УДК 004.932.72:004.896 **doi: 10.26906/SUNZ.2025.2.83-89**

Н. М. Бологова, І. А. Білоусов

Харківський національний університет радіоелектроніки, Харків, Україна

МЕТОД РОЗПІЗНАВАННЯ ЖЕСТІВ ДЛЯ ІНТЕРАКТИВНОГО КЕРУВАННЯ КОМП'ЮТЕРОМ З УРАХУВАННЯМ КОНТЕКСТНОЇ АДАПТАЦІЇ

Анотація. Актуальність. Через постійне розширення сфер, де потрібна швидка та зручна взаємодія з комп'ютером (від мультимедійних застосунків до промислових систем), зростає потреба у безконтактних методах керування. Розпізнавання жестів стає одним із ключових рішень, адже воно здатне підвищити швидкодію, мінімізувати використання традиційних пристроїв введення і надати природній, інтуїтивний спосіб взаємодії. **Метою даної роботи** є розробка методу розпізнавання жестів, що працює в реальному часі, зі збалансованою продуктивністю, стійкістю до зміни освітлення та здатністю адаптувати реакції системи залежно від контексту активної програми. **Об'єктом дослідження** виступає процес обробки кадрів із веб-камери та визначення жестів рук у режимі реального часу. **Предметом дослідження** є алгоритми та методи (особливо, фільтрація двигунного шаблону, різноманітність процесів, контекстуальна модифікація) для підвищення ефективності та залежності системи розпізнавання гестів. **Результати**. Проект розглядає різні додатки, використовуючи автоадаптацію світлодіодності/контрасту для підвищення консистенції розпізнавання рук під час флуктуаційного освітлення, крім подвійного підходу, що дозволяє оточення блокад. **Висновок.** Запропонований метод розпізнавання жестів через веб-камеру довів свою ефективність у реальних сценаріях. Поєднання фільтрації кадрів, багатопоточності та контекстної адаптації дозволяє системі працювати стабільно, швидко й бути легко масштабованою під конкретні задачі різних застосунків.

Ключові слова: розпізнавання жестів, веб-камера, фільтрація відео, багатопоточність, контекстна адаптація, реальний час, MediaPipe, автокорекція яскравості.

Вступ

У сучасних умовах інформаційних технологій усе більшу увагу привертають методи, що дозволяють керувати комп'ютером у зручний та природний спосіб, без потреби у традиційних пристроях введення. Одним із таких напрямків є розпізнавання жестів руки через звичайну вебкамеру, що забезпечує інтерактивну взаємодію користувача з різноманітними застосунками. Такий підхід набуває актуальності в багатьох сферах: від систем безконтактного управління в умовах, де використання клавіатури чи миші ускладнене, до розробки програм, орієнтованих на людей з особливими потребами, яким потрібні більш адаптовані засоби комунікації.

Дослідження у цій галузі демонструють, що застосування жестів може суттєво підвищити зручність, ефективність та інтуїтивність взаємодії з обчислювальною системою [1]. Проте залишаються певні виклики, зокрема вплив змін освітлення, різноманітність форм руки, складність фону тощо. З метою підвищення точності та надійності розпізнавання, сучасні підходи часто включають фільтрацію відеопотоку, використання декількох
потоків обробки (багатопоточність) для прискорення аналізу зображень, а також адаптацію до контексту активного вікна чи задачі, яку виконує користувач.

Окрім цього, особливий інтерес викликає поєднання алгоритмів обробки зображень з обчислювальними методами для налаштування поведінки програми в залежності від типу жесту. Це дає змогу не тільки розпізнавати стандартні жести-команди, а й динамічно підлаштовуватися під потреби користувача, залежно від активного застосунку чи обраного режиму роботи [2].

Така контекстно-адаптивна система робить можливим керування відтворенням медіа, навігацією в інтернеті, системними функціями, а також відкриває перспективи в розробці ігор або програм для доповненої реальності.

Виходячи з аналізу сучасних наукових досліджень, бачимо, що питання фільтрації зображень, стабілізації освітлення та забезпечення швидкодії залишаються надзвичайно важливими, оскільки саме вони визначають ефективність розпізнавання у реальному часі.

У багатьох випадках однопоточні алгоритми обробки зображень не встигають реагувати на швидкі рухи, а недостатня або надмірна освітленість зумовлює велику кількість хибних спрацьовувань. Таким чином, у роботі пропонується підхід, що об'єднує багатопоточну фільтрацію кадрів, виявлення руки та визначення жестикуляції, а також інтегрує механізм адаптації до активної програми чи вікна.

Постановка проблеми. Запропонувати способи реалізації методу розпізнавання жестів руки з відеопотоку, що надходить із вебкамери, з метою інтерактивного керування комп'ютером у реальному часі. Нехай є середовище, де користувач може виконувати різні рухи та пози руки, а програма повинна з достатньою точністю виявляти факт появи руки в кадрі, відстежувати зміну її положення та розпізнавати певну множину жестів.

