Оглавление

В.В. Кугуракова

От составителя

Р.Р. Газизов, Д.И. Костюк, В.В. Кугуракова

КОНФИГУРИРОВАНИЕ ВЫЧИСЛИТЕЛЬНОГО КЛАСТЕРА ДЛЯ РАСПРЕДЕЛЕННОГО РЕНДЕРИНГА ВЫСОКОРЕАЛИСТИЧНЫХ ЗД СЦЕН

А.С. Гришина, В.В. Кугуракова

ФРЕЙМВОРК ДЛЯ ОБЛАЧНОГО ВИДЕОМОНИТОРИНГА ЧЕРЕЗ IP-КАМЕРЫ С ИНТУИТИВНО-ПОНЯТНЫМ ИНТЕРФЕЙСОМ

В.В. Кугуракова, А.М. Степанов

КОНТРОЛЛЕР РЕАЛИСТИЧНОГО ПОВЕДЕНИЯ СТАЙ/СТАД ЖИВОТНЫХ

А.Л. Шайхутдинов

МЕТОДЫ «ОЖИВЛЕНИЯ» MIDI-ПАРТИЙ УДАРНЫХ МУЗЫКАЛЬНЫХ ИНСТРУМЕНТОВ

ОТ СОСТАВИТЕЛЯ

Настоящий тематический выпуск журнала «Электронные библиотеки» содержит новые научные результаты, полученные сотрудниками и студентами Учебно-практической лаборатории визуализации и разработки игр (Digital Media Lab – DML) Высшей школы информационных технологий и информационных систем (ИТИС) Казанского (Приволжского) федерального университета (КФУ). Представленные статьи описывают решение разноплановых задач как в рамках исследований, проводимых в Лаборатории, так и в рамках дипломных проектов выпускников ИТИС. Это разработка новых методов для «оживления» midi-партий ударных музыкальных инструментов, конфигурирование вычислительного кластера для распределенного рендеринга высокореалистичных 3D сцен, создание контроллера реалистичного поведения стай и стад животных, проблемы облачного видеомониторинга через IP-камеры. Сфера применения результатов этих исследований – виртуальная и дополненная реальности в образовании; модернизация музеев с использованием виртуальной и дополненной реальностей; наблюдение за поведением людей и животных, а также видеонаблюдение за объектами с помощью камер.

Составитель тематического выпуска В.В. Кугуракова

УДК 004.75+004.272.26+004.925.3

КОНФИГУРИРОВАНИЕ ВЫЧИСЛИТЕЛЬНОГО КЛАСТЕРА ДЛЯ РАСПРЕДЕЛЕННОГО РЕНДЕРИНГА ВЫСОКОРЕАЛИСТИЧНЫХ ЗД СЦЕН

Р.Р. Газизов[1], Д.И. Костюк[2], В.В. Кугуракова[3]

[1-3]*Казанский (Приволжский) федеральный университет*

[1]starkindustries14579@gmail.com, [2]xdxnxkx@gmail.com, [3]vlada.kugurakova@gmail.com

Аннотация

Описан способ реализации распределенных вычислений с использованием GPU и современных инструментов администрирования и управления расчетами для ферм рендеринга. Проведен сравнительный анализ традиционного метода вычислений на CPU и современных подходов. Рассмотрены различные конфигурации вычислительного кластера. Определен наиболее производительный способ выполнения визуализации.

Ключевые слова: визуализация, распределенный рендеринг, вычислительный кластер, конфигурация.

ВВЕДЕНИЕ

Трёхмерная графика играет в современном мире всё большее значение. Благодаря ей можно создавать фотореалистичные или стилизованные изображения, получать представления о реальных или вымышленных объектах, а также изображения того, чего обычно человек не в силах увидеть невооружённым глазом. Такими примерами могут служить модель галактики или множество форм микроорганизмов. Благодаря этому компьютерная графика активно используется во многих отраслях жизни, таких, как архитектура, промышленность, киноиндустрия, игровая индустрия, рекламная продукции, наука, медицина.

В процессе работы с трёхмерной графикой приходится так или иначе совершать следующие шаги:

- моделирование – создание трёхмерной математической модели сцены и объектов в ней;

- текстурирование – назначение поверхностям моделей растровых или процедурных текстур (подразумевает также настройку свойств материалов – прозрачность, отражения, шероховатость и пр.);
- освещение – установка и настройка источников света;
- анимация – придание движения объектам;
- динамическая симуляция – автоматический расчёт взаимодействия частиц, твёрдых/мягких тел и пр. с моделируемыми силами гравитации, ветра, выталкивания и др., а также друг с другом;
- рендеринг (визуализация) – построение проекции в соответствии с выбранной физической моделью;
- композитинг (компоновка) – доработка изображения.

Рендеринг является важным этапом в работе с трёхмерной графикой, так как от его результата зависят реалистичность и эффектность итоговой работы. Сам по себе этот процесс очень ресурсоёмкий, вследствие чего он занимает продолжительное время вычислений. К примеру, визуализация видеоролика длиной в несколько минут в высоком разрешении может выполняться несколько дней или недель, в зависимости от сложности сцен. Поэтому зачастую не хватает вычислительной мощности одного компьютера для выполнения поставленной задачи визуализации. Возникает необходимость решения этой проблемы.

Один из способов решения трудоемких вычислительных задач – это использование вычислительного кластера или группы компьютеров, предназначенных для распределенных вычислений в параллельном режиме.

Заранее предпросчитанный рендеринг может быть использован в подготовке удалённых планов для крупномасштабных культурно-исторических реконструкций, например, [1].

РАСПРЕДЕЛЕННЫЙ РЕНДЕРИНГ

Техника распределенного рендеринга позволяет распределить задачи визуализации между несколькими компьютерами в сети, то есть производить вычисления мощностями нескольких компьютеров (см. рис. 1). При распределенной визуализации анимации (секвенции) между компьютерами распределяются диапазоны кадров. Каждая машина проводит рендеринг своей части, кадр за кадром, пока не решит поставленную задачу. В случае с визуализацией одного кадра, но очень большого разрешения, можно применить следующий способ:

основная идея состоит в том, что кадр, который подвергается рендерингу, делится на небольшие участки (бакиты) (см. рис. 2); каждой машине в сети раздается некоторое их количество; результаты расчетов собираются в итоговое изображение.

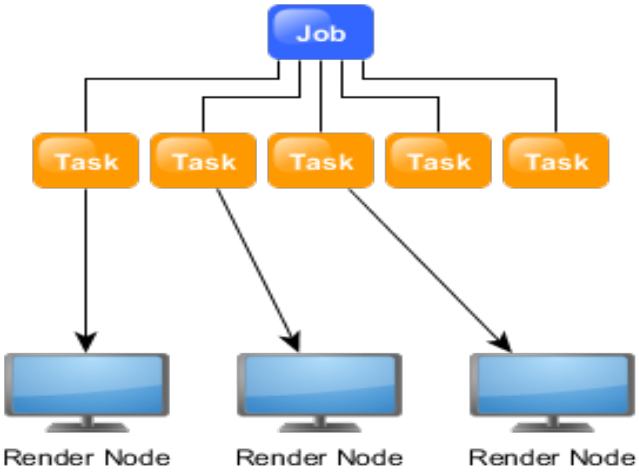

Рис. 1. Распределение работы между нодами

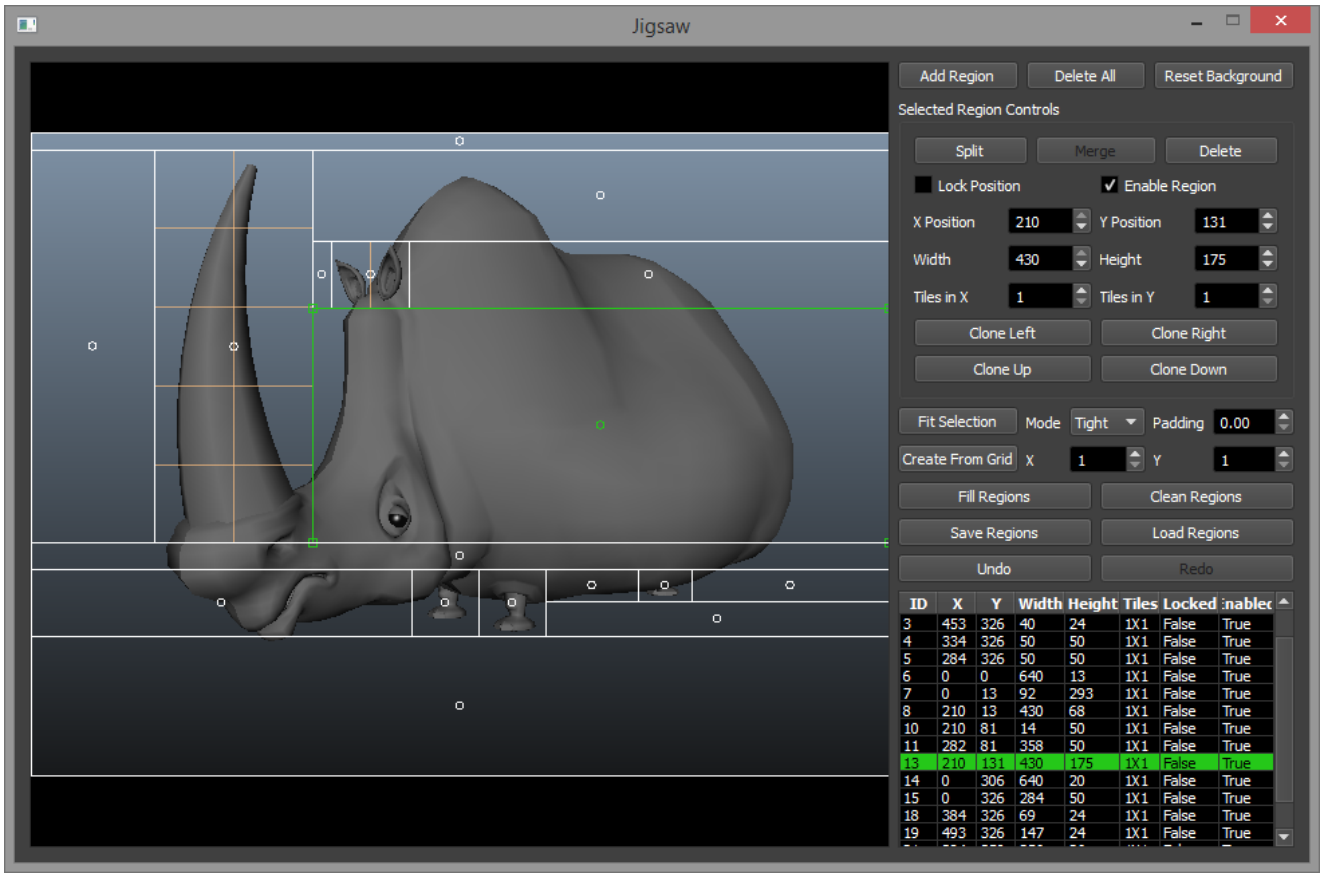

Рис. 2. Разбиение кадра на бакиты

ВЫЧИСЛИТЕЛЬНЫЙ КЛАСТЕР

Что же представляет из себя вычислительный кластер? Каждый вычислительный узел – это компьютер или нода, которые имеют собственную оперативную память, операционную систему и т. д. Чем больше компьютеров находится в вычислительной сети, тем меньше времени требуется на визуализацию сцены [2].

Узлы объединены высокопроизводительной коммуникационной сетью, обладающей минимальными задержками и широкой полосой пропускания информации, то есть компьютеры «видят» друг друга через сетевое окружение. На все машины установлены идентичное программное обеспечение и плагины. Если компьютеры не в прямой зоне досягаемости, то необходимо используются утилиты, обеспечивающие удаленный доступ, для работы с удаленными машинами.

Все необходимые данные для распределенного рендеринга должны храниться на сервере или общем сетевом диске, к которому будут обращаться вычислительные узлы. Этот диск хранит файлы сцен, текстуры, настройки. Он же принимает выходные данные – итоговый результат. Он виден всем машинам под одним и тем же именем.

В настоящее время описанные кластеры находят широкое применение по всему миру.

СРЕДСТВА ВИЗУАЛИЗАЦИИ

Визуализацию можно выполнять как на CPU, так и на GPU. Мы выбрали два наиболее популярных на данный момент решения из специализированного программного обеспечения для визуализации: Redshift (GPU); Arnold (CPU). Каждый из этих визуализаторов (рендеров) поддерживает распределенные вычисления. Они производят визуализацию с использованием алгоритма трассировки луча, что позволяет получить фотореалистичные изображения.

Визуализатор Arnold использует при рендеринге в качестве вычислительной платформы центральный процессор CPU. Преимущества Arnold – это высокая оптимизация, гибкость и система плагинов, к которым относятся процедурные функции, шейдеры, свои драйверы и прочее. Однако визуализация замедляется из-за данных, которые находятся в кэш-памяти, это увеличивает используемые оперативную и дисковую память.

Программа Redshift использует графический процессор GPU. Это первый в мире рендер, полностью ускоренный на GPU. Для его работы необходима видеокарта от Nvidia. Он поддерживает следующие алгоритмы глобального освещения: Brute-force GI; Photon Mapping; Irradiance Cache; Irradiance Point Cloud. При визуализации данные подгружаются из оперативной памяти или жесткого диска в видеопамять графического процессора. Достаточно 1 гигабайта видеопамяти, чтобы провести рендеринг сложных сцен [3, 4].

Достоинством GPU-процессоров является огромная вычислительная мощность. При большом объеме информации, которая обрабатывается, GPU имеет явное преимущество. Поэтому область применения GPU бесконечна.

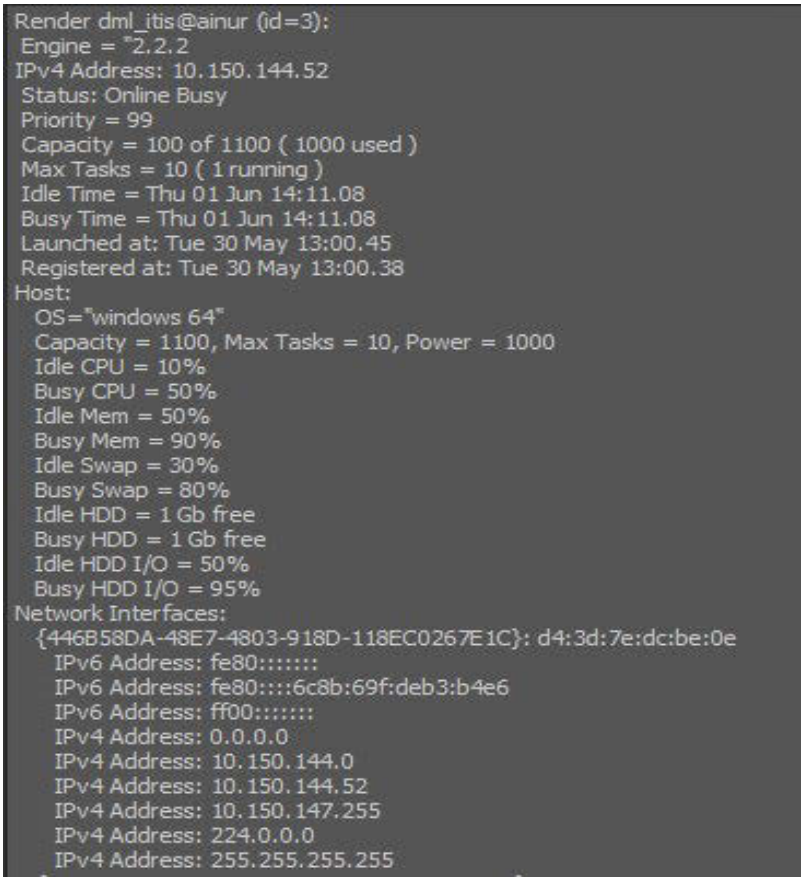

Рис. 3. Основные ресурсы

ИНСТРУМЕНТЫ АДМИНИСТРИРОВАНИЯ И УПРАВЛЕНИЯ РАСЧЕТАМИ

В нашей работе использовалось готовое программное решение Afanasy. С его помощью мы легко создали систему распределенного рендеринга и получили доступ к таким инструментам администрирования, как:

- контроль работы для распределенных вычислений;

- отслеживание ресурсов, используемых нодами (использование центрального процессора, оперативной памяти, жесткого диска, сетевого трафика, скорости работы дисковых операций ввода-вывода (см. рис. 3); это помогает определить, что замедляет процесс визуализации);
- удаленное отслеживание выполняемых задач через Веб и стационарные интерфейсы (см. рис. 4);
- возможности учета собственных нужд программным путем.

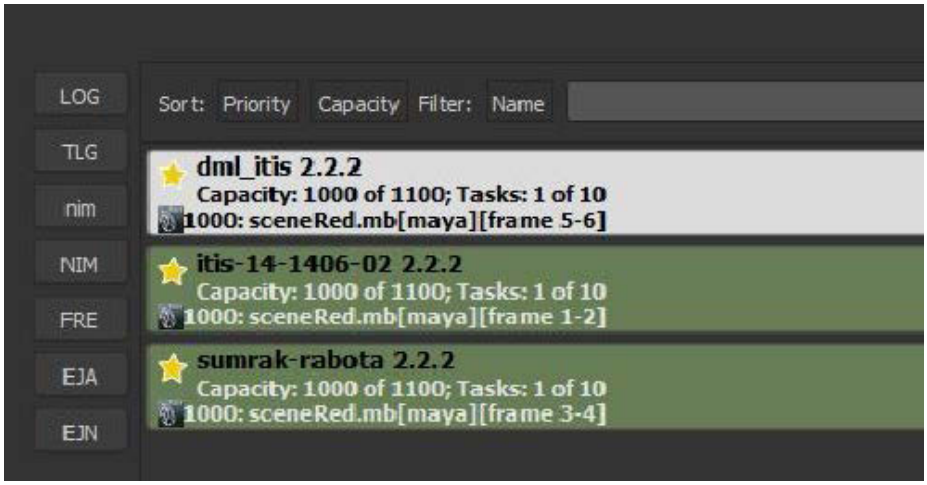

Рис. 4. Перечень задач рендеринга

НАСТРОЙКА РАБОЧЕЙ СИСТЕМЫ

Вычислительный кластер, на котором была проведена визуализация, состоит из трех нодов. На каждом из них CPU – Intel Core i7-4770 3.40GHz 8ГБ Ram и GPU – NVIDIA GeForce GTX 660 1.5 GB. На каждом были установлены программы Arnold, Redshift и программный комплекс Afanasy. После установки программ на все машины были настроены сеть и общий сетевой диск, определены сервер и клиенты. Подробный процесс настройки описан ниже.

*В корневом каталоге установленной программы запустить файл **start.cmd**. На панели задач закрепится меню программы Afanasy. Выбрать в меню **set to server**, указать **IP-адрес** сервера. Указать путь к программе Autodesk Maya, выбрав в меню **software->setup soft->maya**. Указать файл **maya.exe**.*

*Теперь можно запустить сервер. Перейти в каталог **\start\AFANASY** и запустить файл **_afserver.cmd**. Далее через меню программы Afanasy запустить **start watch**. Появится окно для мониторинга процесса рендеринга.*

*Запустить в меню **submit job**. Появится окно для настройки работы, которая будет выполняться.*

*Запустить в меню **local render**. Добавить ноду сервер для визуализации.*

Теперь сервер настроен полностью и готов выполнять работу.

*Аналогично запустить на клиентах **start.cmd**. Указать **IP-адрес** сервера и путь к программе Autodesk Maya.*

*Запустить визуализацию **local render**. Убедиться, что в командной строке указано **render registered**. Также в окне мониторинга во вкладке **renders** появились клиенты.*

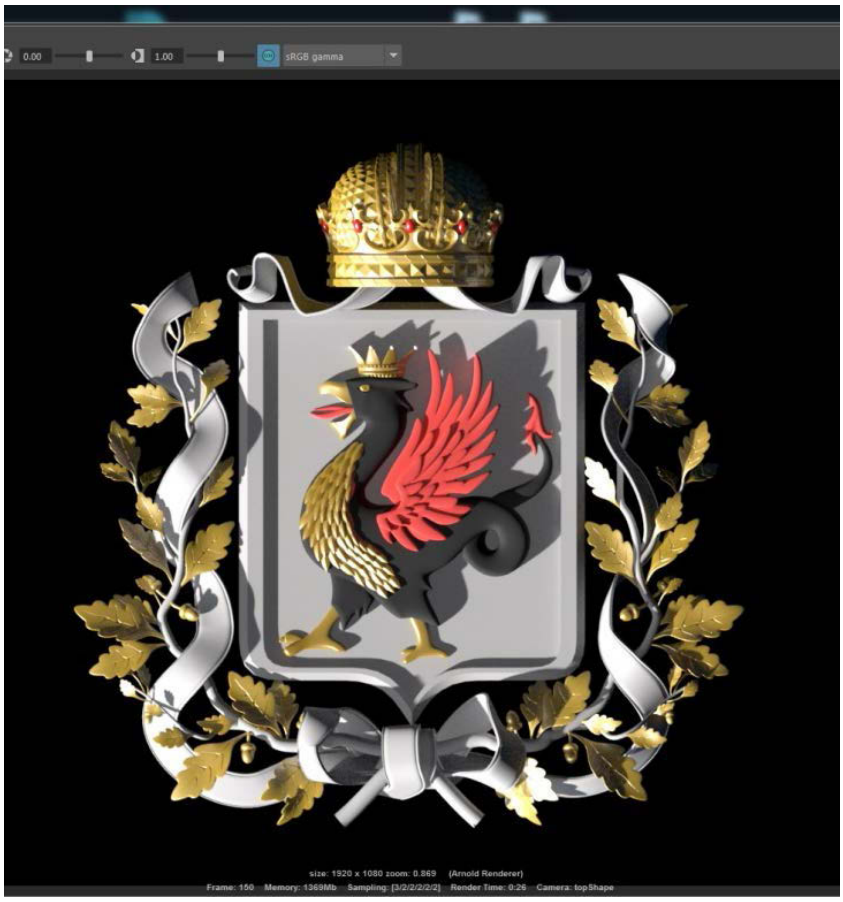

Рис. 5. Сцена с гербом после процедуры рендеринга

АНАЛИЗ ВЫЧИСЛЕНИЙ И РЕЗУЛЬТАТЫ

Оценка производительности выполнялась в несколько этапов. На каждой итерации оценивалось время, затраченное на растеризацию конечной работы. Техническая спецификация приведена в соответствующих таблицах 1 и 2.

Таблица 1. Сравнение затраченного времени на визуализацию сцены с гербом с использованием GPU и CPU

Количество полигонов	635732	635732
Разрешение	1920x1080	1920x1080
Вычислительная платформа	Intel Core i7-4770 3.40GHz 8ГБ Ram	GPU – NVIDIA GeForce GTX 660 1.5 GB 8ГБ Ram.
Визуализатор	Arnold	Redshift
Время выполнения	26 сек	9.05 сек

На первом этапе оценивался рендеринг сцены с достаточно большим количеством полигонов. Конечное изображение с гербом продемонстрировано на рис. 5. Результат рендеринга при использовании разных вычислительных решений представлен в таблице 1. Видно, что рендеринг на видеокарте опережает своего конкурента на CPU практически в 3 раза.

