УДК 004.92

О. Н. Романюк
В. П. Майданюк
Б. В. Бабій

РЕАЛІЗАЦІЯ АНТИАЛІАЙЗИНГУ В ВІДЕОКАРТАХ

Вінницький національний технічний університет

Анотація

Проаналізовано реалізацію антиаліайзингу у сучасних відеокартах на апаратному рівні. Основну увагу приділено роботі графічного конвеєра GPU, зокрема растеризатора, блоків тестів глибини та стенсіла, мультисеплових буферів, шейдерних етапів і механізму злиття результатів. Показано, що антиаліайзинг у GPU є наслідком спеціальної організації пам'яті та обробки даних, а не окремою функцією. Розглянуто особливості обробки підпіксельних семплів, формування маски покриття та використання буферів історії у часових підходахПроаналізовано основні напрями модифікації класичних методів згладжування зображень. Розглянуто просторові, морфологічні, часові та гібридні підходи до зменшення ефекту aliasing у системах реального часу.

Ключові слова: аліайзинг, антиаліайзинг, дискретизація, піксельна сітка, комп'ютерна графіка, GPU, растеризація, згладжування країв, візуальні артефакти.

Abstract

Aliasing is considered as one of the main artifacts of rasterized computer graphics and the implementation of antialiasing in modern video cards at the hardware level is analyzed. The main attention is paid to the operation of the GPU graphics pipeline, in particular the rasterizer, depth and stencil test blocks, multi-sample buffers, shader stages and the result merging mechanism. It is shown that antialiasing in the GPU is a consequence of a special organization of memory and data processing, and not a separate function. The features of processing subpixel samples, formation of a coverage mask and the use of history buffers in temporal approaches are considered. The main directions of modification of classical methods of image smoothing are analyzed. Spatial, morphological, temporal and hybrid approaches to reducing the aliasing effect in real-time systems are considered.

Keywords: aliasing, antialiasing, sampling, pixel grid, computer graphics, GPU, rasterization, antialiasing, visual artifacts.

Вступ

Аліайзинг [1-14] — це ефект спотворення, який виникає під час подання безперервних форм або сигналів у дискретному вигляді, зокрема при відображенні зображень на піксельній сітці. Він проявляється у вигляді зубчастих країв, ламаних контурів або мерехтіння дрібних деталей і є прямим наслідком обмеженої роздільної здатності та недостатньої частоти дискретизації [1]. У контексті комп'ютерної графіки аліайзинг вважається небажаним артефактом, оскільки він знижує візуальну реалістичність сцени та робить цифрове зображення помітно штучним для людського ока.

Результати дослідження

Антиаліайзинг [1-14] у відеокартах реалізується не як єдиний алгоритм, а як сукупність апаратних механізмів у графічному конвеєрі GPU, а також відповідних форматів пам'яті та операцій з обробки результатів рендерингу [2]. Загальна ідея реалізації полягає у тому, що обчислення виконуються не в одній точці на піксель, а в кількох підпіксельних позиціях — семплах, результати яких зберігаються окремо до моменту фінального злиття в один піксель. Альтернативно антиаліайзинг може виконуватися як додатковий повноекранний прохід, де GPU обробляє вже сформований кадр, використовуючи його як текстуру [3].

У випадку мультисеплової реалізації GPU на рівні графічного API отримує параметри render target і буфера depth/stencil із заданою кількістю семплів. Драйвер відеокарти та апаратна частина GPU налаштовують внутрішню організацію пам'яті таким чином, щоб кожному пікселю відповідало N підпіксельних значень. Ці значення не зберігаються у вигляді простого лінійного масиву, а

розміщуються у спеціальній плитковій (tiled) структурі з використанням компресії, що дозволяє зменшити навантаження на відеопам’ять і пропускну здатність [4].

На етапі растеризації GPU визначає, які семпли всередині пікселя покриваються геометричним примітивом. Для цього апаратно формується маска покриття, що описує, які з фіксованих підпіксельних позицій знаходяться всередині трикутника. Оцінка покриття зводиться до швидких геометричних перевірок відносно ребер примітива. Далі виконуються тести глибини та стенсіла, які в мультисепловому режимі здійснюються окремо для кожного семпла. Таким чином, частина семплів може пройти тести, а частина — бути відкинутою, що і створює ефект часткового покриття на межах об’єктів [5].

