RAPORT NAUKOWO-TECHNICZNY

Lekki system rozpoznawania gestów dłoni do sterowania prezentacją

Przegląd modalności pomiarowych, projekt architektury i protokół walidacji empirycznej

CHARAKTER OPRACOWANIA

Analiza naukowo-techniczna oraz projekt badania porównawczego. Wnioski dotyczące doboru technologii mają status uzasadnionych rekomendacji inżynierskich, a nie wyników własnego eksperymentu.

1 listopada 2026 r. • wersja 1.0

Streszczenie

Rozpoznawanie gestów dłoni stanowi alternatywną metodę sterowania prezentacją multimedialną, lecz jego użyteczność zależy przede wszystkim od ograniczenia niezamierzonych aktywacji. Celem opracowania jest porównanie technologii realizujących bezdotykowe przełączanie slajdów oraz zdefiniowanie architektury lekkiego systemu, działającego lokalnie na komputerze bez dedykowanego procesora graficznego. Rozważono analizę ruchu obrazu RGB, śledzenie punktów charakterystycznych dłoni, gotowe klasyfikatory gestów statycznych, stereowizję głębi, specjalizowane śledzenie podczerwone, radar, ultradźwięki i czujniki inercyjne. Zaproponowano modularny potok przetwarzania, regułowy detektor przemieszczenia poziomego i automat stanów z mechanizmem blokady wielokrotnych wywołań. Głównym kandydatem do implementacji jest RGB z MediaPipe Hand Landmarker; OpenCV stanowi wariant kontrolny, natomiast RealSense i Ultraleap alternatywy sprzętowe. Sformułowano protokół oceny skuteczności zdarzeń, fałszywych aktywacji na godzinę, opóźnienia end-to-end, stabilności oraz kosztu obliczeniowego. Wskazano ograniczenia interpretacji danych technicznych i konieczność testów na niezależnych użytkownikach. Nie przedstawia się nowych wyników empirycznych.

Słowa kluczowe: interakcja człowiek–komputer, rozpoznawanie gestów, MediaPipe, OpenCV, kamera RGB, Ultraleap, RealSense, interfejs bezdotykowy, walidacja systemu.

1. Wprowadzenie i definicja problemu

Interfejs bezdotykowy umożliwia zmianę slajdów bez kontaktu z klawiaturą, myszą lub pilotem. Z perspektywy interakcji człowiek–komputer (HCI) nie wystarcza jednak rozpoznanie ruchu ręki: system musi odróżnić polecenie od gestykulacji towarzyszącej wypowiedzi, a następnie wygenerować dokładnie jedno zdarzenie sterujące. Dlatego podstawową jednostką oceny jest zdarzenie przełączenia slajdu, nie pojedyncza poprawnie zaklasyfikowana klatka.

Rozpatrywany system ma działać w tle, na typowym komputerze przenośnym, bez połączenia z usługą chmurową i bez dodatkowego GPU. Jego podstawowe funkcje obejmują rozpoznanie gestu poziomego przesunięcia dłoni (swipe), rozróżnienie kierunku, kontrolę aktywnego okna, wysłanie pojedynczej komendy oraz natychmiastową dezaktywację. Klawiatura lub klasyczny pilot pozostają niezależną drogą sterowania.

Analiza ma charakter przeglądu narracyjnego technologii i projektu metodologicznego. Nie jest metaanalizą ani systematycznym przeglądem publikacji. Parametry czasowe i wydajnościowe przedstawione dalej są wartościami inicjalnymi do strojenia lub celami akceptacyjnymi, nie zmierzonymi rezultatami prototypu.

2. Wymagania operacyjne i kryteria oceny

Zdolność do pracy w czasie rzeczywistym nie może być sprowadzona do nominalnej liczby klatek na sekundę. W zadaniu prezentacyjnym koszt błędnego przełączenia bywa większy niż koszt ponowienia pominiętego gestu. Wymagania należy zatem traktować wielokryterialnie, z odrębnymi miarami trafności, ryzyka błędu, reaktywności i obciążenia systemu.

Tabela 1. Operacjonalizacja kryteriów oceny interfejsu gestowego.