На втором этапе производился рендеринг сцены с содержанием большого количества флюидов на примере с торнадо. Конечное изображение продемонстрировано на рис. 6. Результат рендеринга при использовании разных вычислительных решений представлен в таблице 2.

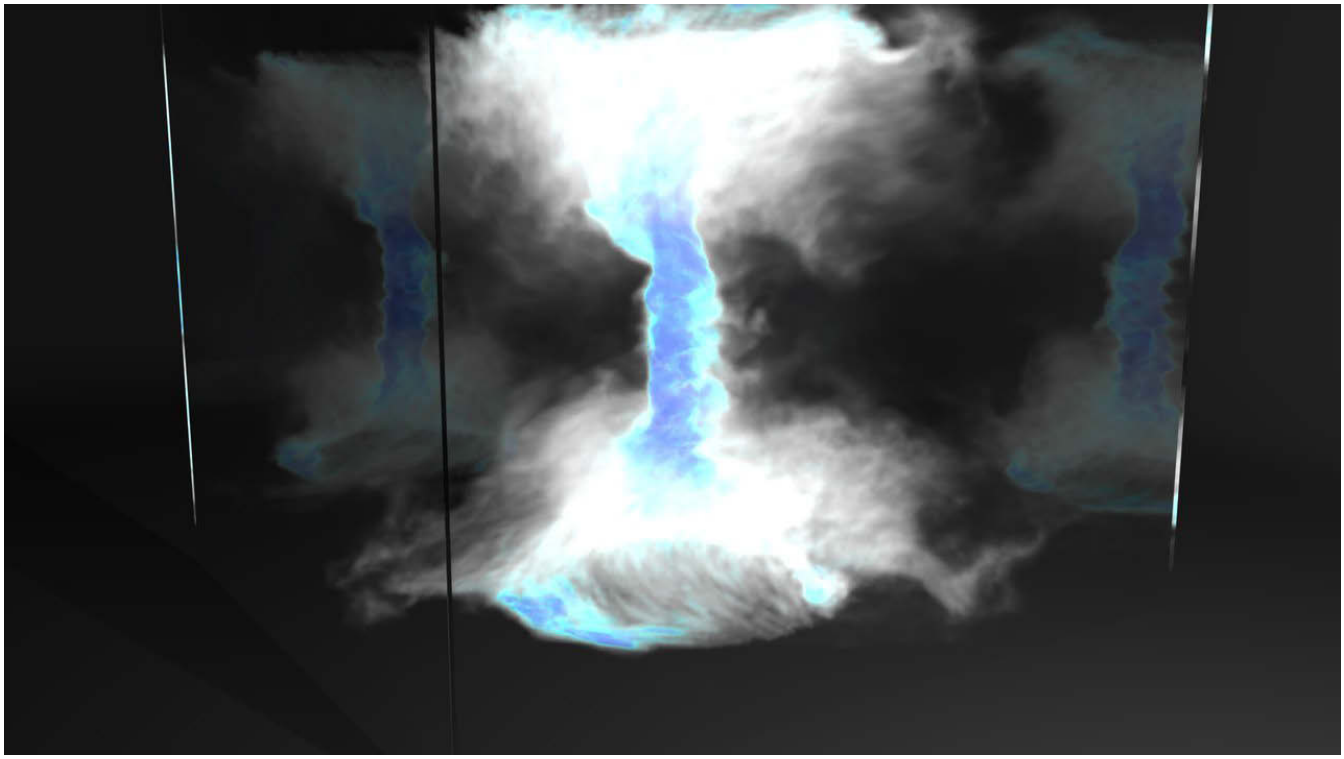

Рис. 6. Рендеринг сцены с торнадо

Таблица 2. Сравнение затраченного времени на визуализацию сцены с торнадо с использованием GPU и CPU

Количество полигонов	(Флюидные симуляции)	(Флюидные симуляции)
Разрешение	1920x1080	1920x1080
Вычислительная платформа	Intel Core i7-4770 3.40GHz 8ГБ Ram	GPU – NVIDIA GeForce GTX 660 1.5 GB 8ГБ Ram.
Визуализатор	Arnold	Redshift
Время выполнения	52 мин 56 сек	1 мин 56 сек

В очередной раз установлено, что вычисления на GPU занимают лидирующую позицию, в данном случае отрыв составляет 27 раз.

На 3-ем этапе использовали систему распределенного рендеринга для визуализации ролика длиной в 50 кадров с тем же торнадо. Результаты представлены в таблице 3. Пример работы системы мониторинга визуализации продемонстрирован на рисунках 7 и 8.

Таблица 3. Сравнение затраченного времени при распределенной визуализации сцены с торнадо с использованием GPU и CPU

Количество нодов	3	3
Разрешение	1920x1080	1920x1080
Количество фреймов	50	50
Вычислительная платформа	Intel Core i7-4770 3.40GHz 8ГБ Ram	GPU – NVIDIA GeForce GTX 660 1.5 GB 8ГБ Ram.
Визуализатор	Arnold	Redshift
Среднее время рендеринга 1 фрейма	53мин	2 мин 08 сек
Время выполнения	14 часов 42 мин 16 сек	36 мин 30 сек
Суммарное затраченное время	44 часа 12 мин 41 сек	1 час 46 мин 30 сек

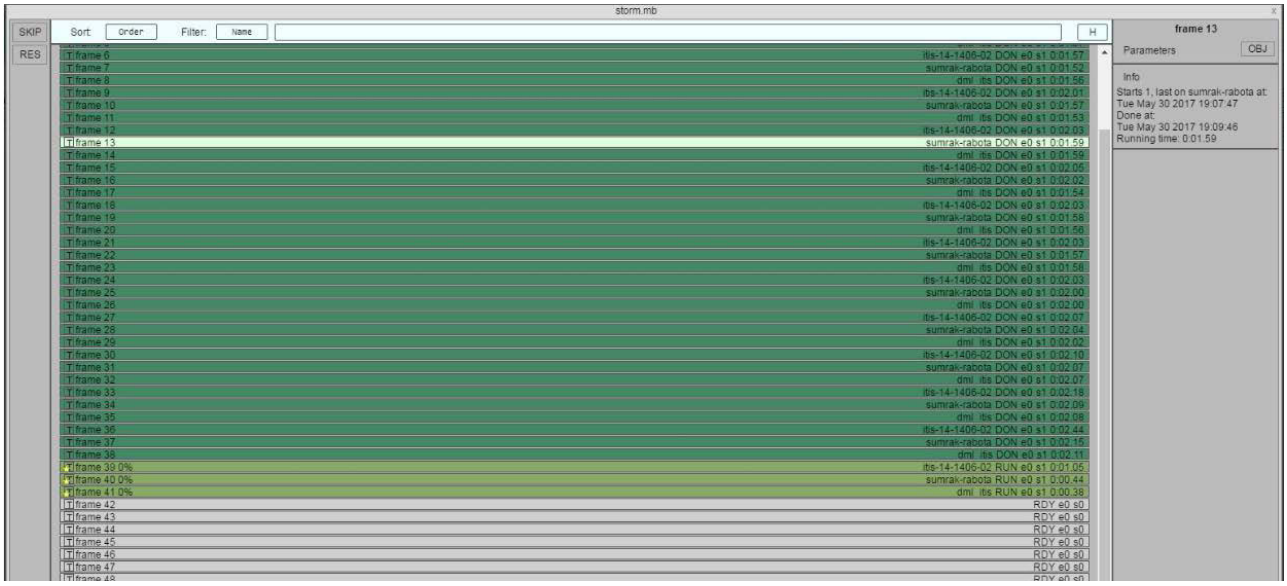

Рис. 7. Отслеживание рендеринга каждого кадра

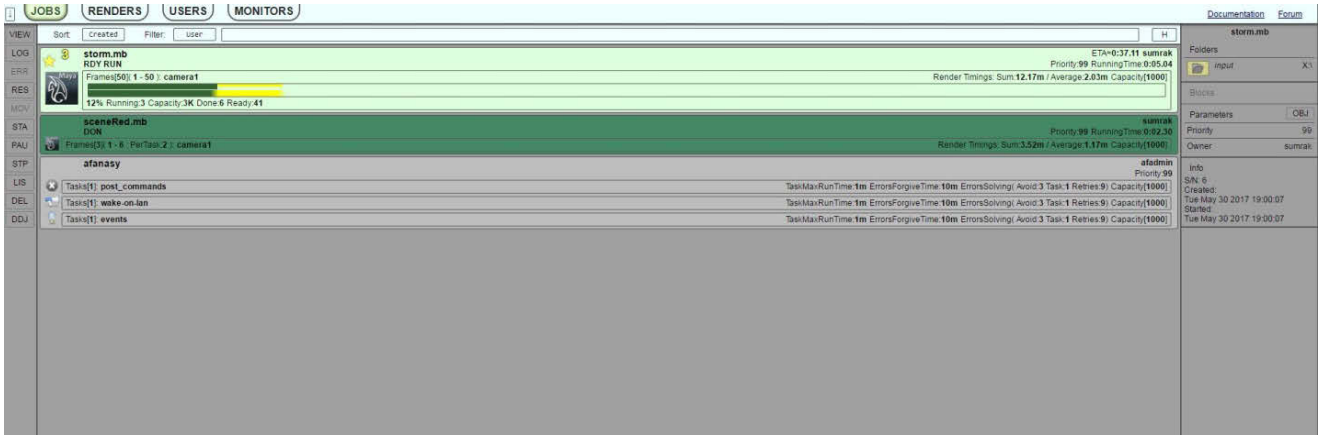

Рис. 8. Мониторинг работы

Исходя из полученных результатов, можно сделать вывод, что визуализация с использованием графического процессора в качестве вычислительной платформы позволяет сократить время на рендеринг примерно в 25 раз. Однако при визуализации ненагруженных сцен разница в скорости незначительна при двух конфигурациях. Использование технологии распределенного рендеринга позволяет сократить время, затрачиваемое на рендеринг, пропорционально количеству задействованных нод.

ЗАКЛЮЧЕНИЕ

Выполнение правильной настройки вычислительного кластера позволило значительно сократить время выполнения визуализации. При выполнении следующей визуализации не потребуется вносить какие-то кардинальные изменения в систему, нужно лишь перенести необходимые данные в общую сетевую папку.

Созданный сервис позволяет получить доступ к ресурсам рендер-фермы, эффективно и удобно взаимодействовать с ней. Он может эффективно использоваться для рендеринга анимационных роликов, высокореалистичных изображений и других проектов.

Вычислительная мощность GPU по сравнению с CPU намного выше. Наличие большего количества вычислительных ядер по сравнению с CPU и наличие собственной памяти с высокоскоростной шиной являются достоинствами графического ускорителя. Благодаря большому количеству ядер GPU выдает высокую производительность на распараллеливаемых задачах.

СПИСОК ЛИТЕРАТУРЫ

1. *Razuvalova E., Nizamutdinov A.* Virtual Reconstruction of Cultural and Historical Monuments of the Middle Volga// Procedia Computer Science. 2015. V. 75. P. 129–136.

2. *Shih M., Rizzi S., Insley J., Uram T., Vishwanath V., Hereld M., Papka M.E., Ma K.* Parallel Distributed, GPU-Accelerated// Advanced Lighting Calculations for Large-Scale Volume Visualization. 2017. P. 47.

3. ITKN.RU Информационные технологии и коммуникации: Вычислительные кластеры, кластер для расчета трехмерных моделей и сцен. http://www.itkn.ru/solutions/clusters

4. *Zheng R., Jia J., Jin H., Lv X., Yang S.* Memory-Based Data Management for Large-Scale Distributed Rendering. 2017. P. 123.

CONFIGURATION OF A COMPUTER CLUSTER FOR DISTRIBUTED RENDERING OF HIGH-REALISTIC 3D SCENES

R.R. Gazizov[1], D.I. Kostyuk[2], V.V. Kugurakova[3]

[1-3] Higher School ITIS. Kazan Federal University

[1]starkindustries14579@gmail.com, [2]xdxnxkx@gmail.com, [3]vlada.kugurakova@gmail.com

Abstract

The article discusses a method for implementing distributed computing using the GPU and modern administration tools and calculation management for the rendering farms. A comparative analysis is carried out between the traditional method of computing on the CPU and the modern approach to the GPU. Various configurations of the computing cluster are considered. The most productive way of rendering is determined.

Keywords*: visualization, distributed rendering, computational cluster, configuration.*

REFERENCES

1. *Razuvalova E., Nizamutdinov A.* Virtual Reconstruction of Cultural and Historical Monuments of the Middle Volga// Procedia Computer Science. 2015. V. 75. P. 129–136.

2. *Shih M., Rizzi S., Insley J., Uram T., Vishwanath V., Hereld M., Papka M.E., Ma K.* Parallel Distributed, GPU-Accelerated// Advanced Lighting Calculations for Large-Scale Volume Visualization. 2017. P. 47.

3. ITKN.RU Информационные технологии и коммуникации: Вычислительные кластеры, кластер для расчета трехмерных моделей и сцен. http://www.itkn.ru/solutions/clusters

4. *Zheng R., Jia J., Jin H., Lv X., Yang S.* Memory-Based Data Management for Large-Scale Distributed Rendering. 2017. P. 123.

СВЕДЕНИЯ ОБ АВТОРАХ

ГАЗИЗОВ Рим Радикович – лаборант-исследователь Высшей школы информационных технологий и информационных систем КФУ. Сфера научных интересов – распределенный рендеринг высокореалистичных 3D сцен и вопросы их моделирования.

Rim Radikovich GAZIZOV – assistant-researcher, KFU, Higher School of Information Technologies and Information Systems. Research interests include distributed rendering of highly realistic 3D scenes and questions of their modeling.

email: starkindustries14579@gmail.com

КОСТЮК Даниил Иванович – младший научный сотрудник, ассистент кафедры интеллектуальной робототехники. Сфера научных интересов – процедурная генерация сцен, моделирование, рендеринг.

KOSTYUK Daniil Ivanovich – junior researcher, assistant of the Department of Intelligent Robotics. Research interests include procedural scenes generation, modeling, rendering.

email: xdxnxkx@gmail.com

КУГУРАКОВА Влада Владимировна – старший преподаватель Высшей школы информационных технологий и информационных систем, руководитель лаборатории «Виртуальные и симуляционные технологии в биомедицине». Сфера научных интересов – реалистичность визуализации и симуляций, иммерсивность VR.

Vlada Vladimirovna KUGURAKOVA, Senior Lecturer of Higher School of Information Technology and Information Systems, Head of Laboratory “Virtual and simulation technologies in biomedicine”. Research interests include realism of visualization and simulation, immersion VR.

email: vlada.kugurakova@gmail.com

Материал поступил в редакцию 30 мая 2017 года

УДК 004.415.2+004.353+004.75

ФРЕЙМВОРК ДЛЯ ОБЛАЧНОГО ВИДЕОМОНИТОРИНГА ЧЕРЕЗ IP-КАМЕРЫ С ИНТУИТИВНО-ПОНЯТНЫМ ИНТЕРФЕЙСОМ

А.С. Гришина[1], В.В. Кугуракова[2]

[1,2] Казанский (Приволжский) федеральный университет

[1]sinerysmile@gmail.com, [2]vlada.kugurakova@gmail.com

Аннотация

Описаны основные моменты процесса создания системы, которая позволяет управлять несколькими камерами одновременно, сохраняя данные на сервере. Система имеет возможность подключать IP-камеры и камеры на мобильных устройствах, предоставлять доступ другим пользователям, а также позволяет осуществлять просмотр видео в онлайн-режиме. Выявлены и описаны «горячие точки» в архитектуре системы. Разработан отдельный Angular-модуль, использованы паттерны проектирования. Описано взаимодействие системы с пользователем. Предложены этапы дальнейшего развития облачного видеомониторинга через IP-камеры с легкодоступным управлением для конечного пользователя.

***Ключевые слова**: IP-камера, система, сервер, приложение, видеонаблюдение.*

ВВЕДЕНИЕ

Инновации не стоят на месте, изо дня в день появляются новые идеи и наработки. С развитием технологии видеозаписи становятся всё более востребованы системы, которые могут выполнять многие функции, такие, как подключение камеры, просмотр и скачивание видео, предоставление доступа к камере и многие другие. Пользователи ищут более удобное и доступное приложение.

Наиболее доступным решением в качестве источника трансляции может стать любая IP-камера. Это бюджетное устройство, которое сможет использовать любой пользователь. Если используется множество устройств, поддерживающих запись видео как источников транслируемого сигнала, а пользователей много, то необходима удобная система управления пользователями, их камерами, их файлами и доступом к ним. Такими камерами могут выступать как веб-камеры,

видеокамеры смартфонов, так и сетевые стримеры и медиа серверы. Просматривать трансляции было бы удобно как с компьютера, так и с мобильных устройств: смартфонов, планшетов. Эта система управления автономными устройствами видеонаблюдения должна решать также такие задачи, как настройка и управление камер, распределение доступа между пользователями с разными ролями, с разграничением возможностей с помощью введения групп доступа.

ДОСТАВКА ВИДЕОПОТОКОВ С ГЕОГРАФИЧЕСКИ УДАЛЕННЫХ КАМЕР

Разработка облачных систем обработки видеопотоков актуальна для сегодняшнего времени, такие системы активно разрабатываются сейчас и будут продолжать разрабатываться, обрастая новыми возможностями, так как, по мере увеличения числа пользователей, увеличиваются и требования самих пользователей к задачам по обработке данных с видео.

Самой важной проблемой доставки видео является защита информации и увеличение ресурсов для сохранения видео на сервер. В статье [1] описан один из методов для обеспечения конфиденциальности и защиты видео в облачных сервисах: представлена POC-реализация распределенного CVSS в облаке, основанная на сети.

Другая проблема – это масштабируемость, для обеспечения надежного получения видео одновременно с множества камер через ненадежные сети, в [2] для ее решения предложены эффективные методы.

Следующая проблема – обеспечение качества обслуживания – освещена в [3], где предложено использование контроллера OpenQoS на основе подхода SDN для доставки мультимедиа.

Исследования [1–3] не охватывают пользовательских проблем по обеспечению комфортного использования большого количества разнородных камер в личных целях, отводя решению этой проблемы менее значительное место. Мы же считаем проблему простого, комфортного, интуитивного, понятного управления пользовательскими трансляциями довольно важной.

НАША КОНЦЕПЦИЯ ПРОСТОГО УПРАВЛЕНИЯ КАМЕРАМИ

Чтобы можно было подключить любую камеру любому пользователю с его мобильного устройства, необходимо создать такую систему управления его ка-

мерами, которая может удовлетворить логичные потребности пользователя по работе со своими видеофайлами. В качестве основного функционала можно выделить следующие задачи:

- управление несколькими камерами одновременно («easily» добавление / удаление доступа к камере, удаленное включение / выключение);
- хранение данных на сервере;
- возможность подключения камер и на мобильных устройствах;
- предоставление доступа другим пользователям.

Система состоит из клиентской и серверной частей, тем самым реализуя технологию «клиент-сервер». Клиентская часть системы реализует пользовательский интерфейс, формирует запросы к серверу и обрабатывает ответы от него. Серверная часть системы получает запрос от клиента, выполняет вычисления, после этого формирует веб-страницу и отправляет её клиенту по сети с использованием протокола HTTP[1]. Рисунок 1 иллюстрирует работу системы.

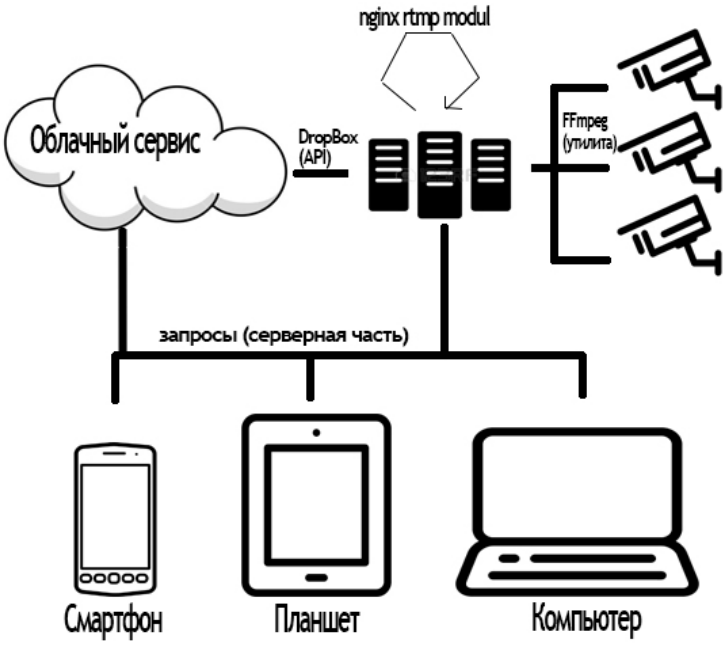

Рис. 1. Схема работы системы

При подключении новой камеры записывается URL-адрес, являющийся для каждой камеры уникальным. После подключения камеры запускается циклическая консольная команда в фоновом режиме, использующая библиотеки

[1] HTTP (англ. HyperText Transfer Protocol — «протокол передачи гипертекста») — протокол прикладного уровня передачи данных (изначально — в виде гипертекстовых документов в формате «HTML», в настоящий момент используется для передачи произвольных данных)

FFMPEG (набор свободных библиотек с открытым исходным кодом [4]), которая записывает на сервер видео указанными промежутками времени. Записанное на сервер видео направляется в облако [2], например, в Dropbox[2]. В облаке создается папка с идентификатором пользователя, и сохраняется видео, которое закрепилось за новой камерой. После того, как видео направлено в облако, на сервере консольная команда удаляет этот фрагмент видео. Для того чтобы просмотреть видео в онлайн режиме, был использован модуль Nginx rtmp [5] и js-библиотека JWplayer. Видео можно просмотреть как из приложения, так из веб-версии. Однако, чтобы скачать видео, стоит использовать веб-версию, потому что в большинстве случаев смартфон не сможет его воспроизвести.

Для организации данного сервиса главной технологией служат докеры[3], они используются для запуска таких процессов, как Nginx и его rtmp-модуль. Новый докер запускается, когда пользователь добавляет новую камеру. Nginx запускается с настройками, в которых переданы hash-данные[4] пользователя, который записан в базу данных, сгенерированных при регистрации. Докер запускается посредством командного интерфейса Laraver. Также в докер-контейнере запускается процесс записи видео с помощью утилиты FFmpeg с последующей записью в облако DropBox.

Таким образом, каждый процесс обработки камеры является изолированным друг от друга и никак не влияет на параллельные процессы. При добавлении камеры в базу данных PostgreSQL также записывается имя докер-контейнера, который создан под данную камеру, а при удалении останавливается докер-контейнер с заданным именем.

Категории, являющиеся «горячими точками»[5] в архитектуре приложения (рис. 2):

- аутентификация и авторизация (Authentication and Authorization);
- композиция (Composition);

[2] Dropbox — файловый хостинг компании Dropbox Inc., включающий персональное облачное хранилище, синхронизацию файлов и программу-клиент

[3] Докер — это открытая платформа для разработки, доставки и эксплуатации приложений

[4] Hash — результат обработки неких данных хеш-функцией (функция, реализующая алгоритм и выполняющая преобразование)

[5] «Горячие точки» (хот-спот, или англ. hotspot) — участок кода в программе, на который приходится большая часть исполняемых инструкций процессора или на исполнение которого процессор затрачивает очень много времени

- управление конфигурацией (Configuration Management);
- связанность и сцепление;
- доступ к данным (Data Access);
- взаимодействие с пользователем (User Experience).

