AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI, INFORMATYKI I INŻYNIERII BIOMEDYCZNEJ

KATEDRA INFORMATYKI STOSOWANEJ

Praca dyplomowa magisterska

System aktywnej obserwacji oczu i głowy kierowcy za pośrednictwem kamery telefonu komórkowego ostrzegający go przed zaśnięciem

System of active observation of the driver's eyes and a head through a mobile phone camera to warn him before falling asleep

Autor: *Michał Panek*
Kierunek studiów: *Informatyka*
Opiekun pracy: *dr hab. Adrian Horzyk*

Kraków, 2015

Oświadczam, świadomy odpowiedzialności karnej za poświadczenie nieprawdy, że niniejszą pracę dyplomową wykonałem osobiście i samodzielnie i nie korzystałem ze źródeł innych niż wymienione w pracy.

Dziękuję mojemu Promotorowi dr hab. Adrianowi Horzykowi za poświęcony czas oraz cenne rady związane z niniejszą pracą, a także za wsparcie i wyrozumiałość.

Spis treści

1. Wprowadzenie

1.1. Wstęp

Rozwój informatyki w dziedzinie przetwarzania obrazów (zarówno pod względem jakości informacji, jak i algorytmiki) przyniósł wiele nowych i stosunkowo tanich rozwiązań dla różnych problemów człowieka. Miniaturyzacja elementów logicznych oraz coraz lepsza energooszczędność urządzeń przenośnych pozwoliła na pojawienie się na rynku konsumenckim smartfonów. Dzięki połączeniu tych dwóch zjawisk można zaobserwować powstawanie nowych rozwiązań kłopotów życia codziennego dostępnych dla każdego użytkownika posiadającego telefon komórkowy. Jednym z problemów współczesnego świata są wypadki powodowane przez sennych kierowców, a receptą na to są systemy aktywnej obserwacji otoczenia przy pomocy kamery umieszczonej w zminiaturyzowanej maszynie cyfrowej. Ze względu na coraz większą liczbę samochodów zapotrzebowanie na tego rodzaju aplikacje jest duże.

1.2. Motywacja

System, który będzie wykrywał senność u kierowców poprawi bezpieczeństwo użytkowników oraz przyczyni się do zwrócenia uwagi na ten niebezpieczny problem. Ponadto zadanie aktywnej obserwacji twarzy i oczu kierowcy daje sposobność zastosowania w praktyce wielu technik programowania oraz użycia rozmaitych narzędzi programistycznych. Można z niego wyodrębnić następujące tematy:

- przetwarzanie i analiza obrazów cyfrowych,
- algorytmika,
- tworzenie aplikacji mobilnych,
- integrację z urządzenia zewnętrznymi (kamera, głośnik).

  Jest to zatem zadanie łączące w sobie wiele dziedzin informatyki. Daje ono również pewność, iż opracowane rozwiązanie będzie przydatne w codziennym życiu.

1.3. Cel pracy

Celem pracy jest zaprojektowanie oraz implementacja systemu pozwalającego na wykrywanie senności oraz jej zapobieganiu poprzez aktywną obserwację twarzy i oczu kierowcy. Należy to rozumieć

jako podjęcie decyzji o wykryciu zaśnięcia na podstawie wystąpienia pewnych oznak fizycznych możliwych do zidentyfikowania na obrazie cyfrowym. Dodatkowo cel ten ma być osiągnięty przy wykorzystaniu kamery oraz mocy obliczeniowej zwykłego smartfona. Wiąże się to z chęcią stworzenia systemu, który byłby ergonomiczny i przenośny. Aplikacja ma działać na urządzeniach wyposażonych w system operacyjny Android, zaś źródłem informacji potrzebnej do działania są ramki obrazu o minimalnej rozdzielczości 640x480 pikseli.

1.4. Zawartość pracy

Praca została podzielona na sześć rozdziałów. Rozdział 1. stanowi wprowadzenie do części właściwej pracy. W rozdziale 2. został zamieszczony opis dziedzinowy problemu, czyli informacje teoretyczne o skali występowania senności wśród kierowców, ich źródłach oraz sposobach przeciwdziałania. Przedstawione zostały również istniejące rozwiązania komercyjne oraz proponowane przez naukowców, opierające się na różnych sposobach implementacji i działania. Rozdział 3. stanowi opis koncepcji systemu wraz z założeniami oraz ograniczeniami, a także z przedstawieniem wykorzystanych technik programistycznych. W rozdziale 4. autor przedstawia kompletny opis zaprojektowanego oraz zaimplementowanego systemu. Jest to opracowanie rozwiązań pośrednich oraz rozwiązania uznanego za ostateczne, użytego do osiągnięcia celu pracy. Rozdział 5. zawiera prezentację istotnych wyników, ich analizę oraz krytyczną ocenę. W rozdziale 6. autor podsumowuje prace związane z tematem oraz przedstawia własne opinie na temat możliwości rozwoju systemu. Zawiera on również rozliczenie z celem, postawionym w rozdziale 1.

2. Opis dziedziny problemu

Rozdział ten podzielony jest na dwie części tematyczne. Pierwsza dotyczy problemu senności wśród kierowców, jego przyczyn, skutków oraz metod zapobiegania opartych na literaturze dziedzinowej. W drugiej części tego rozdziału autor przedstawia rozwiązania komercyjne obecnie stosowane w pojazdach. Opisano również metody opracowane przez pracowników naukowych. Sposoby na wykrywanie senności kierowców można podzielić na wykorzystujące metody wizyjne oraz opierające się na innych sensorach, np. elektrodach przyczepionych do ciała lub czujnikach nacisku. Zaprezentowano także metody detekcji twarzy oraz oczu, które są używane we wcześniej wspomnianych systemach.

2.1. Problem zasypiania wśród kierowców

Senność wpływa w sposób negatywny na zachowania kierowców podczas jazdy i może być jednym z symptomów zmęczenia. Obniżenie sprawności psychofizycznej i zmniejszenie niezawodności w podejmowaniu decyzji przekłada się na większe ryzyko wypadku drogowego.

2.1.1. Dane statystyczne

Zgodnie z rocznym raportem [33] Komendy Głównej Policji w 2014 roku w Polsce było 34 970 wypadków drogowych, zginęło w nich 28 716 osób. Aż 555 wypadków było spowodowanych przez zmęczenie lub zaśnięcie i w ich wyniku śmierć poniosło 77 osób, zaś rannych zostało 846 osób. Podobne liczby, przedstawiane przez oficjalne rządowe statystyki, występują na całym świecie. Trudności w określeniu przyczyn wypadków mogą te wartości jednak zaniżać i problem ten ma w rzeczywistości znacznie większą skalę. Zgodnie z raportem AAA Foundation for Traffic Safety [22]:

- 13% wypadków, które skutkowało ciężkimi obrażeniami wymagających hospitalizacji
- oraz 21% śmiertelnych wypadków

było spowodowanych przez sennych kierowców. National Sleep Foundation (Stany Zjednoczone) [6] podała, że w 2009 roku 54% kierowców odczuwało ospałość, a aż 28% zasnęło podczas jazdy. Statystyki te ilustrują fakt, że senność jest jedną z ważniejszych przyczyn powstawania wypadków drogowych.

2.1.2. Specyfika zasypiania u kierowców

Fazy snu można podzielić na następujące etapy [35]:

– przebudzenie,

– sen wolnofalowy, sen o wolnych ruchach gałek ocznych (ang. NREM - non-rapid eye movement),

– sen paradoksalny, sen o szybkich ruchach gałek ocznych (ang. REM - rapid eye movement).

Dodatkowo faza NREM może być dalej sklasyfikowana, między innymi ze względu na występujące fale mózgowe, na następujące etapy:

– etap 1: przejście z stanu przebudzenia do senności,

– etap 2: sen lekki,

– etap 3: sen głęboki.

