Міністерство освіти і науки України НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ» Кафедра інформатики факультету мультимедійних систем

Дослідження сучасних технологій ігрової індустрії

Керівник курсової роботи

ст. викладач Борозений С. О.

_____

“____” _____ 2022 р.

Виконав студент

Гусар К. Ю.

“____” _____ 2022 р.

Зміст

Вступ

В наш час, процес програмування був максимально спрощений, для збільшення швидкості розробки, зниження ціни виробництва, спрощення навчання нових спеціалістів, тощо. Тому, складні алгоритми зазвичай сховані в низьких рівнях і розробники не контактують з ними напряму. Наприклад, складні математичні розрахунки, для алгоритмів машинного навчання сховані в бібліотеки, фізика в іграх схована в рушій, а алгоритми швидкого пошуку вже давно написані, і винесені в прості методи для використання розробниками. Звичайно, це «винесення в окремий метод» досить значно впливає на швидкодію, але, через відсутність ліпших альтернатив, відмовитись від нього поки майже неможливо.

Дослідження предметної області

Перші комп'ютерні ігри з'явились ще в 1940 році, але це не ті ігри, до яких ми звикли. Вони були дуже простими, наприклад, pong чи хрестики і нолики. Хотілось би зазначити, що розділення індустрії на періоди є досить складною задачею. Так як значну частину існування індустрії саме консолі використовували провідні технології, було прийняте рішення брати консольні покоління як епохи. В епохах де, на мою думку, необхідні додаткові розділення – розділив на свій розсуд.

Історія

1 покоління консолей, 1972 – 1980. Ігри на консолях, рівно як і на інших пристроях технічно були дуже простими. Зазвичай щось по типу Pong, або текстові ігри. Також в цей час з'явились перші 3д ігри. Також половина існуючих тоді консолей продавалась тільки на азіатському ринку, а ті що лишались були доступними далеко не всім. Всі комп'ютери чи , як їх тоді

отримання або оновлення ЕОМ чекали роками. Доступ був у студентів університетів, де стояли такі машини. Також власті ЕОМ мали серйозні підприємства, де відбувалось проектування літальних апаратів, ракет, машин, в військовій справі.. Причому, окрім університетів, ЕОМ були доступні тільки провідним компаніям того часу, а їх кількість вимірювалась одиницями на компанію та десятками на країну, хоча популярність, як і доступність, звісно, росла. Доступ до машини відбувався через термінал. Багато людей працювали з ЕОМ одночасно, кожен через свій термінал. Машини були несумісні між собою, причому не тільки фізично, на рівні заліза, а ще й через різні мови програмування. Програма написана на одній точно не буде працювати на іншій(іншої моделі) та навіть не факт що запрацює на іншій машині тої самої моделі. Зрозуміло, що ні про яку масовість ігор мова йти не могла, гра працювала тільки на тій машині, під яку була написана. Досить цікавий факт, що найбільше ігор в той час створили в лабораторії NASA. Так як в них, на той момент, були найпотужніші комп'ютери. Також, в цей період вийшла перша гра з використанням векторної графіки.

2 покоління консолей, 1976 – 1984. В цей час, провідні технології зазвичай використовувалась тільки в ігрових автоматах. Консолі також розвивались, з'явилось кольорове зображення(не у всіх), самі консолі стали більш доступними(150-200 доларів), хоча ігри все ще лишались досить примітивними. 3d графіка все ще не була доступна звичайним користувачам. В першу чергу технології використовувались в ігрових автоматах, а вже потім з'являлись в консолях або на домашніх комп'ютерах. Ігри на комп'ютерах були більш технологічними, інколи використовувалась 3д графіка. Це пояснюється технічними можливостями: потужність комп'ютерів була набагато вищою ніж в консолей, а ціна перевалювала за 1000 доларів. Якщо говорити про 3d графіку, перші 3д ігри цього періоду використовували моделі з прозорими полігонами. В 1983 році з'явились перші ігри з замальованими

полігонами. З культових ігор можна виділити серію Ultima для пк, Mario bros на консолях.