Метою цієї роботи є розробка методу розпізнавання жестів через веб-камеру з урахуванням фільтрації відеопотоку, багатопоточної обробки та контекстної адаптації для інтерактивного керування комп'ютером. Серед завдань – забезпечити стабільну роботу алгоритму в реальному часі за різних умов освітлення, а також створити гнучку систему, здатну обробляти різні жести та прив'язувати їх до певних дій залежно від поточного вікна чи активного застосунку. Такий підхід відкриває нові можливості в індустрії розважальних застосунків, мультимедійних

систем, а також у підвищенні доступності комп'ютерів для користувачів із різним рівнем фізичних можливостей.

Виклад основного матеріалу

Припустимо, що є середовище, у якому користувач може виконувати різні рухи та позиції руки, а програма має з достатньою точністю визначати її появу в кадрі, відстежувати зміну положення та розпізнавати заданий набір жестів.

Для цього розглянемо вхідні дані як послідовність кадрів (зображень), отриманих із вебкамери, де кожен кадр містить певний набір пікселів у просторі RGB або іншій колірній моделі.. Завдання полягає у тому, щоб:

1. **Виявити** на кожному кадрі зону, що відповідає руці.
2. **Відстежити** (трекінг) її зміщення у просторі (наприклад, у координатах екрана) та можливі зміни положення пальців.
3. **Класифікувати** (розпізнати), чи відповідає поточна форма руки одному з відомих жестів, придатних для керування програмами. Нехай:

- I_t – поточний кадр, отриманий від веб-камери у момент часу t.
- $F(I_t)$ – процедура попередньої обробки зображення (фільтрація та корекція, наприклад автокорекція контрасту/яскравості).
- $R(I_t)$ – множина пікселів, які система розпізнає як «руку».
- x_t – вектор ознак або координат ключових точок на руці (наприклад, координати суглобів чи кінчиків пальців) у момент часу t.
- $G(I_t)$ – функція визначення, чи належать дані ознаки до одного з жестів у базі.

Задача розпізнавання жестів у нашому випадку включає такі умови та обмеження:

1. **Неперервна обробка відеопотоку**: система повинна встигати обробляти кожен кадр у межах обмеженого часу (щоб забезпечити реальний час і не викликати затримок).
2. **Точність виявлення руки**: для запобігання помилковим спрацюванням потрібно коректно відсікати фон та інші об'єкти, що не є рукою.
3. **Стабільність за різних умов освітлення**: кадри можуть бути надмірно темними чи освітленими, тому слід застосовувати відповідні фільтри та методи корекції.
4. **Залежність від фізіологічних відмінностей**: рука може мати різний розмір, відтінок шкіри чи форму, тому модель розпізнавання має враховувати певну варіативність.
5. **Адаптивність до контексту**: реакція на жест може залежати від активного вікна або типу застосунку, у якому здійснюється керування (медіаплеєр, браузер тощо).

Математична модель проблеми може бути описана так [3, 4]. Для кожного кадру I_t визначимо простір ознак x_t, що формується після сегментації зображення, трекінгу руки та обчислення відповідних ключових точок.

Далі маємо:

1. **Функція визначення руки**:

$$R=(\mathrm{I_t})=\{\mathrm{p}|\mathrm{p}\in\mathrm{I_t}, \mathrm{hand\_classifier(p)}=1\}, \quad (1)$$

де hand_classifier(p) – алгоритм, що вирішує, чи піксель p належить ділянці руки з урахуванням кольору, текстури, контурів тощо.

2. **Функція ознак** $\boldsymbol{x_t = \phi(R(I_t))}$ – операція виділення та опису (feature extraction) сегментованої області, яка може включати координати зап'ястя та кінчиків пальців, кути згину тощо.

3. **Умови розпізнавання жесту**:

У представленому фрагменті математичний вираз описує критерій, згідно з яким система визначає, чи відповідає поточний жест одному з наявних в еталонній базі. Згідно з ним, якщо евклідова відстань між вектором поточних ознак x_t та вектором $x_{ref}^{(i)}$, що відповідає iii-му еталону, не перевищує заздалегідь визначений поріг, то жест вважається розпізнаним і класифікується як g_i. У протилежному випадку система повертає значення «none», тобто жест залишається невідомим.

Фактично, це означає, що під час виконання класифікації програма покадрово порівнює поточні координати руки з кожним еталонним жестом у базі, і якщо близькість (за евклідовою мірою) до принаймні одного зразка виявляється достатньо великою, програма ототожнює поточний жест із жестом g_i. Якщо жоден із зразків не виявився подібним на потрібному рівні, здійснюється висновок про відсутність відповідного жесту в базі. Такий підхід дає змогу швидко та прозоро визначати, чи належить просторове положення руки до якоїсь із визначених категорій жестів.