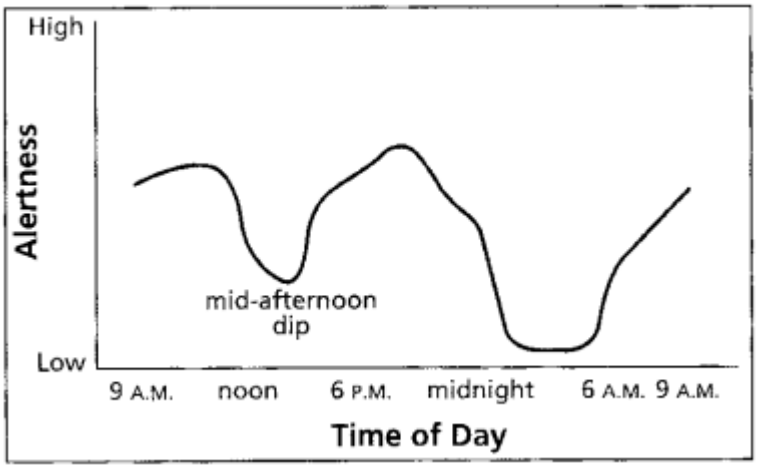
Analiza senności u kierowców polega na badaniu etapu 1 i zapobiegnięciu jego wystąpieniu. Przyczynami zasypiania podczas jazdy mogą być [29]:

– pora doby - rola rytmu dobowego snu i czuwania przedstawione m.in na wykresie 2.1,

– deficyt snu - związany z czasem snu poprzedzającego jazdę, jakością snu, bezsennością lub bezdechem sennym,

– długi czas czuwania wynikający z długości jazdy,

– natężenie ruchu drogowego (niskie i wysokie wymagania dotyczącej jazdy) oraz monotonność jazdy.



Rysunek 2.1: Wykres zależności pory doby od stopnia czujności (Źródło: [28])

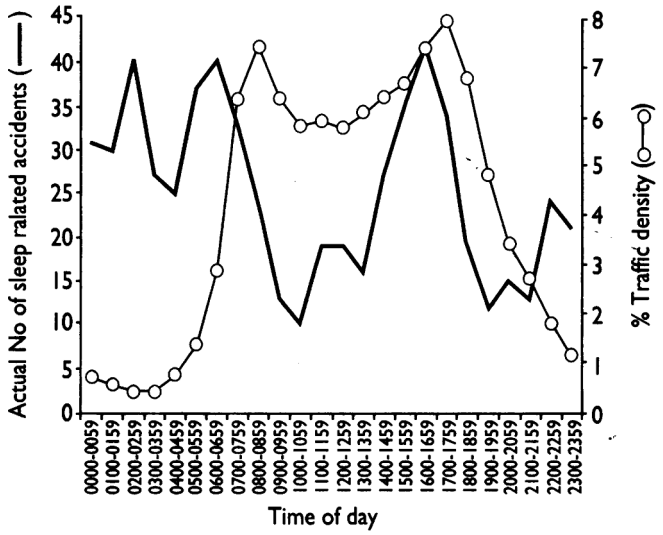
Jako krytyczne godziny w pracy kierowcy podaje się czas od 0:00 do 6:00 oraz między 12:00 a 16:00 [29]. Na wykresie 2.2 została przedstawiona korelacja między porą doby a liczbą wypadków

samochodowych. Uczucia senności, które występują o tych newralgicznych godzinach związane są z obniżoną aktywnością funkcji fizjologicznych, mogą objawiać się w tak zwanych mikrodrzemkach - kilkusekundowych utratach świadomości. Skutkami ospałości są zwolnione reakcje, zmniejszona czujność oraz pogorszone przetwarzanie informacji [39]. Sygnałami, które mogą świadczyć o narastającej senności są między innymi:

– zmiana prowadzenia samochodu - mniej płynne ruchy kierownicą, częstsze zmiany szybkości,

– tryb jazdy bez uwagi,

– opadanie głowy, zamykanie oczu, ziewanie.



Rysunek 2.2: Wykres zależności pory doby od liczby wypadków samochodowych związanych z sennością oraz natężenia ruchu drogowego (Źródło: [25])

2.1.3. Sposoby przeciwdziałania problemowi

Metodą na ograniczenie negatywnych skutków senności u kierowców może być użycie urządzeń technicznych. Wyróżnić tutaj należy systemy wewnętrzne montowane w samochodach wykrywające ospałość i reagujące na to w odpowiedni sposób, np. poprzez alarmy lub wprowadzenie interakcji z kierowcą. Przegląd możliwych rozwiązań znajduje się w następnym podrozdziale. Sposobem na przeciwdziałanie problemowi mogą być również odpowiednio umieszczane na drogach linie wyznaczające pasy ruchu [29]. Wspomniane wyżej rozwiązania nie eliminują jednak całkowicie problemu senności. Najbardziej efektywnym sposobem jest odpowiednia ilość snu przed podróżą. Stwierdzono [39], że np. po 16 godzinnych okresie czuwania potrzebne są około 8 godzin snu i czas ten zwiększa się wraz z długością trwania aktywności. Innymi skutecznymi metodami mogą być [6]:

– unikanie jazdy nocą,

– krótki odpoczynek po 2-3 godzinach jazdy,

– krótka drzemka (ok. 30 minutowa) w razie odczuwania senności,

– odpowiednia wentylacja powietrza w samochodzie,

– spożywanie kofeiny (ok. 200 miligramów).

Wspomniane wcześniej systemy ostrzegające kierowcę przed zaśnięciem w połączeniu z informowaniem o np. konieczności drzemki mogą w sposób pozytywny przyczynić się do przynajmniej częściowego uniknięcia senności za kierownicą.

2.2. Istniejące rozwiązania

2.2.1. Istniejące systemy wykrywające senność

Systemy wykrywające senność u kierowcy można podzielić na dwie grupy ze względy na sposób akwizycji informacji:

– wykorzystujące metodę wizualną, czyli kamerę obserwującą otoczenie kierowcy lub samochodu,

– wykorzystujące sensory montowane w samochodzie takie jak czujniki nacisku, ruchu, mikrofony lub przyczepianie do ciała kierowcy (inwazyjne) - elektrody mierzące funkcje życiowe.

Poniższy opis istniejących rozwiązań został oparty na tym podziale. Można je również sklasyfikować uwzględniając rodzaj wykorzystywanej informacji [35]:

– zachowanie pojazdu - monitorowane są wartości wskazujące na pozycję na pasie jazdy, ruchu kierownicy, nacisku pedału gazu - oparte o sensory,

– zachowanie kierowcy - analizowane są informacje o ziewaniu, mruganiu oczami, zamykaniu oczu, pozycji głowy itp. - metoda wizyjna,

– sygnały fizjologiczne kierowcy - badana jest korelacja pomiędzy szybkością tętna, biciem serca i funkcjami mózgu - oparte o sensory.

