САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

КАФЕДРА КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ И СИСТЕМ

Дворянчикова Валерия Эдуардовна

Выпускная квалификационная работа бакалавра

Визуализация движения мультиагентных систем

Направление 010300

Фундаментальная информатика и информационные технологии

Научный руководитель

к. ф.-м. н, доцент

Погожев С. В.

Санкт-Петербург

2017

Содержание

Введение

Моделирование поведения групп интеллектуальных сущностей является одной из наиболее масштабных задач такого глобального направления современных исследований, как «искусственный интеллект». Группа взаимодействующих между собой сущностей называется *мультиагентной системой* [1]. Стремительное развитие областей робототехники, киноиндустрии и компьютерных игр вызывает актуальность поиска наиболее удачного решения задач по реализации как взаимодействия агентов внутри мультиагентной системы, так и взаимодействия нескольких таких систем между собой.

Из задач, где мультиагентные системы наиболее популярны, стоит отметить: сбор и обработка информации, автоматизация управления сложными системами, компьютерное моделирование. Мультиагентные системы основаны на взаимодействии множества интеллектуальных агентов. Одно из ключевых полезных свойств агентов — это *интеллектуальное поведение* [2], которое может быть заложено в каждого из них в соответствии с общим подходом к решению задачи, в рамках которой требуется взаимодействие многих агентов, работающих параллельно.

На сегодня, реализация поведения системы агентов имеет множество вариантов решений [3], большинство из которых упираются на организацию движений группы через поведение внутренних агентов. В данной работе рассмотрена модель организации движений нескольких таких групп, где в качестве агентов выступают мультиагентные системы, т. е. поведение группы зависит от поведения как внутригрупповых агентов, так и от поведения агентов внешних систем.

Постановка задачи

Задача состоит в построении реалистичной компьютерной модели поведения групп агентов с использованием игрового движка Unity3D, покрывающей реализацию как взаимодействия агентов внутри мультиагентной системы, так и взаимодействия мультиагентных систем между собой.

В основу подхода для решения поставленной задачи положена модель К. Рейнольдса «Boids» [4] с последующим расширением. Выбор обусловлен тем, что, имея простую реализацию, модель позволяет получить реалистичные результаты и позволяет производить расширение под другие задачи, связанные с организацией поведения сущностей в группе (наподобие стаи или роя).

Обзор литературы

Базовые понятия, области применение и возможные реализации мультиагентных систем раскрываются в [1], [2] и [3]. В [1] дано конструктивное определение понятия «мультиагентная система», в [2] подробно освещена тема интеллектуального поведения, реализации которого представлены в [3].

Основные характеристики такого важного понятия как интеллектуальный агент наиболее полно раскрываются в [5]. Источник [6] наиболее глобально рассматривает теорию искусственного интеллекта, в том числе, глубоко открываются важные понятия, относящиеся к мультиагентным системам: координация, соглашения и эмерджентное поведение.

В работах [4], [7] и [11] рассмотрен алгоритм Рейнольдса, который тесно связан с понятием роевого интеллекта (данное понятие было введено Херардо Бени и Ван Цзином в их работе, посвященной системе клеточных роботов [9]). В [4] Рейнольдсом даны общие характеристики алгоритма и наглядная демонстрация модели посредством компьютерного моделирования, а в [11] самым подробным образом им рассмотрены все аспекты данной модели с учетом дальнейших перспектив и возможных дополнений. Построенная Рейнольдсом модель «Boids» является примером *individual-based model* — понятие, раскрытое в [10]. В [7] дано общее описание модели «Boids».

В следующих главах будут подробнее рассмотрены основные понятия мультиагентных систем, модель «Boids» и, особенно подробно, алгоритм Рейнольдса, на котором строится решение поставленной в работе задачи.

Глава 1. Обзор предметной области

1.1. Мультиагентная система

Мультиагентная система (МАС) — это система, образованная несколькими взаимодействующими интеллектуальными агентами [5].

В данной системе агенты имеют несколько важных характеристик:

- автономность — агенты, хотя бы частично, независимы;
- ограниченность представления — ни у одного из агентов нет представления о всей системе, или система слишком сложна, чтобы знание о ней имело практическое применение для агента;
- децентрализация — нет агентов, управляющих всей системой.

Мультиагентное управление позволяет координировать целенаправленную деятельность автономных агентов, планировать их поведение и взаимодействие, адаптироваться к изменяющейся среде и разрешать конфликты между агентами.

Так как отрасль искусственного интеллекта достаточно нова, и мультиагентные системы являются одной из сложнейших задач, то мы располагаем обширным списком тем для исследования в рамках МАС: знания, желания и намерения, координация, организация, коммуникация, согласование, конкуренция, мультиагентное обучение и др.

Так как данная работа тесно связана с понятием *координация,* то рассмотрим его подробнее.

1.2. Координация

Простейший метод, с помощью которого группа агентов может обеспечить координацию при выполнении совместного плана, состоит в принятии определенного *соглашения* [6] до начала совместной деятельности. Соглашением является любое ограничение, касающееся выбора совместных планов,

выходящее за рамки того основного ограничения, в соответствии с которым совместный план должен работать, если ему следуют все агенты. Например, соглашения «придерживаться своей стороны поля» и «один игрок всегда остается у сетки», приведут к исполнению различных планов действий. Некоторые соглашения, такие как вождение по правильной стороне дороги, приняты настолько широко, что считаются общественными законами. В качестве соглашений можно рассматривать и естественные языки.

Соглашения, указанные в предыдущем абзаце, являются характерными для конкретной проблемной области и могут быть реализованы путем внесения в описания действий таких ограничений, которые позволили бы исключить нарушения соответствующего соглашения. Более общий подход состоит в использовании соглашений, независимых от проблемной области. Например, если каждый агент использует один и тот же алгоритм планирования с одними и теми же входными данными, то он может следовать соглашению о выполнении первого найденного осуществимого совместного плана и быть уверенным в том, что другие агенты сделают такой же выбор. Более надежная, но и более дорогостоящая стратегия могла бы состоять в том, чтобы выработать все совместные планы, а затем выбрать, например, тот из них, внешнее представление для вывода на печать которого находится на первом месте в алфавитном порядке.

Соглашения могут также возникать благодаря эволюционным процессам. Например, колонии общественных насекомых выполняют сложные совместные планы, а осуществление подобных действий обеспечивается благодаря общим генетическим характеристикам отдельных особей в этой колонии. Согласованность действий может также быть обусловлена тем фактом, что отход от соглашений ведёт за собой уменьшение способности выживания особей при изменении жизненных условий, в связи с чем, любой осуществимый совместный план может стать шагом к стабильности. Следующий пример [6] рассматривает поведение птиц при образовании стай. Данное поведе-

ние можно описать с помощью модели, в которой каждая птица-агент, или боид (от слова «*bird-oid*») [7], выполняет три перечисленных ниже правила:

1. **Разделение.** Изменять направление полёта относительно соседей (двигаться в обратном направлении от них), при слишком близком приближении к ним.
2. **Соединение в линию.** Придерживаться направления полета, который позволяет занять среднюю позицию относительно соседей.
3. **Выравнивание.** Стараться двигаться в направлении движения соседей.

