УДК 004.85:336.745

А. А. Кадетова

ПРОТОТИП МОБИЛЬНОЙ ИГРЫ В ЖАНРЕ ЭКШЕН С ИСПОЛЬЗОВАНИЕМ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА ДЛЯ УПРАВЛЕНИЯ ПЕРСОНАЖЕМ

Статья посвящена описанию созданного прототипа мобильной игры жанра экшен с использованием искусственного интеллекта, обучаемого на основе генетического алгоритма управления персонажем, на языке программирования Lua в платформе Corona SDK. Приведены разработанные проектные диаграммы приложения в нотации UML: вариантов использования, классов, состояний объектов и популяции. Кратко описан игровой процесс.

Работа посвящена описанию созданного прототипа мобильной игры на Corona SDK в жанре экшен с использованием искусственного интеллекта для управления персонажем. Разработка является продолжением освоения тематики нейронных сетей, описанного в [1].

Крупнейшим рынком в игровой индустрии являются мобильные игры. Широкое развитие за последнее десятилетие получили разные виды нейронных сетей (НС) и методы их обучения. Для описания и обучения небольших НС удобно использовать встраиваемые языки программирования, например, Lua. Они упрощают и ускоряют разработку ПО, а также обеспечивают быстрое и эффективное обучения НС. Существенно облегчающая разработку мобильных приложений на языке Lua платформа Corona SDK использует все его преимущества: плотную интеграцию с языками C/C++; динамическую типизацию; большой набор библиотек расширений.

В процессе проектирования приложения разработаны диаграммы в нотации UML: вариантов использования, классов, состояний объектов и популяции. Диаграмма вариантов использования, приведенная на рисунке 1, иллюстрирует отношения между актёрами и вариантами использования: в проекте один актёр (Игрок) и несколько вариантов использования (просмотр информации, повышение уровня, выбор типа игры).

Как показано на рисунке 1, имеется три типа игры: «Batte!» (здесь игрок управляет персонажем, и его задача победить всех врагов на уровне, увеличив счетчик побед), «AvtoBattle!» (здесь НС управляет персонажем, и ее задача – победить всех врагов на уровне, увеличив счетчик побед), «Train!» (здесь НС учится управлять персонажем, и ее задача – тренировать НС при помощи генетического алгоритма сражаться с врагом).

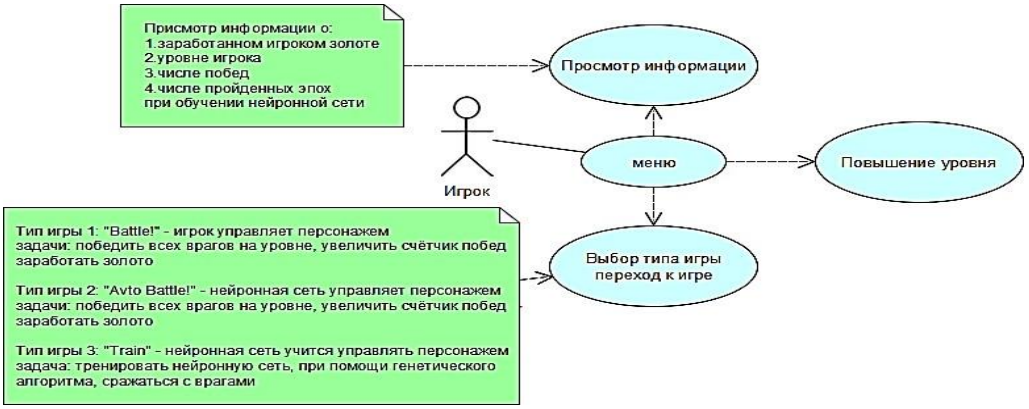

Рисунок 1 – Диаграмма вариантов использования

Диаграмма классов UML, приведенная на рисунке 2, иллюстрирует логическую структуру системы, описывая классы, их атрибуты, операции и отношения между объектами. В проекте присутствуют следующие классы: популяция, нейронная сеть, сцена игры, объект, объект-враг, объект-лезвие, объект-персонаж.

