DOI 10.34185/1562-9945-4-153-2024-11
УДК 004.421.2

І.О. Невкритий, С.В. Антоненко

АНАЛІЗ АЛГОРИТМІВ ВИЯВЛЕННЯ ЗІТКНЕНЬ У ТРИВИМІРНИХ ВІРТУАЛЬНИХ СЕРЕДОВИЩАХ

Annotation. The relevance of the research topic lies in the significant computational challenge posed by real time collision detection in virtual environments, often consuming up to 50% of ex ecution time in some cases, or even more. In real world simulations, three dimensional proto types can be highly complex, containing thousands of primitives. Therefore, developing collision detection algorithms in real time for complex environments remains a relevant scientific prob lem. Maintaining a frame rate of at least 30 FPS is crucial for creating a user friendly simulator in such environments.

The problem of collision detection in virtual environments becomes even more intricate during the narrow phase. The broad phase of collision detection is responsible for discarding pairs of objects that will not collide due to a sufficiently large distance between them. The narrow phase determines whether two objects collide when the distance between them allows for such a possi bility.

This article describes a collision detection algorithm based on the structure of R trees of axis aligned bounding boxes (AABBs) and utilizes the oriented bounding box intersection (OAABB) method to address the narrow phase of collision detection, which finds intersections between two objects. Experimental results demonstrate that the algorithm presented in this article identifies intersections with significant interaction speeds.

Experimental results demonstrate that the algorithm presented in this article identifies intersec tions with significant interaction speeds.

Ключові слова: collision detection, 3D gaming algorithms, computer graphics, R trees, bounding boxes, performance benchmark, innovations, model processing.

Постановка проблеми. При виявленні зіткнень у віртуальних середовищах виникає проблема великого обсягу обчислень, що може забирати 50% часу виконання та більше. У складних тривимірних симуляціях, які містять тисячі примітивів, розробка алгоритмів виявлення зіткнень у реальному часі залишається актуальною науковою проблемою. Досягнення частоти кадрів принаймні 30 FPS є важливим для створення зручних для користувача симуляторів. Виявлення зіткнень звичайно складається з двох фаз: широкої та

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

вузької. Широка фаза відповідає за відкидання пар об'єктів, які не зіткнуться через достатньо велику відстань, тоді як вузька фаза визначає зіткнення, коли об'єкти достатньо близько. Очевидно, ускладнення настає під час вузької фази виявлення зіткнень. У даній роботі представлено алгоритм виявлення зіткнень, що використовує структуру R-дерев орієнтованих обмежуючих паралелепіпедів (AABB) та метод перетину орієнтованих обмежуючих паралелепіпедів (OAABB) для вирішення вузької фази виявлення зіткнень. Експериментальні результати демонструють ефективність алгоритму у виявленні перетинів зі значними швидкостями взаємодії.

Аналіз останніх досліджень та публікацій. Існує кілька загальнодоступних наборів інструментів для вирішення вузької фази виявлення зіткнень на основі ієрархій обмежуючих об’ємів. Вони відрізняються типом реалізованого об'єму обмеження. Наприклад, SOLID та OPCODE використовують орієнтовані паралелепіпеди (AABB). RAPID, V-COLLIDE, PQP, H-COLLIDE використовують орієнтовані обмежуючі паралелепіпеди (OBB). QuickCD та DopTree використовують k-dops; a Swift++ використовує опуклі оболонки (CH). Також існують гібридні підходи, наприклад QuOSPOs, які використовують комбінацію OBB та k-dops [1].

Порівняти різні підходи досить складно, оскільки продуктивність також залежить від форм моделей, типу контакту, розміру моделей і інших факторів. У знаходженні зіткнень різних видів обмежуючих об’ємів використовуються різноманітні алгоритми, що залежать від конкретних вимог і характеристик задачі. Основними перевагами різних підходів є їх швидкодія та точність, а також можливість оновлення обмежуючих об’ємів при руху об'єктів. Такі алгоритми є важливою складовою в розробці віртуальних середовищ та ігрових додатків, де точне виявлення зіткнень відіграє вирішальну роль.