Systemy oparte na informacji wizualnej

Systemy te opierają się na aktywnej obserwacji twarzy, oczu, ust kierowcy za pomocą kamery wizyjnej i na tej podstawie decydują o senności. Jedną z najefektywniejszych metod jest PERCLOS (ang. percentage of closed eyes over time) [42]. Metoda ta polega na obliczeniu stosunku liczby ramek obrazu z wykrytymi zamkniętymi oczami do liczby wszystkich ramek obrazu w określonym oknie czasowym. Innym sposobem jest proste badanie częstości mrugania [17] lub długości zamknięcia oka przez dany czas. W celu detekcji używa się metod opartych o binaryzację obrazu lub jasności ramki do określania, czy oko jest zamknięte, wykrywania krawędzi (rozwarcie powiek [27]) lub analizę wartości projekcji wertykalnej lub horyzontalnej obrazu [19]. Innymi sposobami jest wykorzystanie detekcji okręgów z użyciem transformaty Hougha, metody lokalnych wzorców binarnych (ang. LBP - Local Binary Pattern) [42] lub porównywania wzorców (ang. template matching) [23]. Do klasyfikacji otrzymanych danych używane są

sieci neuronowe [43], maszyna wektorów nośnych (ang. SVM - Support Vector Machine) [16, 21] oraz funkcje senności [34]. Istnieją również rozwiązania oparte o pozycję głowy i oczu oraz detekcję kiwania głową [38]. Algorytmem pomocniczym jest tutaj na przykład użycie modelu aktywnego kształtu, czyli ASM (ang. Active Shape Model) [45]. Jako uzupełnienie przedstawionych rozwiązań często stosuję się dodatkowo rozpoznawanie ziewania (otwartość ust) lub inne metody wykorzystujące mimikę twarzy. Częstym problemem w tego typu rozwiązaniach jest niska jasność analizowanych ramek obrazu. Receptą na to jest używanie diod emitujących światło podczerwone. Sposoby z użyciem metody wizyjnej na wykrywanie senności są stosowane przez Lexusa oraz firmę Seeing Machines. Istnieją również inne rozwiązania dedykowane na urządzenia mobilne [26, 14, 44].

Wadą rozwiązań korzystających z informacji wizualnej jest niska skuteczność działania w zmiennych warunkach oświetleniowych, zaś zaletami stosunkowo niski koszt stworzenia, brak inwazyjności, wygoda stosowania oraz dość wysoka skuteczność działania (na poziomie 90% [35]).

Rysunek 2.3: Wizyjny system detekcji senności w Lexusie (Źródło: [8])

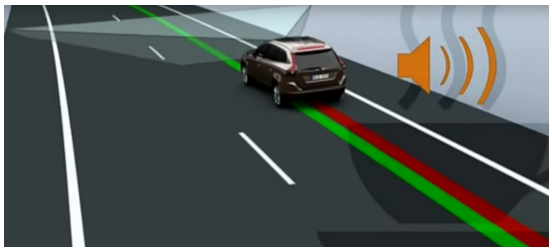
Inne systemy

Systemy te wykorzystują zewnętrzne sensory do podejmowania decyzji o senności kierowcy [35]. Metody bazujące na zachowaniu samochodu to między innymi:

– Steering Wheel Movement (SWM) - analizowane są ruchy kierownicą dzięki sensorowi pomiaru wychylenia kierownicy. Senny kierowca wykonuje mniej mikropoprawek kierownicą w porównaniu do kierowcy wypoczętego.

– Standard Deviation of Lane Position (SDLP) - mierzone jest odchylenie standardowe pozycji na pasie ruchu dzięki wykorzystaniu zewnętrznych kamer.

Środki te stosowane są w pojazdach takich marek jak Nissan, Renault czy Volvo. Wadą wymienionych rozwiązań jest zbyt duża zależność od geometrii drogi.



(a) System detekcji odchylenia standardowego na pasie ruchu

(b) Przykład informacji dla kierowcy o potrzebie odpoczynku

Rysunek 2.4: Driver Alert Control w samochodach Volvo X60 (Źródło: [13])

Metody, które wykorzystują badanie stanu fizjologicznego kierowcy pozwalają na dość wczesne wykrycie senności. Opierają się na badaniu sygnałów takich jak [35]:

– elektrokardiogram (ECG) - dotyczy badania czynności serca; stosunek wysokich do niskich częstotliwości zmniejsza się, gdy kierowca zaczyna być senny,

– elektroencefalogram (EEG) - dotyczy badania czynności mózgu; opiera się na badaniu pasm częstotliwości odpowiedzialnych za sen (fale delta), senność (fale theta), czujność (fale beta), odprężenie (fale alfa),

– elektrookulogram (EoG) - dotyczą badania napięcia elektrycznego pomiędzy siatkówką a rogówką; pozwala na wykrywanie szybkich (REM) i powolnych ruchów oka (SEM), które pojawiają się podczas aktywności i senności kierowcy.

Zaletami metod opierających się na sygnałach fizjologicznych są przede wszystkim wysoka dokładność i niezawodność. Dużą wadą pozostaje jednak niewygoda stosowania oraz inwazyjność. Tą ostatnią niedogodność da się poprawić stosując czujniki i elektrody umieszczone w kierownicy, fotelu kierowcy lub pasach.

Rysunek 2.5: Sensory do badania sygnałów fizjologicznych zamontowane w fotelu kierowcy (Źródło: [7])

Testowanie systemów wykrywających zmęczenie kierowców

Testowanie systemów, które zajmują się zwiększaniem bezpieczeństwa użytkowników nie powinno być przeprowadzane w sposób zagrażający życiu testerów. Zatem testy systemów wykrywających senność kierowcy nie mogą być wykonywane na drogach w ruchu ulicznym. Rozwiązaniem tego problemu jest zastosowanie symulatorów [35]. Zaletami ich stosowania są:

– przeprowadzanie eksperymentów w kontrolowanych warunkach,

– dokładność działania i bezpieczeństwo dla użytkowników,

– łatwość gromadzenia danych.

Istnieje wiele rozwiązań od prostych programów symulujących jazdę samochodem wyświetlanych na monitorach po środowiska z obrazem wyświetlanym dookoła kierowcy z dodatkowymi wrażeniami zewnętrznymi jak np. wibracje. Jedną z wad stosowania symulatorów jest brak wrażenia ryzyka wśród uczestników symulacji.

Rysunek 2.6: Przykład symulatora do testowania jazdy samochodem - NADS (National Advanced Driving Simulator) (Źródło: [10])

2.2.2. Detekcja charakterystycznych elementów

Jak zostało wspomniane w poprzednim rozdziale metoda wizyjna wykrywania senności opiera się na obserwacji twarzy i oczu, żeby tego dokonać potrzebna jest detekcja tych elementów.

Detekcja twarzy

Zadanie detekcji twarzy jest najważniejszych elementem i powinno zostać wykonane jak najmniejszym nakładem obliczeniowym. Jedną z najpopularniejszych i najskuteczniejszych metod jest użycie kaskadowych klasyfikatorów Haara (opis w [41] przykłady użycia w [18, 32]). Kolejną spotykaną w literaturze metodą jest wykrywanie twarzy na podstawie odcienia koloru skóry [40, 20]. Po przekształceniu ramki obrazu do odpowiedniej przestrzeni kolorów wybierane są wartości (na podstawie prostych zakresów lub określonych reguł), które odpowiadają kolorowi skóry twarzy. Inną metodą na lokalizację głowy jest użycie detekcji jej ruchu [9]. Działanie to sprawdza się jednak tylko, gdy tło jest nieruchome. Kolejnym popularnym algorytmem jest analiza głównych składowych dla twarzy [30, 20]. Jest to metoda umożliwiająca wnioskowanie o obiektach opisanych wieloma zmiennymi.

Detekcja oczu

Dużym ułatwieniem wykrywania oczu jest określenie podstawowych zależności geometrii twarzy (np. maksymalny rozmiar oczu, odległość między oczami) i wykorzystanie tych informacji po uprzednim odnalezieniu twarzy. Do detekcji oczu również często stosowane są kaskadowe klasyfikatory Haara[32]. Jedną z innych metod jest użycie transformacji Hougha dla okręgów. Pozwala ona na odnalezienie okręgów opisujących tęczówkę na podstawie obrazu zbinaryzowanego. Innym sposobem jest użycie progowania oraz znalezienie najciemniejszego obszaru będącego przybliżonym środkiem oka [15]. Dostępna literatura [24] opisuje metody związane z gradientem (nagłą zmianą jasności), czyli między innymi model aktywnego konturu (Active Contour Model). Kolejną metodą do detekcji oczu jest wykorzystanie funkcji projekcji [47]. W celu śledzenia ruchu wykrywanego obiektu stosuje się między innymi filtr Kalmana [46].