Если все птицы выполняют одни и те же описанные выше правила, то вся стая обнаруживает *эмерджентное поведение* (от слова *emergent* — появляющийся), совершая полет в виде одного псевдоустойчивого (не распадается со временем) тела приблизительно с постоянной плотностью [6].

Стоит отметить тот факт, что в многоагентных системах может проявляться *самоорганизация* — процесс упорядочения элементов одного уровня в системе за счёт внутренних факторов, без внешнего специфического воздействия — и сложное поведение, даже если стратегия поведения каждого агента достаточно проста [8]. Это составляет основу такого понятия, как роевой интеллект.

1.3. Роевой интеллект

Роевой интеллект (РИ) описывает коллективное поведение децентрализованной самоорганизующейся системы. Термин был введён Херардо Бени и Ван Цзином в 1989 году, в контексте системы клеточных роботов [9].

По аналогии с примерами из координации, системы роевого интеллекта, состоят из множества агентов-боидов, локально взаимодействующих между собой и с окружающей средой. Модель поведения системы чаще всего основана на биологической системе. Каждый агент подчиняется простым правилам и, хотя каждому из них ничто не указывает на то, что ему следует делать, локальные взаимодействия приводят к возникновению интеллекту-

ального глобального поведения, которое не поддается контролю отдельными агентами-боидами. Точной формулировки определения роевого интеллекта всё еще нет. В целом, РИ должен представлять собой многоагентную систему, которая бы обладала самоорганизующимся поведением, долженствующее проявлять разумное поведение [3].

Глава 2. Алгоритм Рейнольдса

Прежде чем приступить к самому алгоритму, рассмотрим природное проявление роевого интеллекта, которое вдохновило К. Рейнольдса на создание компьютерной модели — стая птиц.

Стая птиц представляет собой прекрасный пример коллективного поведения животных. Летая большими группами, птицы почти никогда не сталкиваются в воздухе. Сама стая двигается плавно и скоординировано, словно ей кто-то управляет. А любой, кто вешал в своем дворе кормушку, знает, что если о ней узнала одна птица, то вскоре о ней узнают и все остальные. Стая, наряду с колонией муравьев, косяком рыб или роем пчел, представляет собой роевой интеллект. Птицы в ней действуют согласно определенным — относительно простым — правилам. Каждая из птиц координирует свое движение, беря в расчет положение своих сородичей. А если найден источник пищи, то и вся стая узнает об этом.

В 1986 году Крейг Рейнольдс разработал компьютерную модель скоординированного движения живых существ, беря за основу поведение стаи птиц и косяков рыб. Данная модель основана на трехмерной вычислительной геометрии, обычно использующейся в компьютерной анимации и вспомогательной компьютерной графике. Рейнольдс алгоритмически вывел поведение каждой из птиц в отдельности, а также их общее взаимодействие в стае. Алгоритм, симулирующий поведение стаи существ, назван «Boids». Объекты внутри стаи называются боидами. В базисе алгоритма положены три простых правила (см. рис. 1), описывающие индивидуальные маневры каждого боида с учетом положения и скоростей соседних:

1. *Разделение*: каждый боид должен менять направление, чтобы избежать столкновения с соседями;
2. *Выравнивание*: каждый боид должен стремиться к средней скорости соседей;

3. *Сплоченность*: каждый боид должен стремиться двигаться вблизи центра группы.

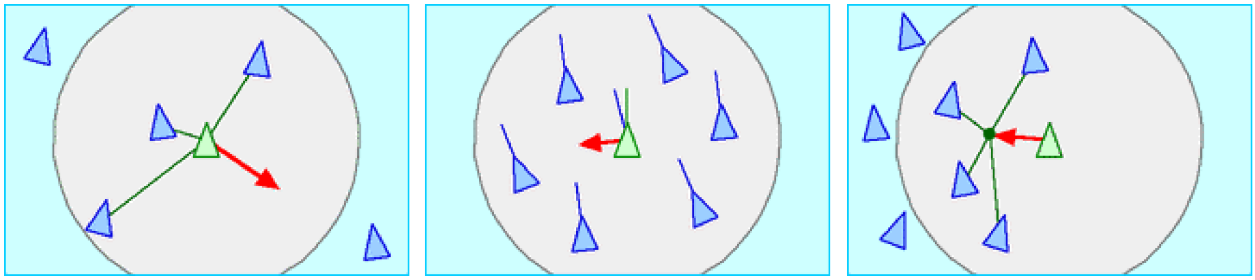

Рис.1. Базовые правила: разделение, выравнивание, сплоченность[1]

В данной модели:

- каждый боид имеет локализованное восприятие мира;
- для каждого боида необходимо хранить координаты и скорость;
- скорость является векторной величиной;
- поведение каждого боида базируется на знаниях о длине и направлении вектора между ним и каждым из боидов в группе.

Пример. Псевдокод общей структуры алгоритма

```
FOR EACH BOID b
    v1 = rule1(b)
    v2 = rule2(b)
    v3 = rule3(b)
        …
    b.velocity = b.velocity + k1*v1 + k2*v2 + …
    b.position = b.position + b.velocity
END
```

На каждой итерации алгоритма направление и длина вектора скорости каждого боида изменяются в соответствие со сведениями, полученными от соседних боидов. После вычисления направления вектора, боид перемещается в новую позицию. Таким образом, обновляются значения координат и

[1] Изображения взяты из статьи Рейнольдса [4], где подробно им рассказано об алгоритме «Boids».

направления векторов скоростей каждого боида стаи. После этого цикл повторяется.

Разделение (или *предотвращение столкновений*) наряду с *согласованием* (*выравниванием*) скорости дополняют друг друга. В комбинации они гарантируют, что члены моделируемой группы могут свободно перемещаться, не сталкиваясь друг с другом. Предупреждение столкновений есть стремление изменить направление движения так, чтобы избежать удара. Реализация правила разделения опирается на знания об относительном положении соседей в стаи и игнорирует их скорость. И наоборот, правило согласования скорости основано только на скорости и игнорирует позицию. Предсказание предотвращения столкновений можно описать так — если боид согласует скорость со своими соседями, то маловероятно, что он будет сталкиваться с любым из них в ближайшее время. При согласовании скорости, расстояние между боидами остается приблизительно инвариантным относительно текущего момента. Предотвращение столкновений, в свою очередь, служит для установления минимального расстояния «разделения», а выравнивание скорости стремится поддерживать его.