Wymiar	Zmienna / sposób pomiaru	Interpretacja
Skuteczność	Recall i precision dla zdarzeń	Wykrycie zamierzonego gestu i właściwej komendy
Fałszywa aktywacja	FP na godzinę bez zamierzonych gestów	Ryzyko samoczynnego przełączenia slajdu
Opóźnienie	P50, P95 i rozkład czasu end-to-end	Responsywność od zakończenia gestu do reakcji prezentacji
Stabilność	Liczba awarii i utrat śledzenia	Ciągłość działania podczas wystąpienia
Zasoby	CPU, RAM, ewentualnie GPU i energia	Koszt utrzymywania aplikacji w tle
Sterowalność	Test bramkowania kontekstu oraz wyłącznika	Możliwość bezpiecznej kontroli działania

Warunkiem poprawnej interpretacji jest uprzednie ustalenie okna czasowego dopasowania polecenia do gestu oraz reguły traktowania wywołań powtórnych. Komenda o nieprawidłowym kierunku nie może być uznana za prawidłową detekcję; podwójne przełączenie po jednym geście należy rejestrować osobno.

3. Modalności i algorytmy percepcji

3.1. Kamera RGB i heurystyczna analiza ruchu

Najprostsza architektura opiera się na akwizycji wideo przez OpenCV, ograniczeniu obszaru zainteresowania (ROI), odejmowaniu tła (np. MOG2) i wyznaczaniu przepływu optycznego. Zaletami są niski koszt sprzętowy, brak potrzeby uczenia modelu i łatwość prototypowania. U podstaw podejścia leżą jednak założenia względnie stabilnej kamery, oświetlenia oraz tła [4, 5].

Ruch postaci w tle, przechodzący uczestnik spotkania, zmiana ekspozycji lub poruszenie kamery mogą wywołać sygnał podobny do gestu. Dlatego wariant heurystyczny jest wartościowym punktem odniesienia i potencjalnym rozwiązaniem w kontrolowanej scenie, lecz wymaga szczególnie starannej oceny fałszywych aktywacji.

3.2. Detekcja punktów dłoni przez MediaPipe

MediaPipe Hand Landmarker zwraca 21 punktów charakterystycznych dłoni i estymację jej strony. Tryby VIDEO i LIVE_STREAM wykorzystują śledzenie międzylatkowe, ograniczając potrzebę ponownego uruchamiania pełnej detekcji. W trybie asynchronicznym zadanie może pominąć klatkę, gdy poprzednia jest jeszcze przetwarzana; dla sterowania na bieżąco rozwiązanie to pozwala ograniczyć narastanie opóźnienia [1, 3].

Trajektorię dłoni można wyznaczać na podstawie nadgarstka i stawów śródręczno-paliczkowych (MCP), co zmniejsza zależność od chwilowego ułożenia palców. MediaPipe Gesture Recognizer dostarcza między innymi klasy Open_Palm i Closed_Fist, lecz nie udostępnia standardowej klasy Swipe. Konieczna jest zatem własna logika czasowa lub klasyfikator sekwencji [2].

3.3. Głębia i specjalizowane śledzenie w podczerwieni

Kamery stereoskopowe RealSense umożliwiają wyznaczenie położenia obiektu w osi głębokości i ograniczenie przestrzeni roboczej dłoni do wybranego przedziału odległości. W rodzinie D435 wykorzystuje się dedykowany układ przetwarzania głębi D4. Nie zwalnia to z odróżnienia świadomego polecenia od przypadkowego ruchu ręki [6, 7].

Ultraleap Leap Motion Controller 2 korzysta ze specjalizowanego śledzenia dłoni w podczerwieni. Potencjalną zaletą jest uzyskanie danych geometrycznych bez implementowania pełnej analizy obrazu RGB; ograniczeniami pozostają koszt urządzenia, konfiguracja runtime, zgodność bindings/SDK i wrażliwość optyczna na intensywne promieniowanie lub odbicia IR [8, 9].