3 покоління консолей, 1983 – 1990. В 1985 році почалось використання спрайтів. Спрайти, звичайно, були в 2д, а 3д(псевдо 3д) ігри можна було зустріти тільки на комп'ютерах або на ігрових автоматах. Самі спрайти збільшувались при зближенні з камерою і віддалялись при віддаленні. Також з'явились онлайн ігри. Дуже знаковим став 1984 рік, це рік створення Tetris. Тетріс отримав просто колосальну популярність, побив рекорд по кількості продаж у всій індустрії, а також показав світу що таке ігрова залежність та які можуть бути побічні ефекти від надто довгої гри. було створене поняття «ефект тетрісу». Окрім тетрісу можна виділити наступні ігри: super mario bros, arcanoid, sim city.

4 покоління консолей, 1987 – 1997. Так як комп'ютерний геймінг досить явно конкурував з консольним, домашні системи стали достатньо потужними щоб використовувати найновіші технології епохи. На консолях виходили такі ігри як metalgear, the legend of zelda 2, street fighter, на пк - Wolfenstein 3D, Doom.

Прорив відбувся в 1994 році. Саме в 1994 почали використовувати метод затінення по гуро. Метод дозволяв створювати плавне затінення на об'єктах з низькою кількістю полігонів. Відповідно, самі об'єкти виглядали набагато плавніше. Технологія отримала широке застосування.

5 покоління консолей, 1993 – 2000. Тут, на мою думку, варто згадати про нововведення в області пк(хоча консолей це також стосувалось). Почали виходити дискретні відеокарти. Графіка стала достатньо складною, щоб створити для її обробки окремий пристрій. Пам'ять вже знаходилась на самій відеокарті та була швидшою за звичайну оперативну. Також чіп був спеціально спроектований для рендеру графіки: ядер було на порядки більше ніж в процесорах, а самі ядра були спеціалізованими. Це дало змогу значно

поліпшити графічну складову ігор. Також, однією з консолей цього покоління стала Sony PlayStation. Консоль виявилась вдалою та стала дуже популярною, а за кілька наступних поколінь повністю витіснила Atari з ринку. Приблизно в цей час виник та зник жанр квестів. Якісні квести були технічно складними, та продуманими. Сюжет не був слабкою частиною цих ігор, а ціллю було погрузити гравця в пригоду, де він грав певну роль: приймав рішення, вирішав задачі, справлявся з різноманітними проблемами.. Але, через популярність, з'явилось безліч охочих заробити на феномені. Через це, немало якісних квестів стали непоміченими в ще більшій кількості відвертих підробок. Таким чином сам жанр отримав дурну репутацію та був забутий. Був ще один популярний жанр – стратегії. Вони ж почались з гри 1994 року warcraft:orcs and humans, дуже швидко розвинулись в окремий жанр. Гра робилась на чистому ентузіазмі та неймовірній самовідданості багатьох розробників та отримала дуже широку популярність.

6 покоління консолей, 1998 – 2005. Починаючи приблизно з 2000 року індустрія почала розвиватись з новими темпами. З'явилась справжня 3д графіка. Все що було до цього моменту – всього лиш імітація 3д. Завжди були певні обмеження, яких неможливо було уникнути через технічні обмеження. Наприклад, в doom, моделі ворогів все ще були всього лиш 2д спрайтами які завжди були повернуті до гравця, а рівні мали свої обмеження. В шостому поколінні консолей справжня 3д графіка почала використовуватись усюди. Звісно, ПК геймінг не відставав, там були ті самі можливості. Але, так як в нашому регіоні консолі не були популярними, особисто я можу назвати більше культових ігор на ПК з цього покоління ніж консольних.

Однією з важливих подій епохи став вихід Unreal Engine в 1998 році. Це дуже революційний, для того часу, ігровий рушій. Він працював з фізикою, графікою, штучним інтелектом(npc), файловою системою, також мав готове середовище розробки. За допомогою цього рушія можна було створювати дзеркальні поверхні, а технологія врапінгу дозволяла створювати гарні

портали(це стало особливістю першої версії рушія), також був введений skybox - коробка з накладеною текстурою неба зсередини. Рушій розвивався, виходили нові версії, приблизно в 2004-2005 році вийшов Unreal Engine 3 з підтримкою багатоядерності, сумісністю з консолями наступного покоління та сучасними системами рендера графіки. Також з'явилась підтримка динамічного освітлення. Це був перелік тільки найбільш значимих нововведень.

