Висновки

Сформульовано та визначено задачу прогнозування подій, що базується на обробці новин за попередній проміжок часу. Зазначено, що задача є комплексною та може передбачати використання такого математичного апарату, як статистичний аналіз, часові ряди, марковські моделі та процеси, асоціативні правила. Проведено огляд публікацій з теми, вказано, які методи розв'язання використовуються частіше інших. Визначено недоліки названих методів та способи, як їх можна уникнути. Запропоновано узагальнену схему розв'язку задачі, що передбачає попередню обробку текстових даних, класифікацію чи кластеризацію даних для їх аналізу, застосування методу прогнозування (заснованого на марковських процесах, часових рядах, асоціативних правилах чи нейронних мережах).

Перелік посилань

1. Discovering and learning sensational episodes of news events / [X. Ao, P. Luo, C. Li та ін.]. // Information Systems. – 2018. – №78. – С. 68–80.
2. Preethi P. Temporal Sentiment Analysis and Causal Rules Extraction from Tweets for Event Prediction / P. Preethi, V. Uma, A. Kumar. // Procedia Computer Science. – 2015. – №48. – С. 84–89.
3. Gerber M. Predicting crime using Twitter and kernel density estimation / Matthew S. Gerber. // Decision Support Systems. – 2014. – №61. – С. 115–125.
4. Realization of a news dissemination agent based on weighted association rules and text mining techniques / [C. Huang, J. Liao, D. Yang та ін.]. // Expert Systems with Applications. – 2010. – С. 6409–6413.
5. Modeling Precursors for Event Forecasting via Nested Multi-Instance Learning / Y.Ning, S. Muthiah, H. Rangwala, N. Ramakrishnan. // the 22nd ACM SIGKDD International Conference. – 2016. – С. 1095–1104.
6. Singh S. Data Mining based Technique for Natural Event Prediction and Disaster Management / S. Singh, R. Khatri. // International Journal of Computer Applications. – 2016. – №139. – С. 34–39.
7. Zhukov D. Stochastic Diffusion Model for Analysis of Dynamics and Forecasting Events in News Feeds / D. Zhukov, E. Andrianova, O. Trifonova. // Symmetry. – 2021. – №13.

УДК 004.415.2.031.43

КАТОЛІКЯН Т. М.

ПРАКТИЧНЕ ВИКОРИСТАННЯ РІЗНИХ МЕТОДІВ АЛГОРИТМІЧНОЇ НА АПАРАТНОЇ ОПТИМІЗАЦІЇ ПРИ РОЗРОБЦІ СИСТЕМИ ДОПОВНЕННЯ РЕАЛЬНОСТІ У РЕАЛЬНОМУ ЧАСІ

У статті розглянуто практичне використання різних методів оптимізації рушія для доповнення реальності у реальному часі. Демонструються ефекти використання різних тактик для прискорення виконання задач з обробки зображень та виявлення на них маркерів доповненої реальності, а також використання тактик прискорення процесу рендерінгу та роботи з камерою. Виконані розрахунки ефективності використання усіх методів.
КЛЮЧОВІ СЛОВА: оптимізація, доповнена реальність, багатопоточність, рендерінг, джоб система

The subject of the article is an application of different methods of optimizations for a real time augmented reality engine. It demonstrates the effects of usage of different tactics to speed up the execution of image processing and AR markers recognition, and tactics of speeding up the rendering process and interaction with a camera device. Performance improvements from all these methods are calculated.
KEYWORDS: optimization, augmented reality, multithreading, rendering, job system

1. Введення

У статті розглянуто практичне використання декількох методів апаратної та алгоритмічної оптимізації, які є особливо доцільними при використанні для рушія доповненої реальності.

Якщо розглянути рушій доповненої реальності як складний багатокомпонентний об'єкт, то його можна поділити на наступні компоненти:

1. Модуль отримання і обробки зображень з абстрактного пристрою (камера)
2. Модуль розпізнавання та розрахунків
3. Модуль рендерінгу
4. Модуль синхронізації

Кожен окремий модуль з наданого списку може бути оптимізований; також, існує простір для глобальної оптимізації взаємодії усіх цих модулів, який полягає у розбитті задач, які виконуються окремими модулями, на набір окремих даних та інструкцій, для можливості подальшого виконання таких «блоків» паралельно. Для цього буде використана оптимізація, що у програмній інженерії має назву «джоб-система» (*Job System*).