Пример. Вариант реализации правил разделения и выравнивания

```
PROCEDURE Rule1(BOID bi)
    Vector v = 0;
    FOR EACH BOID b
        IF b != bi AND |b.position - bi.position| < 100 THEN
            v += bi.position - b.position
        END IF
     RETURN v
END PROCEDURE


PROCEDURE Rule2(BOID bi)
     Vector v_average;
     FOR EACH BOID b
         IF b != bi THEN
             v_average += b.velocity
         END IF
     END
     // N - количество боидов в группе
```

```
    v_average = v_average / N - 1
    RETURN (v_average - bi.velocity)/10
END PROCEDURE
```

Сплоченность — правило, при котором каждый боид стремится быть в центре группы. Поскольку каждый боид имеет локализованное восприятие мира, то «центр стаи» на самом деле означает центр близлежащих соседей. Таким образом, боид стремится двигаться в направлении к центру масс близлежащих боидов. Если боид глубоко внутри стаи, то распределение внутри группы, примерно, однородно, т. е. плотность боидов, приблизительно, одинакова во всех направлениях. В этом случае, центр группы боидов находится примерно в центре стаи. Но если боид находится на границе группы, а соседние боиды располагаются рядом в другой части стаи, то центр масс соседей смещается в направлении к положению боида, находящегося на границе стаи. Здесь траектория полета будет отклоняться несколько в сторону локального центра стаи. Правильно реализованное правило сплоченности позволяет смоделированной стае раздваиваться, чтобы обходить препятствия — пока боид находится рядом с его ближайшими соседями, он не заботится, если остальная часть стаи меняет курс.

Пример. Вариант реализации сплоченности

```
PROCEDURE Rule3(BOID bi)
    Vector v = 0;
    FOR EACH BOID b
        IF b != bi THEN
            v += b.position
        END IF
        END
// N - количество боидов в группе
        v = v / N - 1
        RETURN v
END PROCEDURE
```

Модель «Boids» имеет прямой доступ к геометрической базе данных, которая описывает точное положение, ориентацию и скорость всех объектов в окружающей среде. Информация реальной птицы о мире строго ограничена, поскольку она воспринимает его через несовершенные чувства, а также в

связи с тем, что находящиеся поблизости соседи скрывают тех, кто находится дальше. Это фактор сильно выражен в пастушьих животных, потому что все они вынуждены находиться в одной и той же плоскости. В косяках рыб визуальное восприятие соседних рыб дополнительно ограничивается рассеиванием и поглощением света через воду между ними. Эти факторы в совокупности сильно локализуют информацию, доступную каждому объекту из группы.

На практике скорости, выданные каждым из правил, необходимо уменьшать (с помощью деления на какой-нибудь коэффициент) или ограничивать. Ограничения и коэффициенты подбираются экспериментальным путём для достижения наилучшего результата.

Модель «Boids» является примером *individual-based model* [10] — класса моделирования, используемого для захвата глобального поведения большого числа взаимодействующих автономных агентов.

Алгоритм К. Рейнольдса отлично подходит для симуляции «живых» агентов, когда допустимо «роевое» поведение группы. Несмотря на простоту лежащих в основе программы правил, конечная модель стаи, реализованная на компьютере, выглядела реалистично: птицы образовывали группы, пытались уйти от столкновений и хаотично бросались в разные стороны, повторяя поведение реальной стаи птиц.

В статье [11] Рейнольдс также отметил, что разработанная им модель поведения стаи птиц может быть дополнена введением дополнительных факторов, например, таких, как поиск пищи и/или уклонение от препятствия или боязнь хищников.

Глава 3. Решение поставленной задачи

Для решения задачи и наглядной демонстрации задействованы алгоритм Рейнольдса с дополнительными правилами, позволяющими расширить его на случай взаимодействия между мультиагентными группами, и игровой движок Unity3D для визуальной составляющей.

В модели существуют группы сущностей, которые взаимодействуют между собой и с окружающим пространством. Для решения рассматриваются два вида групп сущностей — «хищники» и «жертвы». Поведение каждой из этих групп моделируется с помощью набора правил поведения, включающего:

1. Базовые правила алгоритма Рейнольдса:
    - разделение;
    - сплоченность;
    - выравнивание.
2. Дополнительные правила для алгоритма Рейнольдса:
    - движение к цели;
    - управление скоростью;
    - обход препятствий.

Взаимодействие групп между собой также реализуется с помощью правил, которые применяются к группе, состоящей из объектов обеих групп. Для поведения системы «хищник-жертва» реализованы правила:

- преследование;
- обход врага.

Каждый объект группы имеет такие характеристики, как *дальнозоркость* и *собственное пространство*, влияющие на природу поведения как самого объекта, так и всей группы, к которой он принадлежит. В связи с чем, подбор значений этих характеристик является важной задачей, решение которой зависит от цели, преследуемой при моделировании определённой

группы объектов. Например, если у какого-либо объекта значение параметра «дальнозоркость» будет больше, чем у других, то в данном случае будет наблюдаться понятие *вожака стаи* — сущность, которая имеет достаточное влияние, чтобы вести за собой группу, — и сама группа изменяет природу поведения из-за данного изменения в параметре. Таким образом, вновь смоделированная группа будет отличаться от модели, где параметр «дальнозоркость» у всех объектов группы идентичен.

Для каждой группы сущностей выбирается определённый набор правил, характеризующий природу объектов. Правила применяются непосредственно к объектам группы и вычисляют векторы направления скорости движения — для каждого конкретного правила свой собственный вектор. После получения всех векторов направления вычисляется результирующий вектор скорости, который отвечает за перемещение объекта в пространстве. Он вычисляется путем суммирования полученных векторов направлений с соответствующими коэффициентами и после прибавляется к координатам объекта для расчета новой позиции. Коэффициенты выбираются опытным путем и отвечают за реалистичность поведения конкретной группы. Также, для реалистичности модели реализовано понятие «вожака стаи» — объект, радиус обзора которого позволяет раньше всех обнаружить отличные от группы сущности и, следовательно, вести за собой группу.

Данные, необходимые для перемещения групп объектов (векторы направлений, результирующая скорость, позиция объекта и т. д.), обновляются с периодом в 0,016 с, что является временем между кадрами в Unity3D.

3.1. Реализация базовых правил

Базовые правила, реализованные при построении данной модели, по своей сути совпадают с базовыми правилами алгоритма Рейнольдса, но имеют несколько изменений в реализации.

Правило «разделение»

Суть данного правила заключается в том, что если в собственное пространство объекта попадает другой объект группы, то объект меняет направление движения на противоположное относительно вторгшегося объекта.

Для реализации данного правила вводится параметр *space*, который отвечает за собственное пространство объекта, и значение которого идентифицируется заранее.

Пример. Псевдокод правила «разделения»

```
PROCEDURE Rule1 (boid bi)
    Vector v1 = 0;
    FOR EACH boid b
        IF b != bi AND |b.position - bi.position| < space THEN
             v1 += bi.position - b.position
        END IF
    END
    RETURN v1
END PROCEDURE
```

Правило «сплоченность»

Для того, чтобы несколько объектов отождествлялись с группой, необходимо, чтобы они двигались совместно, находясь на определённом расстоянии друг от друга. Данное правило Рейнольдс реализовал с использованием понятия *центра масс* — каждый объект стремится к центру всей группы, — в данной модели оно также используется.