Також почався рух флеш ігор (пік популярності приблизно 2005-2007 роки). Флеш ігри відіграли дуже важливу роль в розвитку інтернету, а їх автори створили і популяризували багато цікавих та унікальних ідей які використовуються і зараз.

7 покоління консолей, 2005 – 2012. Вийшов Unity, на даний момент один з найпопулярніших ігрових рушіїв. З кожним оновленням добавлялись нові платформи. Таким чином ігри написані на юніті підтримуються більшістю пристроїв(включаючи мобільні). Також консолі почали програвати ПК в продуктивності вкінці покоління. Консолі були критично слабшими за ПК. Причиною цьому стала бульбашка мобільних ігор. Рух флеш ігор започаткував каталоги, після чого вони з'явились усюди. З'явились каталоги застосунків типу playmarket, де також були ігри. А так як в Unity з'явилась підтримка мобільних пристроїв – ринок мобільних ігор активно розширявся. Настільки активно, що великі компанії почали вважати, що мобільний геймінг повністю захопить індустрію. Тому, PlayStation 3, рівно як і інші консолі з покоління, стали скоріше мультимедійними пристроями, аніж ігровими. Отримали якісні дискові приводи, замість потужних комплектуючих. Але, мобільний ринок, будучи новим для індустрії, через великі вливання коштів швидко здувся. Виявилось, що аудиторія дуже непостійна, а успішні сьогодні проекти завтра можуть стати непотребом.

Загалом, 2000-2012 роки, були часом неймовірного росту технічних можливостей. Технології розвивались просто колосальними темпами, тільки

що куплений новий і дорогий комп’ютер через кілька місяців міг безнадійно застаріти. Також, інтернет з’явився в масовому доступі, з достатньою швидкістю для створення повноцінних онлайн ігор. Процесори з одноядерних замінились на 8 ядерні, також з’явився гіпертредінг: технологія яка дозволяє використовувати 1 фізичне ядро як 2 віртуальних, що в свою чергу, дає змогу більш повноцінно загрузити фізичне ядро. Об’єм пам’яті в комп’ютерах виріс з мегабайт до гігабайт, і ближче до 2010 року 4 гігабайти вважались стандартом, а в 2007 системи вже підтримували до 8 гігабайт оперативної пам’яті. Відеокарти також розвивались, причому не меншими темпами. Об’єм пам’яті виріс з 32 мегабайт до гігабайта. Кількість ядер, пропускна здатність, типи, а відповідно і швидкість пам’яті. Росли частоти, мінялись інтерфейси(з AGP на PCI). З’явились нові апаратні елементи в самих відеокартах, що давало змогу використання нових технологій. Техпроцес з 400 нанометрів впав до 40(в 10 разів! Для порівняння, за останні 4, а то й 6 років йшла боротьба за 7 нанометрів, перехід з 14 на 7). Що дало змогу ускладнити мікроархітектуру чіпів без збільшення енергоживлення. Роздільна здатність картинки також кардинально збільшувалась. З 600*600 до hd. З’явились портативні комп’ютери, а саме ноутбуки. Це все серйозно вплинуло на розвиток індустрії. Напевно один з найбільш значимих моментів – вихід Lineage 2 в 2004 році. Це була перша масова MMORPG. Потім, вийшло оновлення World of Warcraft Wrath of the Lich king в 2008. WoW LK став найбільш популярною онлайн грою того часу (більше 10 мільйонів активних підписок). Саме wow набув найбільшої масовості, отримав статус культової гри, та став прообразом сучасних онлайн ігор через свій успіх і популярність. Як наслідок, більшість ігор які виходять сьогодні мають мультиплеєр.