3.4. Radar, ultradźwięki i jednostki inercyjne

Radar częstotliwości milimetrowych umożliwia analizę mikrogestów bez rejestrowania klasycznego obrazu osoby. Systemy ultradźwiękowe wykorzystują odpowiedź akustyczną, natomiast jednostki IMU rejestrują przyspieszenie i prędkość kątową. Technologie te są mniej zależne od obrazu RGB, ale w omawianej aplikacji zwykle wymagają specjalizowanego sprzętu, integracji lub elementu noszonego przez użytkownika. Wysokich wyników uzyskanych w innych zadaniach laboratoryjnych nie wolno bezpośrednio przenosić na swobodną gestykulację podczas prezentacji [11].

Tabela 2. Jakościowa porównywalność modalności dla funkcji przełączania slajdów.

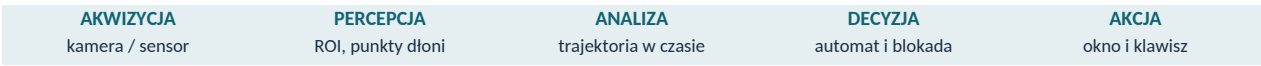
Wariant	Najważniejsza przewaga	Ograniczenie	Przeznaczenie
OpenCV / RGB	Minimalny stos i niski koszt obliczeń	Tło, oświetlenie, ruch kamery	Punkt odniesienia
MediaPipe / RGB	Geometria 21 punktów dłoni	Błędy śledzenia; własny detektor swipe	Wariant podstawowy
Gesture Recognizer	Gotowe gesty statyczne	Brak natywnego swipe	Opcjonalne uzbrajanie
RealSense / depth	Separacja przestrzeni w osi Z	Dodatkowe urządzenie i integracja	Scena złożona
Ultraleap / IR	Śledzenie dedykowane dłoni	Sprzęt, runtime i zgodność SDK	Wariant specjalizowany
Radar	Brak klasycznej rejestracji obrazu	Ograniczona dostępność desktopowa	Kierunek badawczy
Ultradźwięki	Niewielka ilość danych	Niestandardowy tor akustyczny	Kierunek badawczy
IMU	Niezależność od warunków obrazu	Wymaga urządzenia noszonego	Zastosowania szczególne

Porównanie ma charakter jakościowy. Bez jednolitego protokołu badawczego nie można formułować rankingu dokładności ani podawać jednego uniwersalnego obciążenia CPU dla tych technologii.

4. Architektura proponowanego systemu

Architektura powinna rozdzielać percepcję od decyzji o emisji polecenia. Umożliwia to późniejszą wymianę modułu akwizycji (RGB, depth, IR) bez modyfikowania reguł bezpieczeństwa i interakcji. Sugeruje się pięć etapów logicznych, połączonych krótkotrwałym przekazywaniem zdarzeń i znaczników czasu.

Rysunek 1. Potok przetwarzania i podejmowania decyzji o przełączeniu slajdu.



Akwizycja dostarcza klatki i stabilnego znacznika czasu. Percepcja zwraca obecność dłoni oraz jej reprezentację geometryczną. Moduł temporalny utrzymuje krótkie okno trajektorii. Automat stanów rozstrzyga, czy gest został rozpoczęty, potwierdzony albo odrzucony. Warstwa integracyjna dopuszcza wysłanie polecenia

wyłącznie wtedy, gdy aplikacja prezentacyjna ma odpowiedni kontekst wejścia, a system został świadomie aktywowany.

Odczyt z kamery i inferencję należy oddzielić kolejką o ograniczonej pojemności, preferującą najnowszą obserwację. Przy przeciążeniu przetwarzanie każdej zaległej klatki jest mniej uzasadnione niż utrzymanie niskiej latencji. Czynności kosztowne obliczeniowo nie powinny wykonywać się w callbackach obsługujących systemowe wejście [10].