Для реализации данного правила также используется параметр *space*, но в удвоенном значении, который отвечает за максимальное расстояние между объектами и идентифицируется заранее. Определяются все объекты, находящиеся в зоне досягаемости от рассматриваемого, вычисляется центр

масс данных объектов, после чего каждый из них начинает двигаться в сторону полученного центра.

Пример. Псевдокод правила «сплоченность»

```
PROCEDURE Rule2 (boid bi)
    Vector v2 = 0;
    Vector center = 0;
    FOR EACH boid b
        IF b != bi AND |b.position - bi.position| < 2*space THEN
                center += b.position
        END IF
    END
    center = center / N - 1
    v2 = center - bi.position;
   RETURN v2
END PROCEDURE
```

Правило «выравнивание»

Данное правило отвечает за то, чтобы группа визуально двигалась с одинаковой скоростью, как одно целое.

Для реализации вычисляется средняя скорость *соседей* — объектов, находящихся в зоне досягаемости. Для обозначения *зоны досягаемости* вводится соответствующий параметр *sight*, значение которого устанавливается заранее. После вычисления средней скорости, результирующий вектор направления движения сглаживается для более естественного эффекта. Коэффициент сглаживание выбирается опытным путем (при решении данной коэффициент равнялся 0.125).

Пример. Псевдокод правила «выравнивание»:

```
PROCEDURE Rule3 (boid bi)
    Vector v3 = 0;
    FOR EACH boid b
        IF b != bi AND |b.position - bi.position| < sight THEN
           v3 += b.velocity
        END IF
    END
    v3 = v3 / N-1
    RETURN (v3/N-1 - bi.velocity)* 0.125
END PROCEDURE
```

3.2. Реализация дополнительных правил

Правило «движение к цели»

В модель вводятся дополнительные целевые объекты (*food*), олицетворяющие «пищу». Данные объекты будут являться целью, которую нужно достичь одной из рассматриваемых групп.

В данном правиле оказывается влияние на вектор направления тех объектов, в *радиус обзора* которых попал целевой объект, которые в свою очередь влияют на природу поведения соседей, и, в итоге, вся стая осуществляет движение по направлению к целевому объекту. То есть если в поле видимости сущности попадает целевой объект, то вектор направления движения сущности смещается в сторону целевого объекта, и группа, к которой принадлежит сущность осуществляет движение по направлению к данному объекту. В качестве радиуса обзора в модели используется параметр *sight* — максимальное расстояние между объектами одной группы, — умноженный на 2. Данный параметр можно изменять в зависимости от целей, преследуемых при построении модели.

Пример. Псевдокод правила «движение к цели»

```
PROCEDURE Rule4
    Vector v4 = 0;
    FOR EACH boid b
        IF |b.position - food.position| < sight*2 THEN
            v4 += food.potion - b.position
        END IF
    END
    RETURN v4
END PROCEDURE
```

Правило «управление скоростью»

Так как в построенной модели присутствует две группы сущностей, то для того, чтобы их поведение было более естественным, скорости объектов-«хищников» и объектов-«жертв» отличаются и ограничиваются определённой величиной. Другими словами, каждая группа объектов ограничивается

заранее заданной скоростью, которая выбирается в зависимости от природы сущностей, олицетворяющей конкретную группу.

Чтобы различать группы друг от друга используется компонент игрового движка Unity3D — *тег*. Тег — маркер, который может использоваться для идентификации объектов в проекте, созданном в Unity3D [12]. Объекты группы «хищников» имеют тег «Enemy», а объекты группы «жертв» — тег «Prey».

Пример. Псевдокод правила «управление скоростью»:

```
PROCEDURE Rule5
    FOR EACH boid b
        IF b.TAG != Enemy THEN
            boid.v = ClampMagnitude(boid.v, 10)
        ELSE
            boid.v = ClampMagnitude (boid.v, 4)
        END IF
    END
END PROCEDURE
```

В построенной модели жертвы имеют преимущество в скорости перед хищниками.

Правило «обход препятствий»

В среде, где наблюдаются сущности, существуют статичные объекты-препятствия (*border*), при столкновении с которыми направление движения объектов групп меняется на противоположные относительны препятствия. Наличие тега «Object» у статичных объектов позволяет их отличить от динамических объектов различных групп.

Пример. Псевдокод правила «обход препятствий»:

```
PROCEDURE Rule6
    Vector v6 = 0;
    FOR EACH boid b
        IF |b.position - border.position| < 1 THEN
            v6 += border.position - b.position
        END IF
    END
    RETURN v6
```

```
END PROCEDURE
```

Правило «преследование»

Как было сказано ранее, в построенной модели присутствуют две группы сущностей — «хищники» и «жертвы». Естественный отбор гласит «чтобы выжить сильные пожирают слабых», данный момент отражен в работе посредством преследования объектов-«жертв» объектами-«хищниками».

Если в зону досягаемости объекта, помеченного как «хищник» (имеет тег «Enemy»), попадает объект из группы «жертв» (имеет тег «Prey»), то объект, а за ним и вся группа «хищников», начинает двигаться по направлению к «жертве», если же объект «жертва» выходит за пределы видимости объекта из группы «хищников», то преследование им прекращается. Зона досягаемости реализована с помощью параметра *sight*, рассмотренного ранее.

Пример. Псевдокод правила “преследование”:

```
PROCEDURE Rule7(BOID bi)
    Vector v7 = 0;
    BOID prey = NULL;
    FOR EACH BOID b
        distance = b.position - bi.position
        IF b.TAG != Enemy AND bi.TAG == Enemy
        AND |distance|< sight THEN
            prey = b
        ELSE IF b.TAG != Enemy AND bi.TAG == Enemy
        AND |distance| > sight THEN
            prey = NULL
        END IF
        END IF
        IF prey!= NULL AND bi.TAG == Enemy
            v7 += prey.position - bi.position
        END IF
    END
    RETURN v7
END PROCEDURE
```

Правило «обход врага»

Для реализации данного правила вводится параметр *sight_to_enemy*, отвечающий за поле зрение объекта-«жертва» по отношению к объектам-

«хищникам». Данный параметр идентифицируется заранее и может принимать различные значения в зависимости от поставленной задачи.

Если в поле зрения объекта «жертвы» попадает объект «хищник», то направление движение меняется в сторону, противоположную объекту «хищнику» до тех пор, пока «хищник» *видим* — т. е. находится в поле зрения «жертвы».

Пример. Псевдокод правила «побег»:

```
PROCEDURE Rule8(BOID bi)
    Vector v8 = 0;
    FOR EACH BOID b
        distance = b.position - bi.position
        IF b.TAG == Enemy AND bi.TAG != Enemy
        AND |distance|< sight_to_enemy THEN
            v8 += bi.position - b.position
        END IF
    END
    RETURN v8
END PROCEDURE
```

Результаты

В данном разделе представлена наглядные результаты реализация правил поведения, на которых строится модель поведения групп агентов в пределах поставленной в работе задачи.