З’явились карти тіней, простий рендеринг динамічного освітлення(раніше всю інформацію, в тому числі рівень освітлення, записували в текстури), зглажування(розділення піселя на сабпікселі та розрахунок рівня його кольору через середнє значення). З’явились пост ефекти, які накладаються вже після рендеру сцен.

8 покоління консолей 2011 – 2019. Так як ажіотаж навколо мобільного сегменту зник, графіка нового покоління з'явилась на консолях. Хоча консолі все ще були слабшими ніж ПК, за допомогою додаткової оптимізації, в тому числі динамічного розширення, використання сучасної графіки стало можливим. Весь прогрес покоління можна описати як оптимізацію. Якщо раніше оптимізація відбувалась паралельно, то тепер вона стала основним напрямком розвитку. Також технологія віртуальної реальності з'явилась в вільному доступі. Звичайно, технологія далека до ідеалу, обладнання дороге, потребує потужних комп'ютерів для роботи, роздільна здатність картинки досить низька.. Все ж, технологія все ще розвивається, а різниця між обладнанням 8 і 9 покоління значна.

9 покоління консолей 2020 – 2026?. Відбулась значна робота з оптимізації затрат на виробництво. А сталося це через вихід Unreal Engine 5. Принципово новими функціями стали: встроєний рей трейсінг, оптимізація сітки моделей(зменшення кількості полігонів), та перевід моделей з різних форматів. Створення моделей високого розширення більше не є проблемою. 3д сканинг став широко доступним, а імпорт моделей в рушій став дуже простим. У відеокартах з'явились 2 нових типи ядер: тензорні та RT. На RT обраховуються промені, а тензорні були створенні для роботи з нейронними мережами. Через це, оптимізація графіки також пішла вперед. З'явилась така технологія як dlss 2.0. Ця технологія дозволяє збільшувати роздільну здатність моделей на етапі пост обробки. Також повноцінна емуляція поведінки світла в реальному часі стала можливою, так як з'явилась технологія рей трейсинг.

Висновки з розділу

- усе що відбувалось до 1 покоління – дуже рання стадія. Комп'ютерні ігри тільки но почали з'являтись.
- 1 – 2 покоління – епоха відвертих експериментів та створення ринку. Ігри стають більш свідомими.

- 3 – 5 покоління – зародження ігрової індустрії, ігри набувають популярності, масовості. Ні в кого вже не виникає сумнівів що це може приносити прибуток.
- 6 – 7 покоління – активний розвиток технологій, як технічних можливостей, так і програмних. Оптимізується процес виробництва, створюються нові ринки, впроваджуються нові ідеї. Технічна складова дуже схожа на сучасну, моделі мають порівняно мало полігонів, а роздільна здатність текстур досить низька, але це питання часу. База закладена і все ще використовується. Саме тут ігри стали справді популярними.
- 2012 – наш час – сміливо можна назвати оптимізацією. Як і процесу виробництва, так і самих ігор. З'являється безліч технологій які роблять необхідні обрахунки дешевшими, натомість, вільні ресурси можна використати для покращення графіки. Зробити фотореалістичну картинку зараз не є проблемою, але запустити цю картинку зможе далеко не кожен, тому оптимізація справді важлива.

Огляд існуючих технологій

Вступ

Протягом всього часу розробка ігор була і лишається одною з найскладніших IT областей, все ще потребує новаторства в плані алгоритмів. Необхідність знання математики для роботи в індустрії, серйозно знижувалась з часом. Але, все ще значна кількість задач не є вирішеними, рішення інших - є дуже ресурсоємким та потребує оптимізації. Тому в цій статі хотілось би описати кілька цікавих технологій, де використовується математика та алгоритми задля створення базового уявлення про застосування математики в ігровій та прилеглих областях. Також будуть пояснення принципів роботи

алгоритмів по мірі необхідності. На мою думку, найцікавіші алгоритми є саме в області графіки.