5. Formalizacja detektora gestu poziomego

5.1. Reprezentacja trajektorii

Niech $p(t) = (x(t), y(t))$ oznacza uśrednione położenie środka dłoni w układzie współrzędnych obrazu, znormalizowanych do $[0, 1]$. W oknie $W = [t_0, t_1]$, gdzie $t_1 > t_0$, wyznacza się przemieszczenia i średnią prędkość poziomą:

$$\Delta x = x(t_1) - x(t_0) \qquad \Delta y = y(t_1) - y(t_0) \qquad v_x = \Delta x / (t_1 - t_0).$$

Wystąpienie gestu może być modelowane jako koniunkcja trzech warunków: minimalnego przemieszczenia, dominacji ruchu poziomego i minimalnej prędkości. Aby częściowo uwzględnić odległość dłoni od kamery, próg przemieszczenia można dodatkowo odnieść do obserwowanej szerokości dłoni s.

$$G = (|\Delta x| > \max(\theta_d, \alpha s)) \wedge (|\Delta x| > \kappa |\Delta y|) \wedge (|v_x| > \theta_v).$$

Parametry θ_d, θ_v, α oraz κ muszą zostać dobrane na zbiorze kalibracyjnym. Alternatywnie można użyć odpornej estymacji prędkości w oknie przesuwnym zamiast różnicy punktów skrajnych. Znacznik czasu powinien być monotoniczny w obrębie procesu; w wywołaniach MediaPipe VIDEO należy zapewnić ściśle rosnące znaczniki milisekundowe [1].

5.2. Automat stanów i zdarzenie sterujące

Proponowany automat: NIEAKTYWNY → GOTOWY → KANDYDAT → ZATWIERDZONY → BLOKADA → GOTOWY. Do stanu KANDYDAT przechodzi się po stwierdzeniu obecności dłoni i rozpoczęciu zgodnego ruchu. Zatwierdzenie następuje po spełnieniu kryteriów czasowo-przestrzennych i kontekstowych. W stanie ZATWIERDZONY wysyłana jest dokładnie jedna komenda; BLOKADA uniemożliwia kolejne polecenie przez ustalony czas.

Stan kandydata należy zerować po utracie śledzenia, przekroczeniu maksymalnego czasu gestu, zmianie kierunku, przejściu poza ROI lub wyłączeniu sterownika. Ponowne uzbrojenie nie może opierać się na dawnej trajektorii. Przeciwdziała to fałszywemu rozpoznaniu po przerwie w detekcji dłoni.

Dla sterowania dwukierunkowego znak Δx jest mapowany na „następny” i „poprzedni” slajd. Mapowanie trzeba skonfigurować zgodnie z odbiciem lustrzanym podglądu i położeniem kamery. Użycie tego samego klawisza Space dla obu kierunków jest niewłaściwe, jeżeli użytkownik oczekuje dwóch różnych działań.

5.3. Gest uzbrajający

Statyczny gest otwartej dłoni, utrzymywany przykładowo przez 150–250 ms, może ograniczyć niepożądane wyzwolenia przed gestem swipe. Rozwiązanie ma jednak koszt interakcyjny: wydłuża operację, może zwiększać liczbę gestów pominiętych i nie jest z definicji skuteczniejsze. Porównanie konfiguracji z uzbrajaniem i bez niego stanowi odrębny element eksperymentu.

6. Parametry implementacyjne i wydajność

W architekturze CPU-first główny potencjał optymalizacji wynika z ograniczenia objętości przetwarzanych danych. Dla wariantu podstawowego warto rozpocząć od obrazu 640 × 360 lub 640 × 480 pikseli, jednej dłoni, ROI i przetwarzania około 10–15 klatek na sekundę. Są to wyłącznie konfiguracje pilotażowe. Wymagana częstotliwość analizy zależy od czasu i dynamiki gestu, a jej zmniejszanie należy kontrolować przez pomiar odsetka detekcji.

Tabela 3. Parametry startowe i warunki ich walidacji.