Визуальная составляющая компьютерной модели была выполнена с помощью игрового движка Unity3D, а логика программной части написана на языке C#.

Правило «разделение»

На рис. 2 показано как работает правило «разделение» в модели: красной линией у объекта схематично изображён вектор направления движения. В качестве параметра *space* взято значение равное 4 в системе, объектами которой являются боиды-«жертвы», и равное 9 в системе с объектами-«хищниками». В момент, когда объекты находятся на определённом друг от друга расстоянии, вектор направления становится нуль-вектором (все линии сходятся в одной точке (0; 0; 0)).

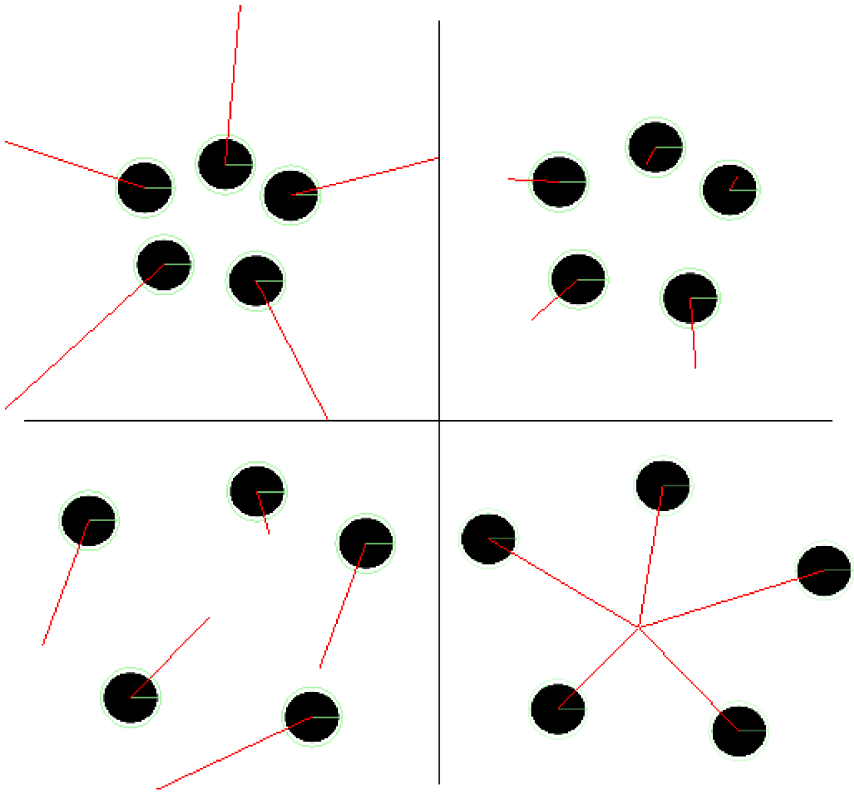

Рис. 2. Правило «разделения»

Правило «сплоченность»

Одно из правил, которое контролирует восприятие группы как единого целого, изображено на рис. 3. Здесь в качестве параметра взято удвоенное значение *space*, т. е. 8 у группы «жертв» и 18 у «хищников». С помощью серых линий изображен вектор направления движения объектов группы. Движение каждого объекта-боида осуществляется к центру масс, который вычисляется каждый кадр (0,016 с).

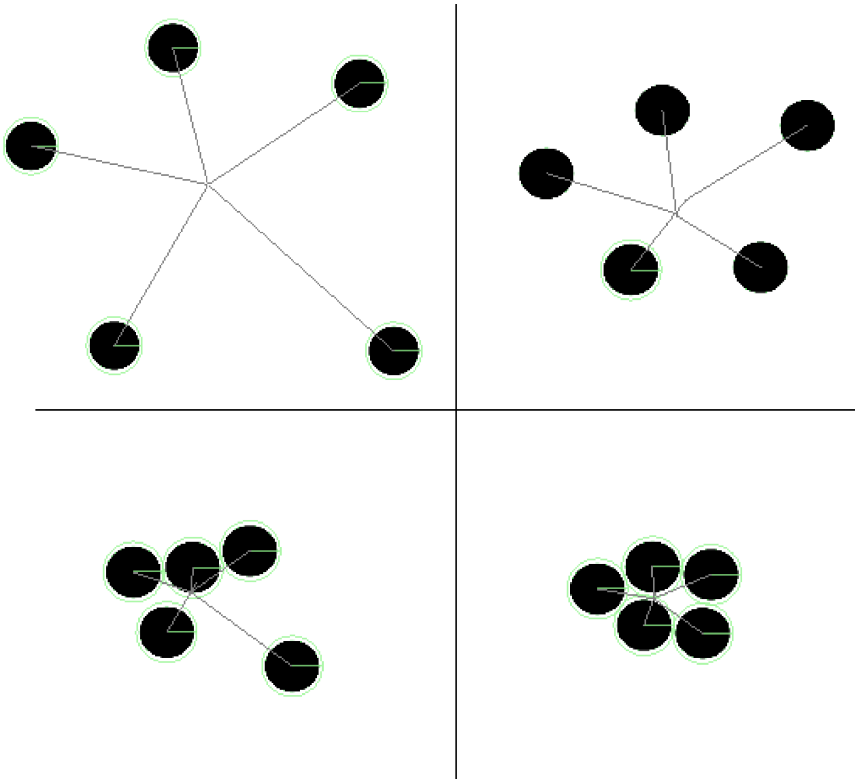

Рис. 3. Правило «сплоченность»

Комбинация правил «сплоченность» и «разделение»

Комбинирование правил «сплоченность» и «разделение» можно увидеть на рис. 4. Объекты, стремятся к центру масс группы, но при достижении определённого расстояние между друг другом прекращают свое стремление — включается в работу правило «разделения», вследствие чего, объекты избегают столкновения друг с другом.

В самом начале объекты находятся на приемлемом расстоянии, т. е. вектор направления движения, отвечающий за выполнение правила «разделение», является нуль-вектором. В то же время объекты стремятся к центру масс (серая линия) пока расстояние между ними это допускает.