Введення основних понять

Шейдери

Саме слово «shader», можна грубо перекласти як «затіняючий», тому з назви можна зробити помилковий висновок шо мова йде про логіку освітлення. Насправді, це не зовсім так. В даний момент, слово «шейдер» означає будь-який алгоритм ціллю якого є задання параметрів об’єкту(таких як освітлення, нерівності..) призначений для обробки відеокартою. Шейдери це досить широке поняття. З самого початку, відеокарти в іграх в основному використовувались для обробки світла(іншими словами, створення та виведення на екран графки, хоча по суті - це те саме), тому що процес досить складний та потребує багатьох потоків для виконання і процесор просто не справляється. Але, з ростом потужностей відеоркат(та створенням cuda ядер), слово набуло нового значення, так як виявилось, що окрім обробки світла, існує дуже багато операцій які вигідніше рахувати саме на відеокартах.

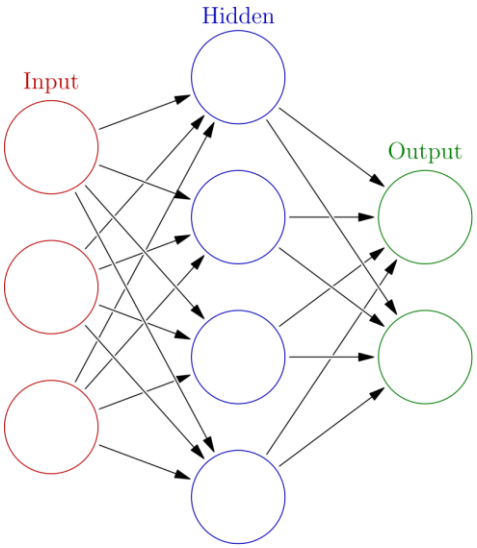
Пост обробка

Поняттям пост обробки називається все, що відбувається після промальовки кадру. Це може бути згладжування, корекція кольору, blur (замилення, розмивання). Пост обробка довгий час активно використовувалась в кіно, після чого з’явилась і в іграх. Кількість технологій пост обробки активно росте.

Нейронні мережі

Вперше поняття штучного нейрону було введене в 50х роках. Тоді йому дали назву «перцептрон». Практичного застосування в цього гіпотетичного поняття не було довгий час. Все змінилось в 2010х роках, коли перші нейронні мережі почали застосовуватись для рішення різноманітних задач. Область застосування обмежується класифікацією інформації та розпізнаванням образів. Це те, в чому нейро мережі обходять всі інші існуючі алгоритми.

Нейрон в цій технології – деяка сутність яка може бути пов'язаною з іншими нейронами. Приймає та видає значення від 0 до 1. Зазвичай об'єднуються в шари, а зв'язок між ними – всі до всіх. Це і є нейронна мережа. Вихідна інформація передається також як набір значень від 0 до 1. Кожен зв'язок має свою вагу, а процес встановлення цих ваг називається навчанням. Коли значення потрапляють в нейрон, зазвичай вони перемножуються на відповідні ваги (значення з зв'язку А на вагу зв'язку А..), проходять через деяку функцію активації, після чого значення йде на вихід нейрону.



Способів навчання є безліч, але більшість з них потребує колосального об'єму даних. Найпростішим алгоритмом є навчання зі вчителем. Є деякий набір набір даних. Вхідні дані, та очікувані вихідні. Необхідно загрузити ці дані в мережу, порівняти правильну відповідь(ту, що міститься в датасеті) з відповіддю мережі. Якщо відповіді сильно відрізняються – необхідна корекція ваг. Ціль – підстроїти їх так, щоб значення функції похибки було мінімальним. Узагалі, всю нейро мережу можна представити як функцію з тисячами, мілінами аргументів. Більшість з них і є ваги. Якщо їх правильно підстроїти, функція починає повертати правильне

значення. Тут задача зводиться до пошуку мінімального значення функції. Але, нажаль, це зовсім не проста задача, так як універсальних алгоритмів для пошуку мінімального значення немає, а сама функція дуже складна.

