ВИКОРИСТАННЯ СПЛАЙНІВ ДЛЯ ПОБУДОВИ ТРАЕКТОРІЇ РУХУ ТРАНСПОРТУ В ІГРОВИХ ДОДАТКАХ UNREAL ENGINE

Пупченко О.О., магістрант,
Цололо С.О., к.т.н., доц., sergii.tsololo@donnto.edu.ua
Донецький національний технічний університет, м Покровск, Україна

У роботі пропонується підхід, заснований на використанні сплайнів для реалізації механізму пересування колісного транспорту (автомобіль) в ігрових додатках на Unreal Engine 4 (UE4). Сплайни в UE4 фактично є кривими Безьє, а розглянутий підхід називається Pure pursuit (пряме переслідування). Він використовується як один з варіантів для управління безпілотними автомобілями.

Існуючі рішення задачі пересування автомобіля на основі сплайнів [2] можуть бути вдосконалені і доповнені. Так, в підході, що пропонується в роботі, роль сплайна буде не тільки у вказівці маршруту, а й в переміщенні двох точок прив'язки (кубів), які впливають на параметри руху транспорту. Для цього в клас сплайна додаються два куба: перший буде регулювати швидкість автомобіля, другий – поворот коліс.

Ці куби рухаються за сплайном і віддаляються від автомобіля на певну дистанцію, яка в подальшому буде задаватися розробником. Так, якщо спочатку об'єкт і обидва куба рухалися по прямій один за одним, то з появою повороту їх положення відносно один одного змінюється (рис. 1, зеленим відображаються вектори на основі кутів між поточним становищем автомобіля і кубами).

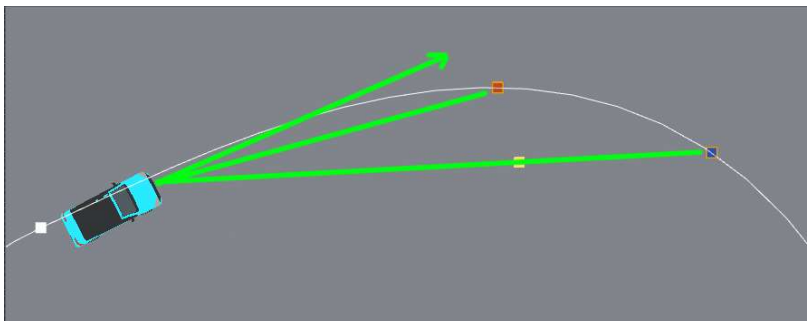


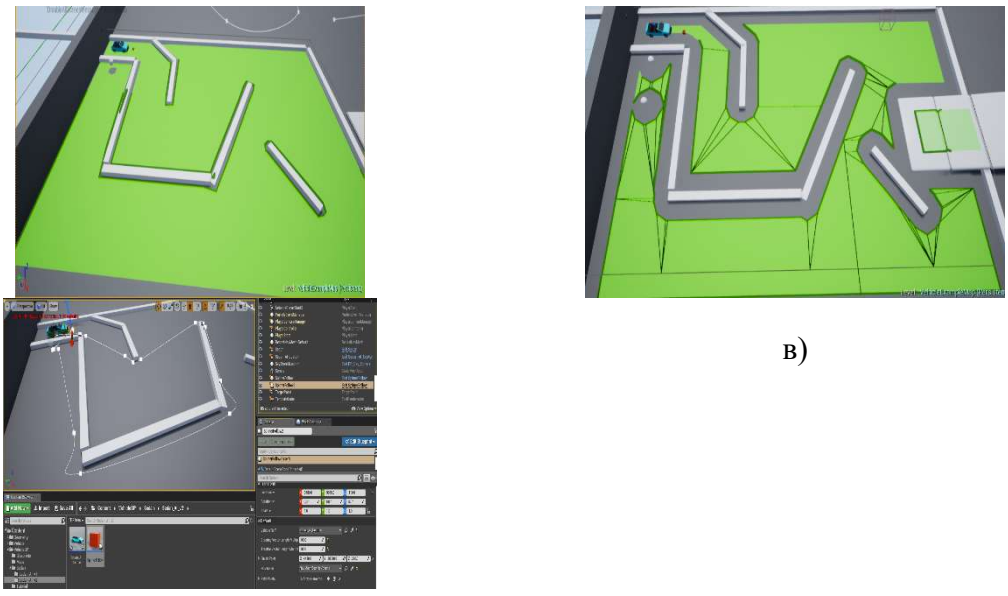
Рисунок 1 – Автомобіль перед поворотом

Обчисливши в класі автомобіля кут між поточним напрямком автомобіля і кожним з кубів, можна задати силу повороту коліс авто (червоний куб) і силу прискорення або гальмування (синій куб). При цьому в разі червоного куба – чим більшим є кут, тим сильніше провернуті колеса, і навпаки – чим меншим є кут, тим колеса будуть менш повертатися. У разі синього куба – чим більшим є кут, тим меншим буде прискорення, і навпаки. Дистанція віддалення кубів від автомобіля забезпечує зміщення кубів (вліво і вправо

щодо машини) на поворотах. При цьому важливе значення має настройка коефіцієнтів – їх некоректні значення можуть привести до поведінкових помилок.