Parametr	Punkt początkowy	Wymagany test
Obraz wejściowy	640 × 360 / 640 × 480 px	Porównać recall i opóźnienie dla kilku rozdzielczości
Częstość analizy	10–15 kl./s	Sprawdzić pominięcia szybkich gestów
Liczba dłoni	1	Zweryfikować zdarzenia przy dwóch widocznych dłoniach
Cooldown	700–1000 ms	Zbadać duplikaty i szybkość kolejnych poleceń
Próg przestrzenny	Normalizowany do rozmiaru obrazu i dłoni	Walidacja dla różnych odległości od kamery
Tryb MediaPipe	VIDEO lub LIVE_STREAM	Pomiar P95 i stalego obciążenia CPU

Wariant OpenCV wymaga filtracji maski pierwszoplanowej i analizy kierunku przepływu optycznego. MediaPipe powinno śledzić jedną dłoń, wykorzystywać punkt środkowy dłoni i zarządzać historią z ograniczonym oknem czasowym. W ścieżce depth należy stosować rzeczywistą skalę głębokości czujnika; surowe wartości formatu Z16 nie mogą być automatycznie interpretowane jako milimetry [7].

Integracja z systemowym wejściem wymaga odrębnych adapterów platformowych. Mechanizmy Windows, macOS oraz sesji X11/Wayland różnią się uprawnieniami i zachowaniem. Należy uwzględnić dostępność sterowania klawiaturą w aktywnym procesie, bez zakładania pełnej przenośności jednej biblioteki [10].

7. Bezpieczeństwo funkcjonalne, prywatność i ergonomia

Najważniejszym zagrożeniem operacyjnym jest wygenerowanie komendy podczas zwykłej gestykulacji prezentera lub poza kontekstem wystąpienia. W konsekwencji projekt powinien mieć domyślny stan nieaktywny, jednoznaczny wskaźnik działania, możliwość natychmiastowego wyłączenia oraz bramkowanie na poziomie wybranej aplikacji i jej trybu prezentacji.

Mechanizmy cooldown, debounce i pojedynczej emisji zdarzenia powinny być realizowane w jednej warstwie decyzyjnej, niezależnej od modelu percepcji. Błędy kamery, brak klatek, niepoprawne znaczniki czasu i utrata fokusu aplikacji powinny prowadzić do stanu bezpiecznego bez generowania wejścia. Opcjonalny dźwięk lub wskaźnik w zasobniku systemowym powinien sygnalizować zatwierdzenie gestu, bez ingerencji w obraz wyświetlany publiczności.

Minimalizacja danych zakłada przetwarzanie obrazu wyłącznie lokalnie, bez trwałego zapisu surowych klatek. Dopuszczalna telemetria eksploatacyjna obejmuje stan automatu, czas zdarzenia, typ komendy, informacje o błędzie i zagregowane wskaźniki wydajności. Testy z udziałem ludzi wymagają odrębnego określenia zasad zgody, anonimizacji i okresu przechowywania pomiarów.

8. Projekt badania empirycznego

8.1. Pytania badawcze i hipotezy

Główne pytanie badawcze dotyczy tego, czy detektor oparty na landmarkach dłoni osiąga mniejszą liczbę niezamierzonych przełączeń niż heurystyczny detektor ruchu obrazu, przy porównywalnym opóźnieniu i akceptowalnym koszcie zasobowym.

Hipoteza H1: MediaPipe z analizą trajektorii zapewni mniejszy wskaźnik FP/h niż OpenCV w scenach z ruchliwym tłem. Hipoteza H2: zastosowanie gestu uzbrajającego zmniejszy liczbę fałszywych aktywacji, ale wydłuży czas obsługi. Hipoteza H3: ograniczenie częstotliwości inferencji zmniejszy obciążenie CPU, lecz po przekroczeniu punktu krytycznego zwiększy pominięcia szybkich gestów. Hipotezy należy przetestować, nie traktując ich jako ustaleń.

8.2. Warunki i procedura

Protokół powinien uwzględniać co najmniej trzy klasy scen: tło statyczne w stabilnym oświetleniu, dynamiczne tło oraz ograniczony kontrast lub zmienne oświetlenie. Każdy uczestnik realizuje bloki celowych gestów w obu kierunkach oraz dłuższe bloki swobodnej prezentacji bez komendy. Należy równoważyć kolejność wariantów, a czas ekspozycji bez gestu podawać oddzielnie od czasu prób aktywnych.