3. Koncepcja systemu

Rozdział ten jest podsumowaniem fazy analizy i zbierania wymagań podczas pracy nad systemem. Zawiera w sobie elementy takie jak założenia oraz ograniczenia aplikacji, także opis stosu technologicznego.

3.1. Założenia systemu

Istotnym elementem podczas pracy nad systemem jest określenie jego założeń. Pozwala to na ustalenie podstawowych wymagań dotyczących aplikacji oraz ułatwia etap implementacji.

Głównym założeniem systemu jest wykorzystanie wbudowanej kamery telefonu komórkowego jako źródła informacji. Użycie zewnętrznego urządzenia akwizującego dane wizualne odebrałoby jedną z ważnych cech systemu - mobilność. Ponadto wszystkie obliczenia związane z przetwarzaniem obrazu są wykonywane przy użyciu procesora urządzenia przenośnego. Pozwala to na efektywne działanie systemu, czyli uściślając: podczas akwizycji ramek o rozdzielczości 640x480 pikseli szybkość przetwarzania to średnio 6 fps (ang. fps - frames per second, czyli liczba ramek na sekundę). Dla porównania dla rozdzielczości 960x720 szybkość ta wynosi ok. 3 fps. Ważnym założeniem aplikacji jest prawidłowe działanie podczas zmiennych warunków oświetleniowych, czyli w trakcie nagłych zmian jasności obrazu. Kolejnym elementem wartym wspomnienia jest poprawna praca bez jakichkolwiek zewnętrznych źródeł oświetlenia, np. diody podczerwieni lub światła widzialnego. Zaimplementowany algorytm bazuje na obserwacji twarzy i oczu kierowcy. System jest w stanie rozpoznawać senność u osób noszących okulary z przezroczystymi soczewkami. Aplikacja działa na urządzeniach mobilnych wyposażonych w system Android w wersji co najmniej 4.0. Wybór ten jest spowodowany popularnością tej platformy wśród posiadaczy smartfonów.

3.2. Ograniczenia systemu

Ograniczenia systemu pozwalają na określenie jak najlepszych warunków do prawidłowego działania aplikacji oraz były wyznacznikiem doboru algorytmów podczas tego implementacji.

Ze względu na brak zewnętrznego oświetlenia system nie radzi sobie dobrze w warunkach zupełnej lub częściowej ciemności obserwowanego otoczenia. Stosowane są metody wyrównywania jasności jednak i one mają swoje granice prawidłowego działania. Rozpoznawanie senności działa na zasadzie detekcji oczu, zatem nie jest możliwe całkowite lub częściowe zasłonięcie rejonu oczu, np. przez okulary przeciwsłoneczne. Kąt obserwacji głowy kierowcy jest również ograniczony i rezultaty działania

systemu są najlepsze z ustawieniem prostopadłym do płaszczyzny twarzy. Ze względu na działanie na urządzeniu mobilnym z określoną pojemnością baterii czas pracy systemu jest ograniczony. Dodatkowo w celu zainstalowania aplikacji potrzebne jest ok. 50 MB wolnego miejsca na dysku (aplikacja wraz z biblioteką OpenCV na urządzenia mobilne).

3.3. Użyty sprzęt

3.3.1. Parametry telefonu komórkowego

Podczas implementacji systemu autor miał do dyspozycji telefon komórkowy Motorola Moto G pierwszej generacji. Podstawowe i istotne z punktu widzenia działania aplikacji parametry tego urządzenia to:

- czterordzeniowy procesor Qualcomm Snapdragon 400 1.2 GHz,
- 1 GB pamięci operacyjnej RAM,
- GPU: Adreno 305,
- akumulator litowo-jonowy o pojemności 2070 mAh.

Takie parametry pozwalają na prawidłowe działanie systemu. Urządzenie to stanowiło również platformę testową.

3.3.2. Kamera

Urządzenie mobilne opisane w poprzednim podrozdziale wyposażone jest w dwie kamery:

- przednią o rozdzielczości 1.3 MP, umożliwiającą akwizycję ramek obrazu z maksymalną szybkością 15 fps i rozdzielczości 960x720 (dla ramki o rozdzielczości 320x240 pikseli szybkość ta wynosi 30 fps)
- tylną o rozdzielczości 5 MP, umożliwiającą akwizycję ramek obrazu z maksymalną szybkością 20 fps i rozdzielczości 960x720 (dla ramki o rozdzielczości 640x480 pikseli szybkość ta wynosi 30 fps)

Dodatkowymi funkcjami, w które wyposażona jest tylna kamera to autobalans bieli oraz autofocus. Pozwalają one na automatyczne rozjaśnianie obrazu w przypadku słabszego oświetlenia oraz ustawianie ostrości obrazu z obserwowanymi obiektami.

3.4. Platforma oraz środowisko programistyczne

Android

Android [2] to najpopularniejszy system operacyjny dedykowany dla urządzeń mobilnych i stanowi 78% całego rynku [12]. Swoją popularność zawdzięcza łatwości obsługi, możliwości integracji z większością urządzeń mobilnych oraz bardzo dobrym wsparciem dla programistów aplikacji na smartfony. Są

to zalety, które spowodowały, że autor wybrał tą platformę do stworzenia systemu zapobiegania senności wśród kierowców. Aplikacja ta jest kompatybilna z wersjami Androida, począwszy od wersji 4.0. Dzięki temu opracowany program jest możliwy do uruchomienia na 95% urządzeniach wyposażonych w ten system [2]. Platforma Android pozwala na implementowanie aplikacji w języku Java i zawiera wiele udogodnień pozwalających na łatwiejsze korzystanie z interfejsów zewnętrznych, np. kamery i głośnika. Zaletami, które okazały się pomocne w trakcie implementacji systemu są łatwa integracja z dodatkowymi bibliotekami programistycznymi (szczególnie specjalizowanymi do przetwarzania obrazów) oraz darmowy dostęp do narzędzi programistycznych.

OpenCV

OpenCV (ang. Open Source Computer Vision [4]) jest darmową biblioteką napisaną w C/C++ i jest specjalnie dedykowana do szeroko pojętego zadania przetwarzania obrazów. Posiada interfejs programistyczny między innymi dla języka Java [5] i odpowiednią wersję do implementacji aplikacji mobilnych na platformę Android (OpenCV4Android [11]). System do wykrywania senności wśród kierowców został stworzony w oparciu o wersję 2.4.9 i używane są w nim funkcje biblioteczne napisane zarówno w języku Java jak w C++ (możliwe do wywołania dzięki mechanizmowi JNI) Biblioteka OpenCV składa się z kilku modułów przeznaczonych między innymi do pobierania ramki obrazu, podstawowych operacji na macierzach, algorytmów stosowanych w przetwarzaniu obrazów (np. filtry), detekcji obiektów oraz rysowania kształtów geometrycznych. W celu zapewnienia prawidłowego działania aplikacji napisanych z użyciem tej biblioteki konieczne jest pobranie ze sklepu Google Play i zainstalowanie modułu OpenCV Manager.

Autor wybrał OpenCV do przetwarzania obrazów spośród wielu innych (między innymi FastCV, Snapdragon SDK, BoostCV) ze względu na szybkość działania, popularność, kompatybilność z platformą Android oraz najlepsze narzędzia do poradzenia sobie z zadaniem wykrywania senności na podstawie obserwacji twarzy i oczu kierowcy.

JNI

JNI (ang. Java Native Interface) to mechanizm pozwalający na uruchamianie kodu napisanego w innych językach (jak np. C++) na wirtualnej maszynie Javy. Do kompilacji na platformie Android [3] stosowany jest Android Native Development Kit (NDK). Komunikacja między językiem natywnym, a językiem Java istnieje dzięki przekazywanym następującym argumentom:

– obiekt typu `JNIEnv*` - wskaźnik dający dostęp do środowiska JNI; udostępnia wiele metod użytkowych, np. konwersję między typami,

– obiekt typu `jobject` - obiekt reprezentujący metodę, która wywołała ten natywny kod,

– obiekty będące argumentami wywołania funkcji w kodzie Javy (z typami np. `jboolean`, `jbyte`, `jstring`, `jint`, `jlong` itp.).