У знаходженні зіткнень між двома тривимірними об'єктами часто використовують ієрархії обмежувальних об'ємів (Bounding Volume Hierarchies, BVH), які організовують тріангульовані дані об'єкта. Таким чином, продуктивність поліпшується за рахунок зменшення кількості пар тестів за допомогою використання обмежуючих об’ємів [2].

Припустимо, що обмежувальні об'єми двох об'єктів організовані у двох ієрархічних деревах обмежуючих об'ємів (BV) A та B. Класична схема ієрархічного виявлення зіткнень - це одночасне рекурсивне обходження двох дерев обмежуючих об’ємів A та B [3]. Це може бути описано алгоритмом, наведеним у лістингу 1 за допомогою псевдокоду.

Лістинг 1. Типова ієрархічна схема обходу для виявлення зіткнень між об’єктами

```
ОбхідДерев (A, B)
Перевести обмежуючий об’єм B у систему координат A
якщо A та B не перетинаються, то
    повернути пусту множину
якщо A та B є листками, то
    перевести трикутник B у систему координат A
    повернути перетин трикутників, що обмежені A та B
інакше
    якщо A є листком і B - внутрішнім вузлом, то
        для всіх дітей B[i]
        ОбхідДерев (A, B[i])
    інакше
        якщо A є внутрішнім вузлом і B - листком, то
            для всіх дітей A[i]
                ОбхідДерев (A[i], B)
        інакше
             для всіх дітей A[i] та B[j]
                ОбхідДерев (A[i], B[j])
```

Однак Захман та Кніттель виявили деякі проблеми у традиційному обході. Зокрема, вони показали, що декілька вузлів у деревах ієрархії A та B відвідуються більше одного разу. Друга проблема полягає в тому, що вузли дерева обмежуючих об’ємів B, наприклад, повинні бути перетворені в локальну систему координат A. Як наслідок першої проблеми, це перетворення повторюється декілька разів для того самого вузла, що знижує продуктивність [4]. Алгоритм, представлений у даній роботі, вирішує ці проблеми.

Мета дослідження. Метою дослідження є розробка ефективного алгоритму для виявлення зіткнень у реальному часі у віртуальних середовищах.

Основна частина. У цьому розділі представлено деталі реалізації ідей авторів для визначення перетинів поверхонь і трикутників для пари тривимірних об'єктів. Для застосувань у віртуальному середовищі тривимірні об'єкти можуть бути визначені як колекція поверхонь, де кожна поверхня індивідуально представлена як колекція трикутників.

Час виявлення зіткнень між двома об'єктами за допомогою BVH залежить від: (1) вартості перетину та оновлення обмежуючих об’ємів; (2) вартості перетину трикутників; та (3) кількості таких операцій.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Підхід, представлений у роботі, підтримується ієрархіями R-дерев орієнтованих обмежуючих паралелепіпедів та перетином орієнтованих обмежуючих паралелепіпедів, щоб зменшити вартість та кількість операцій оновлення та перетину, тим самим покращуючи продуктивність. Вибір типу об'єму обмежень впливає на продуктивність процесу виявлення зіткнень. А використання орієнтованих паралелепіпедів обумовлено тим, що вони швидше перетинаються.

Було вирішено використовувати R-дерева для побудови ієрархій обмежуючих об’ємів та організації тривимірної геометрії об'єктів для покращення продуктивності процесу виявлення зіткнень. R-дерева є хорошим вибором для виявлення зіткнень, оскільки, по-перше, на будь-якому рівні дерева кожний примітив асоційований лише з одним вузлом. По-друге, в R-дереві всі листки знаходяться на одному рівні. По-третє, оскільки глибина R-дерева, що зберігає n примітивів, дорівнює $\log_m n$, де m – мінімальна кількість дітей вузла [5].

Для прискорення процесу виявлення зіткнень кожен об'єкт представлений структурою даних R-дерева у власній локальній системі координат (рисунок 1). Будується ієрархічне дерево, групуючи сусідні поверхні. Листки R-дерева вказують на геометрію поверхонь, які визначають об'єкт. Для двох об'єктів перевіряється зіткнення між поверхнями, які знаходяться в сусідніх зонах, уникуючи порівнянь з тими, які знаходяться далеко.