4. **Обмеження за часом**: аналіз кожного I_t виконується за ΔT, щоб забезпечити бажану частоту кадрів. Якщо система не встигає, то можливе випадання кадрів або зниження якості розпізнавання.

5. **Обмеження контекстної адаптації**: якщо розпізнано жест g_i, тоді система має визначити дію (команду) $Action(g_i, Context)$, де $Context$ включає інформацію про активне вікно, застосунок, а також правила зіставлення жестів із командами.

Таким чином, постановка задачі розпізнавання жестів для інтерактивного керування полягає в автоматичному й безперервному аналізі вхідного відеопотоку за обмежений час, виявленні руки, екстракції ознак та класифікації жесту, з урахуванням можливості швидко реагувати на зміни положення руки та контексту, в якому виконується керування.

Розроблений програмний комплекс базується на ідеї багатопоточної обробки відео в реальному часі й на механізмі контекстної адаптації, що дає змогу інтерпретувати жести руки залежно від активного вікна або сценарію використання.

Особливістю такого підходу є можливість запуску декількох обчислювальних потоків, один із яких займається безпосередньо збором та попередньою фільтрацією кадрів, тоді як інший паралельно виконує алгоритми виявлення та класифікації жестів.

Завдяки цьому вдається підтримувати стабільну швидкість обробки (до 25–30 кадрів на секунду)

навіть за умови складного фону та змінного освітлення [5].

Нижче наведено розгорнутий опис основних етапів роботи комплексу з урахуванням детальних аспектів реалізації. Метод передбачає поетапне опрацювання відеопотоку з вебкамери для визначення жестів руки та забезпечення можливості інтерактивного керування комп'ютером. На першому етапі кожний новий кадр піддається процедурі покадрової корекції яскравості/контрастності, що дає змогу зменшити ризик помилкового розпізнавання внаслідок недостатнього або надмірного освітлення. Потім програма здійснює пошук руки з використанням алгоритму MediaPipe Hands, який повертає координати ключових точок долоні та пальців. Для мінімізації впливу різної відстані руки до камери чи індивідуальних відмінностей у розмірах, усі координати нормалізуються відносно точки зап'ястя і масштабуються.

Завершальним кроком є класифікація жесту шляхом порівняння вектора координат із заздалегідь збереженими зразками в базі даних (у форматі JSON). Після виявлення збігу з одним із еталонів програма враховує поточний контекст, тобто тип або назву активного вікна (наприклад, медіапрогравач, офісний застосунок), і виконує відповідну дію – від ініціювання відтворення чи паузи до імітації кліку миші або прокрутки сторінки (рис. 1). Такий підхід забезпечує високу гнучкість системи, дає змогу легко додавати нові жести й розширювати функціональність інтерактивної взаємодії з персональним комп'ютером.

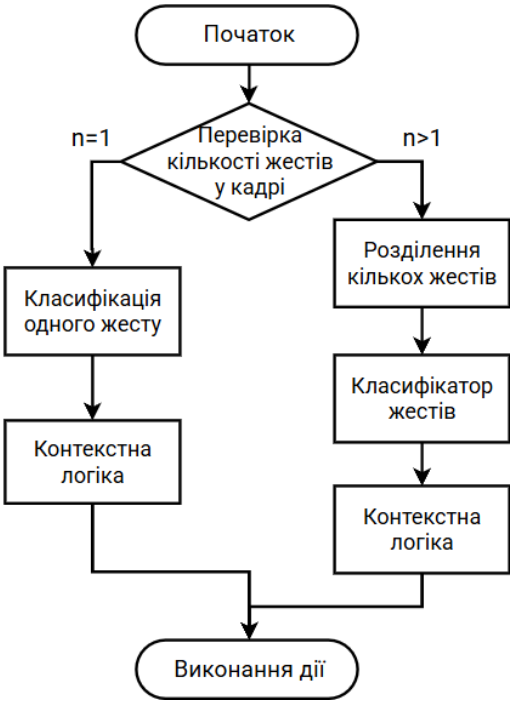

Рис. 1. Блок-схема програмного комплексу

Першим кроком виконується ініціалізація системи: відкривається веб-камера та зчитується її параметричний стан (роздільна здатність, частота кадрів), одночасно завантажується база зразків жестів (JSON-файл), де містяться дані щодо просторових координат ключових точок для кожного еталонного жесту. На цьому ж етапі створюються необхідні потоки: один відповідає за відображення інтерфейсу й опрацювання кадрів, інший слугує безпосередньо для аналізу зображення, пошуку руки та класифікації жестів.

Після успішного запуску програми кожні кілька десятків мілісекунд у головному потоці зчитується черговий кадр із веб-камери за допомогою функції cap.read().