Горячие точки соответствуют разным сквозным функционалам, которые используются при создании приложений, и, соответственно, разным наборам паттернов и практик. Таблица 1 перечисляет ключевые вопросы для каждой горячей точки.

Аутентификация. Для аутентификации разработан отдельный Angular-модуль, работающий со стандартным функционалом Laravel [6]. Авторизовать запросы можно с помощью PHP-сессий, которые передаются из запроса в запрос каждого пользователя. Между запросами передаются данные пользователя, которые изначально при авторизации кладутся в сессию. При обновлении данных синхронизируются данные сессии с уже обновленными данными.

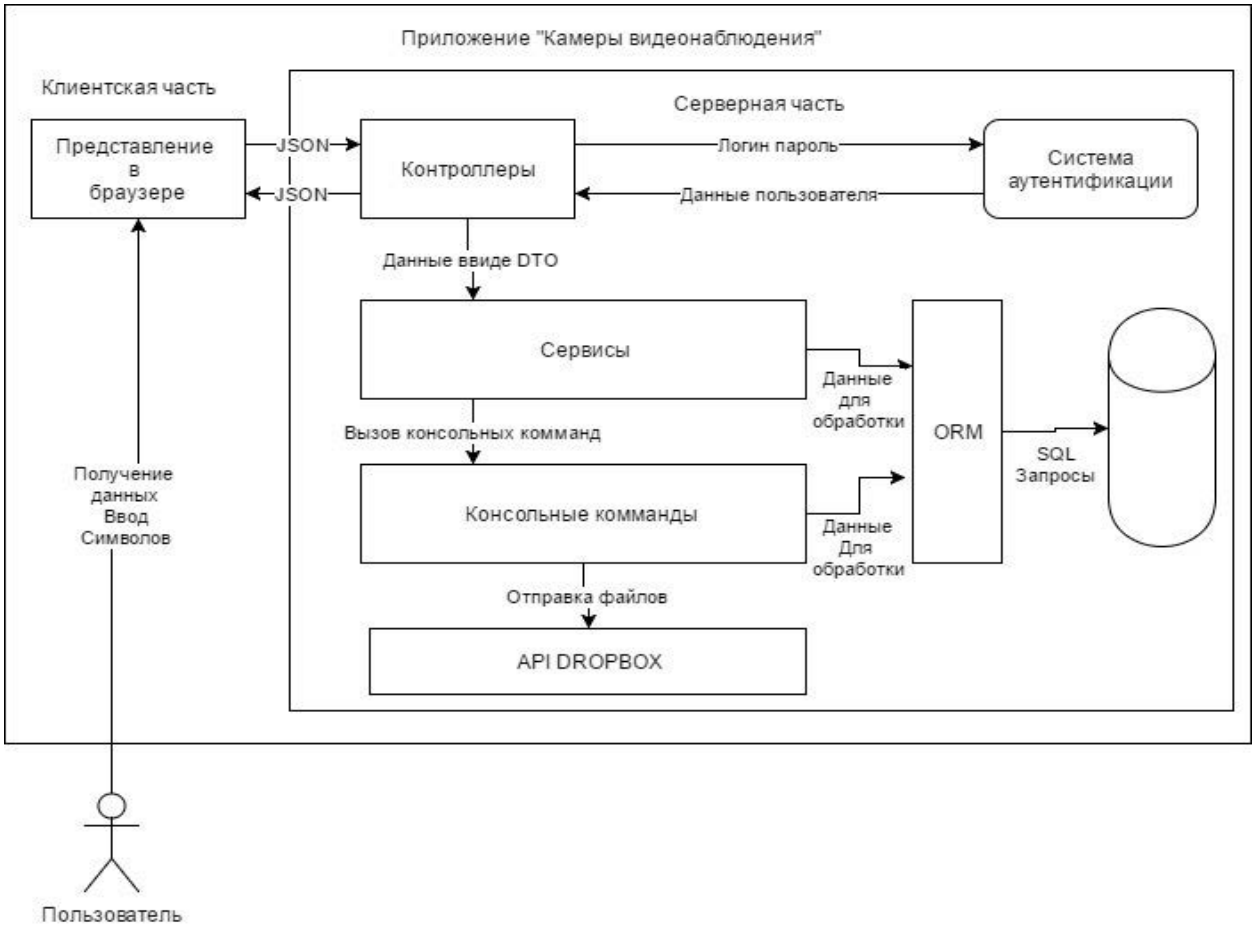

Рис. 2. Архитектура системы

Композиция. Для слабой связанности компонентов приложения используются три вида паттерна проектирования: MVC; Service; Repository. Паттерн проектирования предназначен для того, чтобы отделить представление от логики приложения.

Для разработки с использованием архитектурной модели MVC (англ. Model View Controller – модель-представление-контроллер) используется Laravel framework (с версей php 7.0) [7].

Service – предназначен для того, чтобы отделить входные данные от их обработки. Для реализации данного паттерна использовалась технология сервис контейнеров фреймворка Laravel для более гибкой реализации логики. Данная технология позволяет внедрять реализацию определенных ранее интерфейсов, которые представляют собой набор функций приложения.

Паттерн Repository предназначен для отделения работы с базой данных от логики приложения. Для его реализации использован Eloquent – инструмент, встроенный в Laravel. Также используется Artisan [8] для разработки команд для cmd.

Таблица 1. Ключевые вопросы «горячих точек»

Категория	**Ключевые вопросы**
Аутентификация	- Как хранить идентификационные данные зарегистрированных пользователей? - Как аутентифицировать запросы? - Как передавать идентификационные данные пользователей между слоями приложения?
Композиция	- Как спроектировать структуру приложения? - Как спроектировать слабую связанность между модулями? - Как работать с зависимостями в слабосвязанном режиме?
Управление конфигурацией	- Как определить какая информация подлежит конфигурированию? - Как определить где и каким образом сохранять конфигурационную информацию? - Как работать с секретной информацией?

	- Как работать с конфигурационной информацией в кластере?
Связанность и сцепление	- Как разделить функциональности? - Как структурировать приложение? - Как выбрать подходящее разбиение на слои? - Как задать границы между частями системы?
Доступ к данным	- Как управлять соединениями БД?
Взаимодействие с пользователем	- Как повысить эффективность выполнения задач? - Как улучшить отзывчивость интерфейса? - Как улучшить возможности приложения для пользователя?

Связанность и сцепление. Структура приложения обоснована структурой фреймворка Laravel. Корневой каталог свежеустановленного Laravel содержит ряд папок:

- папка app, содержит код ядра приложения (далее будет рассмотрена подробнее);
- папка bootstrap содержит несколько файлов, которые загружают фреймворк и настраивают автозагрузку, папка cache содержит несколько файлов для оптимизации процесса автозагрузки, сгенерированных фреймворком;
- папка config содержит все конфигурационные файлы приложений;
- папка database содержит миграции и классы для наполнения начальными данными БД; при необходимости эту папку можно использовать для хранения базы данных SQLite;
- папка public содержит фронт-контроллер и ресурсы (изображения, JavaScript, CSS и т. д.);
- папка resources содержит представления, сырые ресурсы (LESS, SASS, CoffeeScript) и «языковые» файлы;
- папка storage содержит скомпилированные Blade-шаблоны, файл-сессии, кэши файлов и другие файлы, создаваемые фреймворком; эта папка делится на подпапки app, framework и logs; в папке app можно хранить любые файлы, используемые вашим приложением; в папке framework хранятся создаваемые

фреймворком файлы и кэш, а в папке logs находятся файлы журналов приложения;

- папка tests содержит автотесты; изначально там уже есть пример PHPUnit;.
- папка vendor содержит Composer-зависимости.

Доступ к данным. За основу был взят PostgreSQL, свободная объектно-реляционная система управления базами данных [9]. Для общения приложения и базы данных используется система объектно-реляционного отображения Eloquent – красивая и простая реализация шаблона ActiveRecord в Laravel для работы с базами данных. Каждая таблица имеет соответствующий класс-модель, который используется для работы с этой таблицей. Модели позволяют запрашивать данные из таблиц, а также вставлять в них новые записи.

Взаимодействие с пользователем.

1. Форма для регистрации. Данная страница предназначена для новых пользователей, прежде всего необходимо зарегистрироваться, чтобы осуществить вход в аккаунт.

2. Форма входа предназначена для пользователей, которые зарегистрированы в системе.

3. Профиль пользователя. На данной странице пользователь может обновить информацию о себе, такую, как Имя, Фамилия, а также почтовый адрес.

4. Страница добавления камер, а также просмотр текущего статуса камеры, Stream-потока камеры.

5. Страница с видеозаписями пользователя. На этой странице пользователь может посмотреть видеозаписи камер, список камер, количество свободного пространства, а также скачать видеозаписи.

6. Меню приложения. Пользователь может выбрать свои дальнейшие действия, а также изменить информацию.

РЕЗУЛЬТАТЫ

Было проведено нагрузочное тестирование, на основании результатов которого можно сделать вывод, что система способна подключать одновременно и получать данные транслирования с достаточно большого количества разных устройств видеонаблюдения. В зависимости от разных конфигураций серверов, на которых расположена серверная часть, это может быть от 100 до 1000 камер одновременно. При увеличении количества пользователей и используемых ими

камер необходимо переходить на другую серверную архитектуру, предлагаемую в [1, 2].

Использовались мобильные устройства и различные камеры – уличные камеры с открытым доступом, камеры с мобильных устройств, а также IP-камеры.

Разработанная система предлагает как простоту шаринга видеопотока (совместного использования карты условного доступа) для заинтересованных пользователей без увеличения нагрузки на сеть, так и использование веб-камер абсолютно разного типа.

Устройства, которые были использованы при тестировании системы:

- Мобильные устройства: Xiaomi Redmi 3S; Samsung Galaxy S5; Sony Xperia Z3; LG G5; HTC One M8; Asus ZenFone 3;
- Камеры видеонаблюдения: CMD IP1080-WB3,6IR V2; VStarcam C7815WIP; VStarcam T7892WIP; IPC-HFW1300S-0360B;
- Онлайн камеры.

На рисунке 3 показаны страница добавления новой камеры и страница подключенных устройств камер видеонаблюдения.

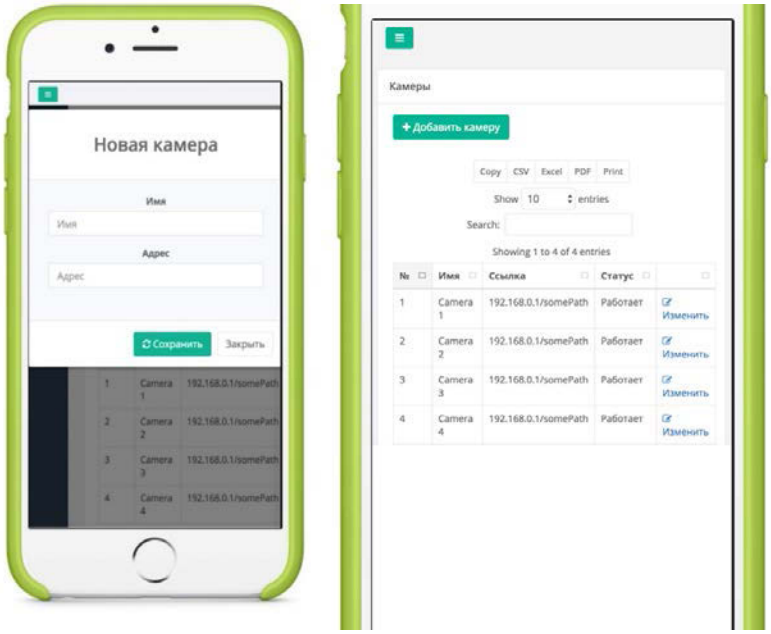

Рис. 3. Добавление новой камеры. Информация о камерах

В интерфейсе для мобильных приложений пользователю доступны: просмотр видеозаписей, доступных камер, информации о них, информация о свободном дисковом пространстве в его персональном облаке, работа с его спис-

ками групп: «друзья», «соседи», «работа» и т. д. Например, на рисунке 4 приведен список видео, записанных с данных камер, которые можно как просмотреть, так и скачать для просмотра. Пользователю доступны все записи с камер, а также свободное дисковое пространство.

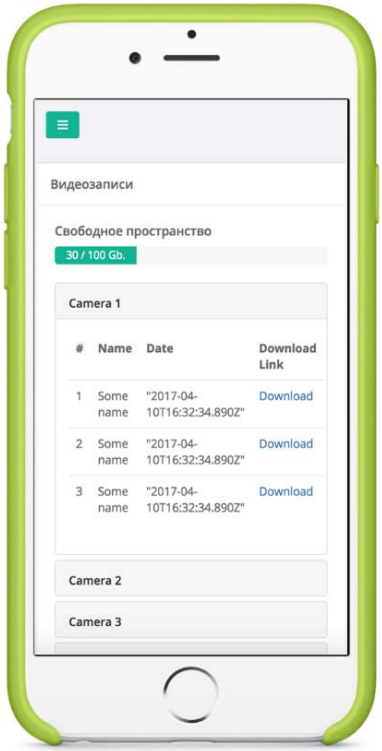

Рис. 4. Страница с видеозаписями пользователя

Проведенные работы положили начало минимально жизнеспособной версии, на основе которой в дальнейшем можно реализовать, например:

- на уровне облачных сервисов такие функции, как:

1. создание службы настройки и обработки видео [10];

2. создание службы распознавания человеческих лиц [11], животных [12], номерных знаков [13], погодных явлений, стихийных бедствий [14];

3. создание функции резервного копирования [10];

4. протоколирование событий по расписанию (праздники, ночное время);

- на клиентском уровне [2]:

1. создание приложения монитора (использовать в качестве мониторинга и инструментов для обнаружения объектов и событий в реальном времени);

2. приложение тревоги (обнаруживается подозрительный человек в наблюдаемой области, о чем отправляется тревожное сообщение);

3. приложение отчета (может быть использовано для составления сводной информации о видеопотоках, камерах или использовании системы).

ЗАКЛЮЧЕНИЕ

Технологии, которые были выбраны в начале разработки системы, отвечающие за получение данных в онлайн режиме, были изменены на другие, более удобные, примером является фреймворк Laravel, который имеет легковесную библиотеку. К такому решение пришли в результате тестирования, которое показало, что эти процессы влияют на отклик приложения.

Разработанная система содержит открытый API[6], на основе которого можно расширять функционал для выполнения новых требований пользователей. Примерами такого функционала могут быть: создание базы биометрических данных на основании обработки изображений лиц; создание групп камер, которые связаны между собой местом наблюдения или другим тегированием; обучение системы распознаванию движения и эмоций; оповещения об энергии камер; интеграция с мессенджерами типа telegram и др.

СПИСОК ЛИТЕРАТУРЫ

1. *Choi K.I., Lee J.H., Lee B.C.* A Distributed Cloud Based Video Storage System with Privacy Protection // Int. Conf. on Advanced Communication Technology – ICACT 2017. P. 830–835.

2. *Sandar N.M., Chaisiri S., Yongchareon S., Liesaputra V.* Cloud-based Video Monitoring Framework: An Approach Based on Software-defined Networking for Addressing Scalability Problems // Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2015. V. 9051.

3. *Egilmez H.E., Dane S.T., Bagci K.T., Tekalp A.M., Koc Univ*. In Signal & Information Processing Association Annual Summit and Conference // APSIPA ASC, 2012 Asia-Pacific – Istanbul, Turkey, 2012. P. 1–8.

4. FFMPEG. https://ffmpeg.org/.

[6] API (программный интерфейс приложения, интерфейс прикладного программирования) (англ. *Application programming interface*, *API* [эй-пи-ай]) — набор готовых классов, процедур, функций, структур и констант, предоставляемых приложением (библиотекой, сервисом) или операционной системой для использования во внешних программных продуктах. Используется программистами при написании всевозможных приложений

5. Nginx-rtmp-module. https://github.com/arut/nginx-rtmp-module.
6. AngularJS. https://angularjs.org/.
7. Laravel. https://github.com/laravel/framework.
8. Artisan. http://laravel.su/docs/5.0/artisan.
9. PostgreSQL. https://www.postgresql.org/.
10. *Dasu A., Panchanathan S.* A Survey of Media Processing Approaches. Circuits and Systems for Video Technology // IEEE Transactions on. 2002. V. 12, No 8. P. 633–645.
11. *Connolly J. F., Granger E., Sabourin R.* An Adaptive Classification System for Video-based Face Recognition // Information Sciences. 2012. V. 192. P. 50–70.
12. *Burghardt T., Cali´c J*. Analysing Animal Behaviour in Wildlife Videos Using ´ Face Detection and Tracking // IEE Proceedings-Vision, Image and Signal Processing. 2006. V. 153, No 3. P. 305–312.
13. *Du S., Ibrahim M., Shehata M., Badawy W.* Automatic License Plate Recognition (ALPR): A State-of-the-art review // Circuits and Systems for Video Technology. IEEE Transactions on. 2013. V. 23, No 2. P. 311–325.
14. *Lai C.L., Yang J.C., Chen Y.H*. A Real Time Video Processing Based Surveillance System for Early Fire and Flood Detection // Instrumentation and Measurement Technology Conference Proceedings. 2007. IMTC 2007. IEEE. P. 1–6.

FRAMEWORK FOR CLOUD-VIDEO MONITORING VIA IP-CAMERAS WITH EASILY ACCESSIBLE CONTROL FOR USERS

A.S. Grishina[1], V.V. Kugurakova[2].

[1,2] Higher School ITIS. Kazan Federal University
[1] sinerysmile@gmail.com, [2] vlada.kugurakova@gmail.com

Abstract

The article describes the main points of the process of creating a system that allows you to manage several cameras simultaneously while saving data on the server. The system has the ability to connect IP cameras and cameras on mobile devices,

provide access to other users, and also allows you to watch video online. Hotspots in the architecture of the system have been identified and described. A separate Angular module was developed and design patterns were used. The interaction of the system with the user is described. The stages of further development of cloud video monitoring through ip-cameras with easily accessible control for the end-user are offered.

Keywords: *IP-camera, system, server, application, video surveillance.*

REFERENCES

1. *Choi K.I., Lee J.H., Lee B.C.* A Distributed Cloud Based Video Storage System with Privacy Protection // Int. Conf. on Advanced Communication Technology – ICACT 2017. P. 830–835.

2. *Sandar N.M., Chaisiri S., Yongchareon S., Liesaputra V.* Cloud-based Video Monitoring Framework: An Approach Based on Software-defined Networking for Addressing Scalability Problems // Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2015. V. 9051.

3. *Egilmez H.E., Dane S.T., Bagci K.T., Tekalp A.M., Koc Univ*. In Signal & Information Processing Association Annual Summit and Conference // APSIPA ASC, 2012 Asia-Pacific – Istanbul, Turkey, 2012. P. 1–8.

4. FFMPEG. https://ffmpeg.org/.

5. Nginx-rtmp-module. https://github.com/arut/nginx-rtmp-module.

6. AngularJS. https://angularjs.org/.

7. Laravel. https://github.com/laravel/framework.

8. Artisan. http://laravel.su/docs/5.0/artisan.

9. PostgreSQL. https://www.postgresql.org/.

10. *Dasu A., Panchanathan S.* A Survey of Media Processing Approaches. Circuits and Systems for Video Technology // IEEE Transactions on. 2002. V. 12, No 8. P. 633–645.

11. *Connolly J. F., Granger E., Sabourin R.* An Adaptive Classification System for Video-based Face Recognition // Information Sciences. 2012. V. 192. P. 50–70.

12. *Burghardt T., Cali´c J*. Analysing Animal Behaviour in Wildlife Videos Using ´ Face Detection and Tracking // IEE Proceedings-Vision, Image and Signal Processing. 2006. V. 153, No 3. P. 305–312.

13. *Du S., Ibrahim M., Shehata M., Badawy W.* Automatic License Plate Recognition (ALPR): A State-of-the-art review // Circuits and Systems for Video Technology. IEEE Transactions on. 2013. V. 23, No 2. P. 311–325.

14. *Lai C.L., Yang J.C., Chen Y.H*. A Real Time Video Processing Based Surveillance System for Early Fire and Flood Detection // Instrumentation and Measurement Technology Conference Proceedings. 2007. IMTC 2007. IEEE. P. 1–6.

СВЕДЕНИЯ ОБ АВТОРАХ

ГРИШИНА Анастасия Сергеевна – бакалавр Высшей школы ИТИС Казанского (Приволжского) Федерального университета. Сфера научных интересов – обработка видеопотока, облачный видеомониторинг, распознавание объектов и другой информации в видеопоследовательностях.

Anastasia Sergeevna GRISHINA, has bachelor's degree of the Higher School of ITIS Kazan (Privolzhsky) Federal University. Research interests include video stream processing, cloud video monitoring, recognition of objects and other information in video sequences.

email: sinerysmile@gmail.com.

КУГУРАКОВА Влада Владимировна – старший преподаватель Высшей школы информационных технологий и информационных систем, руководитель лаборатории «Виртуальные и симуляционные технологии в биомедицине». Сфера научных интересов – реалистичность визуализации и симуляций, иммерсивность VR.

Vlada Vladimirovna KUGURAKOVA, Senior Lecturer of Higher School of Information Technology and Information Systems, Head of Laboratory "Virtual and simulation technologies in biomedicine". Research interests include realism of visualization and simulation, immersion VR.

email: vlada.kugurakova@gmail.com.

Материал поступил в редакцию 2 июня 2017 года

УДК 59.009+308

КОНТРОЛЛЕР РЕАЛИСТИЧНОГО ПОВЕДЕНИЯ СТАЙ/СТАД ЖИВОТНЫХ

В.В. Кугуракова[1], А.М. Степанов[2]

[1-2]Казанский (Приволжский) федеральный университет

[1]vlada.kugurakova@gmail.com, [2]stepanovaleksandrm@gmail.com

Аннотация

Работа посвящена рассмотрению процесса моделирования реалистичного контроллера поведения групп объектов. Проведено исследование основных приемов и принципов, используемых при создании реалистичного контроллера поведения автономных агентов, объединенных в связанные группы. На основе этих данных создан контроллер поведения.

Исследована эффективность поведения групп автономных агентов, рассмотрены возможности использования системы локальных скалярных полей с целью построения максимально точной математической модели, проведён анализ возможности создания иерархической системы мультиагентных подгрупп в рамках группы, проведены эксперименты для оценки корректности разработанного контролера.

Ключевые слова: *контроллер, группа, модель поведения.*

1. АНАЛИЗ ПОВЕДЕНИЯ ЖИВОТНЫХ В РЕАЛЬНОМ МИРЕ

Как возникают скопления животных

При скоплении определенных животных часто образуется стадо. Основополагающим фактором в данном случае являются схожие потребности членов стада. Так, например, если на территории в изобилии присутствует пища, то вероятность объединения животных в стадо возрастает.

В качестве примера рассмотрим птичий «базар». Обычно он образуется в местах активной циркуляции морских вод, в результате чего в изобилии присутствуют как кислород, так и минеральные соли, поднимаемые течениями. Благодаря этому развивается планктон, которым питается рыба. А она является пищей для птиц.

