СБОР И АНАЛИЗ ДАННЫХ, НЕОБХОДИМЫХ ДЛЯ ОБУЧЕНИЯ ИИ NPC В ШУТЕРЕ ОТ ПЕРВОГО ЛИЦА НА ОСНОВЕ ДЕЙСТВИЙ ИГРОКА

Камилов Г.Т.

Камилов Глеб Тимурович – магистрант,
Донской государственный технический университет,
г. Ростов-на-Дону

Аннотация: *в статье анализируются разработка модели способной предсказывать действия игрока и адаптировать поведение NPC в реальном времени.*
Ключевые слова: *искусственный интеллект, разработка игр, программирование, машинное обучение.*

COLLECT AND ANALYZE THE DATA NEEDED TO TRAIN NPC AI IN A PLAYER-BASED FIRST-PERSON SHOOTER

Kamilov G.T.

Kamilov Gleb Timurovich– Master's student,
DON STATE TECHNICAL UNIVERSITY,
ROSTOV-ON-DON

Abstract: *the article analyzes developing a model that can predict player actions and adapt NPC behavior in real time.*
Keywords: *artificial intelligence, game development, programming, machine learning.*

УДК 331.225.3

Сбор данных о действиях игроков является ключевым этапом в разработке адаптивного ИИ для NPC в играх. На этом этапе важно не только собрать данные, но и выбрать подходящий метод их регистрации и хранения. В этой главе рассматриваются основные методы, используемые для сбора данных, а также их особенности и подходы к интеграции в игровую среду.

Самым простым и часто используемым методом является сбор данных в реальном времени, который позволяет отслеживать действия игрока во время игры. Такие данные могут быть собраны непосредственно в игровом процессе через встроенные скрипты и механизмы игрового движка. Преимуществом такого подхода является его оперативность — данные о действиях игрока записываются сразу, без необходимости обработки предварительных отчетов или логов.

Для реализации этого подхода могут использоваться следующие инструменты:

Скрипты на языке программирования: в случае использования игрового движка Unity или Unreal Engine для сбора данных можно создавать скрипты на C# или C++, которые будут автоматически записывать действия игрока в файл или отправлять данные на сервер для дальнейшей обработки. Например, можно отслеживать перемещения игрока, его взаимодействие с объектами, использование оружия и тому подобное.

В этом примере скрипт записывает информацию о действиях игрока в CSV-файл, включая время действия, тип действия и позицию игрока. Это помогает создавать подробные записи о том, как игрок взаимодействует с окружающим миром.

```
using UnityEngine;
using System.IO;

public class PlayerDataCollector : MonoBehaviour
{
    // Метод для записи данных
    void LogData(string action, Vector3 position)
    {
        string logEntry = Time.time + "," + action + "," + position.ToString();
        File.AppendAllText("PlayerActions.csv", logEntry + "\n");
    }

    void Update()
    {
        // Пример: запись перемещения игрока вперед
        if (Input.GetKeyDown(KeyCode.W))
        {
            LogData("Move Forward", transform.position);
        }

        // Пример: взаимодействие с объектом (например, открытие двери)
        if (Input.GetKeyDown(KeyCode.E))
        {
            LogData("Interact with Door", transform.position);
        }
    }
}
```

Рис. 1. Пример сбора данных о перемещении и взаимодействии с объектами в Unity.

Игровые логи могут быть важным инструментом для сбора данных. Логи, которые генерируются игровыми движками, содержат записи о ключевых событиях в игре, таких как события, произошедшие с персонажем, реакции NPC, а также другие действия игрока и изменения в игровом мире.

Пример использования логов:

1. Логи ошибок и событий: с помощью логирования можно отслеживать события, которые происходят во время игры, такие как атаки, убийства, использование способностей, переходы между уровнями.
2. Аналитические логи: применяются для отслеживания более подробных метрик, таких как время нахождения игрока в определенной области карты, частота использования оружия или тактические предпочтения.

Этот скрипт записывает события в лог-файл, что позволяет отслеживать ключевые моменты игры и поведение игрока.

```
using UnityEngine;
using System.IO;

public class GameLogger : MonoBehaviour
{
    void LogEvent(string eventType, string message)
    {
        string logMessage = Time.time + " - " + eventType + ": " + message;
        File.AppendAllText("GameLog.txt", logMessage + "\n");
    }

    void Start()
    {
        LogEvent("GameStart", "The game has started!");
    }

    void Update()
    {
        if (Input.GetKeyDown(KeyCode.F))
        {
            LogEvent("PlayerAction", "Player used 'Fire' action");
        }
    }
}
```

Рис. 2. Пример записи в игровой лог.