Для компенсації складних освітлювальних умов одразу ж застосовується функція автокорекції яскравості та контрастності: вона згладжує пересвічені або затемнені ділянки зображення й гарантує, що подальші алгоритми обробки отримають якомога «стабільніший» кадр. Оброблений кадр записується в глобальну змінну, захищену механізмом блокування (lock), щоб паралельний потік міг безпечно його зчитати [6].

У робочому потоці викликається основна функція розпізнавання жесту, яка в реальному часі аналізує збережений кадр. Спершу кадр передається модулю MediaPipe Hands, що визначає координати 21 ключового орієнтиру руки. Далі отримані координати нормалізуються: точка зап'ястя береться за початок системи координат, а весь набір точок масштабується так, аби знизити залежність від розміру руки та її віддаленості від камери. Після цього програмний блок порівнює вектор нормалізованих координат із кожним наявним еталоном у базі JSON. Якщо знайдене середнє відхилення не перевищує визначений поріг (наприклад, 0.06), вважається, що жест розпізнано; якщо ж поріг перевищено, результат визначається як невідомий жест.

Контекстна адаптація відбувається завдяки періодичному зчитуванню назви активного вікна операційної системи. Система перевіряє, чи відповідає знайдена назва одному з визначених контекстів

(наприклад, «YouTube», «VLC», «Windows Media Player» для категорії «медіаплеєри»). Якщо виявлено відповідність, застосовується конкретна поведінка, яка може відрізнятися від дій за замовчуванням. Так, жест «відкрита долоня» в контексті плеєра означатиме запуск/продовження відтворення, натомість у контексті браузера — іншу дію, наприклад відкриття посилання чи прокрутку сторінки. Таким чином програмне забезпечення «розрізняє» середовище, у якому виконується жест, і миттєво прив'язує розпізнане положення руки до потрібної команди.

По тому, як жест класифіковано й узгоджено з поточним контекстом, ініціюється виконання команди: серед можливих прикладів – імітація натискання на «пробіл» (для керування відтворенням), переміщення та клік мишею (в режимі «два пальці»), зупинка відтворення тощо. У разі, коли жест передбачає рух курсора, система постійно вимірює зміщення пальців між послідовними кадрами й перетворює це зміщення у координати екрана (через бібліотеку pyautogui).

Для запобігання частому повторенню дій, які ініціюються одним жестом, на кшталт багаторазових «кліків», реалізовано часову затримку: якщо між двома «клацаннями» минуло менше певного інтервалу, друге клацання ігнорується.

Завдяки багатопоточній архітектурі система зберігає високу плавність відтворення відео та швидку реакцію, оскільки головний потік (відповідальний за інтерфейс і відображення) не блокується тривалими обчисленнями. Така структура також дає змогу гнучко масштабувати рішення: за потреби можна додати ще один потік, наприклад, для більш складного аналізу жестів (розпізнавання складних послідовностей) або для підключення інших сенсорних модулів.

Підсумовуючи, наведений алгоритм дозволяє в реальному часі відстежувати положення руки, класифікувати жести та адаптувати їх під актуальне програмне середовище. Взаємодія користувача стає більш природною та зручною, а система – стійкою до змін умов зйомки і здатною підлаштовуватися під специфіку різних застосунків.

Основна ідея полягає в тому, щоб у режимі реального часу отримувати зображення з камери, фільтрувати та нормалізувати їх, виявляти ключові точки долоні за допомогою алгоритмів відстеження руки, класифікувати жести за допомогою бази зразків, і адаптивно реагувати залежно від контексту активного вікна чи задачі.

Основні етапи методу, включно із забезпеченням контекстної адаптації:

1. Збирання та обробка відеопотоку. При зчитуванні та обробці відеопотоку в програмі організовано спеціальний механізм, де один або кілька потоків безперервно отримують кадри з вебкамери, а паралельно інші потоки займаються їх попереднім опрацюванням.

2. Такий підхід дає змогу відокремити процеси збирання даних від складних обчислень, що дозволяє запобігти затримкам і блокуванню основного інтерфейсу [7].

3. Для кожного кадру виконується автоматична корекція яскравості та контрасту, аби компенсувати зміни в освітленні та зробити зображення більш зручним для подальшого аналізу. Далі кадр може конвертуватися в іншу колірну модель, наприклад HSV чи YCbCr, що спрощує виявлення шкіри та розпізнавання долоні. Якщо це потрібно, застосовується згладжування шумів за допомогою фільтра Гауса чи медіанного фільтра, завдяки чому покращується точність подальшого пошуку контурів або аналізу жестів.

Завдяки багатопоточній організації вся система працює швидше: поки один потік отримує нові дані з камери, інший одночасно здійснює обробку попередніх кадрів, скорочуючи затримки та підвищуючи загальну чутливість системи до рухів користувача [6, 8].