Autor wykorzystuje ten mechanizm w pracy ze względu na wydajność działania uruchamianego kodu (efektywniejsza praca niektórych algorytmów) oraz brak implementacji niektórych funkcjonalności biblioteki OpenCV w interfejsie Java.

3.5. Stanowisko do pracy z systemem

Ze względu na specyfikę działania algorytmu opierającego się na obserwacji twarzy kierowcy najważniejszym elementem stanowiska jest to, by kamera urządzenia mobilnego była skierowana na osobę monitorowaną. Głowa powinna znaleźć się na środku akwizowanego obrazu. Smartfon powinien zostać umieszczony w odległości ok. 0.3 do 0.7 metra od twarzy oraz pod maksymalnym kątem 10°od kierunku patrzenia kierowcy. Urządzenie powinno zostać zamontowane w pozycji poziomej, obróconej o 90°w lewo w stosunku do pozycji pionowej. Najważniejsze w położeniu telefonu komórkowego jest jednak to, żeby nie zasłaniał widoku drogi ani nie rozpraszał użytkownika. Konieczne może być również podłączenie do zewnętrznego źródła energii elektrycznej, np. wtyczki samochodowej, ze względu na zużycie akumulatora urządzenia.

4. Implementacja systemu

Rozdział ten stanowi opis opracowanego rozwiązania wraz z metodami, które nie znalazły się w nim, ale stanowiły drogę do osiągnięcia postawionego na początku pracy celu. Tekst podzielony jest na elementy projektu systemu wraz z ogólnym przedstawieniem jego działania, a następnie opisane są bardziej szczegółowo poszczególne fragmenty systemu.

4.1. Elementy projektu systemu

4.1.1. Ogólna architektura systemu

Najważniejszymi elementami systemu są:

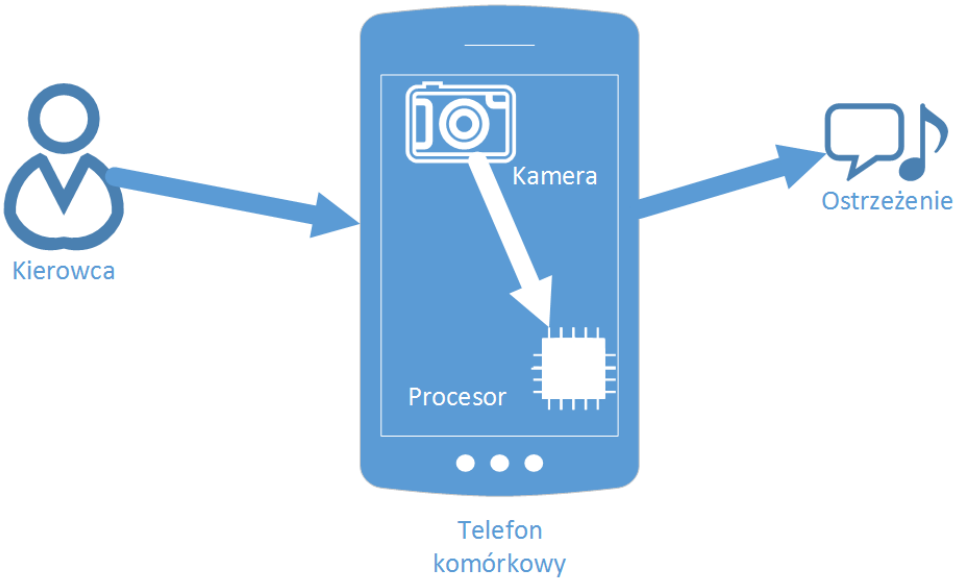
– kierowca - jego twarz oraz oczy są źródłem danych,

– kamera - służy do akwizycji danych,

– aplikacja - algorytm detekcji senności,

– wyświetlacz, głośnik - stanowią wyjście aplikacji, czyli ostrzeganie użytkownika.

Rysunek 4.1 przedstawia schemat interakcji użytkownika z systemem.



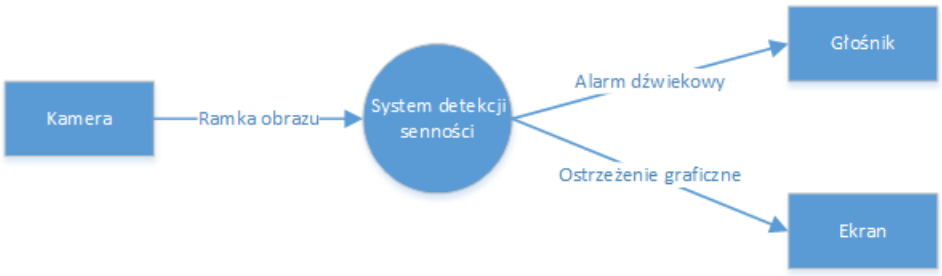
Rysunek 4.1: Schemat przedstawiający interakcję użytkownika z systemem

4.1.2. Opis algorytmu

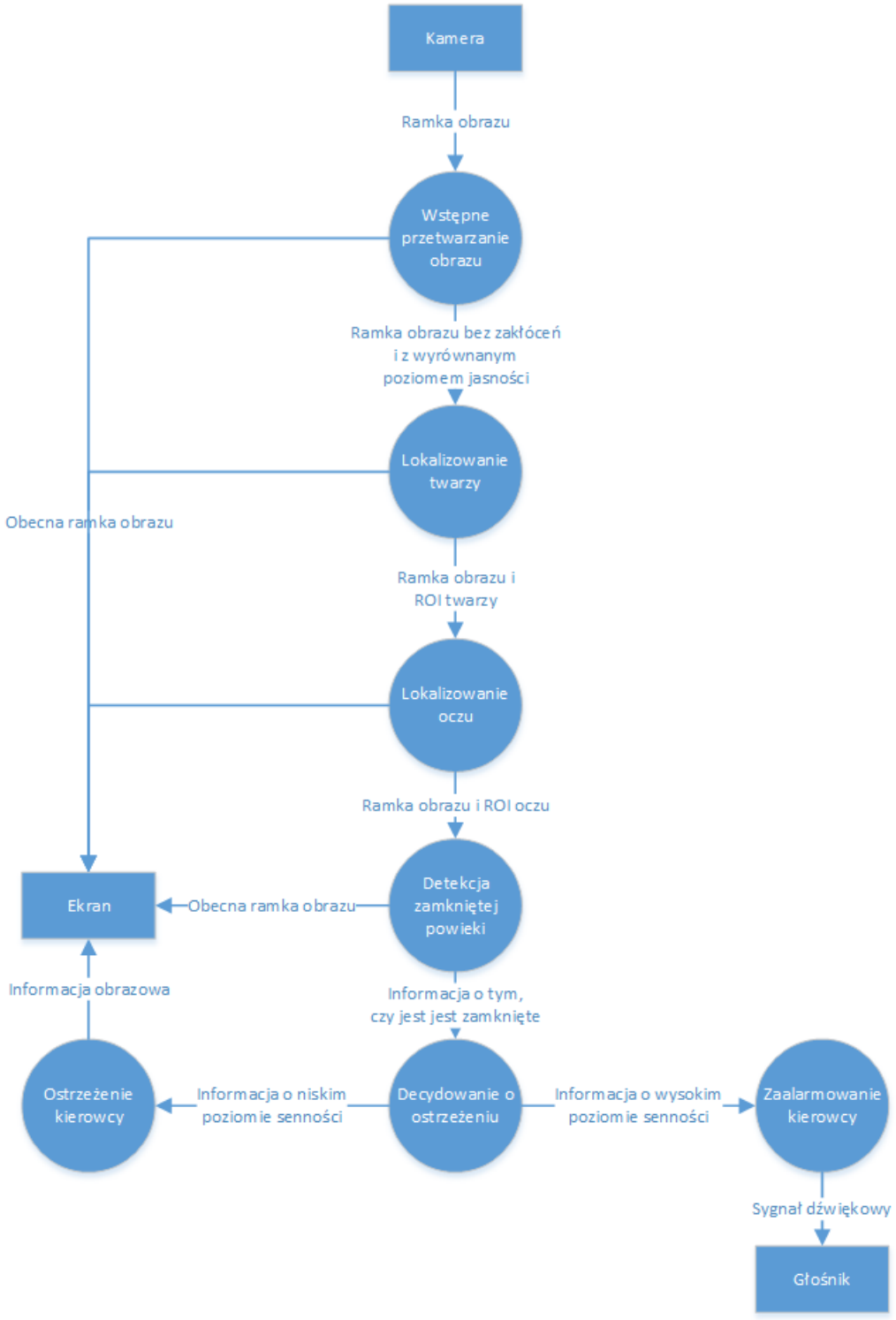
Algorytm wykrywający senność składa się z następujących kroków:

1. Akwizycja ramki obrazu.
2. Przekształcenie macierzy do obrazu w skali szarości.
3. Wstępne przetworzenie obrazu:
   (a) Wyrównanie histogramu metodą adaptacyjnego ograniczenia kontrastu.
   (b) Rozmycie obrazu metodą gaussowską celem usunięcia zakłóceń.
4. Lokalizacja twarzy z użyciem kaskadowych klasyfikatorów Haara na obrazie z uprzednio wykonaną operacją wyrównania histogramu.
5. Lokalizacja oczu z użyciem kaskadowych klasyfikatorów Haara z uprzednio wykonaną operacją wyrównania histogramu.
6. Detekcja zamkniętych oczu jedną z poniższych metod:
   – z użyciem metody binaryzacji obrazu oka,
   – po zastosowaniu metodą Laplasjanu obrazu,
   – z wykorzystaniem analizy średniej jasności w każdym wierszu obrazu,
   – z wykorzystaniem analizy średniej jasności w każdej kolumnie obrazu.
7. Zastosowanie metody wykrywania senności i zaśnięcia.
8. Podjęcie decyzji o uruchomieniu sygnału dźwiękowego lub ostrzeżeniu graficznym.

Poszczególne kroki zostały dokładnie opisane w kolejnych podrozdziałach. Na rysunku 4.2 został przedstawiony ogólny schemat systemu, zaś rysunek 4.3 obrazuje dokładniejszy przepływ danych między najważniejszymi modułami aplikacji.



Rysunek 4.2: Schemat przedstawiający uproszczony przepływ danych



Rysunek 4.3: Schemat przedstawiający przepływ danych między elementami zewnętrznymi i modułami (procesami) w systemie

4.2. Najważniejsze dodatkowe operacje przetwarzania obrazów

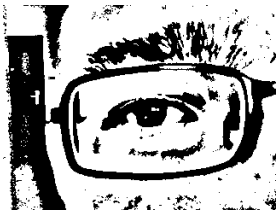
W tym podrozdziale znajdują się opisy operacji przetwarzania obrazów [37], które są podstawowymi elementami składowymi opracowanego rozwiązania. Wskazano również metody z biblioteki OpenCV, które odpowiadają przedstawionym algorytmom. Dla ujednolicenia opisów zdecydowano się, aby piksele białe były opisywane wartością 1, zaś czarne 0. W przypadku implementacji tego podziału w bibliotece OpenCV wartości te wynoszą odpowiednio 255 oraz 0 i wynikają z typów danych użytych do reprezentacji macierzy obrazu.

Binaryzacja

Binaryzacja (inaczej progowanie) pozwala na podzielenie obrazu w skali szarości na taki, który zawiera tylko piksele o wartości 1 oraz 0. Podział ten następuje poprzez zamianę wartości oryginalnej w zależności od wybranego progu. Dzięki temu sposobowi możemy wyodrębnić obiekt, który różni się od innych jasnością pikseli, np. jest ciemniejszy. Operację binaryzacji można przeprowadzić na całej ramce z jednym progiem (binaryzacja jednoprogowa), a także na mniejszych jej elementach dopasowując dla każdego osobny próg binaryzacji - taki proces to binaryzacja adaptacyjna. Obie metody zostały wykorzystane w pracy podczas detekcji zamkniętego oka. W przypadku binaryzacji jednoprogowej próg jest automatycznie dobierany na podstawie średniej jasności obrazu. Przykład działania został przedstawiony na rysunku 4.4. Funkcja biblioteki OpenCV do przeprowadzenia operacji binaryzacji jednoprogowej to `Imgproc.threshold`, zaś jej argumentem jest wartość progu. Do binaryzacji adaptacyjnej użyto metody `Imgproc.adaptiveThreshold`.

(a) Obraz oryginalny

(b) Obraz po operacji binaryzacji jednoprogowej



(c) Obraz po operacji binaryzacji adaptacyjnej

Rysunek 4.4: Operacja binaryzacji (progowania)

Rozmycie gaussowskie

Rozmycie gaussowskie jest jedną z metod usuwania zakłóceń z obrazu i jest to tzw. filtr dolnoprzepustowy. Polega na przeprowadzeniu operacji splotu (konwolucji) funkcji dla każdego piksela w obrazie

w oparciu o jego sąsiedztwo. Wartości pikseli z sąsiedztwa mnożone są przez wartości określone w masce konwolucji (zwykle przyjmującej postać macierzy kwadratowej). Dla rozmycia gaussowskiego postać maski wyznaczana jest przy użyciu funkcji Gaussa: $G(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-x^2+y^2}{2\sigma^2}}$, gdzie σ - odchylenie standardowe. Przykład działania został zaprezentowany na rysunku 4.5. Metoda ta jest używana w pracy w celu usunięcia zakłóceń powstałych w trakcie akwizycji obrazu, a także wynikających z operacji wyrównania histogramu. Funkcja biblioteki OpenCV do przeprowadzenia operacji rozmycia gaussowkiego to `Imgproc.GaussianBlur`. Przyjmuje ona dwa argumenty - rozmiar maski oraz wartość σ.

(a) Obraz oryginalny

(b) Obraz po zastosowaniu filtra gaussowskiego

Rysunek 4.5: Operacja rozmycia gaussowskiego

Wykrywanie krawędzi - laplasjan

Operacja wykrywania krawędzi jest związana z filtrami górnoprzepustowymi mającymi na celu wykrycie nagłych zmian jasności, np. krawędzi. Jest to również zastosowanie funkcji splotu z maską zawierającą dodatkowo wartości ujemne. Laplasjan, wykorzystujący drugą pochodną, jest określany następująco: $L(f) = \frac{\delta^2 f}{\delta x^2} + \frac{\delta^2 f}{\delta y^2}$ i w formie dyskretnej jest sumą różnic wartości poszczególnych pikseli oraz piksela centralnego. Przykład działania został przedstawiony na rysunku 4.6. Funkcja OpenCV odpowiedzialna za wyznaczenie laplasjanu obrazu (`Imgproc.Laplacian`) jako argument przyjmuje między innymi rozmiar kontekstu (maski konwolucji). W czasie swojego działania metoda biblioteczna używa operatorów Sobela, czyli operatorów dyskretnego różniczkowania. Wynikiem działania jest obraz w skali szarości (w przeciwieństwie do następnej metody, czyli wykrywania krawędzi metodą Canny'ego, gdzie daną wyjściową jest obraz zbinaryzowany).