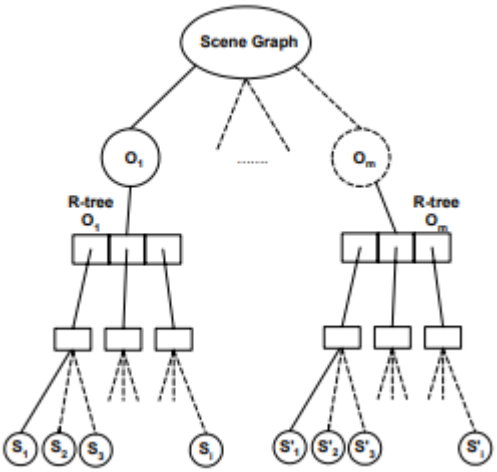

Рисунок 1 - Кожен об’єкт графа сцени представлено R-деревом для організації його поверхонь

Поверхні тривимірної моделі можуть бути складними. Вони можуть містити велику кількість трикутників. З цієї причини реалізація, описана у роботі, також використовує R-дерево для просторової організації трикутників

(рисунок 2), що допоможе швидше відхилити трикутники, які гарантовано не можуть перетинатися. У цьому підході для кожної поверхні обчислюється R-дерево, групуючи сусідні трикутники, для уникнення порівнянь з такими, що знаходяться далеко від зони перетину.

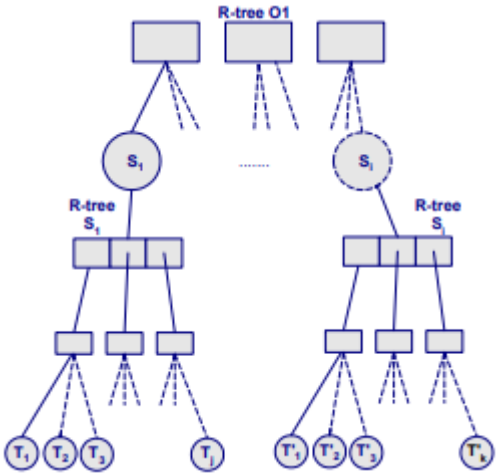

Рисунок 2 - Кожна поверхня 3D-моделі також представлена структурою даних R-дерево для просторової організації її трикутників

Запропонований алгоритм також ґрунтується на використанні перетинів орієнтованих обмежуючих паралелепіпедів (Overlapping Axis-Aligned Bounding Box, OAABB) (A, B), двох геометричних примітивів для покращення продуктивності процесу виявлення зіткнень. OAABB визначається як об'єм, який є спільним для двох орієнтованих обмежуючих паралелепіпедів A та B. OAABB використовується для відфільтрування примітивів, які не можуть перетинатися. Розглянемо приклад на рисунку 3. Поверхня S_2 об'єкта A не може перетинатися з об'єктом B, оскільки вона не перетинає OAABB. Таким чином, можливо швидко відфільтрувати всі поверхні обох об'єктів, які не перетинають OAABB.

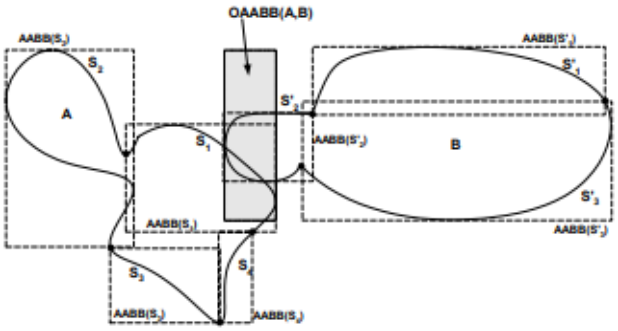

Рисунок 3 - Обмежувальна рамка з вирівнюванням по осі, що перекривається (OAABB), концепція показана в 2D

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Алгоритм для пошуку поверхонь, що перетинаються, представлений у лістингу 2 у вигляді псевдокоду.