Надалі розглянемо оптимізації, що будуть застосовані для кожного модуля окремо.

1. Джоб-система

Основна ідея цього шаблону полягає у тому, що ми створюємо:

- деяке число «потоків» у програмі, які здатні паралельно виконувати обчислення;
- критерії та пріорітети для різних видів обчислень (наприклад, обчислення, що пов'язані з рендерінгом зображення мають найвищий пріорітет та виконуються в окремому потоці);
- потокобезпечну чергу для кожного «типу» обчислень, яка містить дані для виконуваних обчислень

Тобто, при реалізації цієї системи, ми будемо мати змогу відправлями певні обчислення на паралельне виконання, при цьому система розподілу обчислень самостійно знайде вільний (та потрібний) потік для його виконання.

Для реалізації нам потрібно розбити існуючі операції, що виконуються у різних модулях, на окремий та незалежний набір даних та інструкцій. У нашому випадку нам потрібні будуть наступні «джоби»:

- Читання відеопотоку
- Обробка кадру, що надходить з абстрактного пристрою (декодування)
- Робота комп'ютерного зору, розпізнавання маркерів доповненої реальності у кадрі
- Рендерінг попереднього кадру

Варто звернути увагу на те, що виконання рендерінгу попереднього кадру може бути одночасним з обробкою нового кадру. Таким чином, зображення на екрані відставатиме від камери на один кадр. З увахуванням кількості окремих задач, які ми повинні паралелізувати, можна зробити висновок, що чотирьох потоків буде цілком достатньо, якщо врахоувати те, що читання відеопотоку завжди займатиме один потік.

Задача для рендерінгу зображення може виконуватись одночасно із задачею з читання відеопотоку та обробки зображення з камери. У свою чергу, ці 2 задачі можуть виконуватись одночасно з розпізнаванням маркерів.

При практичному розрахунку за допомогою профайлера та використання усереднених даних та пристроїв, отримані наступні результати швидкості виконання різних задач:

Повний кадр – 9.8 мілісекунд;

Очікування завершення попередньої задачі з рендерінгу – 7.2 мілісекунди;

Декодування кадру, що надходить на пристрій – 7.3 мілісекунди;

Рендерінг одного кадру – 9.3 мілісекунди;

Детекція маркерів на зображенні – 5.5 мілісекунд.

Бачимо, що сумарно, операції займають близьмо 30 мілісекунд, але завдяки розглянутій оптимізації ми змогли виконати цілий кадр лише за 9.8 мілісекунд, отримавши приріст більше ніж у 3 рази. При цьому ми жертвуємо лише тим, що постійно відстаємо на один кадр.

2. Оптимізації рендерінгу зображень

Для оптимізації такої задачі, як рендерінг, можуть бути використані різні методи. У нашому випадку, як було описано, ми змогли винести рендерінг у окрему задачу, що може

бути виконана одночасно з обробкою нових даних.

Для того, щоб ми могли одночасно виконувати рендерінг попереднього кадру, та оновлювати дані для наступного, ми використовуємо оптимізацію що має назву подвійний буферінг.

Рис. 1. Схема роботи подвійного буферінгу для рендерінгу

Після початку кадру ми перевіряємо, чи виконується паралельний рендерінг у даний момент. Якщо так, ми повинні дочекатись його завершення. Цього очікування можна було б уникнути використавши потрійний буфер замість подвійного. Так роблять при оптимізації деяких складних навантажених систем, у яких час виконання кадрів може суттємо відрідзятись навіть при роботі програми на системі що має гарні технічні характеристики. Але це створює багато інших проблем. Для нашого випадку буде доцільно використати подвійний буфер та очікувати попередній кадр. Більш того, під час очікування може бути виконана інша корисна робота у потоці синхронізації.

Коли паралельний рендерінг завершується, ми міняємо місцями фронтальний за задній буфер даних, які використовуються для рендерінгу, і після цього починаємо оновлення всіх підсистем рушія, шо призводить до зміни даних у передньому буфері. Потім відбувається кінець кадру.