Аналогичная ситуация и с животными. При хорошем урожае шишек в кедровых лесах животные более активно собираются в местах непосредственного расположения кедра.

В качестве другого фактора объединения животных в стада и стаи может выступать период размножения. Происходит встреча самцов и самок, животные активно разыскивают друг друга, устраивают специфические брачные игры [1].

Представители многих видов животных активно образуют группы. Индивидуальные особенности животных, а именно, окраска, особенности пахучих желез и свойства органов чувств, помогают животным общаться между собой и проще находить друг друга. Данные свойства указывают на эволюционное закрепление всего того, что облегчает образование групп животных.

Размеры скоплений, стай и стад у людей и животных

При ориентировочном анализе количества беспозвоночный особей установлено, что оно может достигать нескольких миллиардов (например, стая мигрирующей саранчи). Скопления птиц на упомянутых ранее птичьих «базарах» исчисляются сотнями тысяч особей. Стада копытных животных, при миграции с одного места на другое, могут достигать численности в несколько десятков тысяч животных [2–4].

Скопления людей в одном месте также могут достигать огромных цифр. Обычно это характерно для таких массовых мероприятий, как парады, паломничества, крупные спортивные мероприятия и праздники [5]. Например, на крупном музыкальном фестивале «Парад Любви» в Германии собралось более миллиона человек.

Скопление людей более 6 человек может перерасти из простого собрания в толпу с характерными целями и признаками. При достижении скоплением людей «критической массы» между ними возникает взаимная связь, что в конечном итоге может привести к активным действиям со стороны этого скопления (в том числе буйным, мятежным, преследующим цели, о которых собиравшиеся не помышляли) [6].

Толпа – это бесструктурное скопление людей, которые не обладают какой-либо общей целью, но связаны между собой определенным сходством эмоционального состояния и общим объектом внимания. Сходство между эмоциональными состояниями людей возникает в процессе превращения скопления людей

в толпу. Введение этого признака суживает разнообразие явлений, которыми можно характеризовать толпу.

Координация движения в стадах и стаях

В отличие от групп мигрирующих животных, в стадах и стаях наблюдается координация поведения особей (например, общее направление движения). При наблюдении небольшой стаи рыб (15–20 особей) становится очевидным, что в рамках самой стаи рыбы постоянно изменяют свое положение. Так, например, голодные особи немного выбиваются вперед, хватают корм и затем постепенно отплывают назад [7, 8].

Стадам парнокопытных животных присущи более простые закономерности, связанные с расположением внутри стада: одна часть животных постоянно стремится быть впереди, другие предпочитают быть боковыми, третьи – центральными. Например, в стаде коров 27% животных предпочитают быть впереди, 10% – сзади, 33% – в гуще стада (центральная часть) [9]. Кроме того, некоторые животные предпочитают находиться на правой стороне, другие, аналогично, на левой (рис. 1). Однако, несмотря на это, значительное число коров встречается то тут, то там, не имея особых предпочтений.

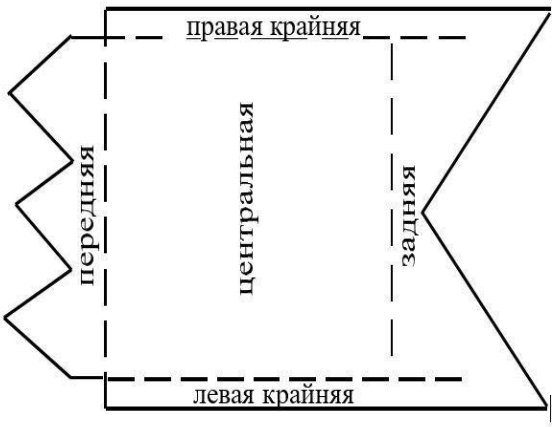

Рис. 1. Предпочтение места в стаде (И.А. Жесткова)

Процесс превращения группы в стадо или стаю

Одной из основных целей пастуха является поддержание животных в форме стада. Так, например, если он видит, что животные разбрелись, то первое, что предпримет пастух, – это сгонит их в плотную группу. Это делается для того, чтобы обрести контроль и власть над животными.

Животные, которые разбрелись по пастбищу, не будут подпускать к себе пастуха на близкое расстояние. Собрав животных воедино, пастух может к ним приблизиться или, например, заставить их двигаться в требуемом направлении. Животные, находясь в стаде, чувствуют безопасность и теряют самостоятельность. Стадо ведет себя как единый организм.

Для возникновения стада в обычных условиях требуется несколько минут. Например, олени, пасущиеся на склонах гор, в обычной ситуации разобщены. Стоит нам вспугнуть одного из них – реакции от других не последует. Однако если мы решим повторить этот эксперимент на равнине, где животные видят друг друга, последует ответная реакция от многих особей. В считанные минуты будет сформировано стадо, и животные будут перемешаться в его рамках, поскольку, находясь в стаде, они чувствуют себя более защищенными. В данном случае некогда свободные в выборе особи ожидают решения вожака стаи и следуют за ним.

Стадо (стая) остается единым до тех пор, пока сохраняются условия, при которых животные образовали группу и сблизились ближе, чем на индивидуальную дистанцию, – испуг, например, от близости человека или при нападении хищников, при прохождении через узкий проход или пребывании в опасной ситуации вроде переправы через реку. Как только животные успокаиваются, начинают кормиться или отдыхать, так стадо рассыпается, поведение обособленных особей вновь становится индивидуальным.

Цели и задачи моделирования толпы

Одной из основных целей моделирования толпы является описание поведения множества агентов в определенных условиях. Наиболее часто в качестве агента выступает именно человек. Аналогичным образом моделируется поведение косяков рыб, стай птиц и стад животных.

В наши дни моделирование толпы является развивающейся областью науки. Оно активно применяется в компьютерной графике, кинематографе, архитектурном и ландшафтном планировании. Необходимость расчета пассажирского и транспортного потоков породила особый класс геоинформационных систем: симуляторов толпы, предоставляющих возможность измерения, оптимизации и визуализации подобных потоков.

В крупных городах наблюдается тенденция увеличения населения [10]. Однако многие городские объекты, магистрали и места массового пользования были спроектированы без учета значительного разрастания. Это приводит к образованию пробок и других транспортных проблем, мешающих корректному функционированию. Кроме того, в случае чрезвычайной ситуации данная проблема может привести к непоправимым последствиям [11].

Благодаря своевременному использованию реалистичной модели поведения толпы можно внести изменения в существующую систему. Благодаря моделированию можно получить следующие данные:

- нахождение областей с высокой плотностью движения;
- подбор оптимального проекта с целью улучшения пропускной способности потока;
- моделирование возможных сценариев поведения толпы при проведении общественных мероприятий (в том числе, в случае ЧП).

2. СОЗДАНИЕ АЛГОРИТМА ПОВЕДЕНИЯ

Поиск объекта

Поиск объекта является базовым элементом алгоритма поведения. В качестве целевой координаты в данном случае выступает определенная точка (либо область) в пространстве. При этом направление движения агента соответствует кратчайшему направлению к целевой точке [12].

При достижении целевой точки (либо области) агент может вызвать какое-либо действие. В этом случае он перейдет к следующей части алгоритма.

Если же никакое действие не будет вызвано, агент сохранит целевое направление и пройдет сквозь цель. После следующей корректировки направления агент развернется и вновь пройдет сквозь цель (рис. 2).

Данный метод алгоритма может быть использован для поиска агентом чего-либо. В рамках стада данный метод должен применяться вкупе с другими методами во избежание разрозненного поведения [13]. Однако в том случае, если животное окончательно отбилось от стада, оно может применять этот метод алгоритма в чистом виде (так как влияние со стороны стада отсутствует).

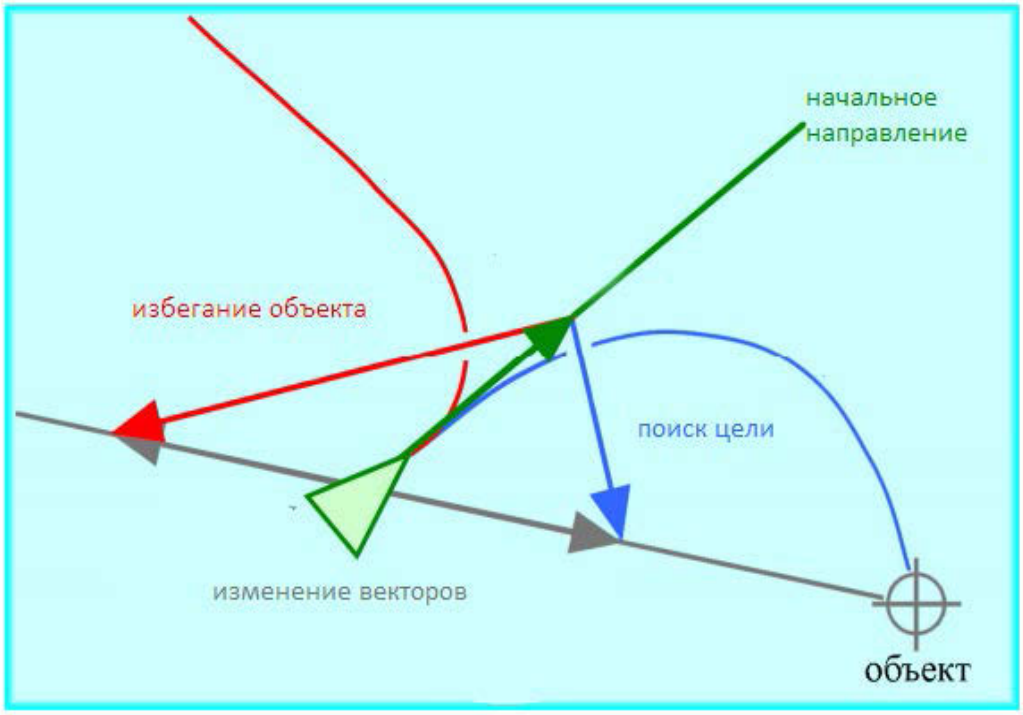

Рис. 2. Поиск и избегание объекта

Избегание объекта

Избегание объекта является элементом алгоритма, противоположным поиску цели. В данном случае агент выбирает направление, противоположное объекту, который он избегает. Действие в данном случае можно вызывать после достаточного отдаления от объекта, который агент старается избежать, либо, например, по прошествии определенного времени, в течение которого агент осуществлял движение в противоположную сторону (время бегства).

Аналогично поиску данный метод должен быть комбинирован с другими методами группового поведения [14]. Например, несколько членов стада приблизились к какому-либо объекту, и он начал издавать отпугивающий звук. В данном случае находящиеся рядом агенты испугаются и начнут движение в противоположную сторону. Далее, при приближении к стаду их движение будет замечено остальными участниками. В зависимости от количества бегущих произойдет лавинообразный эффект: либо бегство заразит все стадо, и оно начнет движение в сторону (если бегущих много); либо бегство постепенно угаснет, частично сдвинув агентов вблизи избегаемого объекта (бегущих мало).

Преследование движущегося объекта

Данный элемент алгоритма имеет черты, похожие на метод поиска объекта. Отличием является то, что в данном случае целевой объект движется. Для более реалистичного поведения мы должны прогнозировать дальнейшее движение нашей цели.

При прогнозировании будем предполагать, что объект сохранит прямолинейное движение. На практике же прогноз не всегда является верным, поскольку движущийся объект может совершать самые разнообразные маневры. Исходя из этого, прогноз должен регулярно проходить корректировки [15].

Поскольку объект находится в стаде, влияние на него будет оказывать как целевой объект, так и окружающие его агенты (стадо). Агенты будут сохранять комфортные для преследования боковой интервал и дистанцию. Основное влияние на вектор движения будет оказывать преследуемый объект, второстепенное влияние окажут окружающие агенты.

Избегание движущегося объекта

Метод алгоритма аналогичен преследованию объекта, за исключением того, что вместо движения за объектом используется бегство от него. Таким же образом производятся прогноз движения и корректировка собственного курса агента [15] (рис. 3). Более того, в данном случае вступают в дело стадные чувства. При возникновении опасности агенты сокращают дистанцию между собой, образуя более плотное скопление. Поскольку агенты находятся на небольшом расстоянии, если объект опасности окажется достаточно близко, стадо начнет синхронное бегство от него (в зависимости от размера стада это могут быть как его часть, так и все оно).

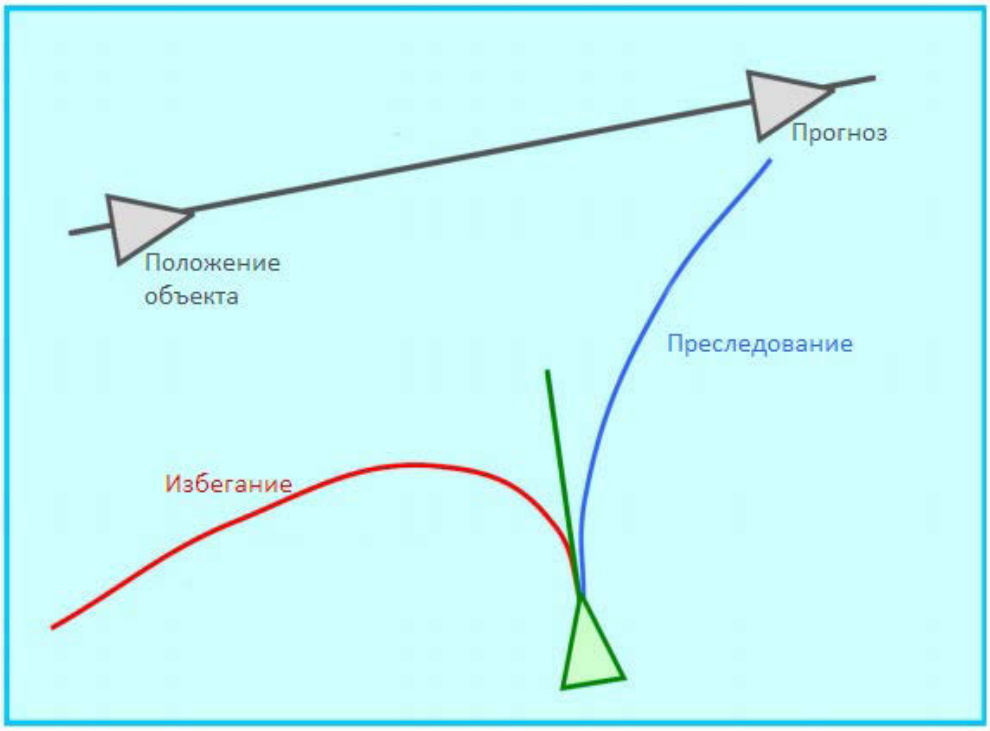

Рис. 3. Поиск и избегание с прогнозом

Бегство от преследователя

Следует упомянуть частный случай избегания динамического объекта, когда бегство осуществляется от преследователя. В данном случае помимо движения в противоположную сторону агент может выполнять разнообразные маневры и уклонения, тем самым усложняя свою поимку [16]. Кроме того, стадный инстинкт призывает агента двигаться к стаду, тем самым вызывая его лавинообразное движение (агенты, к которым приближается преследователь, аналогично обращаются в бегство, затем это бегство наблюдают остальные и присоединяются) [8]. При всем этом агенты осознают состояние окружающей опасности и стараются держаться ближе друг к другу (рис. 4).

Случайное движение

Метод случайного движения моделирует процесс блуждания объекта. Однако нельзя использовать случайное изменение вектора на каждом шагу, поскольку в таком случае движение будет нереалистичным и дерганным. Для достижения большей реалистичности можно с некоторой периодичностью суммировать вектор текущего направления и случайный сгенерированный вектор.

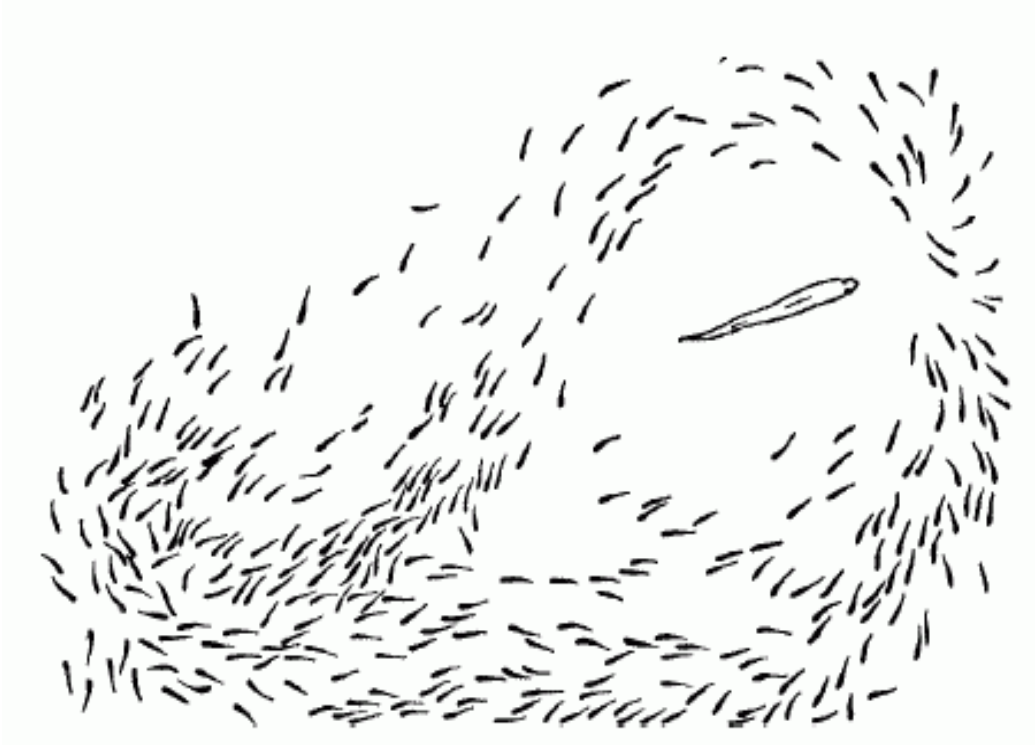

Рис. 4. Пример бегства от хищника в косяке рыб [8]

Для упрощения генерации вектора корректировки движения в качестве области его генерации можно использовать окружность [17]. Благодаря этому можно добиться плавности изменения направления движения, плавности изменения скорости и масштабируемости максимального отклонения. Таким образом, при каждой корректировке посредством текущего вектора движения и сгенерированного вектора изменяются и направление, и скорость движения (рис. 5). Данный метод также рекомендуется комбинировать с методами группового поведения.

Избежание столкновений в толпе

Благодаря этому методу алгоритма агентам в группе удается двигаться в разных направлениях и при этом не сталкиваться. Для лучшего понимания представим толпу людей во время какого-либо массового праздника. Один из агентов (в данном случае человек) движется к своим знакомым (цель) через толпу. При этом он анализирует направления движения окружающих его агентов и соответствующим образом корректирует свой курс. Для проведения корректировки могут использоваться изменение направления движения, ускорение или замедление.

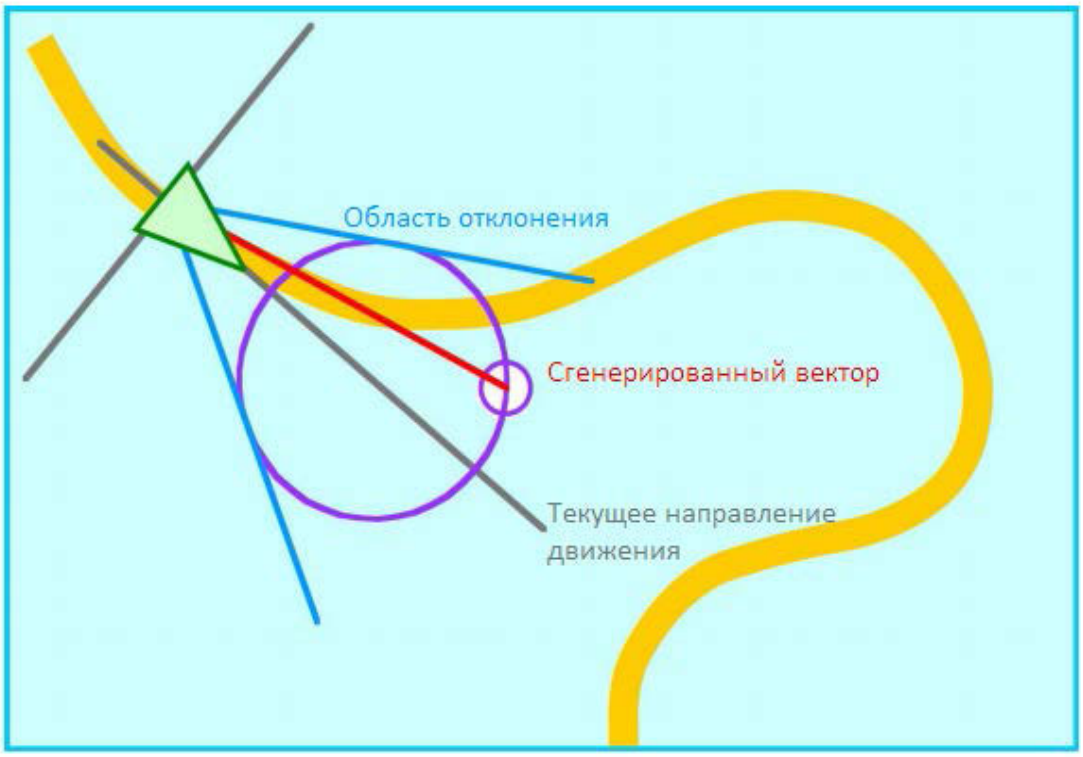

Рис. 5. Пример случайного движения

Когда в поле зрения движущегося агента попадает другой движущийся агент, происходит прогнозирование его движения. Агент корректирует свой курс для избегания столкновения. Аналогично поступает и другой агент. В результате курсу движения обоих агентов изменяются [18, 19].

Если корректировка прошла неуспешно и после повторного прогнозирования движения может возникнуть столкновение, метод запускается повторно и снова вносит корректировки (рис. 6).

Поддержание группового состояния

Целью данного метода является поддержание стада в сплоченном групповом состоянии. Для этого агент анализирует окружающую его область и вычисляет наиболее предпочтительное для него местоположение. Для того чтобы получить это местоположение, нужно вычислить усредненную позицию агентов, находящихся в наблюдаемой области. После успешного вычисления целевой точки агент начинает движение к ней, тем самым корректируя среднюю позицию для других агентов. Благодаря этому члены стада находятся на расстоянии друг от друга, при этом сохраняя целостность группы (рис. 7).