Dobór liczebności próby i czasu obserwacji powinien wynikać z analizy mocy oraz oczekiwanej częstości zdarzeń. Małe badanie pilotażowe pozwala wykryć błędy procedury i oszacować wariancję, lecz nie daje podstaw do szerokiego wnioskowania populacyjnego. Dane służące kalibracji muszą być rozdzielone od danych testowych; szczególnie ważny jest podział między uczestnikami, a nie wyłącznie między losowo wybranymi klatkami.

8.3. Definicje wskaźników

Zdarzenie prawdziwie dodatnie (TP) jest pojedynczą komendą o właściwym kierunku, dopasowaną do intencjonalnego gestu w z góry ustalonym oknie czasowym. Fałszywie ujemne (FN) oznacza gest bez prawidłowej komendy. Fałszywie dodatnie (FP) jest komendą bez przypisanego zamierzonego gestu; obejmuje przypadkowe wywołania podczas bloków kontrolnych. Polecenia zdublowane i błędnego kierunku raportuje się jawnie, a ich reguły punktacji ustala przed analizą.

$$\text{Recall} = TP / (TP + FN) \qquad \text{Precision} = TP / (TP + FP)$$

$$FP/h = \text{liczba FP} / \text{liczba godzin obserwacji bez gestu}$$

Dodatkowe miary obejmują rozkład opóźnienia od zakończenia gestu do faktycznej zmiany slajdu (mediana i 95. percentyl), stabilność śledzenia, liczbę dwukrotnych wywołań, obciążenie CPU/RAM oraz energię w scenariuszu laptopowym. Warto raportować przedziały ufności i analizę warstwową według użytkownika oraz rodzaju sceny.

Przy niewielkiej liczbie fałszywych aktywacji estymator FP/h ma dużą niepewność. Przykładowo zero zdarzeń w jednogodzinnym teście nie dowodzi, że rzeczywiste ryzyko wynosi mniej niż jedno zdarzenie na godzinę; przy prostym modelu Poissona przybliżona górna jednostronna granica 95% to około 3/T zdarzeń na godzinę, gdzie T jest liczbą godzin obserwacji. Długość testu należy dobrać do rygoru wymaganego w praktyce.

8.4. Cele akceptacyjne prototypu

Tabela 4. Przykładowe kryteria akceptacji, wymagające zatwierdzenia przed testem.

Obszar	Cel projektowy	Zasada interpretacji
Fałszywe aktywacje	≤ 1 zdarzenie/h	Długi test ekspozycji i przedział ufności
Skuteczność	≥ 95% prawidłowych komend	Odrębna ocena na niezależnych osobach
Opóźnienie	P95 ≤ 500 ms	Pomiar od końca gestu do zmiany slajdu
Niezawodność	Bez awarii w próbie 60 min	Uzupełnić testami długookresowymi
Zasoby	Bez zauważalnego zakłócania prezentacji	Wartości CPU/RAM ustalić dla platformy

Przedstawione cele nie są wartościami uzyskanymi w eksperymencie ani zewnętrznymi normami. Ostateczne kryteria sukcesu powinny uwzględniać charakter wystąpienia, akceptowalny koszt błędu oraz minimalną wymaganą pewność statystyczną.

9. Implementacja, testowanie i replikowalność

Prace należy prowadzić etapowo: (1) adapter wejścia i bezpieczny wyłącznik, (2) detektor kontrolny OpenCV, (3) detektor MediaPipe z historią trajektorii, (4) wspólny automat stanów i kontrola aktywnego okna, (5) testy interakcji i strojenie progów, (6) testy wieloplatformowe oraz pakowanie aplikacji. Włączenie RealSense lub Ultraleap jest uzasadnione dopiero po pomiarze ograniczeń wariantu RGB.

W repozytorium eksperymentalnym należy utrwalać wersję systemu operacyjnego i Pythona, wersje zależności, sumę kontrolną modelu MediaPipe, parametry kamery i ekspozycji, geometrię ROI, progi, mapowanie kierunków, procedurę adnotacji i konfigurację każdego przebiegu. Raport wynikowy powinien umożliwiać odtworzenie podziału uczestników oraz jednoznaczne przypisanie wyników do wersji implementacji.