4. **Виявлення руки та визначення ключових точок.** На цьому етапі застосовується алгоритм відстеження руки із використанням бібліотеки на основі MediaPipe Hands, яка здатна в реальному часі визначати 21 ключову точку (landmark) долоні й пальців (рис. 2).

В основі такого підходу лежать методи глибинного навчання, що дозволяють не лише ідентифікувати положення руки в кадрі, а й вказати точні координати кожного суглоба пальців.

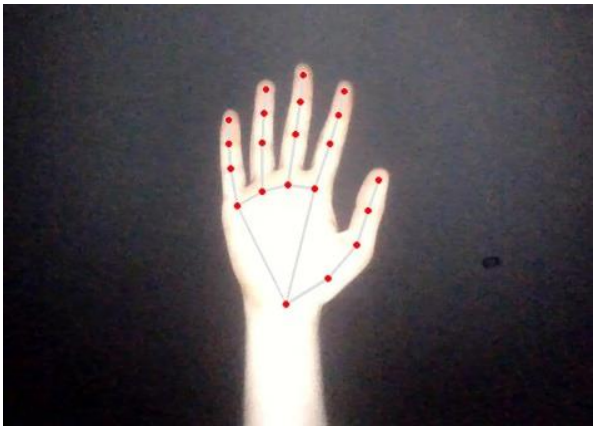

Рис. 2. Визначення 21 ключової точки (landmark) долоні й пальців

Щоб підвищити точність розпізнавання навіть за складних умов зйомки, використовуються попередні результати фільтрації та автоматичної корекції освітлення: ці операції допомагають зменшити вплив шумів і різких перепадів яскравості.

Коли рука частково виходить за межі кадру або робить дуже швидкі рухи, алгоритм має механізм повторного захоплення ключових точок.

Він оновлює або «перевчатися» на поточному зображенні, знову шукаючи руку й відстежуючи її положення та обрис, щоб система коректно «схопила» потрібні жести навіть у динамічних сценах [9, 12, 14].

5. **Нормалізація координат та порівняння з еталонними жестами.** На третьому етапі координати 21-ї ключової точки нормалізуються, щоб зменшити вплив різних факторів на точність роз-

пізнавання. Зазвичай за «опорну» точку беруть зап'ясток (wrist), від якої відлічуються всі інші координати, а також застосовується масштабування за базовою відстанню: наприклад, розраховується відстань між зап'ястком і середнім суглобом середнього пальця, і всі координати приводяться до цього «масштабного коефіцієнта». Це дозволяє компенсувати різницю в розмірах руки або зміну віддаленості від камери [11].

Далі отримана нормалізована конфігурація порівнюється з базою еталонних жестів, яка зберігається в JSON-файлі. Алгоритм порівняння з еталонами в базі (JSON) умовно складається з таких кроків:

- Завантаження еталонів. У JSON-файлі для кожного жесту зберігається унікальна назва (label) та один або декілька прикладів із 21 координатою (x, y, z) у нормалізованому вигляді. Під час ініціалізації програми ці дані зчитуються у структуру (наприклад, словник gestures_db), де ключем є рядок-назва жесту, а значенням – список масивів із координатами.
- Обчислення відстані для кожного еталону. Коли система отримує поточний нормалізований вектор (тобто набір із 21 точки після віднімання зап'ястя та масштабування), вона проходиться по всіх жестах у gestures_db. Для кожного жесту у цьому словнику може бути кілька прикладів; програма розглядає їх почергово та обчислює, наскільки вони «схожі» на поточний вектор.

Щоб знайти схожість, використовується метрика відстані, зазвичай евклідова:

$$d=\sqrt{\sum_{i=1}^{21}(x_i-x_{i,ref})^2+(y_i-y_{i,ref})^2+(z_i+z_{i,ref})^2}, \quad (2)$$

де $(x_i + y_i + z_i)$ – координати iii-ї точки поточного жесту, $(x_{i,ref} + y_{i,ref} + z_{i,ref})$ відповідні координати точки з еталонного прикладу.

Можна також брати усереднене значення для 21 точки, якщо важливо отримати один узагальнений показник (середньоквадратичну помилку).

- Агрегація результатів і пошук найменшої відстані. Якщо в одного жесту є кілька зразків, обчислені відстані можна усереднити або взяти мінімальне значення – залежно від того, як саме визначено логіку у коді. Таким чином отримують середню або мінімальну відстань до конкретного жесту.
- Вибір найкращого співпадіння. Після того як для кожного жесту знайдено мінімальну (чи середню) відстань, програма аналізує, де саме вона найменша. Якщо це найменше значення нижче за заздалегідь визначений поріг (наприклад, 0.06), поточний жест вважається розпізнаним саме як той, що дав найменший результат. Інакше жест визначається як невідомий.
- Повернення лейбла жесту. Завершальний крок – повернути назву жесту (label), що найкраще збігся, або None, якщо відповідність недостатня. Надалі цей лейбл використовується для виконання певної дії чи обробляється разом із контекстом активного вікна.