(a) Obraz oryginalny

(b) Obraz z wykrytymi krawędziami

Rysunek 4.6: Wykrywanie krawędzi z użyciem laplasjanu

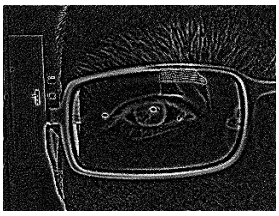
Wykrywanie krawędzi - Canny

Metoda Canny'ego do wykrywania krawędzi jest już bardziej skomplikowanym algorytmem, którego kroki wyglądają następująco:

1. rozmycie obrazu wejściowego metodą Gaussa,
2. szukanie natężenia gradientu metodą Sobela,
3. usuwanie pikseli o niemaksymalnych wartościach,
4. binaryzacja dwuprogowa.

Przykład działania został przedstawiony na rysunku 4.7. Funkcja realizująca wykrywanie krawędzi metodę Canny'ego w OpenCV (`Imgproc.Canny`) oprócz ramki obrazu przyjmuje trzy argumenty reprezentujące progi binaryzacji oraz rozmiar apertury. Dobór progów nie jest łatwym zadaniem, można jednak przyjąć, że dolny próg jest dwa razy mniejszy od górnego. Efektem działania tej metody jest obraz zbinaryzowany z wyznaczonymi krawędziami.

(a) Obraz oryginalny



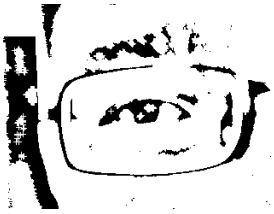
(b) Obraz z wykrytymi krawędziami

Rysunek 4.7: Wykrywanie krawędzi metodą Canny'ego

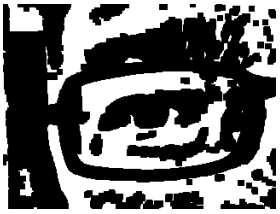
Erozja i dylatacja

Erozja, podobnie jak dylatacja, są podstawowymi operacjami morfologicznymi. W ogólności polegają one na sprawdzeniu konfiguracji punktów elementu strukturalnego z każdym pikselem obrazu. Pierwsza metoda zwana jest również filtrem minimalnym, zaś druga filtrem maksymalnym i obie są możliwe do wykonania na obrazie zbinaryzowanym. Erozja polega na usunięciu pikseli o wartości 1, jeżeli posiadają choć jednego sąsiada o wartości 0. Skutkiem tego jest usunięcie drobnych szczegółów oraz wygładzeniu brzegu figury. Operacja dylatacji jest odwrotna do erozji. Działanie erozji zostało zaprezentowane na rysunku 4.8b, zaś dylatacji na rysunku 4.8c. W bibliotece OpenCV funkcje odpowiadające tym metodom to `Imgproc.erode` oraz `Imgproc.dilate` i oprócz ramki obrazu przyjmują one obiekt będący reprezentacją elementu strukturalnego.

(a) Obraz oryginalny



(b) Obraz po operacji erozji



(c) Obraz po operacji dylatacji

Rysunek 4.8: Operacje morfologiczne

Wyrównanie histogramu

Histogram obrazu w odcieniach szarości jest sumą wszystkich pikseli o danej wartości jasności. Wyrównanie histogramu to operacja na pojedynczym pikselu polegająca na takim jego przekształceniu, aby liczba punktów w danych przedziałach histogramu obrazu była w przybliżeniu taka sama. Widocznym efektem jest zmiana poziomów jasności w ramce obrazu. Operacja ta może dotyczyć histogramu dla całego obrazu lub, podobnie jak w przypadku binaryzacji adaptacyjnej, tylko jego części. Autor w pracy zastosował dwie metody:

– rozciągnięcie histogramu - poprawienie kontrastu w całym obrazie,

– adaptacyjne ograniczenie kontrastu wyrównaniem histogramu (ang. CLAHE - Contrast Limiting Adaptive Histogram Equalization) - wykorzystuje fragmenty obrazu do wyrównania histogramu, ogranicza duże wartości w histogramie, sprawdza się jako dobre rozwiązanie do poprawienia kontrastu w częściach obrazu.

W pracy algorytmy te spełniają znaczącą rolę w uniezależnieniu działania systemu od zmiennych warunków oświetleniowych. Prezentacja działania obu metod została przedstawiona na rysunku 4.9. Klasyczne wyrównanie histogramu posiada swoją implementację w wersji Java biblioteki OpenCV (`Imgproc.equalizeHist`), zaś algorytm CLAHE został napisany z użyciem mechanizmu JNI i wykorzystaniu metod obiektu typu `CLAHE`.

(a) Obraz oryginalny

(b) Obraz po operacji rozciągnięcia histogramu

(c) Obraz po zastosowaniu algorytmu CLAHE

Rysunek 4.9: Wyrównanie histogramu

4.3. Detekcja twarzy

Pierwszym i najważniejszym krokiem w prawidłowym działaniu algorytmu wykrywania senności u kierowcy jest określenie pozycji jego twarzy. Zadanie to warunkuje dalsze prawidłowe działanie systemu. W celu przyspieszenia obliczeń ograniczono region do detekcji głowy. Szerokość wybranego obszaru jest mniejsza o 0.1 szerokości całej ramki obrazu. Ponadto w celu zwiększenia efektywności metod, wyszukiwanie odbywa się na ramce obrazu, z której usunięto zakłócenia metodą rozmycia gaussowskiego oraz na której dokonano operacji wyrównania histogramu z użyciem algorytmu CLAHE.

4.3.1. Sposoby wykrywania twarzy

Jednym ze sposobów na wykrycie twarzy jest wykorzystanie informacji o kolorze skóry. Spośród wielu przestrzeni barw autor wybrał dwie, które posłużyły do dalszych badań:

- HSV (ang. Hue - odcień światła, Saturation - nasycenie koloru, Value - jasność, odcień światła białego),
- YCbCr (Y - luminancja, Cb - chrominancja Y-B, czyli różnicę między luminancją a kolorem niebieskim, Cr - chrominacja Y-R, czyli różnica między luminancją a kolorem czerwonym)

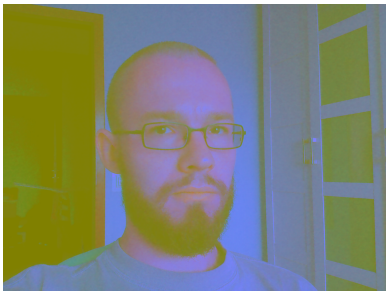
Następnie określono wartości dla odcieni skóry osobno dla każdej przestrzeni:

- $H \in < 0; 25 >$, $S \in < 200; 216 >$, $V \in < 0; 255 >$
- $Y \in < 0; 255 >$, $Cb \in < 133; 173 >$, $Cr \in < 77; 127 >$

Zostały one wykorzystane w funkcji OpenCV `Core.inRange` realizującej binaryzację dwuprogową. Problemami tej metody jest wykrywanie innych części ciała oraz elementów o podobnym kolorze jak

skóra. Metoda nie jest również odporna na modyfikację jasności związaną ze zmianą oświetlenia. Przykładowy efekt działania algorytmu detekcji twarzy z użyciem segmentacji kolorów w przestrzeni YCbCr zaprezentowano na rysunku 4.10.

(a) Obraz oryginalny



(b) Obraz w przestrzeni barw YCbCr

(c) Obraz zbinaryzowany po koniunkcji obrazów binaryzowanych dwuprogowo kanałów Y, Cb i Cr

(d) Obraz po operacji morfologicznej erozji - maska obiektu

Rysunek 4.10: Wykrywanie twarzy w przestrzeni YCbCr

Innym sposobem na detekcję twarzy jest użycie kaskadowych klasyfikatorów Haara. Metoda ta korzysta z uczenia maszynowego do wytrenowania zbioru klasyfikatorów określających występowanie i identyfikację danej cechy. Trenowanie odbywa się na zestawie obrazów z widocznym szukanym obiektem oraz na takich, na których on nie występuje. Klasyfikatory kaskadowe składają się z z podstawowych klasyfikatorów, które są przedstawiane w postaci drzew decyzyjnych. Parametrami wejściowymi, potrzebnymi do wyznaczenia tych struktur, są cechy Haaro-podobne (ang. Haar-like features), które w bibliotece OpenCV przyjmują przykładową postać przedstawioną na rysunku 4.11a.