3. Оптимізації алгоритмів комп'ютерного зору

Для оптимізації обробки зображень та виявлення на них маркерів доповненої реальності ми власноруч напишемо алгоритм, що буде враховувати особливості обраного типу маркерів. У нашому випадку це буде маркер типу *ARTag*. Його особливістю є наступні характеристики:

- Відноситься до поверхневої системи маркерів
- Має квадратні границі
- Розпізнавання базується на граничних лініях, не на кольорах (позбуваємось усіх проблем, зв'язаних з освітленням)

Наш алгоритм виглядатиме наступним чином:

На початку буде проведено розпізнавання границь. Таким чином, буде знайдено перші сегменти ліній. Далі, буде виконане зливання знайдених сегментів ліній у групи. Таким чином, отримаємо лінії повної довжини. Далі, всі отримані лінії будуть розширені базуючись на нашій градієнтній інформації. Далі, згрупуємо лінії у чотирикутники.

Кожен етап з алгоритму виконується послідовно, як частина паралельної операції зі знаходження маркерів. Деякі з операції цього алгоритму також можуть бути виконані паралельно. Наприклад, коли ми займаємося розширенням та групуванням ліній для декількох різних маркерів в одному кадрі. Але, у нашому випадку, виявлення чотирьох маркерів займає близько п'яти мілісекунд, що цілком нас влаштовує, враховуючи те, що у нашій джоб-системі 2 потоки є завантаженими лише на 50%.

Висновки

Було розглянуто основні оптимізації, що було використано для пришвидшення роботи рушія доповненої реальності. При цьому було розглянуто як алгоритмічні рішення (джоб-система, розпізнавання), так і апаратні (рендерінг). Завдяки оптимізаціям вдалося досягти результату у 9.8 мілісекунд для одного фрейму на усередненому пристрої при усереднених вхідних даних, що є дуже гарним результатом. Навіть при таких числах ми маємо період очікування рендерінгу попереднього фрейму в основному потоці, що може бути заповнений виконанням будь-якої корисної роботи протягом 7.2 мілісекунд.

Список джерел

1 Tobias Höllerer, Dieter Schmalstieg “Augmented Reality: Principles and Practice” (2016) – 528 сторінок;
2 Gary Bradski and Adrian Kaehler “Learning OpenCV” (2008) – 571 сторінок;
3 Frahaan Hussain “Learn OpenGL: Beginner's Guide to 3D Rendering and Game Development with OpenGL and C++” (2018) – 208 сторінок
4 Viewing frustum [Електронний ресурс] // Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Viewing_frustum
5 Carnegie Mellon University “Camera matrix” [Електронний ресурс] // Режим доступу до ресурсу: http://www.cs.cmu.edu/~16385/s17/Slides/11.1_Camera_matrix.pdf
6 Camera resectioning [Електронний ресурс] // Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Camera_resectioning

УДК 004.415.25

ПЕТРОВА М.Є.

АРХІТЕКТУРНЕ РІШЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ ТЕХНІЧНОЇ ПІДТРИМКИ КОРИСТУВАЧІВ

У даній системі розглянуто архітектурне рішення програмного забезпечення системи технічної підтримки користувачів та його застосування для підвищення ефективності процесів управління IT-послугами на підприємствах. Продемонстровано роль програмного забезпечення у продуктах для автоматизації Service Desk. Особливу увагу приділено ролі класифікації інцидентів у процесі технічної підтримки користувачів.

КЛЮЧОВІ СЛОВА: Управління IT-послугами, Service Desk, інцидент, послуга, класифікація.

The subject of the article is an architectural solution of customer technical support system software and it’s usage for increasing productivity and effectiveness of information technology service management solutions in enterprise area. The role of this kind of software in Service Desk processes is demonstrated. The process of requests classification in customer support is described in detail.

KEYWORDS: Information technology service management, Service Desk, request, service, classification.

1. Вступ

У даній статті розглянуто архітектурне рішення програмного забезпечення системи технічної підтримки. Продемонстровано можливість підвищення ефективності роботи підприємств шляхом удосконалення архітектури програмного забезпечення системи технічної підтримки користувачів. Також проаналізовано існуючі архітектурні рішення і оцінено ефективність запропонованого рішення. Крім того, в даній статті відображено спосіб підвищення ефективності етапу класифікації запитів у службі технічної підтримки та досягнення вирішення проблеми надмірних заявок.

2. Роль програмного забезпечення Service Desk у ITSM процесах

Протягом останніх років бізнес-процеси активно розвивалися під впливом