ШВИДКЕ ТРАСУВАННЯ ПРОМЕНІВ ТА ПОТЕНЦІЙНИЙ ВПЛИВ НА ГРАФІКУ

Вершинін Мар'ян Володимирович
Студент
Ужгородський національний університет, Україна

Анотація. *Сучасні графічні процесори (GPU), які є майже на кожному персональному комп'ютері, використовують алгоритм z-буфера для обчислення видимості. Трасування променів, альтернатива алгоритму z-буфера, забезпечує вищу візуальну якість, ніж алгоритм z-буфера, але раніше було надто повільним для інтерактивного використання. Проте трасування променів виграло від удосконалення комп'ютерного обладнання, і багато хто вважає, що воно замінить алгоритм z-буфера як графічний механізм на ПК. Якщо така заміна відбудеться, це означатиме фундаментальні зміни як в API, так і в можливостях механізмів тривимірної графіки. В даній роботі розглядаються основи z-буфера та трасування променів, а також, що трасування променів замінить z-буфер у найближчому майбутньому.*

Зараз майже кожен персональний комп'ютер має спеціальний процесор, який забезпечує інтерактивну 3D-графіку. Ці графічні процесори (GPU) реалізують алгоритм z-буфера. Ці графічні процесори можуть інтерактивно відображати кілька мільйонів трикутників із текстурою та освітленням. Широка доступність графічних процесорів революціонізувала роботу в багатьох дисциплінах і створила надзвичайно успішну індустрію відеоігор. Хоча апаратна реалізація алгоритму z-буфера забезпечила чудову інтерактивність за низьку вартість, є три класи програм, які не отримали суттєвої вигоди від цієї революції:

- ті, які мають набори даних набагато більші за кілька мільйонів трикутників;
- ті, які мають неполігональні дані, які важко перетворити на трикутники;
- ті, які потребують високоякісних тіней, ефектів відбиття та заломлення.

У цих класах програм зазвичай використовується алгоритм трасування променів Віттеда [2-4] або розподільний алгоритм трасування променів Кука (DRT), який є на порядок дорожчим за просте трасування променів Віттеда, але дозволяє багато розширених візуальних ефектів. Алгоритм трасування променів краще підходить для величезних наборів даних, ніж алгоритм z-буфера, оскільки він створює зображення в часі, сублінійне за кількістю об'єктів, тоді як z-буфер є лінійним за кількістю об'єктів. Саме більша постійна часу трасування променів і відсутність звичайної апаратної реалізації роблять z-буфер швидшим вибором для невеликих наборів даних. Трасування променів краще підходить для створення тіней, відображень і заломлень, оскільки воно безпосередньо імітує фізику світла. Трасування променів уможливлює такі форми ізольованих запитів видимості, які є проблематичними для алгоритму z-буфера. Трасування променів також забезпечує гнучкість у обчисленні перетину для примітивних об'єктів, що дозволяє безпосередньо представляти неполігональні примітиви, такі як сплайни або криві. На жаль, обчислення цих візуальних ефектів на основі імітації світлових променів обчислюється дорого, особливо на ЦП загального призначення. Алгоритм трасування променів наразі вимагає від десятків до сотень центральних процесорів, щоб бути інтерактивними в повноекранному режимі.

Майже вся тривимірна графіка для настільних комп'ютерів є іграми, і ми віримо, що ця тенденція триватиме. Оскільки трасування променів добре підходить для

підтримки квантового стрибка в здатності ігор підтримувати поверхні вищого порядку, складні моделі та високоякісне освітлення, ми вважаємо, що ігри перейдуть на його використання, коли трасування променів стане достатньо швидким. У цій статті ми стверджуємо, що можливо зробити трасування променів достатньо швидким, і що це означає, що міграція відбудеться. Оскільки така міграція передбачає базову зміну алгоритму, вона матиме значний вплив як на майбутню графіку, так і на курси, пов'язані з іграми. Основна мета цієї статті полягає в тому, щоб проаналізувати ці проблеми та дослідити, як курси, пов'язані з графікою та іграми, можуть вплинути на це.