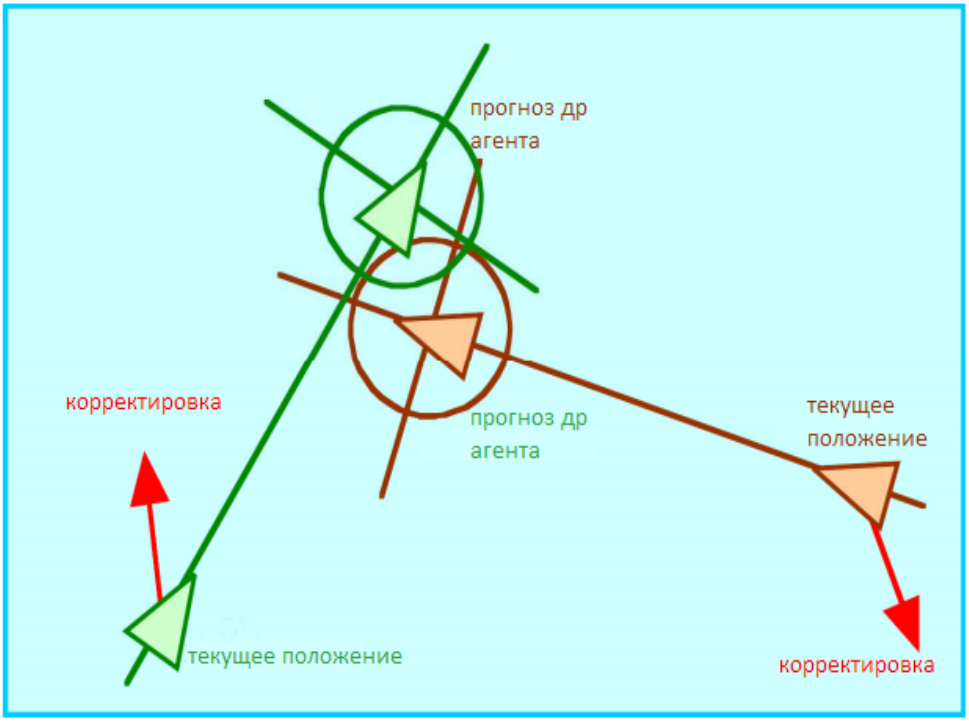

Рис. 6. Избежание столкновений двух объектов

Однако, если агенты будут повсеместно корректировать собственные целевые точки, стадо потеряет общее направление движения. А это неприменимо, когда группа должна осуществлять движение в определенном направлении. Во избежание этого данный метод может применяться совместно с методом сохранения общего направления группы.

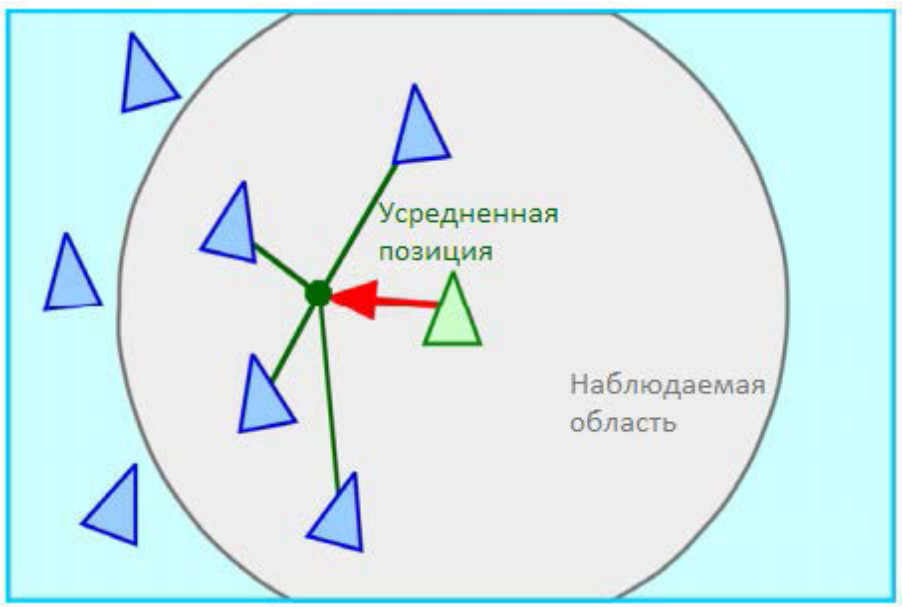

Рис. 7. Корректировка положения агента в группе

Сохранение общего направления группы

Этот метод направлен на поддержание общего направления внутри группы, а также сглаживания отклонений. В качестве примера рассмотрим ситуацию, когда стадо коров идет на пастбище. Коровы, анализируя поведение окружающих собратьев, двигаются в одном направлении. Если вдруг корова пойдет без веской причины в обратном направлении, поведение будет считаться отклоняющимся (рис. 8).

В зависимости от того, насколько жестко происходит корректировка направления, могут быть достигнуты разные результаты. Например, для стада коров не требуется чрезмерно точной корректировки, поскольку коровы находятся на достаточном расстоянии и их скорость невелика. Если рассмотреть бегущий табун лошадей, корректировка должна быть достаточно точной, чтобы сохранить структуру табуна, схожую с реальной. Корректировка должна быть подобрана таким образом, чтобы модель имела наибольшее сходство с реальным миром. Немаловажную роль играет скорость движения группы: чем она выше, тем точнее должна быть корректировка (поскольку на высокой скорости достаточно небольшого промежутка времени, чтобы агент с неверным направлением покинул группу).

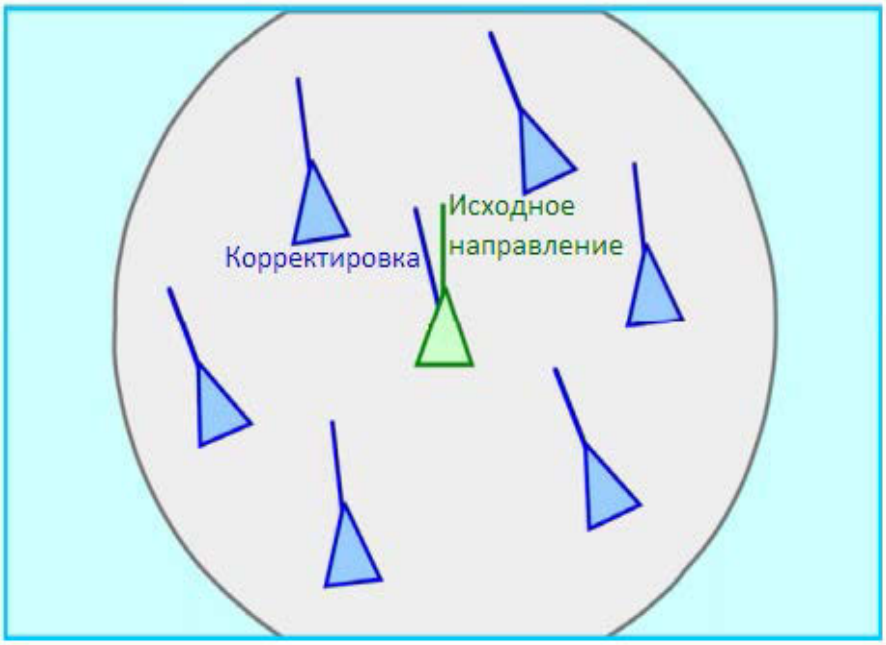

Рис. 8. Корректировка агента в группе

Отделение от стада

В некоторых случаях животные в стаде могут расходиться на большие расстояния. При этом, когда будет достигнуто достаточно большое отделение от других агентов, они перестанут оказывать на него влияние. В этом случае животное с большой вероятностью устремится обратно к сородичам. И иногда агент может окончательно отделиться от стада и вести себя индивидуально до тех пор, пока в его поле зрения не попадет кто-либо (рис. 9).

Данный метод алгоритма поведения имеет место в реальной жизни. Так, например, коровы, находясь на пастбище, расходятся по нему, однако сохраняют зрительный контакт с сородичами. Стоит одной из коров отойти достаточно далеко и отвлечься на время, как она может отделиться от стада. В этом случае, если зрительный контакт сохранился, корова поспешит вернуться в стадо. Если же корова не увидит стада, скорее всего, она просто продолжит пастись в одиночестве.

Данный метод, при комбинировании с методом группирования, может показывать отличный результат поведения, динамически изменяя расстояние между агентами в зависимости от условий. В чистом виде данный метод алгоритма решает задачу моделирования отбивающихся от стада особей (что регулярно происходит с некоторой вероятностью).

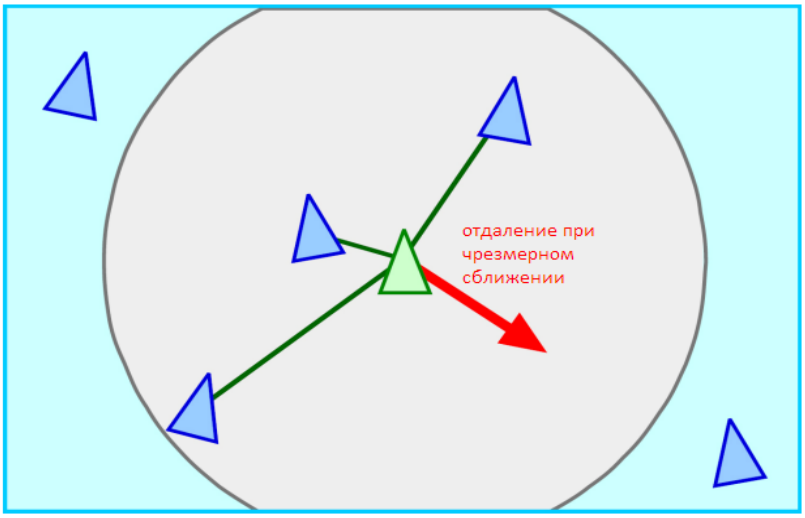

Рис. 9. Пример отдаления агента при чрезмерном сближении

Следование за вожаком

В некоторых группах животных, например, в волчьих стаях или стадах павианов, существует определенная иерархия, которая оказывает влияние на положение животных внутри группы.

В качестве примера рассмотрим стадо павианов. На рисунке 10 представлены два вида расположения животных: походный строй и позиции в случае нападения хищника. Буквам В соответствуют вожаки, С – самки, М – молодняк, Н – самцы низшего ранга [20].

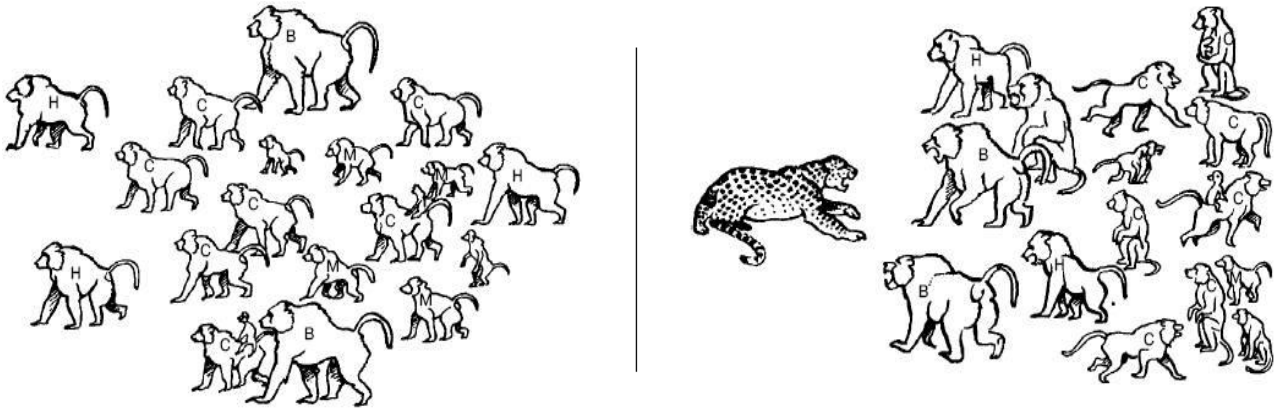

Рис. 10. Расположение животных в стаде павианов

В случае опасности самки с молодняком отходят назад, пропуская вперед самцов. Вожаки занимают передовые позиции как самые сильные представители. За ними следуют остальные самцы.

Таким образом, некоторым животным характерна определенная иерархия, влияющая на положение внутри стада (рис. 11).

В качестве упрощенной классической схемы представим модель следования за вожаком. В данном случае вожак выступает в роли основного координатора стаи и является целью. Остальные члены стада стремятся в область, находящуюся позади вожака (область преследования). Животные, оказавшиеся на пути вожака, стремятся уйти в сторону и присоединиться к следованию.

Таким образом, животные, находящиеся вблизи вожака, будут стремиться занять положение позади него. Кроме того, будет использован модифицированный метод отделения от стада (в качестве фактора, не позволяющего сталкиваться животным). Остальные животные, замечая движение большой группы (возможно даже не наблюдая вожака) будут присоединяться к общей группе и поддерживать движение.

В зависимости от конкретного типа животных данный метод при необходимости должен быть соответствующим образом модифицирован. Так, напри-

мер, могут быть изменены количество вожаков, целевая точка для животных, дистанция, скорость и т. д. [21].

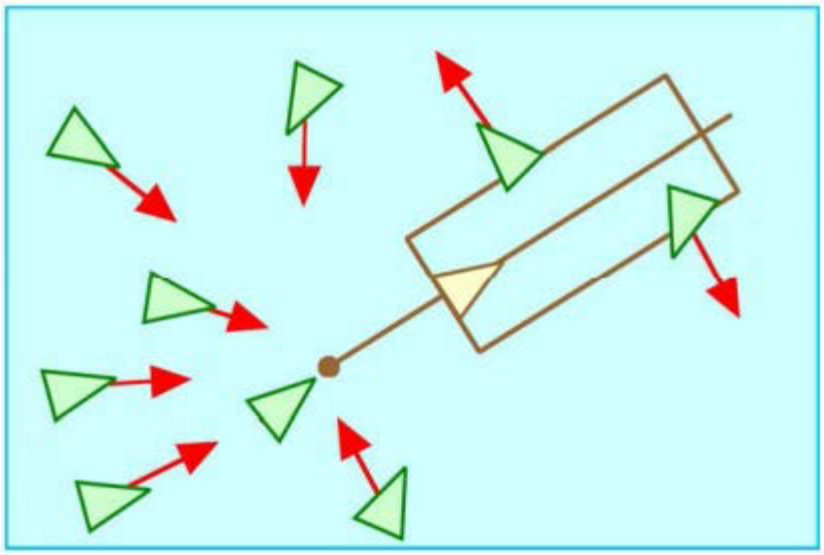

Рис. 11. Следование за вожаком

3. РЕАЛИЗАЦИЯ

Использование модульного подхода

Для того чтобы контроллер поведения был универсальными и легко масштабируемым, был применен модульный подход к реализации. При этом использовалась древовидная иерархия.

В качестве примера рассмотрим контроллер реалистичного поведения, адаптированный под коров. Животное в данном случае выступает в качестве вершины дерева. Следующий уровень – сценарий. В качестве упрощенного примера возьмем два следующих сценария: движение стада на пастбище (с пастбища) и сам процесс нахождения на пастбище. Следующий уровень отвечает за действия агентов для конкретного сценария, то есть что именно будет делать животное. Для сценария движения на пастбище предложены следующие действия: движение в стаде, остановка, отделение от стада (с небольшой вероятностью). Когда животные находятся на пастбище, применимы следующие действия: остановка (как пример – остановка для поедания травы), блуждание, корректировка расстояния до других животных, отделение от стада (с небольшой вероятностью). Далее следует самый последний уровень – методы моделирова-

ния действия. На этом уровне реализуется само действие при помощи комбинации описанных выше методов.

Благодаря такой иерархии и модульному подходу, в случае появления необходимости реализации нового действия мы просто добавляем требуемые модули и реализуем их при помощи методов.

Структура агента

В качестве агента используется заготовка персонажа, prefab (сборная часть). Агенты могут быть расположены на сцене как вручную, так и в автоматическом режиме на уровне генерации при запуске приложения. Второй метод является более предпочтительным, поскольку можно гибко настраивать область генерации животных и их количество.

Агент имеет структуру, схожую с реальным миром, но в некоторой степени упрощенную. На рисунке 12 изображена схема агента.

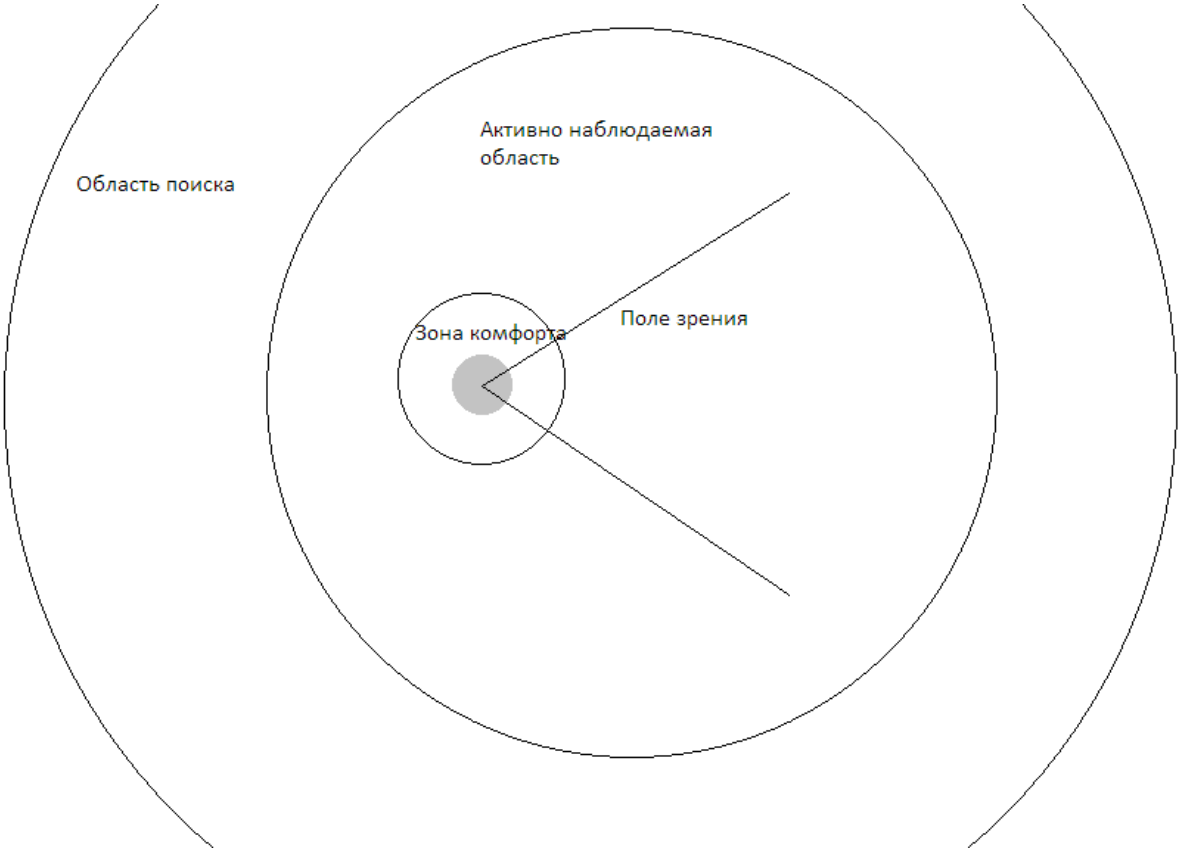

Рис. 12. Структурная схема агента

Зона комфорта является личной зоной агента, и без веской на то причины в реальном мире он старается не подпускать в нее других [22]. В случае опасности или чрезвычайной ситуации это правило может нарушаться, однако, как и

прежде, агент старается этого избежать, и в связи с этим могут возникать коллизии и столкновения (давка) [23].

Поле зрения – это область, наблюдаемая агентом в текущий момент времени. В зависимости от типа животного угол зрения может меняться. У некоторых птиц угол зрения достигает 360 градусов (например, 340 градусов – у голубя) (рис. 13).

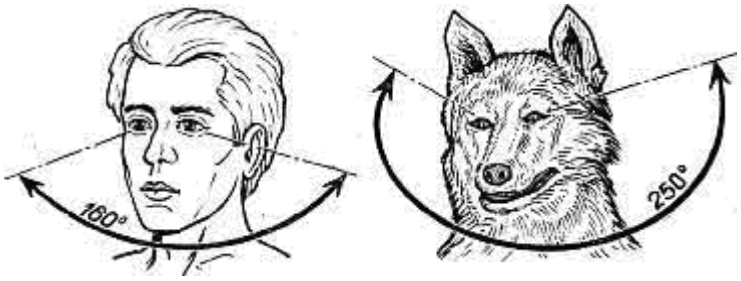

Рис. 13. Сравнение угла зрения человека и волка

Активно наблюдаемая область – это область агента, за которой он активно следит. При этом, в отличие от поля зрения, которое наблюдается в каждый момент времени, данная область обновляется с некоторой периодичностью. Агенты, находясь в данной области, оказывают активное воздействие друг на друга за счет стадных инстинктов. Данная область имеет небольшое смещение в сторону поля зрения (кроме тех случаев, когда поле зрения круговое или близко к нему).

В том случае, если агент в ходе блуждания отдалился от стада, он обращается к следующей по размеру зоне – области поиска. Обнаружив в этой области других животных, агент воспользуется методом поддержания группового состояния (вычислит усредненное значение целевой точки и отправится в нее). Таким образом стадо, как и в реальной жизни, будет сохранять свою целостность. Кроме того, в некоторых случаях животные будут отбиваться от стада (с небольшой вероятностью). Так, например, если животное отошло достаточно далеко, а за время следующего обращения к зоне поиска все животные покинули его.

Область навигации

Игровой движок Unreal Engine 4 поддерживает функцию автоматической постройки карты навигации персонажей. Для этого используется служебный объект «Nav Mesh Bounds Volume». Чтобы построить карту навигации, мы перемещаем данный объект на рабочую область и устанавливаем размеры в соответствии с предполагаемой игровой областью, после чего осуществляем по-

стройку карты. После постройки карты навигации ее можно настроить при помощи служебного объекта «Recast Nav Mesh» (рис. 14).

Рис. 14. Построенная карта навигации

Благодаря использованию карт навигации отпадает необходимость обучать персонажей избегать непроходимые области. Кроме того, присутствует возможность динамически обновлять карту навигации.

Реализация агента

Агент реализуется в качестве дочернего объекта для служебного класса Character. После создания блюпринта для персонажа (Agent) нужно его соответствующим образом настроить.

Поскольку дизайнерская часть не входит в рамки данной работы, для контроллера будут применены стандартная модель персонажа и соответствующая анимация. При дальнейшей реализации и проведении экспериментов такое решение не окажет никакого влияния на результаты (поскольку модель персонажа и анимация не влияют на скрипты).

Следующим этапом является настройка параметров агента. В зависимости от объекта, поведение которого мы моделируем, параметры могут кардинально меняться (скорость, поле зрения, реакция и т. д). Исходя из этого, на данном этапе будут внесены первоначальные изменения для запуска агента, а сами параметры будут задаваться в дальнейшем при генерации самих агентов при помощи класса GameMode, описанного ниже. Также на данном этапе необходимо

выбрать контроллер, который будет реализовывать поведение агента. Для этого был создан класс AgentController, унаследованный от класса AiController.

На данном этапе реализуется описанная ранее структура агента. Для этого к заготовке агента добавляются коллайдеры (Collision), которые будут отслеживать находящихся в них агентов.

Структура создания подразумевает следующий подход: класс, отвечающий за логику игры (GameController), генерирует в соответствующей области (SpawnArea) агентов (Agent). При этом параметры агентов подбираются случайным образом в определенном промежутке. Благодаря этому агенты обладают некоторыми отличиями. Далее вступает в дело непосредственно контроллер (AgentController) (рис. 15).

