Plan choice), Выполнение текущего плана (Plan management) и пр. реализуются также на обучаемых нейрологических модулях.

Когнитивные функции, реализуемые в ИП-слое, формируют параметры для аффективных функций, реализуемых в ФД-слое. Основные из этих функций: Управление ударом (Kicking), Перехват мяча (Interception), Ведение мяча (Dribbling), Защита ворот (Goaltending), Оборона ворот (Defending), Выбивание в открытую зону (Clearing) и пр. являются типовыми и реализованы подобно достаточно удачно выполненным функциям навыков в агенте CMUnited-99 [Stone, 2000].

Наиболее сложным является реализация в КП-слое когнитивных функций для организации игры по сценарным заготовкам. При выборе текущего сценария используется когнитивная функция Распознавания ситуации (Recognition), настраиваемая путем обучения. Периодическая координация агентов при выполнении выбранного сценария реализуется с участием когнитивной функции Координации по обмену (Coordination), распознающей сообщения членов команды изменяющей ход выполнения сценария, если это необходимо.

Предварительные исследования прототипа когнитивного агента показали возможность его обучения не только функции позиции, но и другим сложным поведенческим функциям. Предполагается использование подобного когнитивного агента командой Polytech Санкт-Петербургского ГТУ в чемпионате RoboCup-2002.

Выводы

Чемпионаты по футболу роботов RoboCup, проводимые регулярно во всем мире, являются мощным стимулом развития интеллектуальных систем и робототехники. Особенное значение имеет симуляционная лига, в рамках которой проверяются результаты новейших

разработок в области многоагентных систем, способных реализовать сложное групповое поведение объектов при решении общих задач в условиях противоборства.

Сервер, используемый в симуляционном футболе, обладает разнообразными средствами, позволяющими проводить разносторонние исследования поведения агентов. Он постоянно совершенствуется, чтобы расширять область исследований и усложнять агентов.

Проверенным путем разработки программных агентов-футболистов является использование типовых архитектур агентов, реализованных, например, в Университете Карнеги Меллон (США) или в Санкт-Петербургском ГТУ.

Повышение эффективности агентов-футболистов может быть достигнуто использованием при их проектировании когнитивного подхода, позволяющего реализовать сложные функции восприятия и поведения за счет применения специальной архитектуры с уровнем сценарной координации игры и специальных нейрологических модулей, обучаемых в реальном времени по примерам.

[Станкевич Л.А. и др., 1999] Станкевич Л.А., Городецкий В.И., Ахапкин С.В., Васильев С.В. Футбол роботов – многоагентная среда для исследования группового поведения интеллектуальных роботов// Труды 10-й НТК «Экстремальная робототехника», С-Перебург, Изд-во СПбГТУ, 1999.

[Akhapkin et al., 2001] Akhapkin, S. and Stankevich, L. DrWeb team description. Simulation league, RoboCup-2001, http// www.robocup.org.

[Kitano et al., 1997] Kitano, H. Et al. The RoboCup synthetic agent challenge 97// Proc. of the IJCAI-97, San-Francisco, CA. Morgan Kaufmann, 1997.

[Noda, 1996] Noda, I. Soccer server: a simulator of RoboCup. AI Symposium'95, Japanese Society for AI, 1995.

[RoboCup, 1997] RoboCup web page (1997), http//www.robocup.org.

[Stankevich, 1999] Stankevich, L.A. A cognitive agent for soccer game. Proceeding of First Workshop of Central and Eastern Europe on Multi-agent Systems (CEEMAC'99), Printed by "Anatolyi", S-Petersburg, 1999.

[Stone, 2000] Layered learning in multiagent systems. A winner approach to Robotic Soccer. The MIT Press, 2000.

[Veloso, 1996] M.Veloso, P.Stone “A Layered Approach to learning client behaviors in the RoboCup Soccer Server.” 1996.