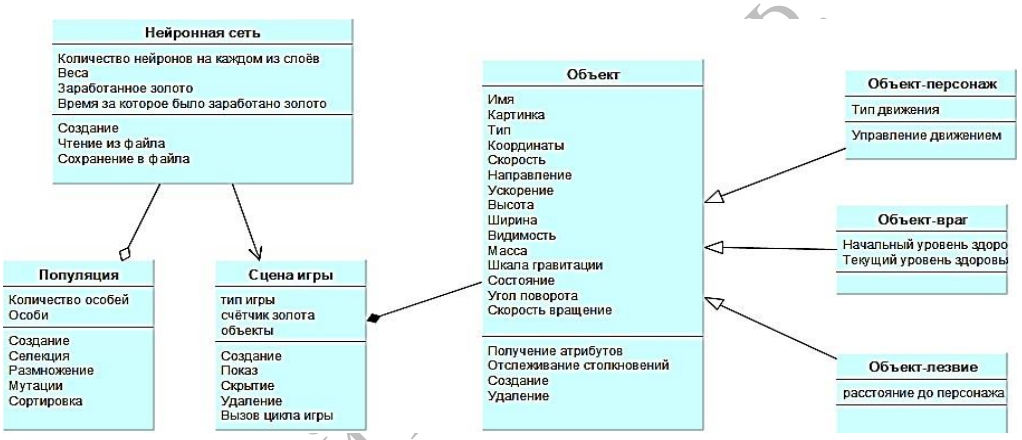

Рисунок 2 – Диаграмма классов

Диаграмма состояний показывает, как объект переходит из одного состояния в другое, и полезна также при моделировании жизненного цикла объекта. Диаграмма состояний описывает процесс изменения состояний только одного экземпляра определенного класса. В приложении такие сущности, как объект, объект-враг, объект-лезвие, объект-персонаж, имеют 3 состояния: «Новый», «Существует», «Не существует». Жизненный цикл этих сущностей можно увидеть на рисунке 3.

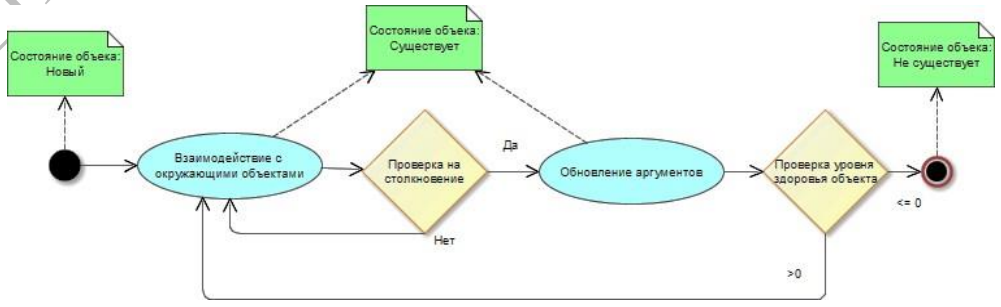

Рисунок 3 – Жизненный цикл объекта

Объектпопуляция также может принимать различные состояния. Диаграмму состояний популяции можно увидеть на рисунке 4.

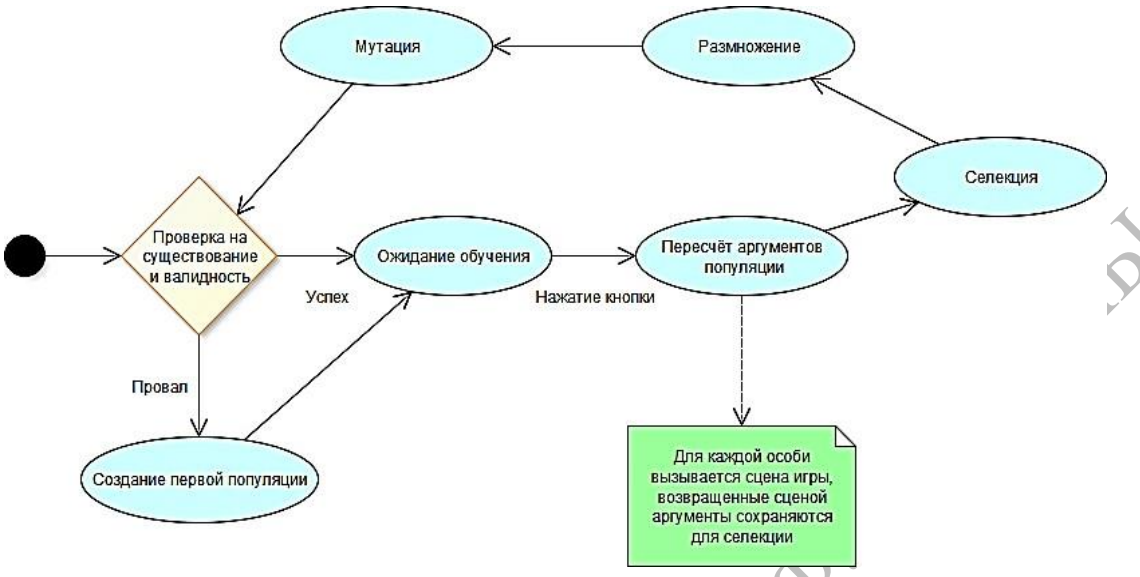

Рисунок 4 – Жизненный цикл популяции

Для удобства разработки и последующего обслуживания проект приложения хранится в файлах Main.lua; Game.lua; Config.lua; Menu.lua. Файл Config.lua содержит параметры приложения; Main.lua – код приложения, из него вызывается основная сцена игры, описание которой содержится в файле Menu.lua. Эта сцена отображает информацию об игроке и позволяет запустить игру. При запуске игры вызывается сцена, хранящаяся в файле Game.lua, в ней происходит игровой процесс, заключающийся в увеличении значения счётчика побед на единицу при прохождении на более высокий уровень игры.