Огляд алгоритмів

Ray Tracing

Ray tracing – алгоритм обробки світла в реальному часі. В даний момент є два основних кардинально різних підходи для обрахування динамічного освітлення: класичний, та ray tracing. Класичний підхід насправді не має нічого спільного з тим як світло працює в реальному світі. Зазвичай, в випадку з класичними шейдерами, тінь це просто проекція об’єкту з певними деформаціями. Для одного джерела світла це справді цілком прийнятне рішення, але як тільки джерел освітлення стає кілька – алгоритм перестає працювати. Також проблемою лишаються бліки та відображення. Зазвичай відображення це просто ще одна камера направлена в інший бік, та проекція цього зображення на об’єкт. А бліки добавляються на етапі пост обробки. Сама технологія існувала вже досить давно, та активно використовувалась в кіно. Єдине що не дозволяло її використання в іграх –ціна обчислень. Не існувало методу для того, щоб в режимі реального часу обраховувати достатню кількість променів світла для створення сцен. При тому, щоб цей метод могла б собі дозволити хоча б половина аудиторії.

Почалось все з написання алгоритмів під вже існуючі відеокарти. Для більш-менш адекватного рендерингу ігор потрібно було кілька найпотужніших відеокарт, хоча це все ще не вирішало проблем з нестачею потужності. Далі інженери добавили пост обробку нейронними мережами, які «дошліфовували» вихідну картинку, заповнювали пробіли (все, що не встигли обрахувати). Таким чином рендеринг динамічних сцен став можливим, але все

ще надто дорогим для використання в індустрії. Мало хто міг би собі дозволити використання технології, відповідно ніхто не хотів інвестувати гроші в інтеграцію в свою гру нової сирої технології. Наступним кроком, було створення графічного процесора з апаратним пристроєм(так званих «RT» ядер) для обрахунку променів. Першою це зробила компанія Nvidia, ставши лідером в цій області.

Зміст алгоритму до смішного простий: джерело світла випускає певну кількість променів, які прораховуються індивідуально, а що найважливіше одночасно. Це автоматично вирішає проблему з відображенням та кількома джерелами освітлення.

Зліва – RTX справа – стандартні шейдери. Справа – бліки на підлозі, можливо навіть прописані картою тіней. Зліва – це явно не так (повторюсь, освітлення динамічне). Також можна помітити різницю в якості туману. Справа це скоріше дим. Туман рівномірний, більше схожий на напівпрозору текстуру. Зліва – він нерівномірний, візуально більш схожий на туман. На мою думку, різниця очевидна.

Хоча користувачами технологія вважається чимось неймовірним, для розробників це всього лиш ще один метод обробки світла (ще й не доступний

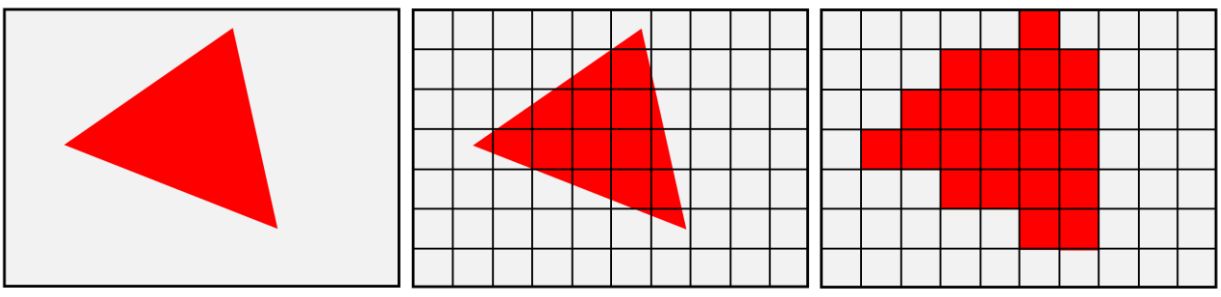
всім, рішення апаратне, є тільки в нових відеокартах). В ньому є не мало плюсів, але метод відносно новий, потрібен час для освоєння. Також розробники ігор вже давно навчились використовувати класичні шейдери, тому є ситуації де технологія виглядає навіть гірше ніж створена розробниками модель, а візуально відрізнити його наявність/відсутність рей трейсинга буває дуже складно. В будь-якому разі, технологія вже була добавлена як стандартна фіча в рушії останнього покоління.