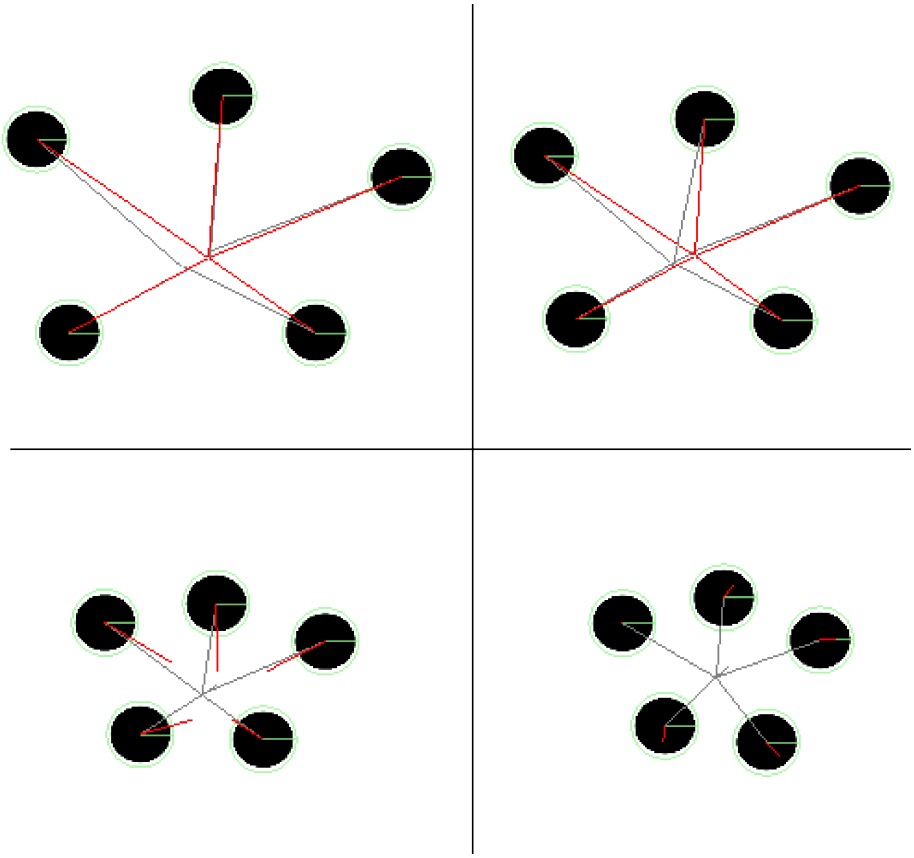

Рис. 4. Комбинация правил «разделение» и «сплоченность»

Правило «выравнивание»

Работа данного правила изображена на рис. 5. Черная линия есть направление движение объекта, а зеленые линии указывают на среднюю скорость группы. Объект считается частью группы, если он попадает в зону досягаемости, за которую отвечает параметр *sight* (дальнозоркость), значение которого равно 13 у группы «жертв» и 10 у группы «хищников». Преимущество у «жертв» взято в качестве примера — оно позволяет им предупредить появление «хищников» раньше, чем хищники обнаружат объекты из группы «жертв».

За счёт приближения черных линий к зелёным, осуществляется выравнивание скоростей.

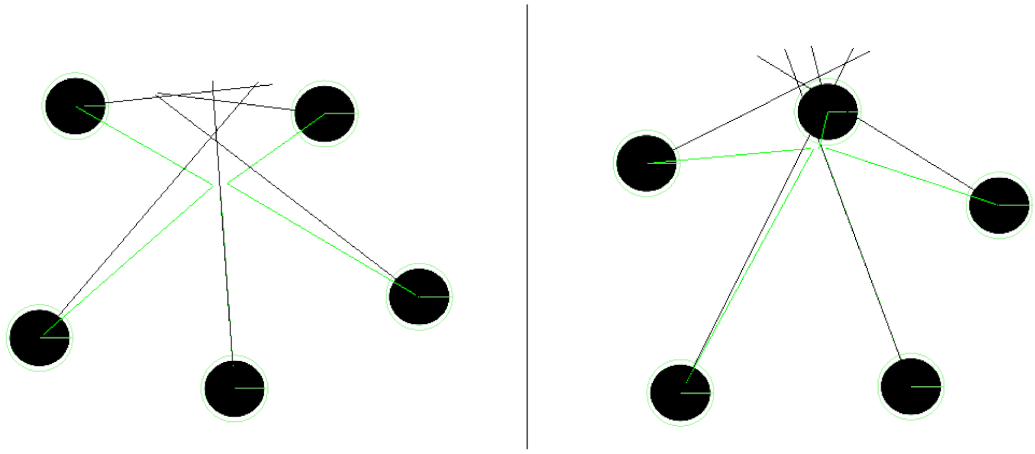

Рис. 5. Правило «выравнивание»

Правило «движение к цели»

Здесь важную роль играет параметр дальнозоркости (*sight*) — чем он больше, тем раньше можно обнаружить целевой объект. В данной работе для группы объектов-«жертв» выбрана пища (зелёные объекты). После её обнаружения каким-либо объектом, вся группа начинает свое движение по направлению к ней (зелёные линии). Это показано на рис. 6.

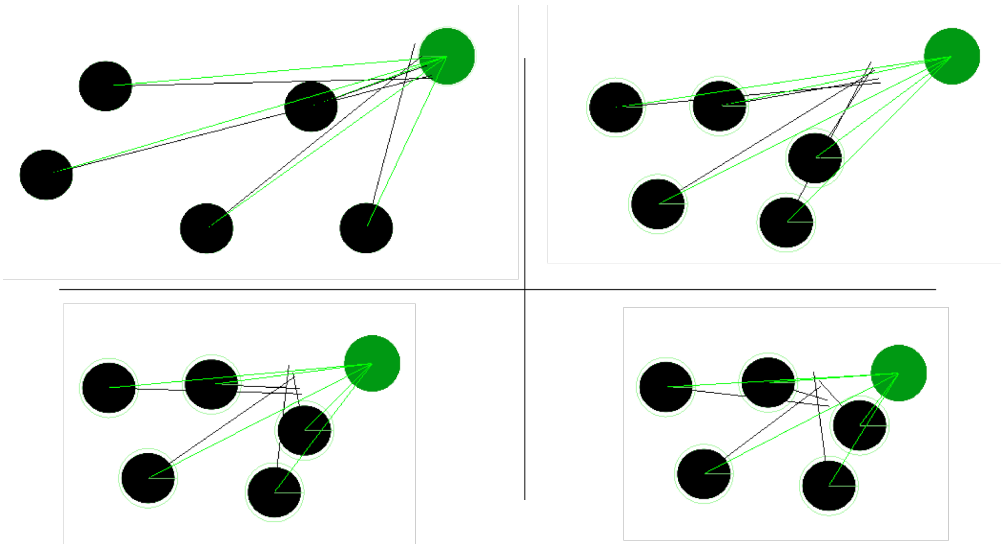

Рис. 6. Правило «движение к цели»

Правило «управление скоростью»

Каждая группа объектов ограничивается заранее заданной скоростью. Группа хищников ограничивается скоростью, равной 4, а группа «жертв» — 10. Группе «жертв» дано преимущество, чтобы позволить им избежать преследование со стороны «хищников».

Если не ограничивать скорость, то она будет очень быстро увеличиваться, что в свою очередь разрушает реалистичность модели поведения группы.

На рис. 7 изображена группа объектов-«жертв», у которой скорость не ограничивалась, а на рис. 8 та же группа, но с ограничением.