Шейдерний етап у GPU тісно інтегрований з маскою покриття. У типовій реалізації фрагментний шейдер виконується один раз на піксель, а отриманий колір застосовується до всіх семплів, які залишилися активними після тестів. Це дозволяє суттєво зменшити обчислювальні витрати. Водночас GPU зберігає пер-семплову інформацію для буферів глибини та стенсіла. У спеціальних режимах можливе виконання шейдера з частотою семплів, коли для кожного активного семпла запускається окремий шейдерний виклик, що підвищує точність, але значно збільшує навантаження на обчислювальні ресурси відеокарти [6].

Після завершення рендерингу у мультисепловий render target необхідним є етап злиття результатів. Він полягає у перетворенні N семплів кожного пікселя в одне фінальне значення кольору. У сучасних GPU ця операція зазвичай прискорюється апаратно і реалізується як окремий шлях обробки даних у блоках виводу. Resolve включає масове зчитування семплів, їх усереднення або іншу визначену специфікацією агрегацію та запис у одно-семплову поверхню. Для підвищення ефективності ця операція виконується з використанням кешів і плиткової організації пам’яті, що мінімізує звернення до відеопам’яті [7].

У реалізаціях, де антиаліайзинг здійснюється як повноекранна фільтрація, GPU виконує додатковий рендер-прохід. У цьому випадку поточний кадр використовується як текстура, а піксельний або обчислювальний шейдер читає значення сусідніх пікселів і формує новий кадр. Апаратно це означає інтенсивне використання текстурних блоків, кешів L1/L2 та блоків запису. Висока продуктивність досягається завдяки локальності доступів до пам’яті та можливості повторного використання даних у кешах або спільній пам’яті обчислювальних груп [8].

Окрему групу становлять часові підходи, де GPU зберігає інформацію з попередніх кадрів. Для цього в пам’яті підтримуються буфери історії та векторів руху. У кожному кадрі виконується перепроєкція даних з попереднього кадру в координати поточного, після чого результати комбінуються з новими значеннями. Це складний шейдерний прохід із великою кількістю текстурних вибірок і строгими вимогами до синхронізації ресурсів між кадрами [9].

Окремо слід зазначити, що ефективність антиаліайзингу в сучасних відеокартах значною мірою визначається балансом між якістю зображення та обчислювальними витратами. Будь-яке згладжування потребує додаткових операцій — або збільшення кількості семплів, або виконання додаткових шейдерних проходів, або зберігання допоміжних буферів. Тому виробники GPU приділяють велику увагу оптимізації апаратних блоків, кешів та форматів пам’яті, щоб мінімізувати вплив антиаліайзингу на продуктивність. У результаті сучасні відеокарти здатні виконувати складні операції згладжування в реальному часі без істотного зниження частоти кадрів, що було неможливо на ранніх етапах розвитку графічного обладнання.

Важливим аспектом є також взаємодія антиаліайзингу з іншими етапами графічного конвеєра. Наприклад, якість згладжування безпосередньо залежить від точності растеризації, коректності інтерполяції атрибутів і стабільності обчислень у шейдерах. Помилки або неточності на будь-якому з цих етапів можуть призвести до появи додаткових артефактів, таких як мерехтіння, розмиття або нестабільні контури в русі. Тому антиаліайзинг не можна розглядати ізольовано — він є частиною комплексної системи формування зображення в GPU.

Перспективним напрямом розвитку антиаліайзингу є подальше поєднання апаратних можливостей GPU з алгоритмічними та адаптивними підходами. Уже сьогодні відеокарти активно використовують інформацію з попередніх кадрів, вектори руху та допоміжні буфери для покращення якості згладжування. У майбутньому очікується ще тісніша інтеграція антиаліайзингу з методами реконструкції зображень і машинного навчання, де GPU виконуватиме не лише класичні графічні операції, а й складні обчислення для відновлення візуально правдоподібної сцени.