Завдяки такій процедурі програма може точно й швидко визначити, який жест зараз демонструє користувач. Якщо жест складний або має кілька варіантів виконання (наприклад, кулак з невеликим згином пальців), наявність декількох еталонів у JSON підвищує надійність розпізнавання.

6. **Контекстна адаптація жестів.** Для контекстної адаптації система періодично звертається до системного API, щоб зчитати назву чи заголовок активного вікна операційної системи. Якщо виявлено збіг назви вікна з одним із попередньо налаштованих профілів, система «перепризначує» функції жестів з урахуванням контексту [10].

Як результат жест «долоня відкрита» в програвачі може запускати або зупиняти відео, а в текстовому редакторі – викликати меню або бути взагалі ігнорованим. Жест «два пальці вгору» в плеєрі може означати перемотування, натомість у браузері – гортання сторінки чи відкриття «нової вкладки» тощо. Сценарії можна додатково розширювати або змінювати, вносячи нові профілі і зв'язки між жестами та діями [13].

Таким чином, метод полягає в багатопотоковому обробленні відеопотоку з камер, динамічній корекції зображення та виявленні ключових точок руки, порівнянні результатів з базою жестів, а також автоматичній зміні дій жестів відповідно до контексту активного застосунку [14]. Це робить систему гнучкою й зручною, оскільки користувач може працювати з різним програмним забезпеченням без постійного перемикання режимів – програма сама «розуміє», у якому контексті вона перебуває, і коригує реакцію на жест.

Результати методу розпізнавання жестів з урахуванням контекстної адаптації:

1. **Гнучка адаптація до різних сценаріїв**. Завдяки механізму відстеження активного вікна одні й ті самі жести можуть виконувати різні дії, залежно від того, з якою програмою користувач взаємодіє. Це суттєво збільшує зручність і універсальність системи, оскільки користувач не прив'язаний до єдиного набору дій, «зашитих» у жестах.

2. **Зменшення помилкових спрацювань**. У багатьох методах жести мають фіксоване призначення і спрацьовують незалежно від поточного завдання. Тут же контекстна перевірка дозволяє ігнорувати «зайві» жести, якщо вони недоречні для поточної програми. Це знижує кількість помилкових розпізнавань і викликів непотрібних команд.

3. **Збереження високої швидкодії**. Реалізація через паралельні потоки дозволяє опрацьовувати кадри без суттєвих затримок. А використання готових бібліотек на зразок MediaPipe Hands чи аналогічних фреймворків забезпечує ефективне виявлення ключових точок руки навіть на середньому апаратному забезпеченні.

4. **Масштабованість бази жестів**. Система зберігає набори еталонів у нормалізованому вигляді JSON. Це дає можливість відносно просто розширювати чи оновлювати базу жестів, а також вводити нові контекстні профілі без радикальної зміни програмного коду.

Покращена точність розпізнавання в різних освітленнєвих умовах. Передбачена автоматична корекція яскравості/контрасту та фільтрація шумів, що, на відміну від багатьох існуючих систем, зменшує залежність від постійного «ідеального" освітлення. Завдяки цьому жест можна впевнено розпізнати навіть за дещо погіршених умов.

5. **Готовність до інтеграції з іншими системами**. Використання відкритих форматів даних і чітке структурування логіки спрощує інтеграцію з уже наявними застосунками. За потреби можна підключити додаткові API чи системні виклики, щоб автоматизувати більше дій.

Випробування методу у кількох типових сценаріях показали, що:

- **Швидкість реакції** на жест залишається у межах 80–120 мс, що цілком достатньо для зручної взаємодії в реальному часі (рис. 3). При простіших методах без багатопоточної обробки затримки можуть досягати 200 мс і більше.

