УДК 004.5:004.93

МУЛЬТИМОДАЛЬНЕ КЕРУВАННЯ В БРАУЗЕРНИХ ВІДЕОІГРАХ НА ОСНОВІ КОМП'ЮТЕРНОГО ЗОРУ

Бойченко А. М.[1]; Рувінська В. М., к.т.н., проф.[2]
[1, 2] Національний університет «Одеська політехніка»
[1, 2] Україна, Одеса
[1] 7344339@stud.op.edu.ua; [2] victoriya.ruvinskaya@gmail.com

Анотація. *Досліджено проблему безконтактного керування браузерними відеоіграми в умовах обмежень однопотокової моделі JavaScript. Запропоновано асинхронну архітектуру на базі технології Web Worker. Розроблено шестиетапну модель прогресії модальностей вводу від клавіатури до жестів перед вебкамерою. Експериментально підтверджено стабільні 60 FPS рендерингу на ПК і смартфоні.*
Ключові слова: *мультимодальне керування; комп'ютерний зір; браузерна відеогра; доступність відеоігор.*

Вступна частина. Браузерні відеоігри жанру Endless Runner традиційно прив'язані до монотонного контактного вводу - клавіатури, миші або сенсорного екрана, що обмежує ігровий інтерес та унеможливлює взаємодію для осіб із руховими порушеннями верхніх кінцівок [1]. Безконтактне керування за допомогою стандартної вебкамери та бібліотеки MediaPipe [2] є доступною альтернативою спеціалізованим контролерам захоплення руху, проте його реалізація в браузері пов'язана з трьома інженерними викликами: висока обчислювальна складність моделей комп'ютерного зору, що призводить до блокування головного потоку і порушує плавність гри, а шуми вебкамери та тремтіння рук провокують хибні спрацьовування жестів.

Метою дослідження є проєктування та розроблення браузерної відеогри з мультимодальним керуванням, що поєднує комп'ютерний зір із традиційними методами вводу для підвищення інтерактивності, ігрової залученості та розширення доступності геймплею при збереженні стабільних 60 кадрів в секунду (FPS).

Основна частина. Однопотокова модель виконання JavaScript створює фундаментальний конфлікт між ігровим рушієм та моделями комп'ютерного зору. Графічний рушій Three.js для забезпечення плавної картинки вимагає суворого дотримання бюджету кадру ≤ 16.67 мс. У той же час обробка моделями відеокадру з вебкамери (детекція долоні та локалізація 21 просторової точки скелета кисті) на комп'ютерах з потужністю 1.5 TFLOPS (FP32) триває від 18 до 42 мс. Виконання обох процесів у головному потоці неминуче монополізує процесорний час, викликаючи затримки черги подій браузера та падіння частоти кадрів до неприйнятних < 20 FPS.

Для розв'язання цієї проблеми використано технологію Web Worker - фоновий потік операційної системи, що функціонує паралельно з головним потоком JavaScript. Проте реалізація комп'ютерного зору у Web Worker має суттєві архітектурні нюанси, вказані нижче, через які переважна більшість стандартних бібліотек не здатні функціонувати у фоновому середовищі безпосередньо:

1. Відсутність доступу до об'єктної моделі документа (DOM): фоновий потік Web Worker повністю позбавлений доступу до глобальних об'єктів window та document, а також до медіа-елементів HTMLVideoElement і HTMLCanvasElement. Класичні браузерні збірки комп'ютерного зору (наприклад, поширені дистрибутиви OpenCV.js чи ранні версії TensorFlow.js) жорстко розраховані на прямі маніпуляції з DOM-вузлами для зчитування відеопотоку та буферизації кадрів. У зв'язку з цим вибір було зупинено на сучасній модульній бібліотеці MediaPipe Tasks Vision, що підтримує роботу з низькорівневим позаекранним контекстом OffscreenCanvas та бінарними даними без залежності від DOM;

2. Накладні витрати на серіалізацію: штатний механізм обміну повідомленнями postMessage за замовчуванням створює глибоку копію переданих об’єктів у пам’яті. Попіксельне копіювання повнорозмірних відеокадрів створює критичні накладні витрати CPU. Цю проблему вирішено застосуванням концепції нульового копіювання: отриманий відеокадр упаковується в об’єкт ImageBitmap і надсилається до Web Worker через передачу права власності на буфер у пам’яті. Це скоротило міжпотокову затримку передавання кадру до ≤ 0.5 мс на ПК та мобільних пристроях.