Лістинг 2. Алгоритм для знаходження поверхонь та трикутників, що перетинаються.

```
ПошукЗіткнень (A, B)
1: AABB_B(A) = M_B_A × AABB_A(A) // оновлення координатної системи BV
2: Якщо (AABB_B(A) не перетинається з AABB_B(B)) вихід
3: Визначити OAABB_B(A, B)
4: Обхід R-дерева (SBV(B), OAABB_B(A,B))
5: Для кожної поверхні з SBV(B), що перетинає OAABB_B(A,B)
6:          Обхід R-дерева (TBV(B), OAABB_B(A,B))
7:          Для кожного трикутника T(B) з TBV(B), що перетинає OAABB_B(A,B)
8:               Перенести геометрію трикутника T(B) в систему координат A
9:               Обчислити новий оптимальний AABB_A(T(B)) для T(B)
10:         Обхід R-дерева (SBV(A), AABB_A(T(B)))
11:         Для кожної поверхні з SBV(A)
12:              Обхід R-дерева (TBV(A), AABB_A(T(B)))
13:                   Перетин T(A) та T(B)
```

Процес виявлення зіткнень спочатку перевіряє, чи об'єкти A і B є відокремленими (рядки 1-2 у лістингу 2). Об'ємні обмеження кожного об'єкта спочатку обчислюються у локальній системі координат об'єкта, $AABB_A(A)$ та $AABB_B(B)$ відповідно. Матриця трансформації, що перетворює локальне представлення об'єкта A у локальну систему координат об'єкта B, визначена як M_{B_A}. Обмежуючий об'єм об'єкта A оновлюється до системи координат об'єкта B, обчислюючи обмежуючий об'єм $AABB_B(A)$. Якщо ця пара AABB не перетинається, тоді відповідні два об'єкти не перетинаються, і процес завершується. Якщо вони перетинаються, то система визначає орієнтований обмежуючий паралелепіпед перетину, $OAABB_B(A,B)$ двох об'єктів (рядок 3 у лістингу 2), який визначений у локальній системі координат об'єкта B.

Наступним кроком процесу виявлення зіткнень є визначення поверхонь об'єкта B, що перетинають OAABB (рядок 4 у лістингу 2). Як вже зазначалося, поверхні об'єкта B організовані в дереві поверхневих об'ємів, яке називається SBV(B). Поверхні B зберігаються в листках дерева SBV(B). За допомогою обходження цього дерева в глибину, фільтруються поверхні об'єкта B, які не перети-

нають $OAABB_B(A,B)$. Лише поверхні у листках, що перетинають $OAABB_B(A,B)$, є кандидатами на зіткнення [6].

Кожна поверхня об'єкта B представлена колекцією трикутників. Трикутники кожної поверхні також представлені у власній структурі даних дерева трикутникових обмежень, організуючи їх у підрегіони. Наступний крок процесу виявлення зіткнень полягає в обчисленні дерева трикутникових обмежень TBV(B) кожної кандидатної поверхні об'єкта B (рядки 5 і 6 у лістингу 2). На цьому етапі визначаються трикутники об'єкта B, що перетинають OAABB.

Використовуючи OAABB, менеджер виявлення зіткнень фільтрує поверхні та трикутники об'єкта B, які не можуть перетинатися.

Трикутники об'єкта B, які перетинають OAABB, перетворюються у систему координат об'єкта A (рядок 8). На цьому етапі також визначається оптимальний обмежуючий об'єм трикутника (рядок 9). Новий оптимальний AABB будується шляхом обчислення нових мінімальних і максимальних значень по осях x, y та z, визначаючи нові розміри обмежуючого об'єму. Цей етап обчислює оптимальний обмежуючий об'єм, оскільки він є швидшим, ніж обчислення мінімального AABB. Крім того, оптимальний AABB краще обмежує трикутник, ніж мінімальний AABB.

Потім процес виявлення зіткнень спускається до дерева поверхневих обмежень SBV(A) для об'єкта A (рядок 10 у лістингу 2). На цьому етапі він знаходить поверхні об'єкта A, що перетинають оптимальне $AABB_A(T(B))$ трикутника об'єкта B.

Щоб визначити, чи перетинається пара поверхонь, необхідно знайти пару трикутників, що перетинаються. Потім процес виявлення зіткнень переходить до дерева трикутникових обмежень TBV(A) кожної кандидатної поверхні об'єкта A (рядки 11 і 12 у лістингу 2). На цьому етапі визначаються трикутники об'єкта A, об'ємні обмеження яких перетинають оптимальне AABB об'єкта B.