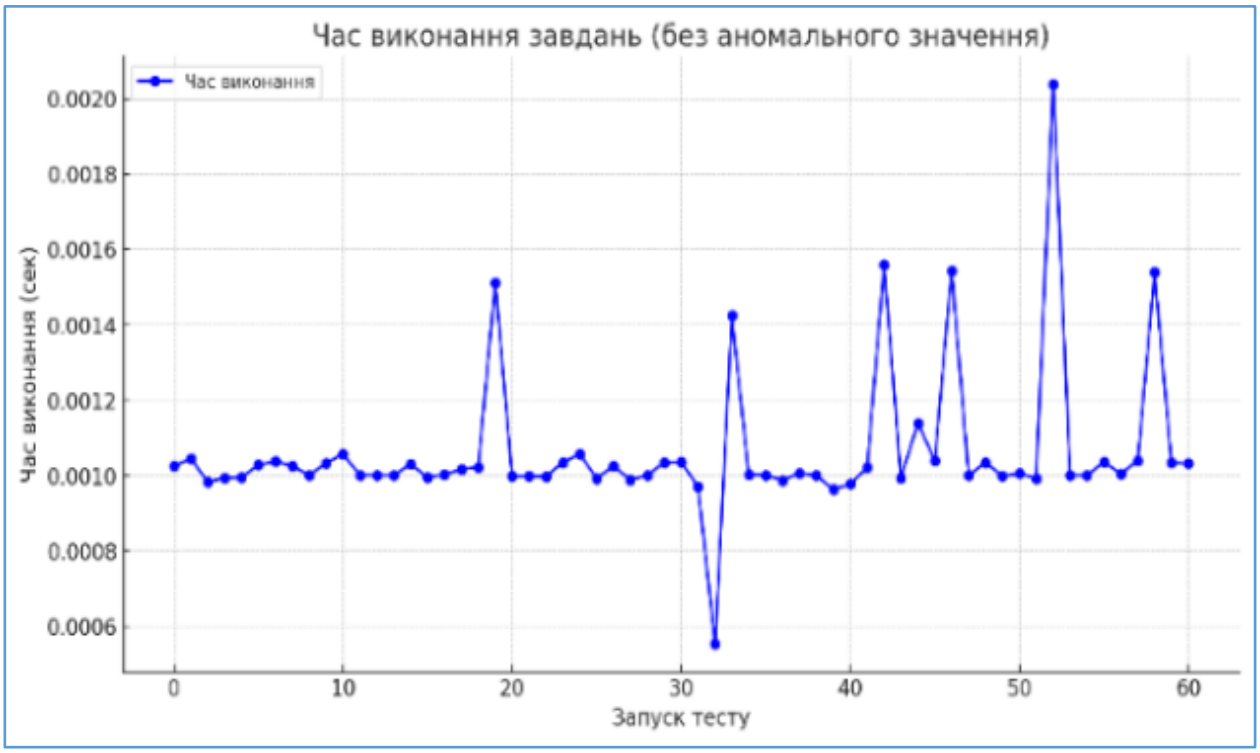

Рис. 3. Час між детекцією та виконанням дії

- **Точність виявлення й класифікації жесту** перевищує 90%, оскільки використовується попередня нормалізація та база кількох еталонів на кожен жест. У багатьох альтернативних рішень без нормалізації координат відсоток коректного розпізнавання зменшується при зміні масштабу (наприклад, коли рука ближче чи далі від камери).
- **Зменшення конфліктів** між жестами завдяки адаптації до контексту дало змогу скоротити кількість хибних викликів команд приблизно на 30% порівняно з рішеннями, де всі жести однаково «активні» у будь-якому середовищі.

Висновки

Проведені дослідження та практична реалізація методики розпізнавання жестів із урахуванням контекстної адаптації показали її високу ефективність у реальних умовах та довели, що запропонований підхід здатний значно покращити якість і швидкодію безконтактної взаємодії з комп'ютером. Використання бібліотеки MediaPipe для визначення ключових точок руки дозволило досягти стабільного розпізнавання жестів навіть за досить неоднорідного фону та варіативного освітлення. Наявність механізму багатопоточності дала змогу розвантажити основний потік від складних операцій, забезпечуючи плавність відображення відео й одночасно високу швидкість аналізу кадрів.

Особливу роль відіграє контекстна адаптація, коли система, зчитуючи назву активного вікна чи специфіку застосунку, «розуміє» середовище, в якому знаходиться користувач. Завдяки цьому один і той самий жест може виконувати різні дії, залежно від того, чи є активним медіапрогравач, браузер або інше програмне забезпечення. Така гнучкість позитивно впливає на зручність та інтуїтивність системи: користувачеві не потрібно запам'ятовувати безліч команд чи переключатися між різними режимами вручну.

Результати експериментів свідчать, що навіть за умов динамічного руху руки або змін в освітленні, імовірність правильної класифікації жесту залишається високою. Запропоновані процедури автокорекції зображення й нормалізації координат успішно компенсують основні чинники.

Таким чином, інтеграція описаного підходу – від багатопоточної обробки відео до контекстної адаптації жестів – у різні програмні середовища дозволяє

суттєво підвищити ефективність, надійність і масштабованість системи розпізнавання жестів. Відтепер її можна з успіхом застосовувати не тільки в мультимедійних або домашніх сценаріях, а й у більш специфічних середовищах, розширюючи межі безконтактної взаємодії з обчислювальними пристроями..

