ОПТИМИЗАЦИЯ ОБНАРУЖЕНИЯ КОЛЛИЗИЙ МЕЖДУ ОБЪЕКТАМИ В ТРЕХМЕРНОМ ПРОСТРАНСТВЕ

Абрашин Т.Ю., Радецкий А.Н.

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Оношко Д.Е. – ст. преподаватель

Выполнен сравнительный анализ алгоритмов обнаружения коллизий между объектами и их практическая реализация на примере простой трёхмерной игры. Проведены замеры производительности, подтверждающие эффективность выбранных методов.

В современных трёхмерных играх, где динамика и интерактивность являются основой игрового процесса, корректная обработка коллизий выступает не только гарантом реалистичности, но и критически важным фактором для обеспечения стабильной производительности. Алгоритмы обнаружения и разрешения столкновений напрямую влияют на вычислительную нагрузку, потребление ресурсов и, как следствие, на частоту кадров (FPS), задержку ввода (input lag) и другие показатели производительности. Неоптимизированные методы обработки коллизий способны стать «бутылочным горлышком» даже для мощного аппаратного обеспечения, особенно в сценах с высокой плотностью объектов, сложной геометрией или динамически меняющейся средой. Это делает выбор и реализацию эффективных алгоритмов ключевой задачей. Целью данной работы являлась минимизация количества необходимых вычислительных ресурсов для эффективного обнаружения и обработки коллизий объектов при разработке 3D-игры в жанре танкового шутера.

Среди основных групп алгоритмов обнаружения коллизий можно выделить следующие: алгоритмы разделения осей, алгоритмы иерархий ограничивающих объёмов и алгоритмы сеток разбиения пространства. Алгоритм разделения осей (SAT) является одним из наиболее распространённых. Он демонстрирует высокую точность обнаружения коллизий, включая поддержку сложных геометрических форм, однако требует оптимизации для сцен с большим количеством объектов из-за снижения производительности. Алгоритм иерархий ограничивающих объемов (BVH) показывает большую производительность, однако значительно меньшую точность по сравнению с SAT. Алгоритм сеток разбиения пространства используется в ситуациях, когда требуется обработать коллизии для больших сцен, однако имеет низкую точность для объектов, имеющих сложную геометрическую форму [1].

Важным аспектом реализации данной работы стал выбор типа ограничивающего объема. Выбор OBB (Oriented Bounding Box) вместо AABB (Axis-Aligned Bounding Box) обусловлен необходимостью точного учёта ориентации моделей. AABB, выровненные по глобальным осям, быстро вычисляются, но при вращении объектов (например, башни танка) их габариты «раздуваются», вызывая ложные коллизии.

Несмотря на сравнительно небольшое количество объектов, сами объекты имели сложную геометрическую форму. В связи с этим среди всех алгоритмов был выбран SAT. Его возможные проблемы с производительностью компенсируются небольшим количеством объектов и выполненными оптимизациями.

Работа алгоритма SAT основывается на следующем математическом утверждении: объекты A и B не пересекаются, если существует ось v, на которой их проекции $[a_{min}, a_{max}]$ и $[b_{min}, b_{max}]$ не перекрываются, где a_{min}, b_{min} – минимальные координаты проекций вершин объектов A и B на ось, a_{max} , b_{max} – максимальные координаты проекций вершин объектов A и B на ось [2]. Алгоритм проецирует объекты на потенциальные разделяющие оси. Для 3D-объектов (в частности, OBB) проверяются 15 осей: локальные оси объектов и векторные произведения их рёбер. На каждой оси вычисляются проекции вершин объектов (минимальные и максимальные значения). Если на любой оси проекции не перекрываются, объекты не пересекаются. Если перекрытие есть – коллизия обнаружена.

Для данного алгоритма была определена следующая оптимизация: изначально проводится грубая проверка на пересечение AABB, если пересечение отсутствует, можно сделать вывод, что коллизия отсутствует, иначе проводится более точная проверка на основании пересечения OBB. Преимуществом данного подхода является более высокая скорость проверки на пересечение для более простого ограничивающего объема. Данная оптимизация помогает отсечь большое количество потенциальных пар объектов.