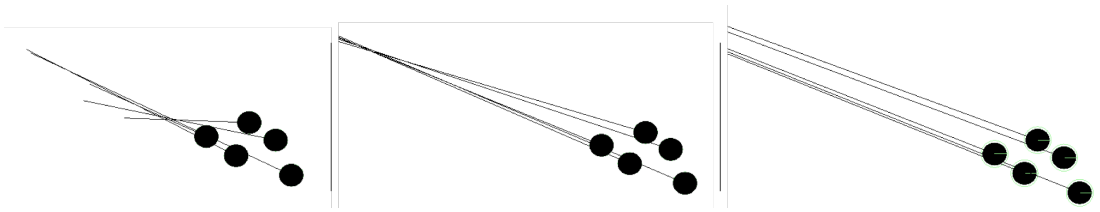

Рис. 7. Поведение группы «жертв» без ограничения скорости

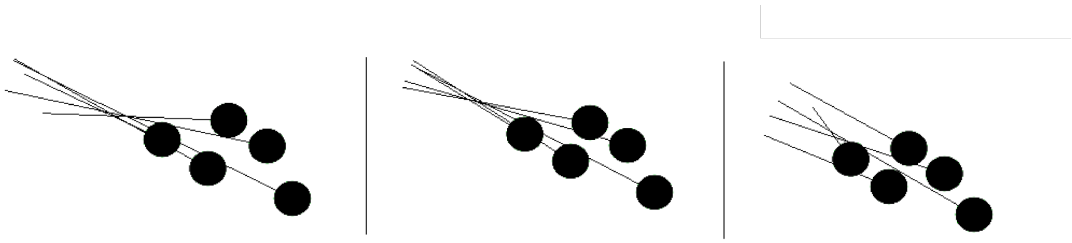

Рис. 8. Поведение группы «жертв» с ограничением скорости

Данное правило не дает в итоге вектор направления, для дальнейшего использования при вычислении результирующего вектора. Оно ограничивает сам результирующий вектор.

Правило «обход препятствий»

Для правила «обход препятствия» также нет вектора направления движения. Как и в правиле «ограничение скорости», обрабатывается сам результирующий вектор, который непосредственно меняет свое направление при

столкновении со статичными объектами. В качестве таких объектов в данной работе рассматриваются стены. По своей сути, в реализации оно похоже на правило «обход врага», что позволяет для его понимания использовать рис. 9.

Правило «обход врага»

Для реализации данного правила используется параметр *sight_to_enemy*, равный 2 и отвечающий за поле зрение объекта-«жертва» по отношению к объектам-«хищникам».

Как видно на рис. 9, если в поле зрения объекта «жертвы» (черный объект) попадает объект «хищник» (красный объект), то направление движение (серая линия) отклоняется в сторону, противоположную объекту «хищнику» до тех пор, пока «хищник» находится на расстоянии, равном параметру *sight_to_enemy*.

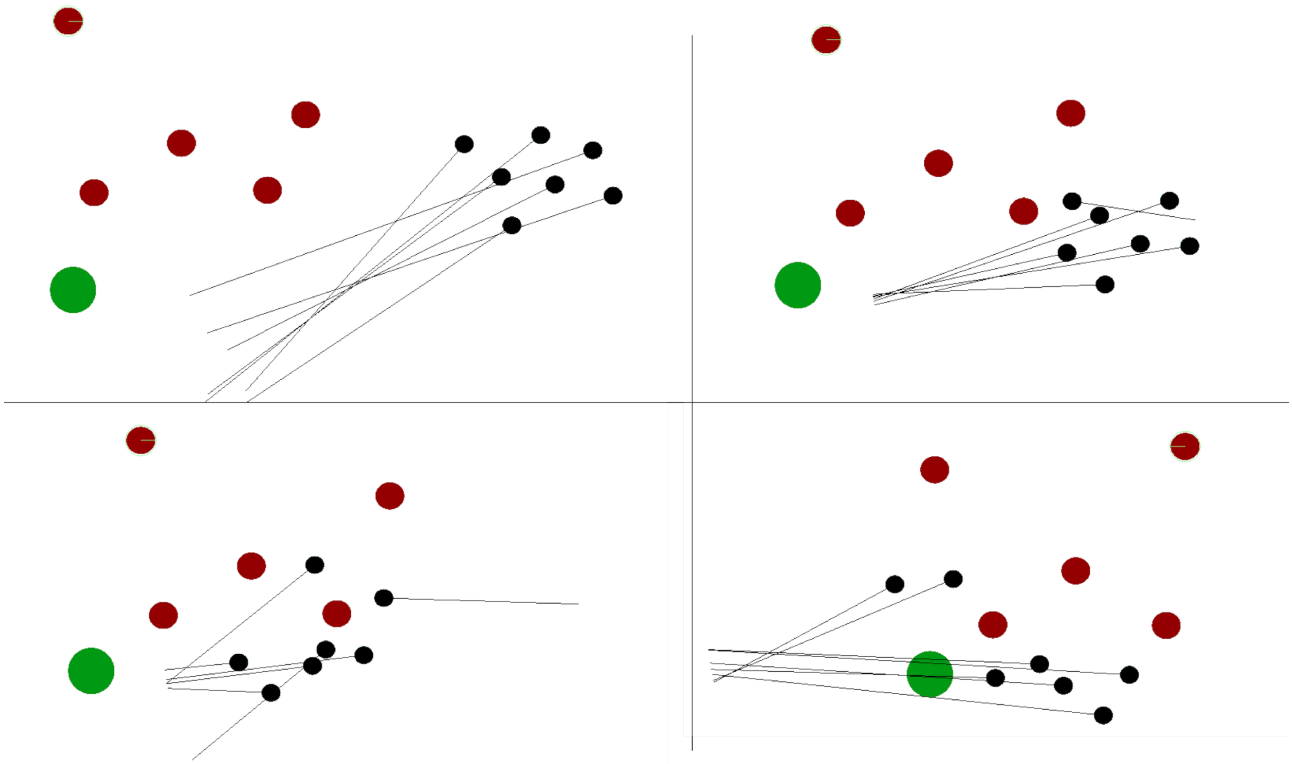

Рис. 9. Правило «обход врага»

Правило «преследование»

На рис. 10 продемонстрирован результат работы правила «преследование»: если в зону досягаемости объекта-«хищника», попадает объект-«жертва», то объект, а за ним и вся группа «хищников», начинает двигаться

по направлению к «жертве» (красные линии), если же объект «жертва» выходит за пределы видимости объекта из группы «хищников», то преследование им прекращается. Зона досягаемости реализована с помощью параметра *sight*, равного у группы хищников 10.