Для безперервної підтримки ігрового інтересу, урізноманітнення взаємодії та поступового ускладнення сенсомоторних навичок впроваджено шестиетапну модель прогресивного керування: від клавіатури, миші та свайпів через зональне керування нахилом голови і долонею до розпізнавання конфігурацій пальців і жестів рукою перед вебкамерою. Після проходження всіх етапів відбувається циклічний перезапуск із приростом швидкості персонажа.

Тремтіння ключових точок через шум матриці вебкамери та мікрорухи рук усуває адаптивний фільтр 1-Euro [3]. У стані спокою він знижує координатний шум, при цьому під час швидких жестів пропускає рух без затримок. Хибні спрацьовування під час повернення руки блокуються паузою вводу, коли користувач повертає її у вихідне положення.

Для узгодження двох потоків, та оскільки камера і гра працюють на різній частоті, кожній розпізнаній події присвоюється унікальний номер, завдяки чому гра обробляє одну фізичну дію рівно один раз.

Проведено заміри продуктивності на ноутбуці з потужністю 3 TFLOPS та смартфоні з потужністю 1.8 TFLOPS у браузері Google Chrome при денному освітленні (табл. 1).

Таблиця 1 - Показники швидкодії системи

Параметр системи	Ноутбук	Смартфон	Норматив
Частота гри	60 ± 0.4 FPS	60.0 ± 0.7 FPS	≥ 60 FPS
Час кадру гри	3 - 5 мс	4 - 7 мс	≤ 16.67 мс
Час обробки моделей комп’ютерного зору	18 - 24 мс	30 - 42 мс	≤ 33.33 мс

Виміряні показники підтверджують архітектурні рішення – обробка відеопотоку з вебкамери моделями комп’ютерного зору не впливає на основну ігрову продуктивність, гра відчувається плавно та без затримок кадрів.

Висновки. Реалізована архітектура браузерної відеогри, з винесенням обробки відеопотоку з вебкамери моделями комп’ютерного зору у Web Worker із нульовим копіюванням буферів кадру дозволило усунути блокування головного потоку та забезпечити цільову продуктивність. Запропонована шестиетапна модель взаємодії у поєднанні з фільтрацією тремору алгоритмом 1-Euro та блокуванням хибних жестів підвищує інтерактивність гри й залученість гравця завдяки поступовій зміні модальностей від клавіатури до жестів перед вебкамерою. Доведено життєздатність використання стандартної побутової вебкамери як альтернативного контролера, що розширює доступність ігрового процесу для користувачів із порушеннями моторики рук без потреби у спеціалізованому коштовному обладнанні.

Використання інструментів ШІ. Під час підготовки тез застосовувалася велика мовна модель Google Gemini 3.8 Flash. Мета використання: перевірка орфографії та пунктуації (у всіх текстових розділах роботи), а також допоміжний аналіз й агрегація масиву логів продуктивності (показників FPS, затримок рендерингу й інференсу) для зведення підсумкових метрик у табл. 1 (розділ «Основна частина»). Збір первинних вимірювань, програмна реалізація та формулювання висновків виконані авторами самостійно.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] Wentzel, J., Junuzovic, S., Devine, J., Porter, J. R., & Mott, M. (2022). Understanding how people with limited mobility use multi-modal input. In Proceedings of the 2022 CHI Conference on Human

Factors in Computing Systems (CHI '22), Article 545, 1–16. https://doi.org/10.1145/3491102.3517458
[2] Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C.-L., & Grundmann, M. (2020). MediaPipe Hands: On-device real-time hand tracking. arXiv preprint arXiv:2006.10214. https://doi.org/10.48550/arXiv.2006.10214
[3] Casiez, G., Roussel, N., & Vogel, D. (2012). 1 € filter: A simple speed-based low-pass filter for noisy input in interactive systems. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2527–2530. https://doi.org/10.1145/2207676.2208639
Boichenko A. M.[1], Ruvinska V. M.[2]
Multimodal Control in Browser Video Games Based on Computer Vision
Abstract. *The study investigates the problem of contactless control in browser video games under the constraints of the JavaScript single-threaded execution model. An asynchronous architecture based on Web Worker technology is proposed. A six-stage input modality progression model ranging from keyboard controls to webcam gestures has been developed. Stable 60 FPS rendering on both PC and smartphone is experimentally confirmed.*
Keywords: *multimodal control; computer vision; browser video game; video game accessibility.*