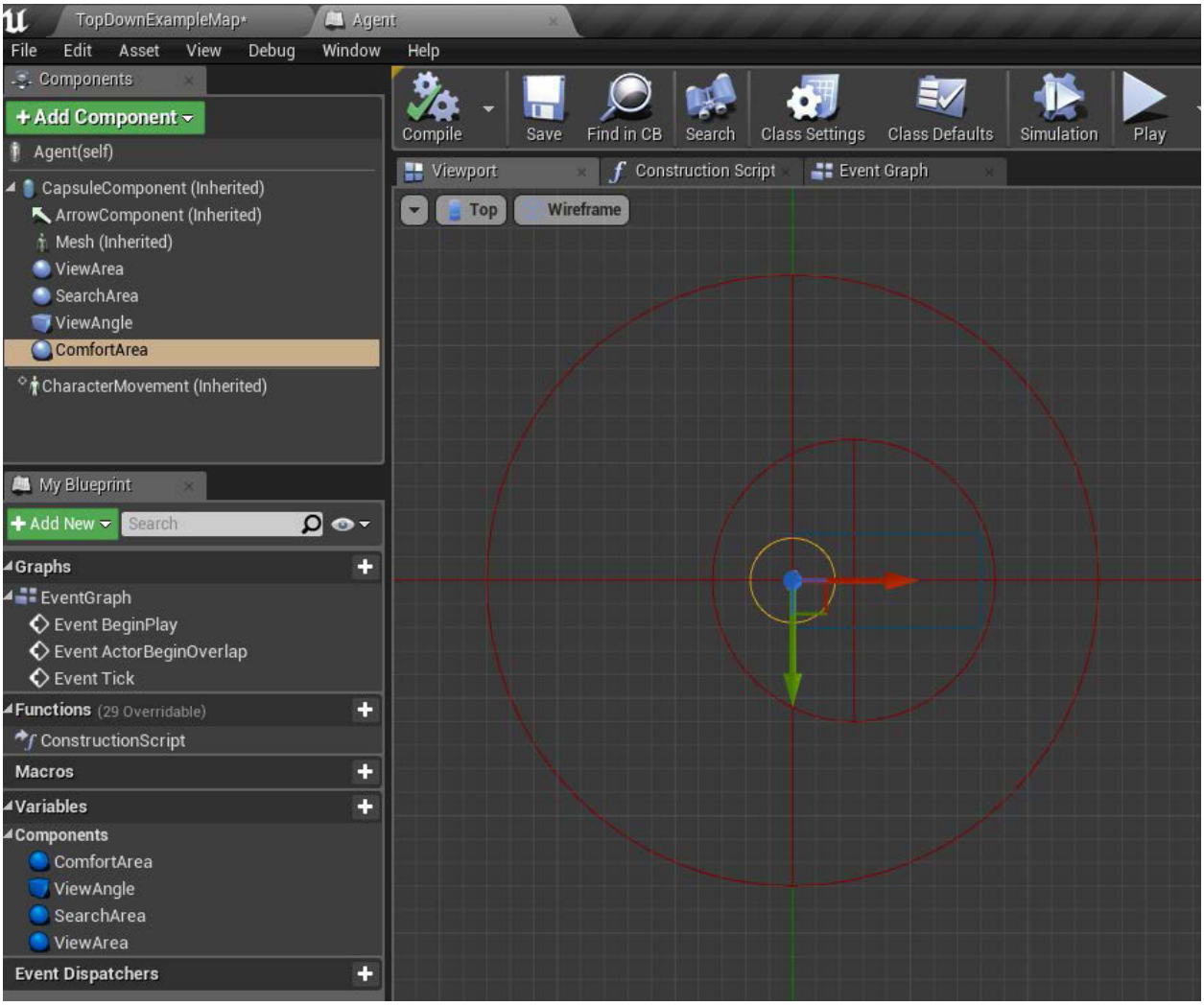

Рис. 15. Игровая структура агента (вид сверху)

Сценарий и дерево поведения

Для хранения ключевой информации, используемой контроллером для принятия того или иного решения, используется класс AgentBlackboard.

Информация анализируется агентом, принимается решение, какое действие должно быть выполнено, затем данная информация передается в ключи класса AgentBlackboard. После обновления ключа вызывается один из методов алгоритма, рассмотренных ранее, о в соответствии со структурой дерева поведения ActionTree. Методы алгоритма также обращаются к классу AgentBlackboard, тем самым завершая выполнение одного метода и переход к другому. Общая структура рассмотрена на рисунке 16.

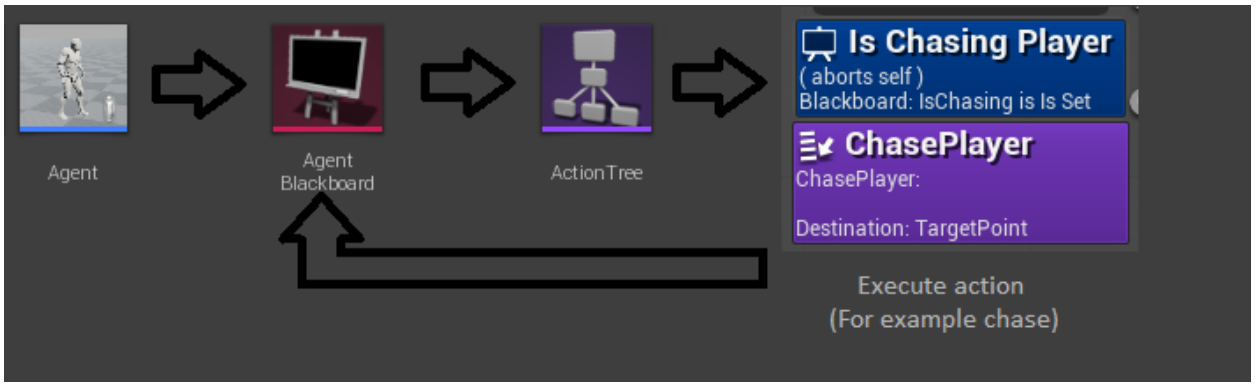

Рис. 16. Структура вызова методов алгоритма

Игровой контроллер агента (AgentController) в данной реализации отвечает за обращение к дереву поведения (ActionTree) и блекборду (Agent Blackboard).

4. ЭКСПЕРИМЕНТЫ

Реакция агентов на отдаление

В качестве примера рассмотрим ситуацию, когда один из агентов начнет отдаляться от общей группы. На данном примере мы управляем агентом с зеленой стрелкой, окружности представляют собой отслеживаемую область, красные стрелки – направления агентов (рис. 17). Для получения более выраженного эффекта было решено использовать небольшое количество агентов (так как, согласно методу, вычисляется усредненная позиция).

Когда мы начинаем отдаляться от других агентов, усредненные позиции смещаются, и в определенный момент агенты начинают корректировать свой курс, тем самым поддерживая связь (рис. 18).

Если продолжить движение в сторону, агенты будут повторять корректировку и изменять курс. Однако если удаляться достаточно быстро, корректировка не будет успевать за движением, и агент может покинуть отслеживаемую область и переместиться в область поиска.

Если вместо этого остановиться, агенты высчитают новые усредненные положения и, приблизившись к ним, перейдут к другому действию (в данном примере – к блужданию) (рис. 19).

Поскольку для более выраженной реакции мы используем всего 4 агента, они достаточно быстро распределяются. Когда агент достаточно сильно отдалится от других, он обратится к области поиска и вернется ближе к группе.

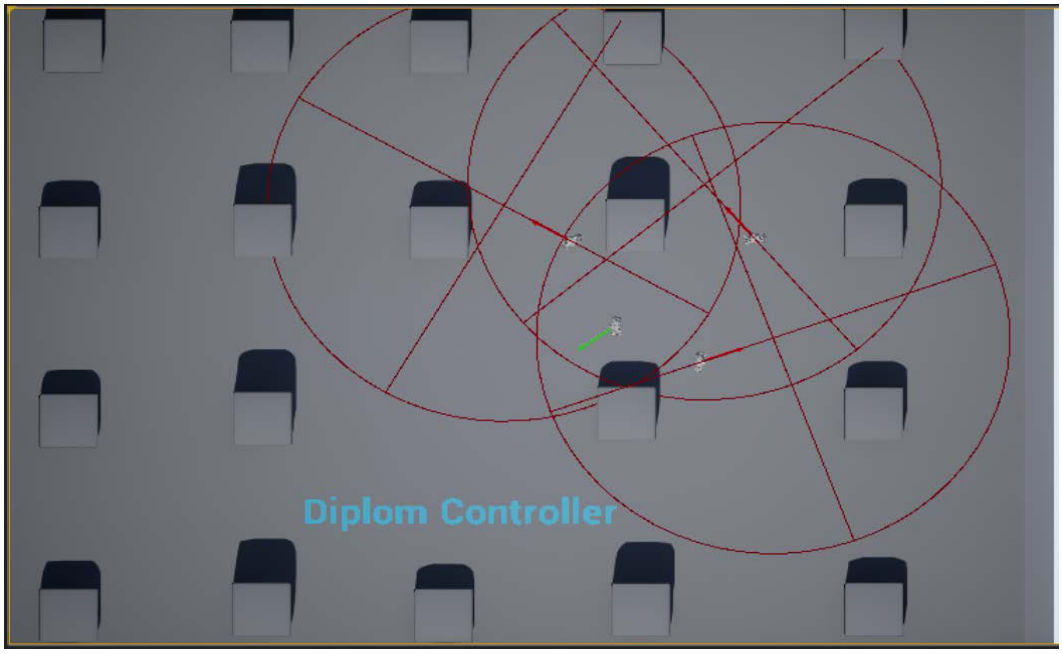

Рис. 17. Эксперимент на отдаление, начальная ситуация

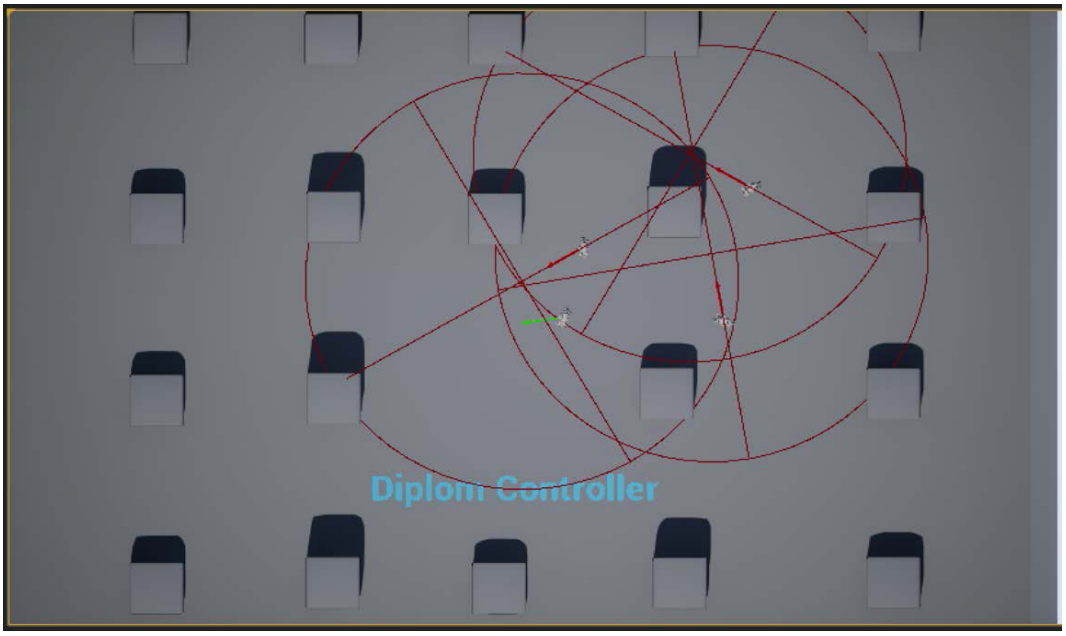

Рис. 18. Эксперимент на отдаление, корректировка

При использовании групп большего размера структура сохраняется лучше, поскольку в области действия находится больше агентов (по этой же причине корректировка происходит незначительно). В больших группах данный метод имеет место на окраинах, не давая расходиться агентам далеко.

После остановки нашего агента остальные персонажи скорректировали курс, однако из-за препятствий они пошли в обход. После сближения агенты вновь перейдут на блуждание.

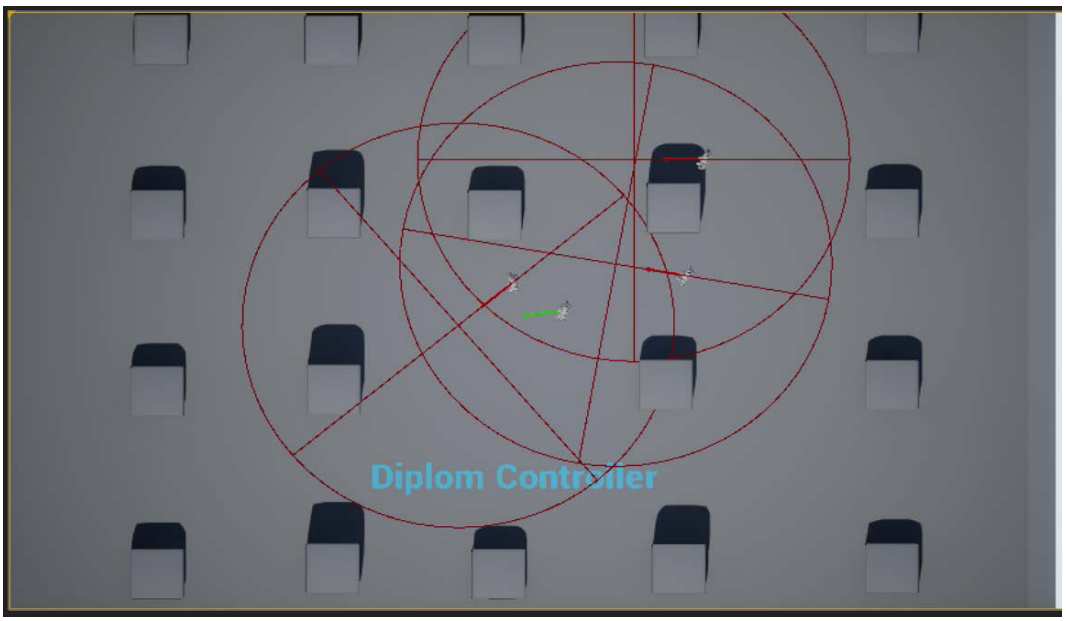

Рис. 19. Эксперимент на отдаление, переход агентов на блуждание

Крайне левый агент, благодаря отсутствию препятствий, после корректировки почти сразу оказался рядом и продолжил движение. В связи с этим он стал отдаляться от других агентов. Через некоторое время он вновь скорректирует свой курс по наблюдаемой области либо по области поиска (если успеет отдалиться достаточно далеко).

Данный эксперимент подтверждает правильность реакции агентов на ситуацию и качественную реализацию метода корректировки. Реакция агентов сравнивалась с описанием поведения по работам [24].

Реакция агентов на попадание в зону комфорта

Рассмотрим ситуацию, когда агент попадает в зону комфорта другого агента (может произойти как взаимная коллизия, так и нет, поскольку при генерации агентов зона комфорта также варьируется). В данном случае агент вычисляет вектор корректировки, направленный в противоположную от коллизии сторону, и объединяет с текущим вектором движения. Таким образом происходят корректировка курса и отдаление от коллизии. В программной реализации это выглядит аналогичным образом (рис. 20).

Управляемый агент изначально стоит неподвижно, и в тот момент, когда рядом проходит агент, наш персонаж пересекает зону комфорта. В ответ на попадание другого агента в коллайдер зоны комфорта вызывается действие и происходит корректировка согласно описанному ранее методу. Агент корректирует курс и начинает движение в соответствии с полученным направлением. Через некоторое время осуществляется повторная проверка, и, если все хорошо, агент переходит к другому действию (рис. 21).

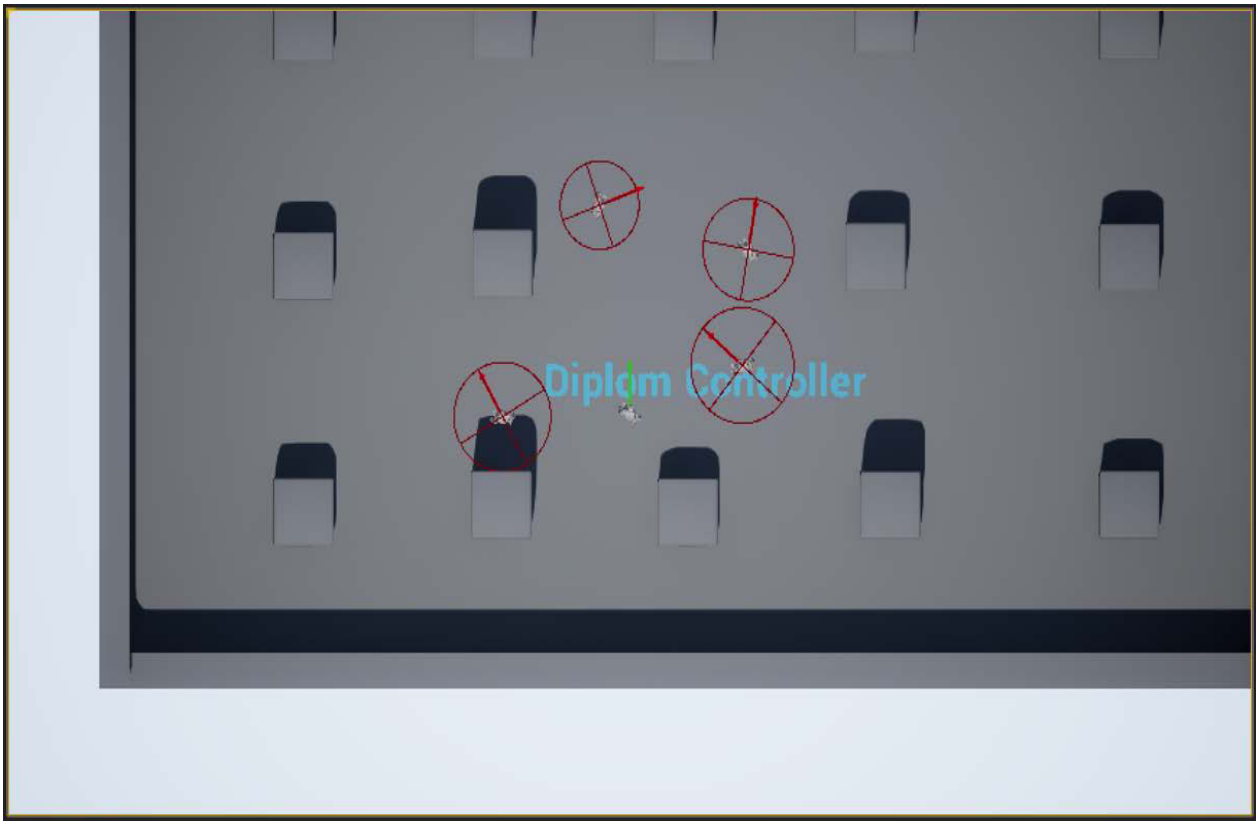

Рис. 20. Эксперимент с зоной комфорта, начальная ситуация

На рисунке 22 изображен процесс корректировки, агент изменяет направление в ответ на попадание в зону комфорта.

Рис. 21. Эксперимент с зоной комфорта, коллизия

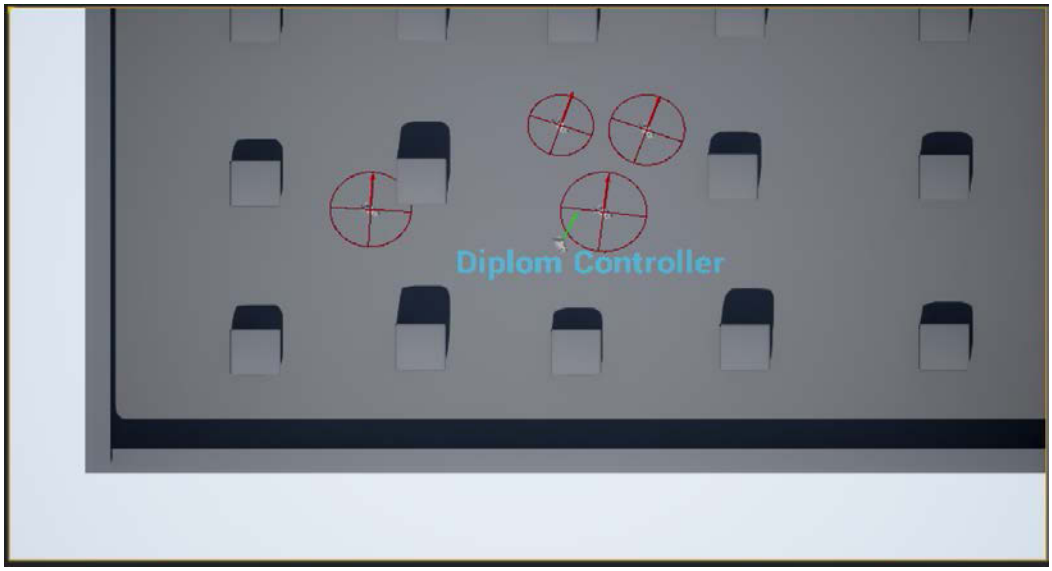

Рис. 22. Эксперимент с зоной комфорта, корректировка

Данный эксперимент также подтвердил правильность реакции на попадание объекта в зону комфорта [24].

Эксперимент на избежание столкновений

Смоделируем ситуацию, когда пути двух агентов будут пересекаться. В ответ на это контроллер, проведя анализ направления, скорректирует курс агента. Спустя небольшой промежуток времени T (используется для изменения направления) будет проведен повторный анализ, и в случае необходимости курс будет повторно корректироваться. Если после нескольких неуспешных корректировок агенты окажутся на расстоянии, меньшем, чем агент пройдет за это время T, то он остановится во избежание столкновения, после чего агент обратится к методу, который используется при попадании в зону комфорта. Поскольку в этот момент времени агент неподвижен, он сразу развернется в противоположную сторону.

На рисунке 23 изображена начальная ситуация: агент, которым мы управляем, изменил направление движения таким образом, что его путь по прогнозу пересекается с другим агентом. В ответ на это искомый агент изменяет свое направление. Для эксперимента мы будем специально ускорять нашего агента таким образом, чтобы он вновь шел пересекающимся курсом.

После корректировки курса агент изменил направление и продолжил движение вниз карты (рис. 24). Если в данный момент повторно изменить

направление нашего агента, наблюдаемый агент не успеет сманеврировать, остановится и развернется (рис. 25).