Останнім кроком є продовження перетину між парами кандидатних трикутників (рядок 13), реалізоване за допомогою алгоритму Guige, P. and Devillers [7]. Якщо є пара трикутників що перетинаються, то відповідні поверхні перетинаються.

Результати експерименту. У цьому розділі представлені результати оцінки продуктивності запропонованого алгоритму виявлення зіткнень.

Представлено приклад моделювання процесу на підприємстві. Усі експерименти, описані в наступних абзацах, виконувалися на процесорі Intel Core i5-8300H, з об'ємом оперативної пам'яті 16 Гбайт при тактовій частоті 4 ГГц.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Розглядається реальний застосунок з обслуговування віртуальних середовищ. На рисунку 5 представлено скріншот з додатку Siemens Kineo software [8], який симулює діяльність 6-осевого робота по відвантаженню товару. Siemens Kineo був використаний в якості симулятора, для можливості завантаження моделі до 3D сцени, яку згодом можна буде конвертувати в набір трикутників і протестувати швидкодію розробленого алгоритму. Наступний приклад використовується для тестування виявлення зіткнень в віртуальних середовищах, де важливо знаходити деталі, що перетинаються з іншими, щоб скоригувати сценарій поведінки для фізичного робота. Даний сценарій має загальну кількість 28 деталей, 2252 поверхні і 30535 трикутників.

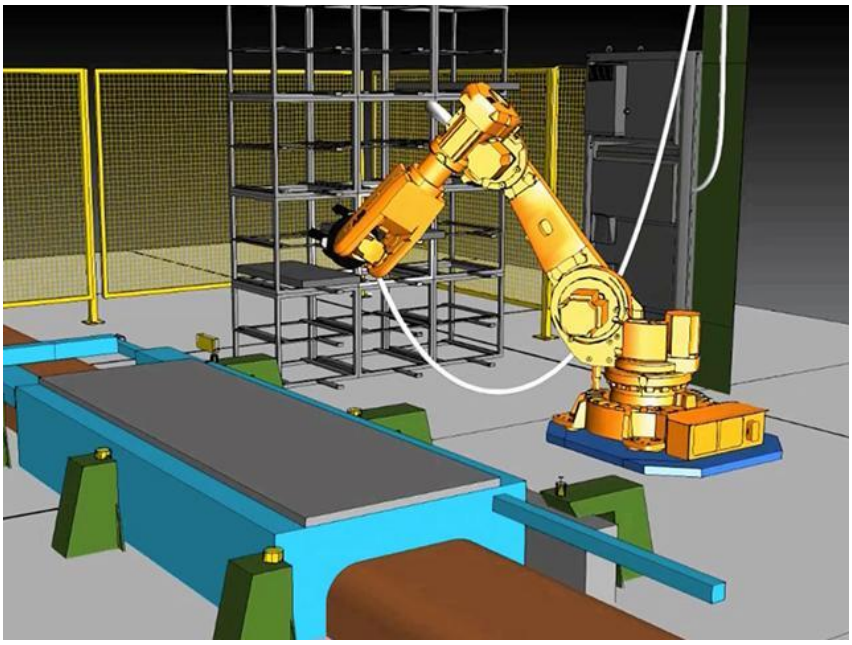

Рисунок 5 - 3D симуляція вантажного 6-осевого робота

Запропонований алгоритм виявлення зіткнень досягає інтерактивних показників у реальних промислових застосуваннях, як це було бажано.

У таблиці 1 для порівняння наводяться часові показники, отримані авторами за двома підходами: з використанням перетинаючого орієнтованого обмежуючого паралелепіпеда (OAABB) та без нього. Відзначається значне покращення продуктивності саме при використанні перетинаючого орієнтованого обмежуючого паралелепіпеда.

Таблиця 1

Час виявлення зіткнень для знаходження поверхонь, що перетинаються, для процесу виробництва, у мілісекундах

Час у мілісекундах для визначення		
Використання OAABB	Так	Ні
(1) перша поверхня, що перетинається	0.04	0.17
(2) усі поверхні, що перетинаються	0.068	1.65
(3) всі поверхні та трикутники, що перетинаються	1.04	1.9

Це покращення пояснюється за допомогою даних, наведених у таблиці 2. Вартість виявлення зіткнень залежить від кількох чинників. Вибір типу об'ємного обмеження впливає на кількість та вартість виконання перетинів об'ємних обмежень. Для AABB та k-dops вартість та кількість оновлення об'ємних обмежень також впливають на продуктивність. Таблиця 2 показує ефективне зменшення кількості тестів AABB та операцій оновлення об'ємних обмежень, що пояснює кращу продуктивність, отриману за допомогою OAABB.

