УДК 004.021

АЛГОРИТМ ПРЕСЛЕДОВАНИЯ ИГРОКА

В. Г. КРАСИЛЬНИКОВ, А. П. ПРЕСНУХИН,
А. А. ДУНАЕВА, канд. техн. наук, доц. Д. В. КОЧКИН
(Вологодский государственный университет, Россия)

Аннотация. *В данной статье содержится материал об алгоритме преследования игрока враждебным юнитом на языке программирования GDScript.*

Ключевые слова: *Godot Engine, GDScript, алгоритм, преследование, chase.*

Введение. Метод преследования в играх обычно означает стратегию, когда игровой персонаж или игрок занимается поиском и преследованием другого объекта, игрока или цели. Этот метод обычно применяется в шутерах, экшен-играх или играх про выживание, в частности Roguelike, где игроки должны находить и преследовать своих противников или наоборот для достижения цели или победы, что и будет рассмотрено далее.

Основная часть. По словам Майка Дейлли из YoYo Games, игрокам интересны простые концепции и незамысловатый геймплей, что характеризует большинство видеоигр с 2D-графикой [1]. По этой причине популярность жанра Roguelike вновь начинает расти.

Roguelike-игры отличаются от других случайной генерацией уровней и перманентной смертью персонажа с полной потерей прогресса. Но у данного жанра есть и другие, свойственные только ему, черты, которые прописаны в так называемой Берлинской Интерпретации [2]. В подобных играх вражеские юниты (боевые единицы) случайным образом появляются в «комнате» подземелья и преследуют свою цель – уничтожение игрока. Монстры следуют заложенному в них разработчиком алгоритму, продвигаясь к игроку и огибая встречающиеся препятствия. Проанализируем метод преследования на примере Godot Engine.

Godot Engine – это универсальный 2D и 3D игровой движок, спроектированный для поддержки всех видов проектов. Годо опирается на парадигму объектно-ориентированного программирования. Понимание таких понятий, как классы и объекты, поможет эффективно кодировать. Разработка игр осуществляется при помощи специально созданного и встроенного в Godot языка программирования с лёгким синтаксисом – GDScript [3].

GDScript – это высокоуровневый, объектно-ориентированный, императивный язык программирования с последовательной типизацией, созданный для Godot. Он использует синтаксис на основе отступов, аналогичный такому языку, как Python. Его цель – оптимизировать и тесно интегрировать с Godot Engine, обеспечивая большую гибкость при создании контента.

Перейдём к реализации алгоритма преследования. Для начала необходимо задать ориентацию к следующей точке пути и, при необходимости, перевернуть спрайт, чтобы корректно отобразить движение, в зависимости от его направления. Данные действия будут отражены в функциях chase() и update_sprite_orientation(), представленных на рисунке 1. В первой функции проверяется, если объект не достиг своей цели, то переменная vector_to_next_point получает направление движения к следующей точке пути, вычитая из позиции агента его текущее глобальное положение; mov_direction устанавливает направление движения к следующей точке, после чего вызывает следующая функция update_sprite_orientation(). В её теле проверяется ориентация в пространстве и, в зависимости от направления движения, переворачивается влево или вправо для корректного отображения движения (объект повёрнут лицом к цели).

```
func chase() -> void:
    if not navigation_agent.is_target_reached():
        var vector_to_next_point: Vector2 = navigation_agent.get_next_path_position() - global_position
        mov_direction = vector_to_next_point

        update_sprite_orientation(vector_to_next_point.x)

func update_sprite_orientation(x_direction: float) -> void:
    if x_direction > 0 and not animated_sprite.flip_h:
        animated_sprite.flip_h = true
    elif x_direction < 0 and animated_sprite.flip_h:
        animated_sprite.flip_h = false
```

Рисунок 1. – Функции chase() и update_sprite_orientation()

Следующим шагом реализации становятся функции _on_PathTimer_timeout() и _get_path_to_player(), представленные на рисунке 2. В теле первой функции проверяется, существует ли объект player(игрок). Если да, то вызывается функция _get_path_to_player(), которая устанавливает позицию игрока в качестве целевой позиции для перемещения к нему, иначе таймер останавливается и движение прекращается (устанавливается в нулевое направление – Vector2.ZERO).

```
func _on_PathTimer_timeout() -> void:
    if is_instance_valid(player):
        _get_path_to_player()
    else:
        path_timer.stop()
        mov_direction = Vector2.ZERO


func _get_path_to_player() -> void:
    navigation_agent.target_position = player.position
```

Рисунок 2. – Функции _on_PathTimer_timeout() и _get_path_to_player()

Вывод. Язык GDScript схож с Python, поэтому лёгок в освоении и позволяет быстро разрабатывать программные продукты. Исходя из этого, видно, что создать алгоритм преследования на движке Годо довольно просто.

ЛИТЕРАТУРА

1. Плоскость восприятия: плюсы и минусы создания 2D-игр [Электронный ресурс]. – Режим доступа: https://dtf.ru/gamedev/32593-ploskost-vospriyatiya-plyusy-i-minusy-sozdaniya-2d-igr. – Дата доступа: 09.03.2024.
2. Roguelike-игры: история возникновения, общая характеристика, секреты популярности [Электронный ресурс]. – Режим доступа: https://gb.ru/blog/roguelike-igry/. – Дата доступа: 09.03.2024.
3. Документация Godot – ветка 4.2 [Электронный ресурс]. – Режим доступа: https://docs.godotengine.org/ru/4.x/index.html. – Дата доступа: 09.03.2024.