DLSS

DLSS, а саме, DLSS 2.0, Deep Learning Super Sampling, дослівно – технологія глибокого навчання для згладжування. Цей алгоритм має дві цілі – збільшення роздільної здатності та згладжування. DLSS є алгоритмом пост обробки, тобто, працює вже з готовим кадром. Навчання відбувалось на суперкомп'ютерах, а датасетом був набір згладжених зображень надвисокої роздільної здатності та відповідні їм зображення низької роздільної здатності.

Це дозволяє економити багато ресурсів. Замість роботи з тяжкими 4к моделями, відеокарта працює з моделями hd якості. А DLSS на останніх етапах рендерингу кадру піднімає якість зображення та згладжує там необхідно.

Згладжування, саме по собі також було затратною процедурою. Для чого воно потрібно? Поглянемо на картинку знизу. Ми маємо ідеальну фігуру – трикутник. Але, так як зображення на моніторі складається з пікселей, ми не можемо відобразити цей трикутник таким, який він насправді є. Якщо просто зафарбувати пікселі, в яких є частина трикутника, зображення стане нечитабельним.



Через це в іграх з’являються так звані «драбинки». Візуальні нерівності на краях рівних об’єктів. Для того щоб це виправити необхідне згладжування. Алгоритм знаходить ці «незаповнені» пікселі, та відповідно до рівня заповненості визначає відтінок. Більш прозорий колір для об’єкту, там де його менше, та насичений там де більше. Алгоритмів є декілька, але тут я поясню на прикладі SSAA. Піксель розділяється на певну кількість сабпікселів. Наприклад 1 піксель на 4 однакових за розміром сабпікселів. Далі, якщо об’єкт заповнює якийсь з пікселей, колір стає більш насиченим. Операція насправді складна, потребує значний відсоток ресурсів відеокарти, а більшість алгоритмів навіть дорожчі ніж SSAA. DLSS успішно вирішує цю і проблему, за значно меншу вартість.

Процедурна генерація ландшафту

Напевно наступна по масштабу, але не менш цікава область це процедурна генерація. Генерувати можна будь-що, починаючи від анімацій, закінчуючи ландшафтом: гори, річки, печери, а також хмари… список можна легко продовжити. Складність полягає в виборі правильного алгоритму, для необхідних цілей. Наприклад, для генерації ландшафта зазвичай використовується шум. Причому кілька шарів шуму різної перідоичності. Напевно, найцікавішим прикладом використання алгоритмів процедурної генерації є гра Minecraft. Хоча світ досить спрощений (замість плавного ландшафту використовуються вокселі або куби, причому мало чи не метрові) розробники працювали над генерацією протягом більше ніж десяти років. За останній рік було багато роботи над генерацією ландшафту, завдяки чому ландшафт насправді став дуже реалістичним. Якщо коротко: в світах Майнкрафту немає нічого, що б поставили вручну розробники, згенерованим є абсолютно все. Інший цікавий приклад – інді гра Astroneer. Ситуація дуже схожа з майнкрафтом, процедурно генерується все, що оточує гравця, але ландшафт складається вже не з вокселів, а з трикутників. Використовується так званий полігональний ландшафт, який навіть можна деформувати.

А алгоритм перетворення шуму в будь-які матеріальні об’єкти зазвичай виглядає наступним чином: функції шуму мають мінімальне значення -1 та максимальне 1. Ми визначаємо мінімальну та максимальну висоту так, що найнижчій точці буде відповідати значення шуму(далі - value) -1, а найвищій – 1. Далі, з певною періодичністю, value конвертуємо в абсолютні координати нашого 3д простору, value проектується на висоту. З’єднуємо найближчі точки, і будуємо полігони. Для збільшення деталізації, ми накладаємо наступну функцію шуму з меншим періодом, функція проектується на вже згенеровану висоту. Таким чином отримуємо більше деталей. В випадку з річками чи підземними пустотами необхідно використання шумів з іншими параметрами, але загальний алгоритм не сильно відрізняється. За потреби використовуються функції згладжування, обробки водою(емуляція результату ерозії), постобробки загалом.