Таблиця 2

Середня кількість операцій на крок для визначення перетинів

Кількість операцій	Перший трикутник		Всі трикутники	
Використання OAABB	Так	Ні	Так	Ні
AABB тести	1523	3251	18324	34526
AABB онов-лення	145	816	631	4125

Також важливо порівняти продуктивність цього алгоритму з іншими інструментами виявлення зіткнень. У таблиці 3 представлено порівняння реалізації запропонованого алгоритму (в таблиці позначений як ПошукЗіткнень) з алгоритмами PQP, RAPID, OPCODE та Dop Tree алгоритмами. Представлені результати були отримані при визначенні першого трикутника, що перетинається, а також для всіх трикутників, що перетинаються. Таблиця 3 свідчить про те, що розроблений алгоритм є ефективним у визначенні перетинів.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Таблиця 3

Час пошуку перетинів для розробленого алгоритму
та його порівняння з іншими методами

Час на пошук перетинів (мілісекунди)	Перший трикутник	Всі трикутники
PQP	0.55	5.17
RAPID	0.12	3.87
Dop Tree	0.06	18.25
OPCODE	1.64	2.54
ПошукЗіткнень	0.15	1.56

Висновки. У роботі представлено новий алгоритм виявлення зіткнень, який обчислює поверхні та трикутники, що перетинаються, вирішуючи проблему вузької фази виявлення зіткнень.

Описано використання R-дерев та OAABB для реалізації ефективного методу знаходження зіткнень в реальному часі для застосунків віртуальної реальності.

Алгоритм виявлення зіткнень, що представлений у роботі, використовує орієнтовані обмежуючі паралелепіпеди разом зі структурою R-дерева, щоб відфільтрувати обмежуючі об’єми примітивів, які не можуть перетинатися. Два примітиви перетинаються, якщо відповідні обмежуючі об’єми також перетинаються з OAABB. Представлений алгоритм обходу зменшує кількість відвідувань вузлів до одного для об'єкта A, покращуючи загальну продуктивність та вирішуючи проблеми кількох відвідувань, що притаманні традиційній схемі обходу. OAABB також сприяє значному зменшенню кількості операцій оновлення обмежуючих об’ємів.

ЛІТЕРАТУРА

1. Огляд алгоритмів розпізнання зіткнень 3Д об’єктів. Режим доступу: https://www.inesc-id.pt/ficheiros/publicacoes/7101.pdf

2. Алгоритми розпізнання зіткнень широкої фази. Режим доступу: https://d2l.ai/chapter_computer-vision/bounding-box.html

3. Класична схема ієрархічного виявлення зіткнень. Режим доступу: https://cgvr.cs.uni-bremen.de/papers/wscg03/wscg_ext.pdf

4. [Zachmann03] G. Zachman, and G. Knittel, “An Architecture for Hierarchical Collision Detection,” Proceedings of WSCG' 2003, the 11th International Conference in

Central Europe on Computer Graphics, Visualization and Computer Vision'2003, Czech Republic, 2003.
5. Ефективні операції з R деревами. Режим доступу: https://www.bartoszsypytkowski.com/r-tree/
6. Пошук перетину об’єктів за допомогою R дерев. Режим доступу: https://www.researchgate.net/publication/236611860_A_Collision_Detection_Algorithm_for_Polygonal_Models_that_uses_Sequential_and_Parallel_Techniques_for_Improving_Performance
7. [Guige03]Guige, P. and Devillers, O., 2003. Fast and Robust Triangle-Triangle Overlap Test using Orientation Predicates. Journal of Graphics Tools, 8, 1, 25-42
8. Огляд застосунку Siemens Kineo. Режим доступу: https://plm.sw.siemens.com/en-US/plm-components/kineo/kineoworks/