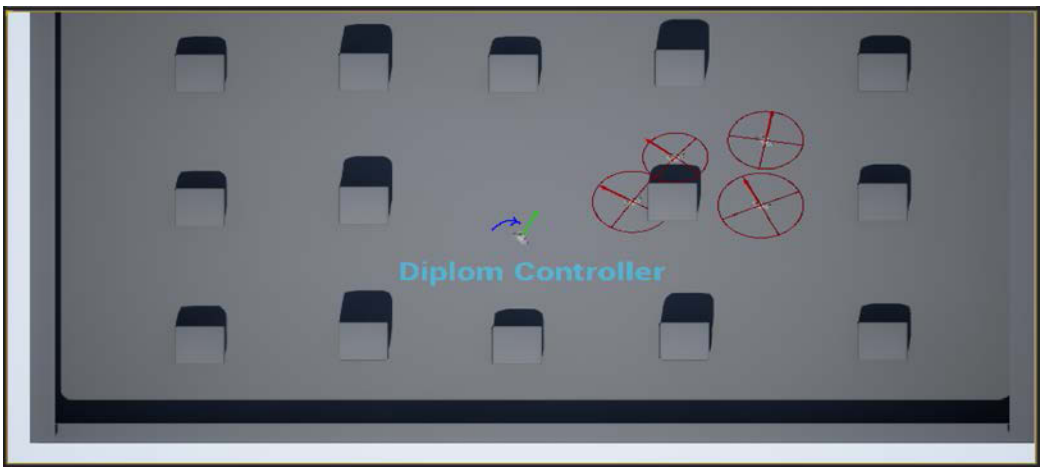

Рис. 23. Эксперимент на избежание столкновений, начальная ситуация

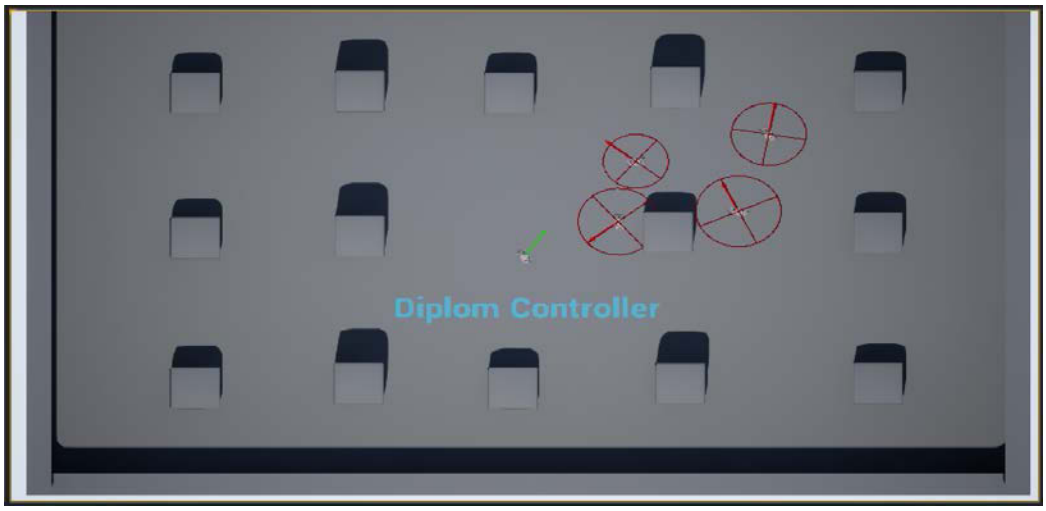

Рис. 24. Эксперимент на избежание столкновений, после корректировки

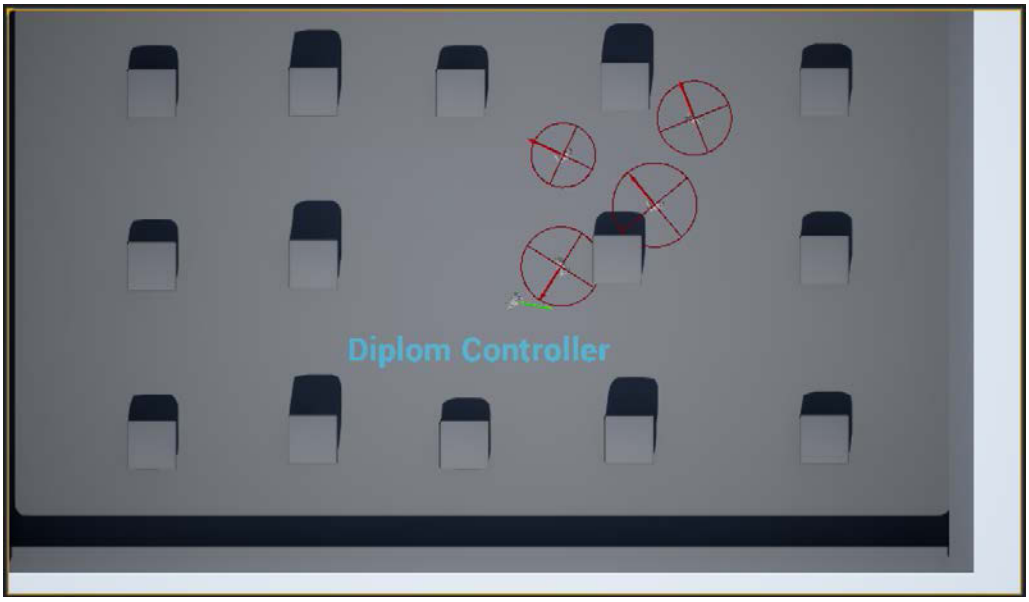

Рис. 25. Эксперимент на избежание столкновений, агент вынужден остановиться и развернуться

Сравнение с другими реализациями поведения толпы

Помимо описанных выше экспериментов был проведён сравнительный анализ полученного контроллера с другими методами реализации, предложенными в рамках следующих научных работ:

- Апробация различных методов поведения толпы Р.В. Гребенникова; при проведении анализа было выявлено, что предложенный метод во многих случаях демонстрирует схожее поведение как для толпы, так и для группы [24].
- Метод Р.В. Гребенникова имитационного моделирования групповой динамики толпы. Данная работа был представлена на международном научном форуме Ломоносов 2010 и зарекомендовала себя как эффективная система моделирования и оценки групповой динамики [25]. При сравнении с динамикой стада, полученной в результате работы контроллера, мы пришли к выводу, что предложенная модель частично схожа с методом Гребенникова. Отличия связаны в первую очередь с тем, что в предложенном контроллере поведения стада используется меньшее количество отслеживаемых параметров (данное решение позволяет моделировать большие группы объектов, не прибегая к большим вычислительным ресурсам).
- Обзор и анализ существующих математических моделей поведения толпы с точки зрения социальной психологии [26, 27], проведена оценка эффективности предложенного контроллера с рассматриваемыми моделями. При сравне-

нии с определенными методами возникали некоторые отличия, связанные с разными подходами как социальной психологии. Однако в общем и целом методы показали реалистичное поведение, хотя и обладали некоторыми отличиями.

ЗАКЛЮЧЕНИЕ

Были проведены исследования в области поведения животных, выявлены основополагающие факторы, влияющие на формирование и поддержание групп животных, проведена декомпозиция поведения до базовых методов и указаны способы комбинации этих методов для достижения реалистичных результатов.

В качестве платформы реализации применен игровой движок Unreal Engine 4 в связке с BlueprintVS. Полученный в результате контроллер реалистичного поведения обладает большой практической значимостью и может быть применен во множестве областей: моделирование групп динамических объектов для кинематографа, моделирование поведения толпы в экстренных ситуациях, моделирование объектов как части транспортно-логистической системы, моделирование поведения животных для лесной промышленности или при пожарах [28] и наводнениях [29], моделирование безопасных мест массового пользования, создание реалистичной симуляции поведения домашних животных в виртуальных исторических реконструкциях, например [30], или при разработке компьютерных игр.

В дальнейшем данный контроллер может быть дополнен методами, реализующими поведение животных в частных случаях. Кроме того, можно спроектировать несколько контроллеров с разными уровнями проработки этих случаев и предоставить несколько готовых решений в данной области.

ЛИТЕРАТУРА

1. *Богомолова Е.М., Курочкин Ю.Л., Минаев А.Н.* Системная организация брачного поведения лосей // 111 съезд Всесоюз. териол. об-ва: Тез. докл. М.: Наука, 1982. Т. 2, С. 115–116.

2. *Zhao D.L., Yang L.Z., Li J.* Exit Dynamics of Occupant Evacuation in Anemergency// Physica A 363. 2006. P. 501–512.

3. *Якушкин Г.Д., Мичурин Л.Н., Павлов Б.М., Зырянов В.А.* Численность и миграции диких северных оленей на Таймыре // Труды IX Межд. конгресса биологов-охотоведов, 1970. С. 335–338.

4. *Estes R.* Behavioural Study of East African Ungulates. 1963–1965 // Nat. Geol. Soc. Res. Repts, Projects. 1969. P. 45–47; *Estes R.D.* Social Organization of the African Bovids // IUCN Publ. 1974, V. 1, No 24. P. 165–205.

5. *Smith R.A.* Engineering for Crowd Safety. Amsterdam: Elsevier, 1993. 442 p.

6. *Sullivan T.J.* The "Critical Mass" in Crowd Behavior: Crowd Size, Contagion and the Evolution of Riots // Humboldt J. of Social Relations. 1977. V. 4, No 2. P. 46–59.

7. *Breder Jr.* C.M., *Halpern F.* Innate and Acquired Behavior Affecting the Aggregation of Fishes // Physiological Zoology. 1946. V. 19, No 2. P. 154–190.

8. *Павлов Д.С., Касумян А.О.* Стайное поведение рыб. М.: Московский университет, 2003. 74 с.

9. *Баскин Л.М., Бальчаускас Л., Жесткова И.А*. Пространственная структура стада коров // Докл. ВАСХНИЛ, 1989. Вып. 3. С. 36–38.

10. *Капица С.* Модель роста населения земли и экономического развития человечества // Вопросы экономики. 2000. №12.

11. *Копылов В.А.* Исследование параметров движения людей при вынужденной эвакуации. Дисс. канд. техн. наук. М., 1974.

12. *Reynolds C.W.* Flocks, Herds, and Schools: A Distributed Behavioral Model// Computer Graphics. 1987. V. 21, No 4. P. 25–34 (ACM SIGGRAPH'87 Conf. Proc., Anaheim, California, July 1987).

13. *Reynolds C.W.* Not Bumping Into Things// The Notes for the SIGGRAPH 88 Course Developments in Physically-Based Modeling, P. G1–G13, published by ACM SIGGRAPH.

14. *Isaacs Rufus.* Differential Games: A Mathematical Theory with Application to Warfare and Pursuit// Control and Optimization, John Wiley and Sons, New York.

15. *Isaacs Rufus.* Differential Games: A Mathematical Theory with Application to Warfare and Pursuit// Control and Optimization, John Wiley and Sons, New York.

16. *Cliff Dave, Miller Geoffrey*. Co-Evolution of Pursuit and Evasion II: Simulation Methods and Results, from Animals to Animats 4// Proc. of the Fourth Int. Conf. on Simulation of Adaptive Behavior (SAB96), Maes, Mataric, Meyer, Pollack, and Wilson editors, ISBN 0-262-63178-4, MIT Press.

17. *Алексеев П.Г.* Разработка модели поведения персонажей // Квалификационная работа бакалавра, 2011. С. 23–28.

18. *Гребенников Р.В.* Модель поведения толпы на основе локальных потенциальных полей // Вестник Воронежского гос. ун-та. Системный анализ и информационные технологии. 2009. Т. 1. С. 46–50.

19. *Гребенников Р.В.* Модель поведения толпы на основе локального планирования пути // Вестник Воронежского гос. техн. ун-та. 2009. Т. 5, Часть 9. С. 77–81.

20. *Дерягина М.А.* Манипуляционная активность приматов. М.: Изд-во «Наука», 1986. 110 с.

21. *Хайнд Р.* Поведение животных. М.: Изд-во «Мир», 1975. С. 245–286.

22. *Аптуков А.М., Брацун Д.А.* Моделирование групповой динамики толпы, паникующей в ограниченном пространстве// Вестник Пермского университета. Серия: Математика. Механика. Информатика, 2009. №3. С. 18–23.

23. *Богданов К.Ю.* Динамика паникующей толпы // Квант, 2005. №5. С. 2–7.

24. *Гребенников Р.В., Тюкачев Н.А.* Апробация различных методов поведения толпы// Сборник работ участников конференции «ИТ-2010». Н. Новгород, 2010. С. 10–12.

25. *Гребенников Р.В.* Метод имитационного моделирования групповой динамики толпы // Материалы Межд. молодежного науч. форума «ЛОМОНОСОВ-2010», Вычислительная математика и кибернетика. М.: МГУ, 2010. С. 47–48.

26. *Гребенников Р.В.* Обзор и анализ существующих математических моделей поведения толпы с точки зрения социальной психологии // Сб. работ участников конференции «Гибридный Интеллект 2010». Воронеж: МИКТ, 2010. С. 13–19.

27. *Гребенников Р.В.* Обзор классических методов моделирования поведения толпы // Межвузовский сб. науч. трудов. Серия: Оптимизация и моделирование в автоматизированных системах. Воронеж: ВГТУ, 2010. С. 50–53.

28. *Гиниятов А.А., Кугуракова В.В., Якушев Р.С.* Разработка симуляционного приложения для моделирования лесных пожаров с учётом погодных условий и формы ландшафта // Электронные библиотеки. 2016. Т. 19, №3. С. 180–192.

29. *Римова Л.З., Кугуракова В.В., Якушев Р.С.* Разработка симуляционного приложения для моделирования разрушений от наводнений с многофакторным учётом // Электронные библиотеки. 2016. Т. 19, №3. С. 238–250.

30. *Хафизов А.Р., Баранов В.С., Сергеев А.С., Кугуракова В.В., Ситдиков А.Г.* Археологические объекты болгарского городища X–XV вв., как материал для создания виртуальной культурно-исторической реконструкции// Электронные библиотеки. 2015. Т. 18, №5. С. 269–282.

CONTROLLER OF REALISTIC BEHAVIOR STAY AND STAGE OF ANIMALS

V.V. Kugurakova[1], A.M. Stepanov[2]

[1-2]*Higher School ITIS. Kazan Federal University*

[1] vlada.kugurakova@gmail.com, [2]stepanovaleksandrm@gmail.com

Abstract

The work is aimed at considering the process of modeling a realistic controller of the behavior of groups of objects. The main techniques and principles used to create a realistic controller of the behavior of autonomous agents, united in related groups, are investigated. Based on this data, a behavior controller has been created.

The following works were performed: on the calculation of the effectiveness of the behavior of groups of autonomous agents; on the possibility of using a system of local scalar fields with the aim of constructing the most accurate mathematical model; on the analysis of the possibility of creating a hierarchical system of multi-agent subgroups within the group, with the aim of realizing the movement of these same groups, as an integral part; to use a structured approach to create an efficient controller architecture; on conducting practical experiments to evaluate the correctness of the data obtained.

To achieve the goals set, the computational methods of optimization theory, mathematical statistics, and analytical methods of the mathematical modeling apparatus are involved.

Keywords*: controller, group, behavior model.*

REFERENCES

1. *Bogomolova E.M., Kurochkin YU.L., Minaev A.N.* Sistemnaya organizaciya brachnogo povedeniya losej // 111 s"ezd Vsesoyuz. teriol. ob-va: Tez. dokl. M.: Nauka, 1982. T. 2, S. 115–116.

2. *Zhao D.L., Yang L.Z., Li J.* Exit Dynamics of Occupant Evacuation in Anemergency// Physica A 363. 2006. P. 501–512.

3. *Yakushkin G.D., Michurin L.N., Pavlov B.M., Zyryanov V.A.* CHislennost' i migracii dikih severnyh olenej na Tajmyre // Trudy IX Mezhd. kongressa bio-logov-ohotovedov, 1970. S. 335–338.

4. *Estes R.* Behavioural Study of East African Ungulates. 1963–1965 // Nat. Geol. Soc. Res. Repts, Projects. 1969. P. 45–47; *Estes R.D.* Social Organization of the African Bovids // IUCN Publ. 1974, V. 1, No 24. P. 165–205.

5. *Smith R.A.* Engineering for Crowd Safety. Amsterdam: Elsevier, 1993. 442 p.

6. *Sullivan T.J.* The “Critical Mass” in Crowd Behavior: Crowd Size, Contagion and the Evolution of Riots // Humboldt J. of Social Relations. 1977. V. 4, No 2. P. 46–59.

7. *Breder Jr.* C.M., *Halpern F.* Innate and Acquired Behavior Affecting the Aggregation of Fishes // Physiological Zoology. 1946. V. 19, No 2. P. 154–190.

8. *Pavlov D.S., Kasumyan A.O.* Stajnoe povedenie ryb. M.: Moskovskij universitet, 2003. 74 s.

9. *Baskin L.M., Bal'chauskas L., Zhestkova I.A.* Prostranstvennaya struktu-ra stada korov // Dokl. VASKHNIL, 1989. Vyp. 3. S. 36–38.

10. *Kapica S.* Model' rosta naseleniya zemli i ehkonomicheskogo razvitiya chelovechestva // Voprosy ehkonomiki. 2000. №12.

11. *Kopylov V.A.* Issledovanie parametrov dvizheniya lyudej pri vynuzh-dennoj ehvakuacii. Diss. kand. tekhn. nauk. M., 1974.

12. *Reynolds C.W.* Flocks, Herds, and Schools: A Distributed Behavioral Model// Computer Graphics. 1987. V. 21, No 4. P. 25–34 (ACM SIGGRAPH'87 Conf. Proc., Anaheim, California, July 1987).

13. *Reynolds C.W.* Not Bumping Into Things// The Notes for the SIGGRAPH 88 Course Developments in Physically-Based Modeling, P. G1–G13, published by ACM SIGGRAPH.

14. *Isaacs Rufus.* Differential Games: A Mathematical Theory with Application to Warfare and Pursuit// Control and Optimization, John Wiley and Sons, New York.

15. *Isaacs Rufus.* Differential Games: A Mathematical Theory with Application to Warfare and Pursuit// Control and Optimization, John Wiley and Sons, New York.

16. *Cliff Dave, Miller Geoffrey*. Co-Evolution of Pursuit and Evasion II: Simulation Methods and Results, from Animals to Animats 4// Proc. of the Fourth Int. Conf.

on Simulation of Adaptive Behavior (SAB96), Maes, Mataric, Meyer, Pollack, and Wilson editors, ISBN 0-262-63178-4, MIT Press.

17. *Alekseev P.G.* Razrabotka modeli povedeniya personazhej // Kvalifikacionnaya rabota bakalavra, 2011. S. 23–28.

18. *Grebennikov R.V.* Model' povedeniya tolpy na osnove lokal'nyh potencial'nyh polej // Vestnik Voronezhskogo gos. un-ta. Sistemnyj analiz i informacionnye tekhnologii. 2009. T. 1. S. 46–50.

19. *Grebennikov R.V.* Model' povedeniya tolpy na osnove lokal'nogo planirovaniya puti // Vestnik Voronezhskogo gos. tekhn. un-ta. 2009. T. 5, Chast' 9. S. 77–81.

20. *Deryagina M.A.* Manipulyacionnaya aktivnost' primatov. M.: Izd-vo «Nauka», 1986. 110 s.

21. *Hajnd R.* Povedenie zhivotnyh. M.: Izd-vo «Mir», 1975. S. 245–286.

22. *Aptukov A.M., Bracun D.A.* Modelirovanie gruppovoj dinamiki tolpy, panikuyushchej v ogranichennom prostranstve// Vestnik Permskogo universiteta. Seriya: Matematika. Mekhanika. Informatika, 2009. №3. S. 18–23.

23. *Bogdanov K.Yu.* Dinamika panikuyushchej tolpy // Kvant, 2005. №5. S. 2–7.

24. *Grebennikov R.V., Tyukachev N.A.* Aprobaciya razlichnyh metodov povedeniya tolpy// Sbornik rabot uchastnikov konferencii «IT-2010». N. Novgorod, 2010. S. 10–12.

25. *Grebennikov R.V.* Metod imitacionnogo modelirovaniya gruppovoj dinamiki tolpy // Materialy Mezhd. molodezhnogo nauch. foruma «LOMONOSOV-2010», Vychislitel'naya matematika i kibernetika. M.: MGU, 2010. S. 47–48.

26. *Grebennikov R.V.* Obzor i analiz sushchestvuyushchih matematicheskih modelej povedeniya tolpy s tochki zreniya social'noj psihologii // Sb. rabot uchastnikov konferencii «Gibridnyj Intellekt 2010». Voronezh: MIKT, 2010. S. 13–19.

27. *Grebennikov R.V.* Obzor klassicheskih metodov modelirovaniya povedeniya tolpy // Mezhvuzovskij sb. nauch. trudov. Seriya: Optimizaciya i modelirovanie v avtomatizirovannyh sistemah. Voronezh: VGTU, 2010. S. 50–53.

28. *Giniyatov A.A., Kugurakova V.V., Yakushev R.S.* Razrabotka simulyacionnogo prilozheniya dlya modelirovaniya lesnyh pozharov s uchyotom pogodnyh uslovij i formy landshafta // Elektronnye biblioteki. 2016. T. 19, №3. S. 180–192.

29. *Rimova L.Z., Kugurakova V.V., Yakushev R.S.* Razrabotka simulyacionnogo prilozheniya dlya modelirovaniya razrushenij ot navodnenij s mnogofaktornym uchyotom // Elektronnye biblioteki. 2016. T. 19, №3. C. 238–250.

30. *Hafizov A.R., Baranov V.S., Sergeev A.S., Kugurakova V.V., Sitdikov A.G.* Arheologicheskie ob"ekty bolgarskogo gorodishcha X–XV vv., kak material dlya sozdaniya virtual'noj kul'turno-istoricheskoj rekonstrukcii// Elektronnye biblioteki. 2015. T. 18, №5. S. 269–282.

СВЕДЕНИЯ ОБ АВТОРАХ

КУГУРАКОВА Влада Владимировна – старший преподаватель Высшей школы информационных технологий и информационных систем, руководитель лаборатории «Виртуальные и симуляционные технологии в биомедицине». Сфера научных интересов – реалистичность визуализации и симуляций, иммерсивность VR.

Vlada Vladimirovna KUGURAKOVA, Senior Lecturer of Higher School of Information Technology and Information Systems, Head of Laboratory ”Virtual and simulation technologies in biomedicine”. Research interests include realism of visualization and simulation, immersion VR.

email: vlada.kugurakova@gmail.com.

***СТЕПАНОВ* Александр Михайлович** – бакалавр Высшей школы ИТИС Казанского (Приволжского) Федерального университета. Сфера интересов: разработка игр.

Alexander Mikhailovich STEPANOV – bachelor of the Higher School of ITIS Kazan (Privolzhsky) Federal University. Research interests game development.

email: stepanovaleksandrm@gmail.com

Материал поступил в редакцию 4 июня 2017 года

УДК 78.021.4+789.983

МЕТОДЫ «ОЖИВЛЕНИЯ» MIDI-ПАРТИЙ УДАРНЫХ МУЗЫКАЛЬНЫХ ИНСТРУМЕНТОВ

А.Л. Шайхутдинов[1]

[1]*Казанский (Приволжский) федеральный университет*

[1]azat2701@yandex.ru

Аннотация

Обсуждены некоторые недостатки гауссовой гуманизации и рассмотрены вопросы о том, каким образом модели артикуляции, проявляемые ударниками, можно эмулировать с помощью вероятностной модели. Разработан авторский алгоритм «оживления» midi-партий ударных музыкальных инструментов. Запрограммированы различные зависимости и закономерности, проявляющиеся при игре на барабанах. Создана зависимость параметров нот не только от параметров предыдущих нот, но и от последующих нот соответствующих частей ударной установки. Создана зависимость силы удара от положения ноты в такте. Таким образом, акцентируя ноты в сильной доле, была создана зависимость громкости ноты от совпадения с нотами других частей ударной установки. Для увеличения динамичности и живости партии было реализовано нарастание громкости хэта перед ударом по малому барабану. Проведено сравнение амплитуд соответствующих нот партий, насколько партия, оживленная с использованием определенного метода, отличается от партии, сыгранной профессиональным барабанщиком. При прослушивании тестов партии, обработанные с использованием модифицированного метода, объективно имеют больше общего с живым исполнением, чем с результатом, полученным при использовании гауссового метода или квантованных партий.