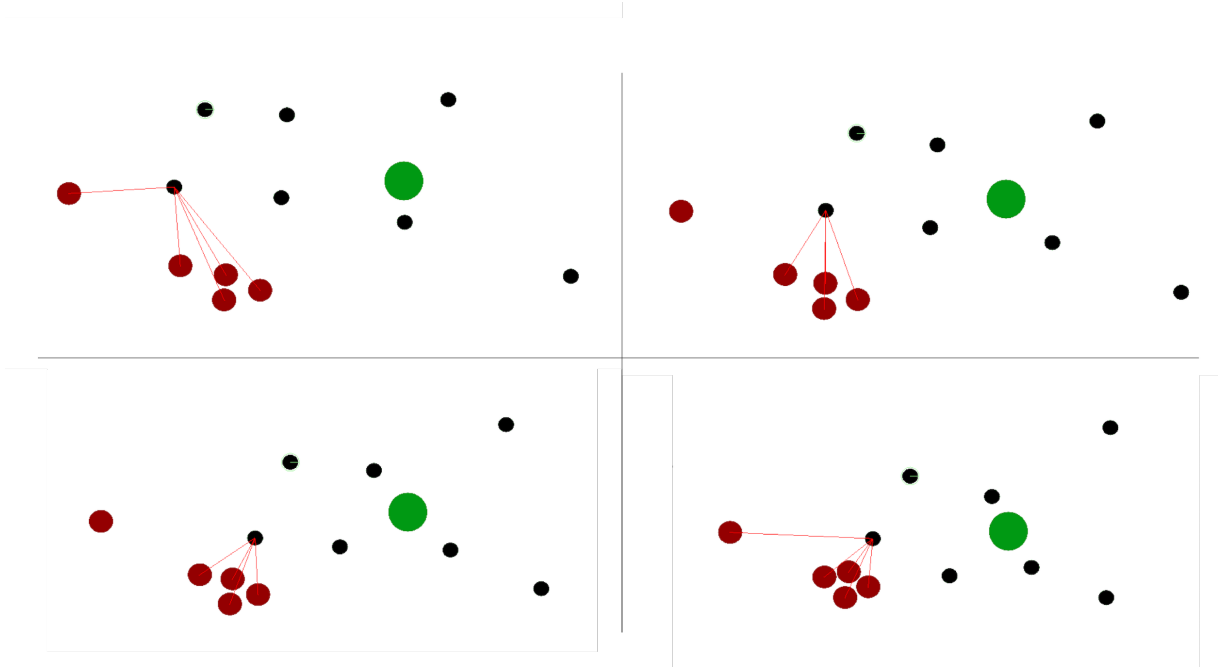

Рис. 10. Правило «преследование»

Так как параметр *space* у группы «хищников» довольно-таки большой, то они располагаются на значительном расстояние друг от друга. Этот факт косвенно влияет на реализации правила поведения — так как «хищники» держаться обособленно друг от друга, то они могут охотиться (преследовать «жертву») не только группой, но и поодиночке. Данный момент продемонстрирован на рис. 11.

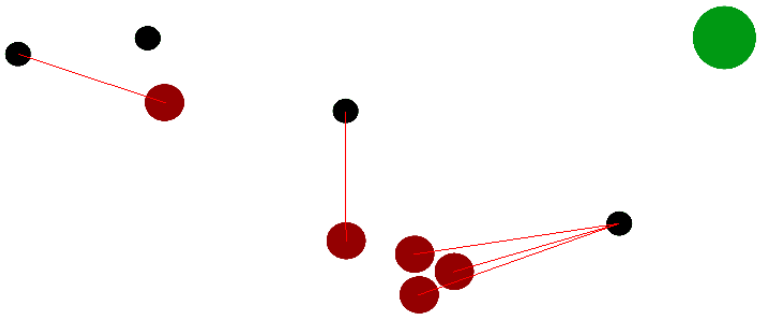

Рис. 11. Правило «преследования»: одиночное и групповое преследование

В последних правилах, где демонстрируются обе группы, помимо введения необходимых параметров, используются коэффициенты, на которые умножаются полученные после каждого правила векторы направления движения. Все коэффициенты подобраны опытным путем и использованы для придания большей реалистичности модели.

Конечное уравнение для результирующего вектора направления движения имеет вид:

```
v = 3*v1 + 0.5*v2 + 2*v3 + 5*v4 + 5*v5 + v7,
```

где v1...v7 векторы направления движения, полученные:

v1 — в результате правила «разделение»,

v2 — в результате правила «сплоченность»,

v3 — в результате правила «выравнивание»,

v4 — в результате правила «обход врага»,

v5 — в результате правила «преследование»,

v7 — в результате правила «движение к цели».

С кодом готового проекта, собранного по заявленной в работе задаче, можно ознакомится по ссылке [13].

Заключение

В рамках данной работы была решена задача по построению компьютерной модели поведения групп агентов, покрывающей реализацию как взаимодействия агентов внутри мультиагентной системы, так и взаимодействия мультиагентных систем между собой. При решении были задействованы алгоритм Рейнольдса, легший в основу построенной модели и игрового движок Unity3D, использовавшийся для компьютерного моделирования.

Полученная модель реалистично отображает взаимодействие как между объектами внутри конкретной группы (мультиагентной системы), так и между объектами различных мультиагентных систем. Модель в перспективе может быть использована для визуализации поведения между любым количеством систем и дополнена другими правилами, такими как:

- поиск укрытий;
- построение наиболее эффективного маршрута передвижения;
- разделение на несколько групп, для более эффективного поиска пищи и побега/преследования.

Человеческое общество можно также считать мультиагентной системой, что позволит в дальнейшем, при совершенствовании, расширить данную модель на случай взаимодействия нескольких групп людей между собой и применять для моделирование самых различных жизненных ситуаций.

Список литературы

1. Интеллектуальный агент. Персонализация и многоагентные системы. http://koriolan404.narod.ru/tipis/5.html (дата обращения: 14.03.17).
2. Бугайченко Д. Ю., Соловьев И. П. Абстрактная архитектура интеллектуального агента и методы ее реализации // Системное программирование. / Под ред. А. Н. Терехова, Д. Ю. Булычева. СПб.: СПбГУ, 2005. С. 36–67.
3. Роевой интеллект. https://ru.wikipedia.org/wiki/Роевой_интеллект (дата обращения: 14.03.17).
4. Boids: Background and Update. http: //www.red3d.com/cwr/boids/ (дата обращения: 12.03.17).
5. Многоагентная система. https://ru.wikipedia.org/wiki/Многоагентная_система (дата обращения: 12.03.17).
6. Рассел С., Норвиг П. Искусственный интеллект: современный подход. М.: Вильямс, 2007. 1408 с.
7. Boids. https://en.wikipedia.org/wiki/Boids (дата обращения: 12.03.17).
8. История и математика: мегаисторические аспекты / Под ред. Л. Е. Гринин, А. В. Коротаев. Волгоград: Учитель, 2016. 256 с.
9. Beni G., Wang J. Swarm Intelligence in Cellular Robotic Systems, Proceed // NATO Advanced Workshop on Robots and Biological Systems, 1989. Vol. 102, P. 703–712.
10. Individual-Based Models. http://www.red3d.com/cwr/ibm.html (дата обращения: 04.05.17)
11. Reynolds W. C. Flocks, herds, and schools: a distributed behavioral model // Computer Graphics / Ed. by Maureen C. Stone. Anaheim, California: SIGGRAPH, 1987. P. 25–34.

12. Теги и слои. https://docs.unity3d.com/ru/530/Manual/classTagManager.html (дата обращения: 28.04.17)
13. Код программы данной работы. https://github.com/dvoryanchikova/01.git