Искусственный интеллект в приложении используется для управления персонажем. С помощью генетического алгоритма НС [2] обучается управлять персонажем. На вход НС поступают координаты ближайших врагов, выходом НС является направление движения точкой (x, y), где x и y принадлежат отрезку [-1;1]. НС имеет произвольную архитектуру: количество скрытых слоев (countHideLayers), количество нейронов на входном слое (coutInputNeuron) и количество нейронов на скрытых слоях (coutHideNeuron) задаётся параметрически в сцене Menu. В качестве функции активации используется гиперболический тангенс. Выход НС используется для движения персонажа, движение происходит дискретным образом каждые 50 миллисекунд. Генами для генетического алгоритма являются веса НС, в гене содержатся два параметра x и y, при создании первой популяции x и y имеют нормальное распределение на отрезке [-5,5]. Хромосомой для генетического алгоритма является набор весов нейронной сети. Количество особей в популяции (countIndivid) также задается в сцене Menu. Игровой процесс заключается в увеличении значения счётчика побед. Значение на счётчике увеличивается на единицу каждый раз при прохождении игроком уровня игры. На каждом из них присутствует 50 врагов, уровень жизни которых имеет равномерное распределение на промежутке $[2; 6+2*m]$, где m – число побед, что постепенно увеличивает сложность игры. Для победы над врагом необходимо снизить его уровень жизни до 0. При победе над всеми врагами уровень считается пройденным. Изображение врага, меча, персонажа можно увидеть на рисунках 5, 6, 7.

Рисунок 5 – Иконка врага

Рисунок 6 – Иконка меча

Рисунок 7 – Иконка персонажа

Уровень жизни врага снижается при столкновении с одним из мечей. Мечи вращаются вокруг персонажа на некотором расстоянии, которое зависит от уровня персонажа. Скорость вращения этих мечей также зависит от уровня игрока. Наносимый мечом урон начинается с 1 и увеличивается на 1 за каждые 4 уровня игрока. Уровень игрока можно повысить, тратя золото. Золото выпадает после победы над врагом и его число равно уровню жизни врага. Цена за повышение уровня игрока вычисляется

по формуле: $P = -75 \cdot L + 200 \cdot 7^L / 5$, где: P – цена; L – уровень игрока.

Для запуска игры на Android необходимо установить собранный файл, после чего игра полностью готова к использованию. Если игра собирается под Windows, то достаточно запустить exe-файл, а установка дополнительного окружения не требуется. После первого запуска открывается меню игры с информацией: количество золота и число побед равны 0, уровень персонажа равен 1, для поднятия уровня персонажа необходимо 205 единиц золота, в обучении было пройдено 0 эпох. Персонаж на экране управляется пальцем, враги двигаются к персонажу с 3-х сторон экрана. Также на экране изображён счётчик золота, заработанного персонажем на данном уровне игры. При столкновении врага с персонажем уровень считается проваленным игрок получает уведомление о проигрыше. При уничтожении всех врагов уровень игры считается пройденным, и игроку демонстрируется уведомление о победе. После чего осуществляется переход в меню игры.

При нажатии на «Train!» начинается обучение нейронной сети. Персонаж на экране управляется согласно выходу нейронной сети. Счётчик золота на экране показывает количество заработанного персонажем золота, счётчик особей – номер текущей особи в популяции. При столкновении врага с персонажем уровень считается проваленным, игроку показывается уведомление о проигрыше. Заработанное при обучении золото не сохраняется. При нажатии на «AvtoBattle!» аналогично запускается игра, но с автоматическим управлением. Для автоматического управления при уведомлении о проигрыше заработанное золото сохраняется. После чего следует переход в меню игры. При победе над врагом счётчик золота увеличивается на первоначальный уровень здоровья врага. Полученная игра была протестирована. Прогресс в обучении нейронной сети виден после 20-й эпохи, а первые победы при автоматическом управлении – на 100-й эпохе.

Литература

1 Кадетова, А. А. Реализация однослойного персептрона на языке Python / А. А. Кадетова, Н. Б. Осипенко // Новые математические методы и компьютерные технологии в проектировании, производстве и научных исследованиях : материалы XXIII Республиканской научной конференции студентов и аспирантов, Гомель, 23– 25 марта 2020 г. / ГГУ им. Ф. Скорины ; редкол.: С. П. Жогаль (гл. ред.) [и др.]. – Гомель : ГГУ имени Ф. Скорины, 2020. – С. 365–366.

2 Николенко, С. Глубокое обучение / С. Николенко, А. Кадурин, Е. Архангельская. – Санкт-Петербург : Питер, 2018. – 480 с.