В дополнение к стандартным методам сбора данных, в некоторых случаях могут использоваться сенсоры и дополнительные устройства для отслеживания поведения игрока. Например, для игр с виртуальной реальностью (VR) или дополненной реальностью (AR) можно использовать датчики движения, камеры и сенсоры, которые фиксируют физические действия игрока.

Примером может служить использование гладких поверхностей и жестов для игр в VR, где действия игрока, такие как движения рук или голова, могут быть собраны и проанализированы.

После сбора данных важно правильно их обработать, чтобы они стали пригодными для обучения моделей ИИ. Это этап, на котором данные очищаются от шума, нормализуются и приводятся к нужному формату для последующего анализа. Эффективная предварительная обработка данных позволяет улучшить качество моделей и ускорить процесс обучения.

На этом этапе удаляются дубликаты, ошибки и пропуски в данных. Например, если в процессе сбора данных о перемещениях игрока возникают ошибки в координатах или записи о действиях с пропусками, их необходимо удалить или заполнить.

В этом примере данные очищаются от пустых значений и дубликатов, что помогает улучшить качество последующего анализа.

```python
import pandas as pd

# Загрузка данных
data = pd.read_csv('PlayerActions.csv')

# Удаление строк с пропущенными значениями
clean_data = data.dropna()

# Удаление дубликатов
clean_data = clean_data.drop_duplicates()

# Сохранение очищенных данных
clean_data.to_csv('CleanedPlayerActions.csv', index=False)
```

Рис. 3. Пример очистки данных в Python.

Нормализация данных — это процесс приведения значений признаков (например, координат или времени действия) к единому масштабу. Это помогает улучшить результаты обучения, особенно в моделях машинного обучения, чувствительных к масштабу и типу данных.

Этот пример показывает, как можно привести данные о позициях игрока к единому масштабу, что улучшает обучение ИИ.

```python
import pandas as pd
from sklearn.preprocessing import MinMaxScaler

# Пример данных с разными масштабами
data = pd.DataFrame({
    'position_x': [10, 20, 30, 40],
    'position_y': [5, 15, 25, 35]
})

scaler = MinMaxScaler()
normalized_data = scaler.fit_transform(data)

# Преобразование обратно в DataFrame
normalized_df = pd.DataFrame(normalized_data, columns=data.columns)
print(normalized_df)
```

Рис. 4. Пример нормализации данных с использованием Python.

Данные о действиях игрока могут содержать категориальные признаки, такие как тип оружия, действия с объектами или использование определенных способностей. Чтобы такие данные могли быть использованы в моделях машинного обучения, их необходимо закодировать в числовой формат.

Этот код преобразует текстовые данные о действиях игрока в числовые значения, которые могут быть использованы в модели.

```python
import pandas as pd
from sklearn.preprocessing import LabelEncoder

# Пример категориальных данных
data = pd.DataFrame({
    'action': ['Move Forward', 'Jump', 'Shoot', 'Crouch']
})

encoder = LabelEncoder()
encoded_data = encoder.fit_transform(data['action'])

# Результат
print(encoded_data)
```

Рис. 5. Пример кодирования категориальных данных в Python.

Важной частью сбора данных является анализ взаимодействий игрока с NPC. Эти взаимодействия могут включать боевые действия, тактические маневры, выбор оружия или предметов, а также стратегии уклонения и нападения. Сбор таких данных позволяет ИИ NPC адаптироваться к поведению игрока и предсказать его действия, что делает игру более динамичной и интересной.

Для анализа взаимодействий с NPC можно использовать те же методы сбора данных, что и для других действий игрока, но с учетом специфики поведения NPC, таких как реакция на атакующие действия или уклонение от угроз.

Список литературы / References

1. Понимание сбора данных в играх и его влияние на искусственный интеллект // Gamasutra [Электронный ресурс]. — URL: https://www.gamasutra.com/blogs/JamesOlsen/20190502/344350/Understanding_Gameplay_Data_Collection_and_Its_Impact_on_Artificial_Intelligence.php (дата обращения: 07.11.2024).
2. Обучение ИИ с использованием алгоритмов обучения с подкреплением для адаптивных NPC // Arxiv [Электронный ресурс]. — URL: https://arxiv.org/abs/1909.07810 (дата обращения: 24.11.2024).
3. *Мейер С.В.* Применение машинного обучения для создания адаптивных NPC в видеоиграх / С.В. Мейер. — М.: Энергоатомиздат, 2020. — 212 с.