Zaleca się testy jednostkowe automatu stanów na syntetycznych trajektoriach, testy negatywne (brak dłoni, odwrócenie ruchu, niemonotoniczne znaczniki czasu), testy integracyjne wejścia oraz próbę ciągłą na docelowym komputerze. W szczególności poprawność działania biblioteki do emisji klawiszy nie powinna być utożsamiana z poprawnym przełączeniem slajdu w konkretnym programie.

10. Ograniczenia i kierunki dalszych badań

Nie istnieją obecnie podstawy do podania porównywalnych wartości dokładności i opóźnienia wszystkich wymienionych modalności dla jednakowego zadania oraz jednakowych użytkowników. Deklaracje producentów, wyniki benchmarków modeli i jakościowe oceny konfiguracji stanowią różne rodzaje przesłanek. W szczególności opóźnienie inferencji pojedynczej klatki nie jest tożsame z czasem od wykonania gestu do zmiany slajdu.

Przedstawiona formalizacja gestu dotyczy ruchu poziomego i nie rozwiązuje wszystkich przypadków zasłonięcia dłoni, zmiany pozycji prezentera, ułożenia poza polem widzenia ani dwuznaczności gestykulacji. Rozwój rozwiązania może obejmować adaptację progów do użytkownika, kalibrację geometryczną stanowiska, niezależną ocenę intencjonalności gestu oraz personalizację modelu. Dalsze zwiększanie złożoności percepcji powinno być uzależnione od wykazanego zysku użytkowego.

11. Wnioski

Najbardziej racjonalnym wariantem początkowym dla lokalnego, lekkiego systemu przełączania slajdów jest kamera RGB współpracująca z MediaPipe Hand Landmarker i prostym detektorem trajektorii. OpenCV pozostaje potrzebnym punktem odniesienia, a RealSense i Ultraleap alternatywami dla sytuacji wymagających bardziej stabilnej separacji dłoni lub specjalizowanego sensora. Radar, ultradźwięki i IMU mają potencjał w odrębnych scenariuszach, lecz zwiększają nakład integracyjny.

O jakości całego interfejsu w większym stopniu niż nominalna dokładność klasyfikacji klatek decydują: liczba niezamierzonych przełączeń, jednoznaczna semantyka gestu, automat stanów gwarantujący pojedynczą komendę, kontrola kontekstu i mierzone end-to-end opóźnienie. Zasadność doboru technologii może zostać potwierdzona dopiero przez porównawczy eksperyment obejmujący swobodną gestykulację, niezależnych użytkowników i zróżnicowane warunki środowiskowe.

Piśmiennictwo

[1] Google AI Edge. Hand Landmarker: Python guide. https://ai.google.dev/edge/mediapipe/solutions/vision/hand_landmarker/python

[2] Google AI Edge. Gesture Recognizer: task guide. https://ai.google.dev/edge/mediapipe/solutions/vision/gesture_recognizer

[3] Google Research. MediaPipe Hands: On-device real-time hand tracking. https://research.google/pubs/mediapipe-hands-on-device-real-time-hand-tracking/

[4] OpenCV. Optical Flow. https://docs.opencv.org/4.x/d4/dee/tutorial_optical_flow.html

[5] OpenCV. Background Subtraction. https://docs.opencv.org/4.x/d1/dc5/tutorial_background_subtraction.html

[6] RealSense. Stereo Depth Camera D435: product specifications. https://www.realsenseai.com/products/stereo-depth-camera-d435/

[7] RealSense. librealsense: Python wrapper. https://github.com/realsenseai/librealsense/blob/master/wrappers/python/readme.md

[8] Ultraleap. Hand Tracking: Getting Started. https://docs.ultraleap.com/hand-tracking/getting-started.html

[9] Ultraleap. LeapC Python bindings: tracking event example. https://github.com/ultraleap/leapc-python-bindings/blob/main/examples/tracking_event_example.py

[10] pynput. Platform limitations. https://pynput.readthedocs.io/en/latest/limitations.html

[11] Google Research. Soli: Radar-based perception and interaction in Pixel 4. https://research.google/blog/soli-radar-based-perception-and-interaction-in-pixel-4/