REFERENCES

1. Review of algorithms for 3D object collision detection. Access mode: https://www.inesc-id.pt/ficheiros/publicacoes/7101.pdf
2. Algorithms for broad-phase collision detection. Access mode: https://d2l.ai/chapter_computer-vision/bounding-box.html
3. Classical hierarchical collision detection scheme. Access mode: https://cgvr.cs.uni-bremen.de/papers/wscg03/wscg_ext.pdf
4. [Zachmann03] G. Zachman, and G. Knittel, "An Architecture for Hierarchical Collision Detection," Proceedings of WSCG' 2003, the 11th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2003, Czech Republic, 2003.
5. Efficient operations with R-trees. Access mode: https://www.bartoszsypytkowski.com/r-tree/
6. Object intersection search using R-trees. Access mode: https://www.researchgate.net/publication/236611860_A_Collision_Detection_Algorithm_for_Polygonal_Models_that_uses_Sequential_and_Parallel_Techniques_for_Improving_Performance
7. [Guige03] Guige, P. and Devillers, O., 2003. Fast and Robust Triangle-Triangle Overlap Test using Orientation Predicates. Journal of Graphics Tools, 8, 1, 25-42
8. Review of Siemens Kineo application. Access mode: https://plm.sw.siemens.com/en-US/plm-components/kineo/kineoworks/

Received 10.05.2024.
Accepted 14.05.2024.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Analysis of collision detection algorithms in three-dimensional virtual environments

The problem addressed in this paper concerns the real-time detection of collisions in virtual environments, which can consume a significant portion of computational resources, particularly in complex simulations with numerous objects. Ensuring a minimum frame rate of 30 FPS is crucial for user-friendly simulations. The focus is on the narrow phase of collision detection, where objects in close proximity are examined for potential collisions. This paper proposes a collision detection algorithm leveraging Directed Bounding Parallels (AABB) R-tree structure and Oriented Bounding Parallelepiped Intersection (OAABB) method to address this phase efficiently. Experimental results demonstrate the effectiveness of the algorithm in detecting intersections with high interaction rates.

Various publicly available toolkits exist for narrow phase collision detection, employing different bounding volume hierarchies and constraint volumes. Comparing these approaches is challenging due to performance variations influenced by factors such as object shapes, contact types, and model sizes. Nonetheless, these algorithms are crucial for accurately detecting collisions in virtual environments and gaming applications.

Bounding Volume Hierarchies (BVH), commonly used for collision detection, organize object geometry and improve performance by reducing test pair numbers through volume constraints. The proposed algorithm utilizes R-tree hierarchies of directed bounding parallelepipeds and OAABB to enhance collision detection efficiency by minimizing update and intersect operations. The choice of constraint volume type impacts collision detection performance, with oriented parallelepipeds offering faster intersections.

To expedite collision detection, each feature is represented by an R-tree data structure in its local coordinate system. Hierarchical trees are constructed by grouping adjacent surfaces, with leaves indicating surface geometry. Additionally, R-trees are employed to spatially organize triangles within surfaces to quickly discard non-intersecting triangles.

The algorithm leverages OAABB, a concept involving the common volume between two oriented bounding parallelepipeds, to filter out primitives that cannot intersect. Traversal algorithms are utilized to reduce the number of node visits, addressing inefficiencies observed in traditional schemes. OAABB contributes significantly to reducing volume refresh operations.

Experimental evaluations demonstrate the proposed algorithm's efficacy in real-world industrial applications, achieving interactive performance. Comparisons with alternative methods highlight the algorithm's effectiveness in collision detection. Overall,

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

the proposed approach offers a robust solution to the narrow phase collision detection problem in virtual environments.

Keywords: collision detection, 3D gaming algorithms, computer graphics, R trees, bounding boxes, performance benchmark, innovations, model processing.

Невкритий Іван Олександрович – аспірант 3-го року навчання Дніпровського національного університету імені Олеся Гончара.

Антоненко Світлана Валентинівна – кандидат технічних наук, доцент, доцент кафедри математичного забезпечення ЕОМ Дніпровського національного університету імені Олеся Гончара.

Nekvrytyi Ivan - graduate student of Dnipro National University named after Oles Honchar.

Antonenko Svitlana - candidate of technical sciences, associate professor, associate professor of the department of computer mathematical support of Dnipro National University named after Oles Honchar.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)