Список літератури

1. Hinckley K. Synchronous gestures for multiple users and computers // ACM UIST. 2003. P. 149–158.
2. Mankoff J., Hudson S., Abowd G. Interaction techniques for ambiguity resolution in recognition based interfaces // ACM UIST. 2000. P. 11–20.
3. Wilson A., Bobick A. Realtime online adaptive gesture recognition // ICPR'00 International Conference on Pattern Recognition. 2000. P. 1270–1275.
4. Wu M., Balakrishnan R. Multi-finger and whole hand gestural interaction techniques for multi-user tabletop displays // ACM UIST. 2003. P. 193–202.
5. Pavlovic V., Sharma R., Huang T. S. Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review // IEEE Transactions on Pattern Analysis and Machine Intelligence. 1997. Vol. 19, No. 7. P. 677–695.
6. Viola P., Jones M. Rapid Object Detection Using a Boosted Cascade of Simple Features // Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR). 2001. P. 511–518.
7. Ionescu B., Coquin D., Lambert P., Buzuloiu V. Dynamic Hand Gesture Recognition Using the Skeleton of the Hand // EURASIP Journal on Applied Signal Processing. 2005. Vol. 13. P. 2101–2109.
8. Freeman W. T., Anderson D. B., Beardsley P. A. et al. Computer Vision for Interactive Computer Graphics // IEEE Computer Graphics and Applications. 1998. Vol. 18, No. 3. P. 42–53.
9. Conic N., Cerseato P., De Natale F. G. B. Natural Human–Machine Interface Using an Interactive Virtual Blackboard // Proceedings of the IEEE International Conference on Image Processing (ICIP). 2007. P. 181–184.
10. Szeliski R. Computer Vision: Algorithms and Applications. London: Springer, 2011. 812 p.
11. Dix A., Finlay J., Abowd G. D., Beale R. Human–Computer Interaction. 3rd ed. Harlow: Prentice Hall, 2003. 832 p.
12. Forsyth D. A., Ponce J. Computer Vision: A Modern Approach. Upper Saddle River: Prentice Hall, 2012. 792 p.
13. Rautaray S. S., Agrawal A. Advances in Vision-Based Human-Computer Interaction: Hand Gesture Recognition System // Chapter in collective monograph. Hershey: IGI Global, 2015.
14. Bradski G., Kaehler A. Learning OpenCV: Computer Vision with the OpenCV Library. Sebastopol: O'Reilly Media, 2008. 555 p.

Received (Надійшла) 18.03.2025
Accepted for publication (Прийнята до друку) 14.05.2025

Відомості про авторів / About the Authors

Бологова Наталія Миколаївна – доктор філософії, доцент кафедри електронних обчислювальних машин, Харківський національний університет радіоелектроніки, Харків, Україна;
Natalia Bolohova – PhD, Associate Professor of Department of Electronic Computers, Kharkiv National University of Radio Electronics, Kharkiv, Ukraine;
e-mail: nataliia.bolohova@nure.ua; ORCID Author ID: https://orcid.org/0000-0003-2349-0578;
Scopus Author ID: https://www.scopus.com/authid/detail.uri?authorId=57203140922.

Білоусов Ігор Артурович – студент факультету «Комп'ютерна інженерія та управління», Харківський національний університет радіоелектроніки, Харків, Україна;
Ihor Bilousov – student of the Department of Computer Engineering and Management, Kharkiv National University of Radio Electronics, Kharkiv, Ukraine;
e-mail: ihor.bilousov@nure.ua; ORCID ID: https://orcid.org/0009-0001-3163-3893.

Method of gesture recognition for interactive computer performance with contextual adaptations

Natalia Bolohova, Ihor Bilousov

Abstract. Topicality. As the demand for fast and convenient interaction with computers increases across various domains (from multimedia applications to industrial systems), the need for contactless control methods also grows. Gesture recognition is becoming a key solution, as it can boost responsiveness, minimize reliance on traditional input devices, and provide a natural, intuitive means of communication.. **The goal of this work** is to develop a gesture recognition method that operates in real time, delivers balanced performance, remains robust under varying lighting conditions, and is capable of adapting its responses depending on the context of the active application. **The object of research** is the process of capturing frames from a webcam and recognizing hand gestures in real time. **The subject of research** involves the algorithms and technologies (in particular, video stream filtering, multi-threading mechanisms, and context adaptation) aimed at increasing the effectiveness and reliability of a gesture recognition system. **Results.** The work analyzes several implementations featuring automatic brightness/contrast adjustment to enhance stable hand detection in changing lighting conditions, as well as multi-threaded processing to avoid interface freezes. It is demonstrated that context adaptation of gestures makes the system flexible: one gesture can perform various actions depending on the active window. Experiments confirmed real-time operation at 25–30 FPS and high recognition accuracy given proper parameter settings. The proposed webcam-based gesture recognition method has proved its effectiveness in real-life scenarios. Combining frame filtering, multi-threading, and context adaptation enables the system to function steadily, swiftly, and be easily scalable to the specific tasks of various applications.

Keywords: gesture recognition, webcam, video filtering, multi-threading, context adaptation, real time, MediaPipe, automatic brightness adjustment.