В останні роки спеціальне обладнання також широко згадувалося в дослідженнях трасування променів. SaarCOR — це конвеєр трасування променів, що складається з блоку генерації/затінення променів, блоку обходу SIMD із чотирма ширинами, блоку списку, блоку перетворення та тестового блоку перетину. Механізм T&I — це архітектура апаратного прискорення тестів проходження та перетину променів. У цій архітектурі використовується метод орієнтації на глибину, щоб зменшити пропускну здатність пам'яті. Він пропонує трифазний перетин променів і трикутника та архітектуру приховування затримки, визначену як блок накопичення променів. SGRT — мобільний графічний процесор із трасуванням променів у реальному часі. В основному він включає дві ключові характеристики: (1) ефективний паралельний конвеєрний обхідний блок; (2) гнучкі та високопродуктивні ядра для затінення та генерації променів. RayCore в основному включає блоки трасування променів (RTU), засновані на уніфікованому конвеєрі обходу та перетину, а також блок побудови дерева (TBU) для динамічних сцен. HART використовує гетерогенні апаратні ресурси: спеціальне обладнання для трасування променів для оновлення BVH і обходу променів і ЦП для реконструкції BVH. Він також використовує PrimAABB для планування обходу. Лі та ін. оптимізували SGRT і запропонували архітектуру обходу двох AABB з двома блоками тестування ray-AABB. Результати експерименту показали, що двоконвеєрна архітектура була в 2,9 рази швидшою за одноконвеєрну архітектуру. Копта та ін. запропонував STRaTA (Streaming Treelet Ray Tracing Architecture), щоб зменшити споживання енергії на масивних паралельних графічних процесорах. Війтанен та ін. застосував MBVH (Multi Bounding Volume Hierarchy) до архітектури прискорювача трасування променів із фіксованою функцією. Завдяки первинним променям енергоефективність покращується на 15%, а продуктивність на площу – на 20%. Іншим підходом до реалізації є кілька потоків, як-от інший підхід до апаратно-прискореної трасування променів, який починається зі зміни порядку операцій рендерингу, запропонований Костянтином Шкурком та ін.. Подвійний підхід організовує доступ до пам'яті трасування променів у два передбачуваних потоки даних. E. Vasiou та ін. представив Mach-RT (Many Chip-Ray Tracing), нову апаратну архітектуру, яка використовується для прискорення трасування променів. Основний підхід поєднує схему впорядкування променів, яка мінімізує доступ до даних сцени, із значним вбудованим буфером, що діє як порядкомп'ютерне сховище, розподілене на кількох мікросхемах.

З точки зору наявних результатів академічних досліджень і комерційних графічних процесорів, прискорення трасування променів поступово покращило обчислювальні можливості разом із прогресом алгоритму та розвитком промислових технологій. Через різні алгоритми, прийняті різними конструкціями, і різні цілі апаратної архітектури трасування променів, багато стратегій мають значні відмінності в продуктивності та накладних витратах на апаратні ресурси. Однак вони мають певні обмеження щодо продуктивності/площі, тому ми зосереджуємося на більш ефективній апаратній архітектурі для трасування променів.

Висновки: Представлено дослідження ефективного механізму RT апаратної архітектури трасування променів, аналізуючи відповідну літературу, алгоритми та реалізації рівня RTL. Кілька стеків використовуються для зберігання інформації про проходження променя. Метод трифазного розриву використовується для раннього виконання розриву циклу та наближення методу зворотної величини для досягнення апаратної оптимізації та підвищення продуктивності. Експериментальні результати дослідників показують, що продуктивність/площа (MRPS/мм 2) архітектури приблизно в 2,4 раза вище, ніж найкращі результати інших академічних досліджень. Ці результати вказують на те, що ця архітектура може забезпечити ефективне трасування променів.

Список використаних джерел:

1. Catmull, E. A Subdivision Algorithm for Computer Display of Curved Surfaces; The University of Utah: Salt Lake City, UT, USA, 1974. [Google Scholar]
2. Whitted, T. An improved illumination model for shaded display. ACM Siggraph Comput. Graph. 1979, 13, 14. [Google Scholar] [CrossRef]
3. Schmid, J.; Uludag, Y.; Deligiannis, J. It just works: Raytraced reflections in “Battlefield V”. In Proceedings of the GPU Technology Conference, San Francisco, CA, USA, 19–22 March 2019. [Google Scholar]
4. Christensen, P.; Fong, J.; Shade, J.; Wooten, W.; Schubert, B.; Kensler, A.; Friedman, S.; Kilpatrick, C.; Ramshaw, C.; Bannister, M. RenderMan: An Advanced Path-Tracing Architecture for Movie Rendering. ACM Trans. Graph. 2018, 37, 1–21. [Google Scholar] [CrossRef]