У класі *Spline* було вказане посилання на автомобіль і зазначена дистанція для кожного з кубів. Посилання на автомобіль потрібно для визначення його положення і для передачі з *Spline* в клас автомобіля положення кубів в просторі. Переміщення кубів здійснюється в класі *Spline*. У класі *Spline* кожен відлік часу відстежується положення автомобіля, і куби переміщаються на задану дистанцію. У класі автомобіля було видалено базовий ручний контроль автомобілем і замінений іншою логікою, яка задає автоматичну їзду за сплайном.

Одним із шляхів розвитку запропонованого підходу є автоматична генерація маршруту без виставлення точок сплайна вручну. Для цього знадобиться *NavMeshBoundsVolume* – зона, в якій автоматично визначається, де можна пересуватися, а де – ні через зовнішні перешкоди, (рис. 2а, зелена зона – *NavMeshBoundsVolume*). За допомогою цієї зони можна побудувати маршрут, за яким буде рухатися автомобіль.



а) б) в)

Рисунок 2 – Локація: а) з *NavMeshBoundsVolume*; б) з маршрутом; в) з налаштованим *NavMeshBoundVolume*

Функція «*AI Move to*» теж використовує дану зону для прорахунку шляху – на рис. 2а можна побачити, що маршрут не побудований до початку гри. Результати створення шляху на основі *NavMeshBoundsVolume* показаний на рис. 2б. Необхідно відзначити, що можливість зміни параметрів (дистанція до кубів повороту і прискорення) дозволяє поліпшувати проходження маршруту. Так, в подальшому можна використовувати навчену нейронну мережу для аналізу і поліпшення проходження маршрутів.

Також необхідно відзначити, що функція автоматичного прорахунку маршруту з точки А в точку Б вимагає доопрацювання – в поточній реалізації будується найкоротший шлях для проходження (рис.). Це добре для людиноподібних класів, проте погано для автомобілів, які можуть врізатися в стіни і не проходити повороти. Тому у випадку з автомобілями слід вибирати точки для маршруту далі від перешкод. Варіантів вирішення проблеми багато. Якісь алгоритми пропонують розбити простір на клітини і використовувати графи: алгоритм пошуку в ширину, алгоритм Дейкстри, алгоритм *A star*. Якийсь інтерес представляє і стохастичний алгоритм. У той же час UE4 дуже функціональний, тому спочатку можна спробувати більш швидкий шлях для досягнення мети. Для цього необхідно налаштувати параметри побудови *NavMeshBoundVolume* і досягти прийнятних результатів, як показано на рис. 2в.

Ще одним спірним питанням є дублювання точок *Spline* на поворотах (рис. 2б). Однак це вимагає додаткового розгляду і є предметом подальшого поліпшення запропонованого методу.

Література

1. UE4 Tutorial: Add spline-meshes procedurally [електронний ресурс]. – Режим доступу: https://www.youtube.com/watch?v=7YUxM0NDWRY
2. Unreal Engine 4 Vehicle Spline Path [електронний ресурс]. – Режим доступу: https://www.youtube.com/watch?v=FVyUVJra3KI

Анотація

Розглянуто підхід прямого слідування (pure pursuit) для керування транспортом і показана його повна реалізація на движку UE4. Були продемонстровані можливості движка UE4 для полегшення побудови маршрутів в автоматичному режимі. Були запропоновані різноманітні варіанти для вирішення питання з автоматичною побудовою шляху, виділені сильні і слабкі сторони виконаної роботи.

Ключові слова: UE4, автомобіль, побудова шляху, spline, NavMeshBoundsVolume.

Аннотация

Рассмотрен подход управления прямого следования (pure pursuit) и показана его полная реализация на движке UE4. Были продемонстрированы возможности движка UE4 для облегчения постройки маршрутов в автоматическом режиме. Были предложены разнообразные варианты для решения вопроса с автоматическим прокладыванием пути, выделены сильные и слабые стороны проделанной работы.

Ключевые слова: UE4, автомобиль, построение пути, spline, NavMeshBoundsVolume.

Abstract

A pure pursuit management approach considered and its full implementation on the UE4 engine is shown. The capabilities of the UE4 engine to facilitate the construction of routes in automatic mode demonstrated. A variety of options have been proposed to address the issue with automatic paving. The strengths and weaknesses of the work done were highlighted.

Keywords: UE4, car, path finding, spline, NavMeshBoundsVolume.