Отже, антиаліайзинг у відеокартах слід розглядати як одну з ключових технологій, що забезпечує перехід від суто технічного відображення геометрії до візуально переконливих і реалістичних зображень. Його апаратна реалізація в GPU є результатом багаторічної еволюції графічних конвеєрів, пам'яті та шейдерних обчислень і залишається важливим напрямом подальшого розвитку комп'ютерної графіки.

Висновки

Таким чином, антиаліайзинг у відеокартах визначається апаратною організацією обробки семплів, пам'яті та шейдерних проходів. Саме ці механізми дозволяють значно зменшити візуальні артефакти та підвищити перцептивну реалістичність зображень у сучасній комп'ютерній графіці.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Mitchell D. P., Netravali A. N. Reconstruction filters in computer graphics // ACM SIGGRAPH Computer Graphics. 1988. Vol. 22, № 4. P. 221–228.
2. Blinn J. F. What we need around here is more aliasing // IEEE Computer Graphics and Applications. 1989. Vol. 9, № 1. P. 75–79.
3. Akenine-Möller T., Haines E., Hoffman N. *Real-Time Rendering*. 4th ed. Boca Raton : CRC Press, 2018. 1200 p.
4. McGuire M., Luebke D. *Hardware-Accelerated Rendering*. NVIDIA Developer Documentation, 2016.
5. Khronos Group. *OpenGL 4.6 Core Profile Specification*. [Електронний ресурс].
6. Khronos Group. *Multisampling*. OpenGL Wiki. [Електронний ресурс].
7. Microsoft. *Direct3D 12 Graphics Command List Documentation*. [Електронний ресурс].
8. Laine S., Karras T. Efficient Sparse Voxel Octrees // IEEE Transactions on Visualization and Computer Graphics. 2011. Vol. 17, № 8.
9. Karis B. High-Quality Temporal Supersampling // Advances in Real-Time Rendering in Games. SIGGRAPH, 2014.
10. Романюк О. Н., Мельник О. В.. Класифікація методів антиаліазингу. Вісник Вінницького національного технічного університету. 2016. № 4. С. 89–95.
11. Романюк О., Мельник О., Романюк С., Коробейнікова Т., Прозор О. Антиаліайзинг зображення крокової траєкторії відрізка прямої на гексагональному растрі. Modern Engineering and Innovative Technologies. 2022. № 1(22-01).
12. Романюк О. Н., Курінний М. С. Антиаліазинг зображення траєкторії гіперболи. Інформаційні технології та комп'ютерна інженерія. 2022. № 2. С. 42–48.
13. Романюк О. Н., Курінний М. С. Методи та засоби антиаліазингу контурів об'єктів у системах комп'ютерної графіки. Вінниця: УНІВЕРСУМ-Вінниця, 2006. 212 с.
14. Романюк О. Н., Курінний М. С., Романюк С. О., Коробейнікова Т. І., Романюк О.В. Модифікація методу оцінювальної функції для антиаліайзингу векторів /На шляху до індустрії 4.0:інформаційні технології, моделювання, штучний інтелект, автоматизація. Монографія.Одеса, 2021. -С.409-422.

Романюк Олександр Никифорович- д-р техн. наук, професор, завідувач кафедри програмного забезпечення, Вінницький національний технічний університет, м. Вінниця, e-mail: rom8591@gmail.com.

Майданюк Володимир Павлович - канд. техн. наук, доцент кафедри програмного забезпечення, Вінницький національний технічний університет, м. Вінниця, e-mail: maidaniuk2000@gmail.com.

Бабій Богдан Валентинович – студент гр. 4ПІ-22Б, Вінницький національний технічний університет, м. Вінниця, e-mail: rom8591@gmail.com..

Romanyuk Oleksandr Nikiforovych- Dr. of Technical Sciences, Professor, Head of the Department of Software, Vinnytsia National Technical University, Vinnytsia, e-mail: rom8591@gmail.com.

Maidaniuk Volodymyr Pavlovych - Candidate of Technical Sciences, Associate Professor of the Department of Software, Vinnytsia National Technical University, Vinnytsia, e-mail: maidaniuk2000@gmail.com.

Babiy Bohdan Valentinovich – student of group 4PI-22B, Vinnytsia National Technical University, Vinnytsia, e-mail: rom8591@gmail.com.