Процедурна генерація анімацій

Також, дуже цікаві результати приносять процедурно генеровані анімації. Найбільш розповсюдженим прикладом, напевно, буде вода. Підхід з часом набуває все більшого розповсюдження, хоча набагато частіше використовуються намальовані художником анімації. Зазвичай генерується не вся анімація, а деякі окремі елементи. Часто поведінка ніг персонажа описується алгоритмом, це дуже гарно видно на нерівностях ландшафту(картинка зліва, обидві ноги сильно зігнуті в колінах, також кут між гомілкою і ступнею – гострий). Так зразу можу згадати два приклади: один хороший, другий, скоріше поганий(виглядає дуже комічно, це приклад простого алгоритму). The last guardian та human fall flat. В першому випадку, гра розроблялась протягом 7 років. Тоді технології процедурної анімації не були поширеними. Це гра про хлопчика і казкового створіння з іменем

Тріко(схожого на грифона, мабуть). Хлопчик годує це створіння за допомогою бочок з їжею, а воно їх ловить, причому з будь-якого положення, виглядає дуже реалістично. Тріко це поєднання цікавого і складного штучного інтелекту з процедурними анімаціями. За рахунок цього, протягом гри, не виникає сумнівів в його живості. Рівно як і анімації хлопчика, де він біжить, лазить по скелям виглядають надзвичайно живо. Дуже багато процедурної анімації малих об'єктів типу пір'я, одягу, волосся.

Другий приклад – скоріше інді гра. Вона зовсім не серйозна і досить проста. Гравець керує абстрактним гуманоїдним створінням, причому окремо правою і лівою рукою. Ходьба візуально нагадує переміщення п'яної людини, а захвати руками виглядають дуже комічно.

Одним з підходів є використання Regdoll фізики. Якщо коротко – в модель вводяться всі необхідні обмеження(степінь вільності суглобів, наприклад) з використанням фізики твердих тіл. Плюсом в даному випадку є реалістичні падіння, а мінусом – непередбачуваність. З даним типом анімацій можуть виникати дуже комічні ситуації. Також, він не дуже підходить для анімацій окрім падіння, об'єкт фактично перетворюється на тряпку. І керувати ним стає дуже складно. Подібні анімації не будуть реалістичними, а скоріше розхлябаними. Кінцівки рухаються штучно, другою силою яка на них впливає є гравітація. Саме так анімується human fall flat.

Для того щоб вирішити цю проблему використовується інверсна кінематика. Створена вона була саме для керування роботами-маніпуляторами. Зазвичай, задається якась цільова точка якої має досягнути

«кінцівка» метод ж визначається алгоритмами закладеними в програмному забезпеченні, а саме, інверсною кінематикою. Прикладами використання цього підходу будуть Rain World, а також, вищезазначений The last guardian. Проблема непередбачуваності нікуди не зникає, тому баги з фізикою створінь в цих іграх – буденність.

Висновки

Так як ціллю роботи було ознайомлення з існуючими алгоритмами, а також з прикладними областями застосування математики в межах розробки комп’ютерних ігор, в ході написання курсової роботи було проаналізовано безліч існуючих алгоритмів. Акцент був зроблений саме на новітніх алгоритмах, за ними майбутнє.

Під час проведення роботи я більш тісно познайомився з деякими алгоритмами, вибудував більш чітку загальну картину того що відбувалось та відбувається в індустрії зараз. А також, зміг знайти одну з областей практичного застосування отриманих в університеті знань.

Був проведений детальний аналіз предметної області. Частину інформації довелось опустити через те, що були важливіші технології, які також необхідно було описати. Область виявилась значно складнішою ніж мені здавалось першочергово.

Список джерел

Стаття про історію 3д графіки в іграх

Покоління консолей, вікіпедія

Шейдери, вікіпедія

Пост обробка

Штучні нейронні мережі

Ragdoll, вікіпедія

Стаття про процедурну анімацію

Стаття про згладжування

Anti-aliasing, вікіпедія

Мікро архітектура тьюринг

DLSS, вікіпедія