Я. М. Кукар

АЛГОРИТМЫ ПОИСКА ОПТИМАЛЬНОГО ПУТИ В ИГРОВОМ ПРИЛОЖЕНИИ. NAVMESH В UNITY.

Рассматривается система навигации NavMesh в Unity и её принцип работы. Рассматриваются альтернативы данной системе, в частности алгоритм поиска пути A.*

Введение

В мире игровых приложений, навигация и поиск оптимального пути являются ключевыми аспектами, которые влияют на реалистичность и сложность игрового процесса.

NavMesh, или Навигационная Сетка, - это инструмент, который позволяет создавать и редактировать сетку навигации для персонажей в игровом мире. Он предоставляет возможность для персонажей двигаться по сложной территории, избегая препятствий и находя оптимальный путь к цели.

Однако, несмотря на все преимущества NavMesh, существуют и другие альтернативные методы поиска пути, которые могут быть более подходящими в определенных ситуациях. Одним из таких методов является алгоритм поиска пути A*, который широко используется в игровой индустрии из-за своей эффективности и гибкости.

I. NavMesh

Система навигации NavMesh позволяет объяснить игровым персонажам, как добраться до определённой точки уровня, избегая всевозможные препятствия и используя созданные игроком механики.

Для работы с ней Unity предлагает следующие четыре компонента:

- NavMesh Obstacle - движущиеся препятствия, которые подвержены физике или могут изменять своё положение каким-либо другим образом должны иметь данный компонент для того, чтоб агенты знали о их существовании и могли их обойти;
- NavMesh - основной компонент, который является структурой данных, описывающей все поверхности игрового уровня для которого была создана. Обращаясь к ней персонажи могут просчитать свой путь до определённой точки;
- NavMeshAgent - это компонент, который закрепляется за игровыми объектами, которые должны в последствии передвигаться по уровню. Такие игровые объекты называются агентами;
- Off-mesh Link - данный компонент необходим для создания особенных путей на уровне, например с помощью него могут быть сделаны: подъём по лестнице, вход в портал, прыжок с платформы.[1]

С помощью данного алгоритма можно быстро,не тратя много мощностей компьютера,найти путь до точки назначения.

II. Принципы работы

В Unity агенты движутся по выпуклым многоугольникам, описывающим поверхности, по которым они могут перемещаться. Для нахождения пути между двумя точками, сначала определяются полигоны, ближайшие к начальной и конечной точкам. Затем, с помощью алгоритма A*, находится путь, проходящий через соседние полигоны. Эта последовательность полигонов называется коридором, по которому агент перемещается, корректируя свой путь при необходимости. Для предотвращения столкновений с препятствиями используется технология RVO, которая позволяет скорректировать скорость агента. После этого рассчитывается конечная скорость с учетом ускорения динамической модели. Полученную скорость можно передать в систему анимации Mecanim или доверить навигационной системе. Чтобы создать навигационную сетку NavMesh, необходимо определить поверхности, по которым агенты могут перемещаться.

На практике же применять NavMesh можно в следующих случаях:

1. Определение проходимости и непроходимости областей: Navmesh позволяет разработчикам игр определить области, где персонажи могут свободно передвигаться, и области, где передвижение запрещено. Например, камни, стены или вода могут быть помечены как непроходимые, тогда как трава, дороги или площадки могут быть помечены как проходимые. Это делает движение персонажей в игре более реалистичным и интуитивно понятным для игрока.

2. Определение пути персонажей: Navmesh позволяет определить оптимальный путь для персонажей от одной точки до другой, учитывая проходимые и непроходимые области. Например, если персонажу нужно дойти от точки А до точки Б, navmesh будет использоваться для определения наименее препятствованного пути, чтобы персонаж мог обойти стены, прыгнуть через пропасти или перебраться через мосты.

3. Обход препятствий и избегание столкновений: Navmesh также используется для об-

хода препятствий и избегания столкновений с другими объектами в игре. Например, если персонаж идет вправо и на его пути возникает стена, navmesh позволит персонажу обойти стену, выбрав более подходящий путь. Это создает впечатление, что персонаж разумно реагирует на окружающую среду.

С помощью данного алгоритма можно быстро,не тратя много мощностей компьютера,найти путь до точки назначения.

III. Альтернативы NavMesh Unity

В Unity есть несколько альтернатив системе NavMesh для навигации персонажей. Одной из наиболее популярных является библиотека A*. Она предлагает множество функций, включая простое избегание препятствий и возможность размещать объекты на пути персонажей во время выполнения игры.

Алгоритм A* предлагает несколько преимуществ по сравнению с системой NavMesh в Unity:

- Динамическое обновление: A* может легко обрабатывать изменения в среде в реальном времени, такие как добавление или удаление препятствий, В то время как NavMesh требует перестроения сетки при изменении среды;
- Гибкость: A* может быть использован в различных типах сред, включая 3D-среды, в то время как NavMesh обычно используется для навигации на плоскости;
- Избегание препятствий: A* предлагает простые методы для избегания препятствий.

Система NavMesh в Unity в свою очередь имеет несколько преимуществ по сравнению с алгоритмом A*:

- Высокая производительность: NavMesh обычно обеспечивает более высокую производительность по сравнению с A*, особенно в больших и сложных средах;
- Точность: NavMesh обеспечивает более точное представление проходимых поверхностей, что позволяет создавать более реалистичные и плавные пути;
- Интеграция с Unity: NavMesh тесно интегрирован с Unity и предлагает удобные инструменты для визуализации и отладки;
- Поддержка сложной геометрии: NavMesh может легко обрабатывать сложную 3D-геометрию, включая ступени, склоны и неровные поверхности;
- Автоматическое создание: Unity предоставляет инструменты для автоматического создания NavMesh из геометрии уровня, что упрощает процесс настройки навигации.

Таким образом NavMesh может быть более удобен, чем A* в следующих случаях:

1. Сложная трехмерная среда: NavMesh хорошо подходит для сложных трехмерных сред, таких как горы, здания или леса. Он может автоматически генерировать навигационную сетку, которая учитывает препятствия и высоту террейна.

2. Большое количество агентов: Если в игре много агентов, которые должны перемещаться одновременно, NavMesh может быть более эффективным. Он позволяет агентам двигаться вместе, избегая друг друга и препятствий.

3. Встроенная поддержка в Unity: NavMesh является частью Unity и интегрируется с другими системами Unity, такими как система анимации. Это может упростить разработку и отладку.

IV. Вывод

Система навигации NavMesh в Unity представляет собой мощный инструмент для создания сложных и реалистичных сценариев движения в играх. Она позволяет разработчикам создавать динамические и адаптивные среды, в которых персонажи могут двигаться с учетом препятствий и изменений в окружающей среде.

Однако, как и любой инструмент, NavMesh имеет свои ограничения и не всегда может быть оптимальным решением. В частности, алгоритм поиска пути A* представляет собой эффективную альтернативу. Однако оба инструмента имеют свои преимущества и могут быть использованы в различных сценариях для достижения наилучших результатов, поэтому выбор между NavMesh и A* должен основываться на конкретных требованиях проекта

1. Habr [Электронный ресурс] /Использование NavMesh для навигации ИИ в Unity - Режим доступа: https://habr.com/ru/articles/646039/ - Дата доступа: 10.03.2024

Кукар Ярослав Михайлович, студент 1 курса факультета ИТиУ БГУИРа, Yarik.ghost@mail.ru.
Научный руководитель: Рязанцев Дмитрий Дмитриевич, ассистент кафедры ВМиП БГУИРа, d.riazantsev@bsuir.by.