Еще одной оптимизацией является ранний выход: если обнаружена первая разделяющая ось, на которой проекции объектов не пересекаются, алгоритм немедленно останавливается, и остальные оси не проверяются.

Для реализации стрельбы был выбран метод HitScan. Метод HitScan (или Hit-Scan) – это техника, используемая в играх и симуляциях для определения попадания снаряда или луча в цель. Луч

начинается в точке происхождения (обычно ствол оружия) и направляется вдоль вектора, определённого направлением выстрела [3].

В отличие от симуляции физически точной траектории снаряда (например, с учетом гравитации и сопротивления воздуха), HitScan мгновенно проверяет, пересекает ли луч (линия) цель в момент выстрела. Этот метод широко используется в шутерах от первого лица (FPS) и других играх, где важна быстрота и простота расчетов. Данный метод был выбран по причине его высокой производительности, точности и предсказуемости. Увидеть принцип действия данного метода можно на рисунке 1.

Рисунок 1 – Схема работы метода HitScan

Hitscan основан на геометрической трассировке луча, задаваемого параметрическим уравнением:

$$R = O + t \cdot D,\ t \in [0,\infty) \quad (1)$$

где O – точка выстрела, D – нормализованный вектор направления.

Для определения коллизии луч проверяется на пересечение с ограничивающими объемами объектов. При этом из всех пересечений выбирается ближайшая точка (с наименьшим значением параметра t) чтобы определить первый объект, пораженный выстрелом.

В данном случае также была проведена оптимизация вычислений за счет изначальной грубой проверки на пересечение с AABB и раннего выхода в случае обнаружения оси, по которой оно отсутствует. Помимо этого, была определена еще одна оптимизация, связанная с возможностью локализации сцены: деления пространства на подпространства определенных размеров с целью проверки только ближайших объектов. Дополнительной оптимизацией стало ограничение дистанции выстрела, исключающее проверку коллизий с удалёнными объектами.

На основании практической реализации алгоритмов обнаружения коллизий на тестовом оборудовании с применением оптимизаций и без них были получены следующие результаты:

– Для сцены с 30 объектами. Без оптимизаций средний FPS составил 70–75. При включении оптимизаций данный показатель вырос до 73-79. Различия в производительности оказались незначимыми (4%), что связано с малой вычислительной нагрузкой при небольшом числе объектов.

– Для сцены с 200 объектами. Без оптимизаций средний FPS снижался до 54-55. С оптимизациями FPS стабилизировался на уровне 65-68. Это соответствует приросту производительности на 22%.

Практическая реализация алгоритмов SAT и HitScan в рамках трёхмерной игры с динамическими объектами сложной геометрии (танки) подтвердила их эффективность в условиях ограниченных вычислительных ресурсов. Внедрение SAT с двухэтапной проверкой (AABB → OBB) и ранним выходом позволило достичь высокой точности обнаружения коллизий при сокращении времени расчётов. Это особенно важно для объектов с нестандартной формой, где традиционные методы (например, BVH) демонстрируют значительные погрешности.

HitScan, оптимизированный за счёт локализации сцены и ограничения дальности, обеспечил мгновенную проверку попаданий без просадок FPS, что критично для шутеров с высокой интенсивностью стрельбы. Однако метод требует дополнительной доработки для учёта баллистических эффектов (гравитация, отскоки), что ограничивает его применение в симуляторах.

Реализация подтвердила, что даже при ограниченных вычислительных ресурсах данные методы обеспечивают баланс между точностью и производительностью, что делает их применимыми в широком спектре игр – от тактических шутеров до симуляторов техники.

Список использованных источников:

1. Ericson C. Real-Time Collision Detection. Morgan Kaufmann, 2005.
2. Bourg M.D., Bywalec B. – Physics for Game Developers.
3. Collision Detection in Interactive 3D Environments / Gino van den Bergen