Ключевые слова: *ноты, алгоритм, midi-партия, ударные инструменты.*

ВВЕДЕНИЕ

В программируемой электронной музыке музыкальные партии обычно записываются с использованием секвенсоров и пианоролла. В этих системах музыкальный ритм сохраняется при использовании сетки с равными делениями.

Поскольку ноты располагаются по сетке, они квантуются по выбранным интервалам. По ощущениям квантованные ноты звучат слишком ровно и не звучат так же, как если бы на музыкальном инструменте играл человек. Для того чтобы эмулировать воспринимаемую естественность стиля человеческой игры, к последовательности может быть применен гуманизатор.

В статье исследователей из Бирмингема [1] дано следующее определение гуманизации: гуманизация – это процесс, который влияет на различные параметры сигнала для того, чтобы создать менее роботизированный результат. В партиях ударных инструментов она в первую очередь влияет на момент возникновения ноты по отношению к метрономной сетке и на амплитуду или громкость каждой соответствующей ноты. В существующих системах гуманизации используются гауссовские и равномерные генераторы случайных значений, которые в недостаточной степени оживляют партии музыкальных инструментов.

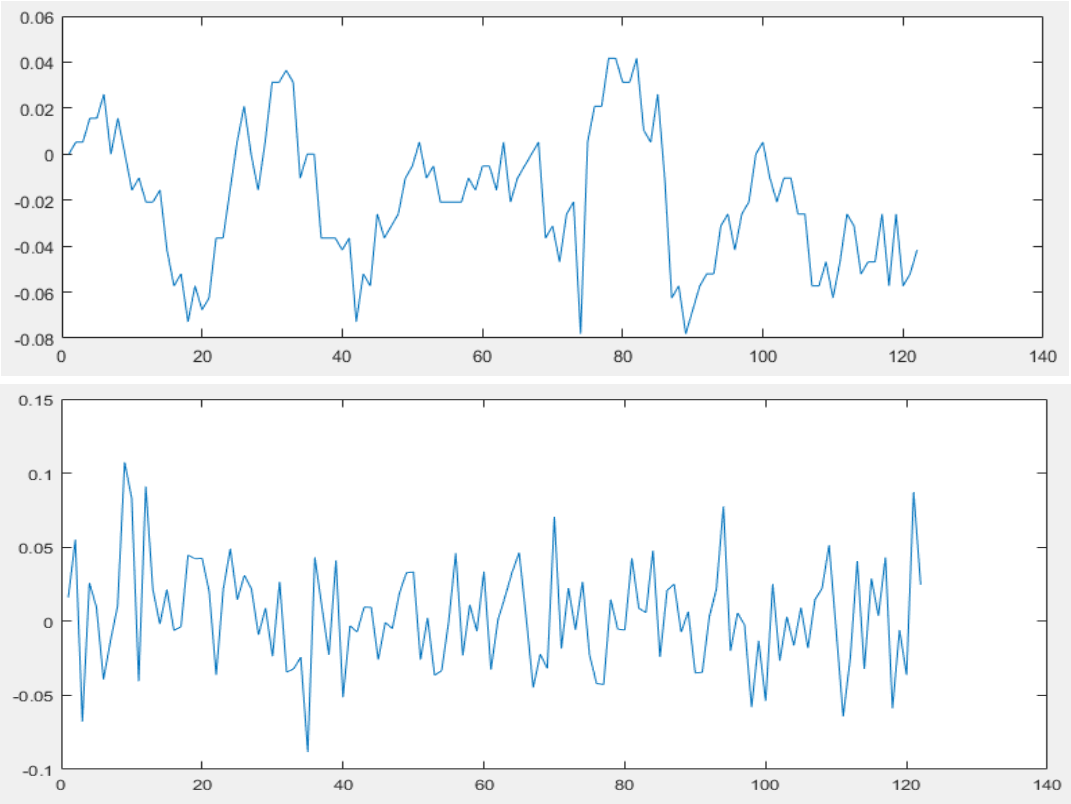

Рис. 1. Отклонения нот от метрономной сетки (по вертикали) и количество сыгранных нот (по горизонтали) одинаковых партий, сыгранной человеком (выше) и гуманизированной гауссовым методом (ниже)

В качестве примера, для сравнения, на рисунке 1 изображены отклонения от метрономной сетки нот партии, сыгранной человеком на клавиатуре компьютера, и той же партии, только квантованной, гуманизированной гауссовым ме-

тодом. По рисунку можно сделать вывод, что человеческая партия заметно отличается от искусственной, с применением существующих технологий.

В статье [2] рассказано о нюансах человеческой игры и программировании партий ударных музыкальных инструментов. В процессе написания MIDI-партий при создании цифрового музыкального контента важной проблемой часто являются синтетичность и неестественность звучания, а основной сложностью – оживление партий, придание партиям «человечности», их гуманизация.

Для обработки midi-партий был использован набор инструментов пакета MATLAB – MIDI Toolbox. В статье [3] показано, как использовать данный набор инструментов. Считывание MIDI-файла в программу происходит посредством функции readmidi, например, так: nmat = readmidi('laksin.mid'). После этого файл представляется в виде так называемой нотной матрицы со следующими столбцами: первая колонка отображает начало ноты, исходя из доли, вторая – продолжительность ноты в долях, третья – канал midi, в котором звучит нота, четвертая – высоту ноты, пятая – силу нажатия, а шестая и седьмая – то же, что первая и вторая, только в секундной системе отсчета.

В качестве синтезатора, для воспроизведения midi-партий ударных, был использован плагин EZdrummer 2. В этом плагине есть готовые образцы midi-партий, записанные на электронных барабанных установках профессиональными барабанщиками, – паттерны, грувы, которые можно использовать для создания своего цифрового музыкального контента, собирая по частям для различных разделов музыкального произведения.

«ОЖИВЛЕНИЕ» MIDI-ПАРТИЙ УДАРНОЙ УСТАНОВКИ

Для написания партий барабанов даны следующие советы, указанные в статьях [4] и [5], следуя которым можно достичь наиболее эмоциональной и живо звучащей партии.

- Не все ноты должны располагаться ровно по метрономной сетке, небольшие отклонения – частое явление;

- Динамика барабанной партии. Каждая новая нота должна отличаться от предыдущей по громкости, также включая высоту тона и тембр;

- Учет физических ограничений. Извлечение с помощью рук более двух звуков одновременно звучит неестественно;

- Украшения деталями (например, ноты-призраки);

- Грамотное использование частей ударной установки;
- Изменения темпа в течение музыкального произведения.

При гауссовой гуманизации каждый из параметров модулируется независимо друг от друга, используя распределение, определенное в уравнении (1), где среднее (μ) и стандартное (σ) отклонения назначаются пользователем опытным путем. Момент возникновения ноты (x=t) представляет собой позицию, соответствующую делению метрономной сетки (значение μ устанавливается в '0'), тогда как σ –параметрическая и представляет собой количество «изменчивости» в последовательности. Точно так же для изменения амплитуды (x=a), μ установлен на произвольное значение средней громкости, а σ представляет изменчивость в динамическом диапазоне.

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad . \qquad (1)$$

Хотя этот метод модуляции параметров увеличивает воспринимаемую хаотичность последовательности, он не обязательно увеличивает количество человекоподобного выражения или артикуляции, содержащиеся в последовательности. Так как распределения в гауссовой модели присваиваются мгновенно, предыдущие события не влияют на текущие события. Это значит, что при гауссовом распределении с μ=0 вероятности получения одного и того же значения слухового временного интервала различения с опережением и задержкой будут равны, т. е. P(-tn+1)=P(tn+1). В действительности это маловероятно. В исследовании [6] продемонстрирована структура, по которой видно, что вероятность *n*-го события условно зависит от *n*-1-го.

СОБСТВЕННАЯ МОДИФИКАЦИЯ

Для создания алгоритма оживления партий были изучены описанные выше методы. Для примера был выбран один из технически сложных образцов среди грувов плагина EZdrummer 2. Таким образом, мы имеем партию, сыгранную профессиональным барабанщиком, которую позже можно будет сравнивать с партией, имеющей те же ноты, но к которым было применено оживление. К партии сначала было применено квантование для последующего применения методов оживления и были выровнены громкости нот (для нот-призраков малого барабана была выделена отдельная высота, и их громкость была снижена примерно на половину) для создания образца квантованной партии для субъек-

тивного сравнения с результатами оживления различными методами. Партия была экспортирована в форматах MIDI и mp3.

Для обработки партии в MATLAB считывается MIDI-файл с квантованной партией ударных инструментов.

Создание алгоритма изменения громкости нот

Как уже известно, переменная nmat представляет собой матрицу, в которой строки соответствуют нотам, а столбцы – параметрам. Для изменения громкости нот создается цикл для изменения параметров нот с первой до последней:

```
for i = 1:length(nmat(:,3))
```

Чтобы партия звучала наиболее живо, громкость нот различных ударных музыкальных инструментов изменяется по разным правилам. Для этого сначала определяется часть ударной установки, например, если текущая нота – удар по большому барабану, то условие выглядит следующим образом:

```
if(nmat(i,4) >= 34 && nmat(i,4) <= 36)
```

Для придания динамики барабанной партии для всех нот, которые должны звучать максимально громко, можно изменить громкость при помощи следующей строки:

```
nmat(i,5) = 127 - rand * 10;
```

В партиях часто встречаются двойные удары по большому барабану. Чаще всего, если второй из них в сильной доле, то первый удар тише.

Для оживления партии большого барабана снижается громкость предыдущего удара, находящегося не в сильной доле, стоящего близко к текущему. Чтобы реализовать эту идею, для каждой ноты должен быть известен индекс предыдущей ноты – удара по большому барабану.

Величины изменения громкости нот были определены сравнением с соответствующими нотами версии грува перед квантованием.

При игре на райде барабанщики часто акцентируют сильные доли и подчеркивают большой барабан. Таким образом, если текущая нота – райд, то если она в слабой доле, то звучит тише, но только если не совпадает с большим барабаном.

Барабанщики при совпадении хай-хэта с малым барабаном увеличивают

громкость хай-хэта, подчеркивая малый барабан. При этом громкость обычно нарастает и в предыдущих ударах по хай-хэту. Чтобы создать нарастание громкости в предыдущих ударах по хай-хэту, для каждой ноты известны индексы двух предыдущих ударов по хай-хэту.

Как и на райде, так и при игре на хай-хэте, барабанщик сильные доли играет громче.

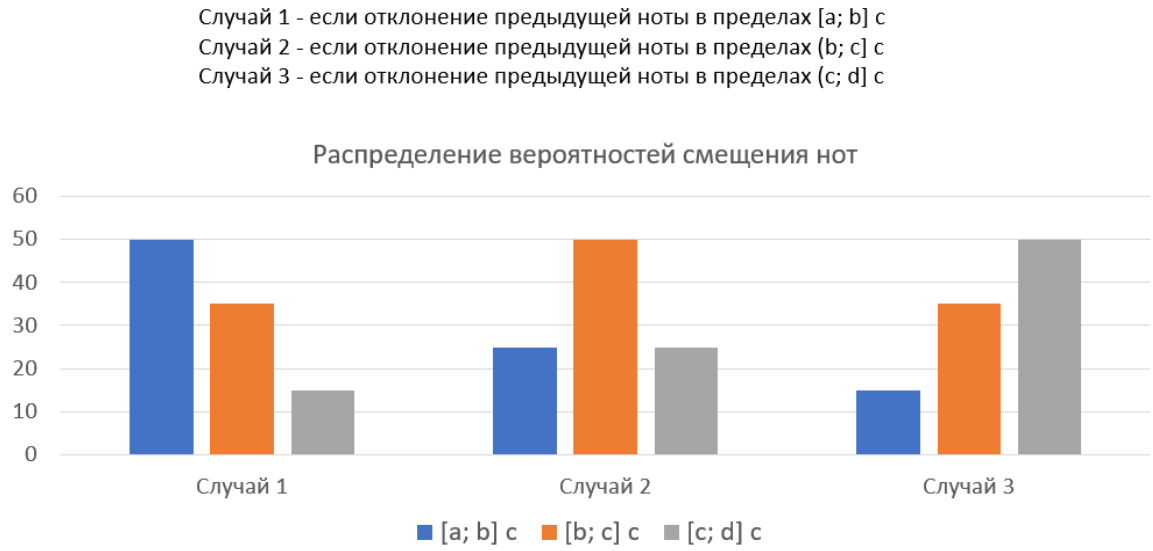

Рис. 2. Распределение вероятностей смещения нот

Создание алгоритма сдвигов нот

На рисунке 1 можно заметить, что в партии, сыгранной человеком, у сдвигов есть определенная зависимость от предыдущих сдвигов, в отличие от гуманизированной гауссовым методом.

Было предположено, что каждая новая нота должна сдвигаться с разной вероятностью на определенные диапазоны отклонения от метрономной сетки в зависимости от предыдущего отклонения (рис. 2). Так как чаще всего наиболее ровные партии не имеют отклонения более 15 миллисекунд, сразу после считывания MIDI-файла были определены границы или диапазоны отклонений от метрономной сетки следующим образом: a=–0.015; b=–0.005; c=0.005; d=0.015.

Для сдвигов нот создается цикл для изменения параметров нот со второй до последней:

```
for i = 2:length(nmat(:,6))
```

В конце итерации происходит смещение текущей ноты в секундной системе отсчета:

nmat(i,6)=nmat(i,6)+t(i).

Для экспорта матрицы в MIDI-файл используется следующая команда:

writemidi(nmat,'modified.mid').

СОЗДАНИЕ ПРИМЕРОВ

Для создания примера гауссовой гуманизации был использован следующий код:

```
nmat = readmidi('q.mid');
nmat(:,6)=normrnd(nmat(:,6),0.007)
nmat(:,5)=normrnd(nmat(:,5),5)
writemidi(nmat,'gauss.mid');
```

После экспорта, для создания примеров, файлы были импортированы в цифровую звуковую рабочую станцию. Таким образом партии приобрели звучание барабанной установки плагина EZdrummer 2. Из цифровой звуковой рабочей станции были экспортированы mp3-файлы.

СРАВНЕНИЕ РЕЗУЛЬТАТОВ

Для сравнения результатов с помощью команд MATLAB был получен график (рис. 3), показывающий, насколько близка динамика партий, гуманизированных гауссовым и модифицированным методами, к партии, сыгранной профессиональным барабанщиком. На графике видно, что при использовании модифицированного метода разница в динамике партий ближе к нулю.

Модифицированный метод в части изменения громкости нот определенно превосходит гауссовый. При прослушивании тестов партии, обработанные с использованием модифицированного метода, объективно звучат гораздо живее, чем с использованием гауссового метода и квантованные партии. Насчет оценки части сдвигов нот есть затруднения, так как оценивать сдвиги нот по похожести на игру живого барабанщика гораздо сложнее. На графике отклонений (рис. 4) можно заметить зависимость модифицированного метода от отклонений предыдущих нот.

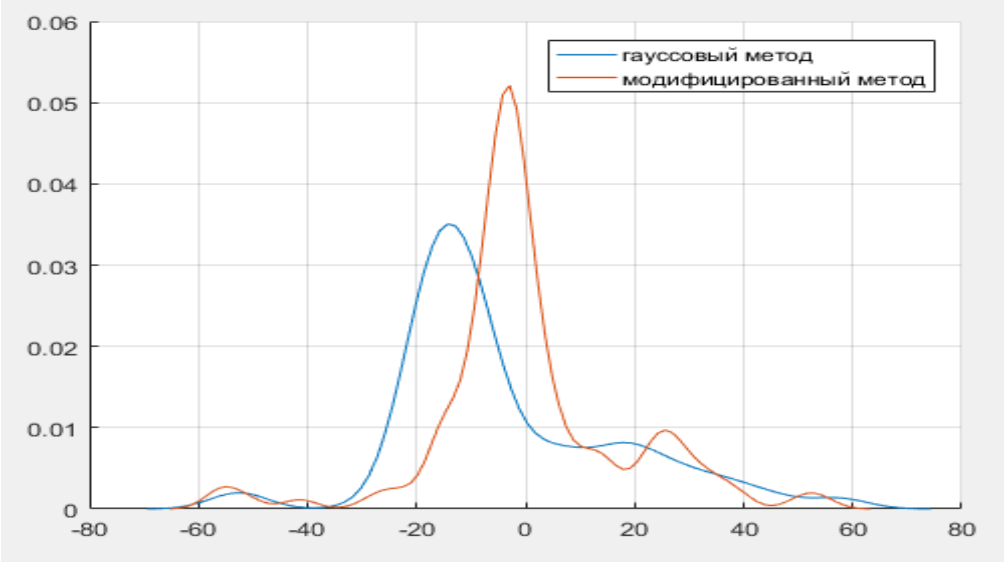

Рис. 3. Сравнение динамики методов

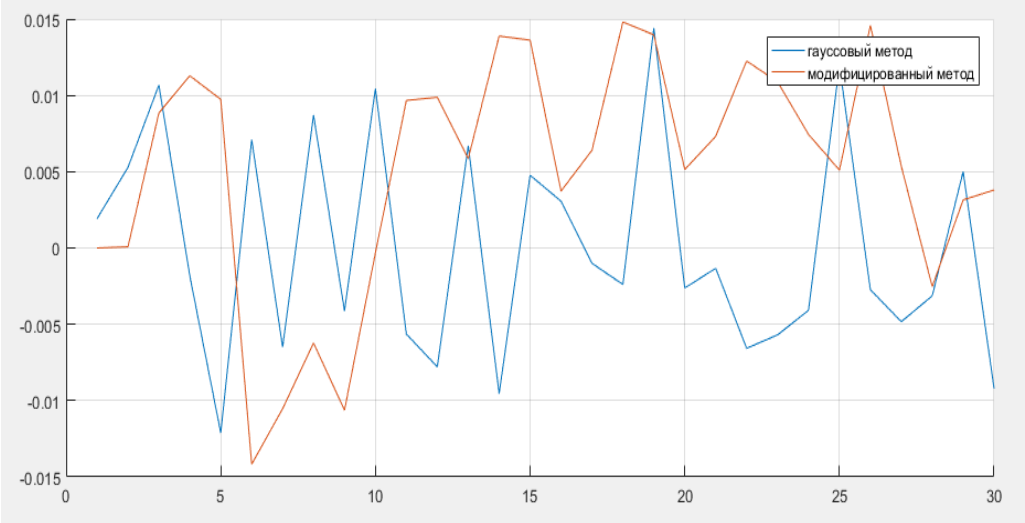

Рис. 4. Сравнение отклонений нот методов

Алгоритм изменения громкости нот модифицированного метода оптимизирован не для всех композиций. Для улучшения алгоритма его можно оптимизировать для остальных композиций, например, таких, в которых используются нестандартные размеры или которые сыграны в специфическом стиле.

ЛИТЕРАТУРА

1. *Stables R.I., Bullock J., Williams I.* Perceptually Relevant Models for Drum Pattern Humanisation// Proceedings of 131st Audio Engineering Society Convention, New York, 2011, AES.

2. Журнал «Звуковые виртуальные студии». 2009. №1. http://zstudio-n.narod.ru/zvs5.html

3. *Eerola T., Toiviainen P.* Mir in Matlab: The Midi Toolbox// Proceedings of 5th Int. Conf. on Music Information Retrieval, Barcelona, Spain, 2004, ISMIR.

4. Программирование ударных: самые распространенные ошибки. https://samesound.ru/studio/midi-work/302-programmirovanie-udarnyh-samye-rasprostranennye-oshibki

5. 20 советов по оживлению MIDI-барабанов. http://recording-studio.ru/2008/07/03/20-sovetov-po-ozhivleniyu-midi-barabanov/

6. *Stables R.I., Athwal C., Cade R.* Drum Pattern Humanisation using a Recursive Bayesian Framework// Proceedings of 133st Audio Engineering Society Convention, San Francisco, 2012, AES.

METHODS OF “LIVING” THE MIDI PARTY OF SHOCK MUSICAL INSTRUMENTS

A.L. Shaikhutdinov[1]

[1]*Higher School ITIS. Kazan Federal University*

[1]azat2701@yandex.ru

Abstract

This study discusses some of the shortcomings of Gaussian humanization and discusses how the articulation models developed by drummers can be emulated using a probabilistic model. An author's algorithm for “revitalizing” midi-parts of percussion instruments was developed. Various dependencies and regularities, which are manifested when playing drums, are programmed. The dependence of the notes parameters not only on the parameters of the previous notes, but also on the subsequent notes of the corresponding parts of the drum set was created. The dependence of the impact force on the position of the note in the measure is created. Thus, emphasizing notes in a strong lobe. Also, the dependence of the note's volume on the coincidence with the notes of other parts of the drum set was created. To increase the dynamism and liveliness of the party, the volume of the hat was increased before the impact on the small drum. A comparison was made of the amplitudes of the corresponding notes of the parts, as far as the party, animated using a certain method, is different from the one played by a professional drummer. When listening to tests,

batches processed using a modified method objectively sound lively than using Gaussian and quantized batches.

***Keywords**: notes, algorithm, midi-part, percussion instruments.*

REFERENCES

1. *Stables R.I., Bullock J., Williams I.* Perceptually Relevant Models for Drum Pattern Humanisation// Proceedings of 131st Audio Engineering Society Convention, New York, 2011, AES.

2. Zhurnal «Zvukovyye virtual'nyye studii». 2009. №1. http://zstudio-n.narod.ru/zvs5.html

3. *Eerola T., Toiviainen P.* Mir in Matlab: The Midi Toolbox// Proceedings of 5th Int. Conf. on Music Information Retrieval, Barcelona, Spain, 2004, ISMIR.

4. Programmirovaniye udarnykh: samyye rasprostranennyye oshibki. https://samesound.ru/studio/midi-work/302-programmirovanie-udarnyh-samye-rasprostranennye-oshibki

5. 20 Sovetov Po Ozhivleniyu MIDI Barabanov. http://recording-studio.ru/2008/ 07/03/20-sovetov-po-ozhivleniyu-midi-barabanov/

6. *Stables R.I., Athwal C., Cade R.* Drum Pattern Humanisation using a Recursive Bayesian Framework// Proceedings of 133st Audio Engineering Society Convention, San Francisco, 2012, AES.

СВЕДЕНИЯ ОБ АВТОРЕ

Шайхутдинов Азат Ленарович –бакалавр Высшей школы ИТИС, Казанский (Приволжский) федеральный университет.

Azat Lenarovich SHAIKHUTDINOV – has bachelor's degree of the Higher School of ITIS Kazan (Privolzhsky) Federal University.
Email: azat2701@yandex.ru

Материал поступил в редакцию 2 июня 2017 года