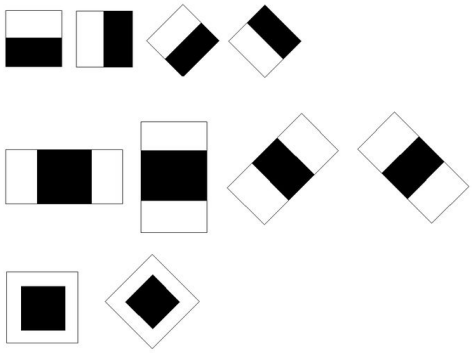
W celu jak najlepszego użycia dostępnych funkcji bibliotecznych w OpenCV określono w odpowiedni sposób wartości argumentów wejściowych. Najważniejsze z nich to:

– minimalny rozmiar twarzy: wysokość i szerokość określono jako 0.35 wysokości przetwarzanej ramki obrazu,

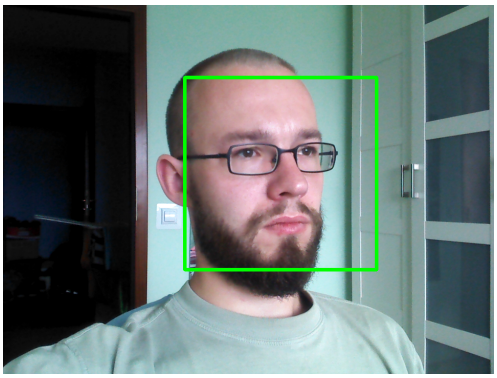
– maksymalny rozmiar twarzy: wysokość i szerokość określono jako 0.75 wysokości przetwarzanej ramki obrazu.

Ze względu na nakład obliczeniowy oraz zdarzające się braki detekcji twarzy autor zastosował rozwiązanie eliminujące te problemy. Po odnalezieniu prostokąta jego pozycja jest zapamiętywana przez następne

10 ramek obrazu po czym następuje ponowna detekcja. Wyszukiwanie twarzy trwa aż do momentu jej ponownego odnalezienia. Rezultat działania kaskadowych klasyfikatorów Haara został przedstawiony na rysunku 4.11b.



(a) Przykładowe cechy Haaro - podobne (rzędami od góry: cechy krawędzi, cechy linii, cechy otoczenia)



(b) Wynik działania kaskadowych klasyfikatorów Haara dla twarzy

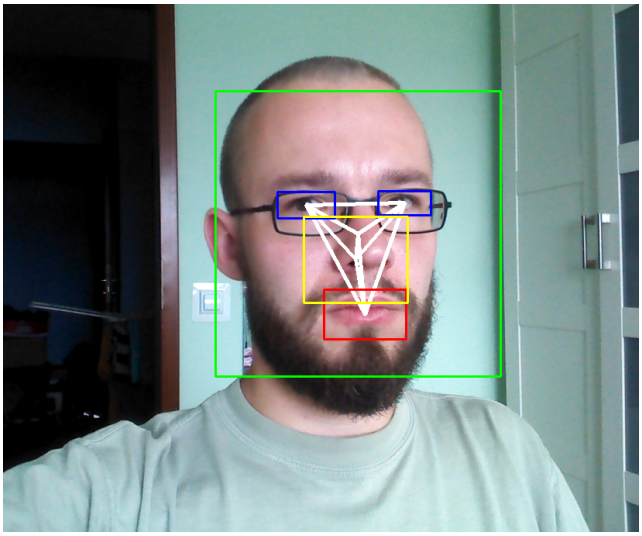
Rysunek 4.11: Kaskadowe klasyfikatory Haara

Jako ostateczne rozwiązanie autor wybrał metodę z użyciem kaskadowych klasyfikatorów Haara ze względu na potwierdzoną skuteczność [36] oraz dobrą szybkość działania.

4.3.2. Określanie pozycji twarzy

Na podstawie badań literaturowych zauważono, że określanie pozycji twarzy również może przyczynić się do wykrywania senności u użytkownika systemu. Przykładem wykrywalnego symptomu jest między innymi szybkie opadanie głowy. Do ustalenia położenia twarzy potrzebny jest jej model, który można uzyskać na wiele sposobów.

Pierwszym z nich jest wykrycie najważniejszych elementów charakterystycznych twarzy: oczu, nosa oraz ust i na podstawie ich położeń określanie pozycji głowy. Autor zaimplementował takie rozwiązanie stosując kaskadowe klasyfikatory Haara. Po wykryciu prostokątów opisujących wykryte obiekty połączono środki otrzymanych prostokątów, które określiły tzw. trójkąt twarzy. Detekcja każdego elementu została ograniczona do przybliżonej pozycji na twarzy użytkownika. Rozwiązanie to pokazano na rysunku 4.12. Wadą takiego tego sposobu jest nieprawidłowe działanie w przypadku braku wykrycia choć jednego elementu. Innym problemem jest częste błędne lub nadmierne wykrywanie ust. Rozwiązanie to również powoduje wprowadzenia dużego obciążenia obliczeniowego, przez co szybkość działania systemu spadła do ok. 2 fps.



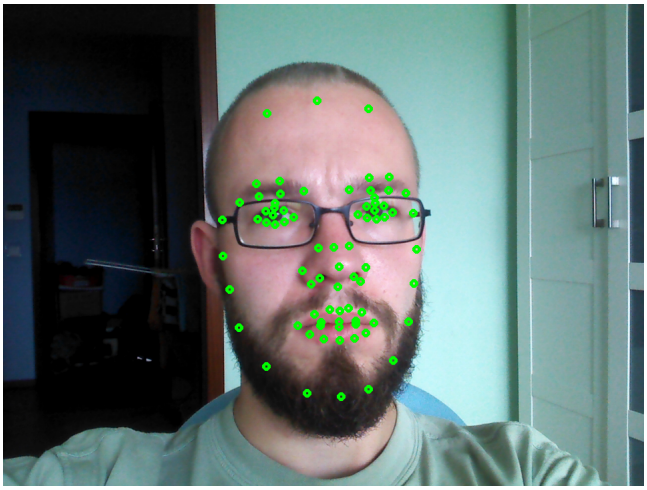
Rysunek 4.12: Położenie elementów charakterystycznych twarzy określających jej położenie

Drugim zaimplementowanym rozwiązaniem jest użycie modelu aktywnego kształtu (ang. ASM - Active Shape Model). Jest to struktura zawierająca informację o średnim kształcie obiektu określonego typu. Posiada również dane, które opisują najbardziej charakterystyczne modyfikacje tego kształtu (uzyskane na podstawie zbioru uczącego). Do implementacji tego algorytmu użyto biblioteki STASM ([31, 1], której funkcje zostały wywołane dzięki mechanizmowi JNI przy użyciu dwóch zaimplementowanych sposobów:

– bezparametrowa metoda do określania modelu aktywnego kształtu dla ramki obrazu, która sama wykrywała charakterystyczne punkty, a następnie wyznaczała model,

– metoda, która przyjmowała jako argumenty punkty początkowe (np. wykryte wcześniej środki prostokątów określających położenie elementów charakterystycznych) a następnie estymowała położenie pozostałych punktów modelu.

Przykład działania został przedstawiony na rysunku 4.13.



Rysunek 4.13: Active Shape Model

Podobnie jak w przypadku pierwszego rozwiązania, tak i podczas użycia metody z użyciem modelu aktywnego kształtu nastąpił duży spadek szybkości działania i w związku z tym zaniechano zastosowania określania pozycji twarzy jako sposobu na wykrywanie senności.

4.4. Detekcja oczu

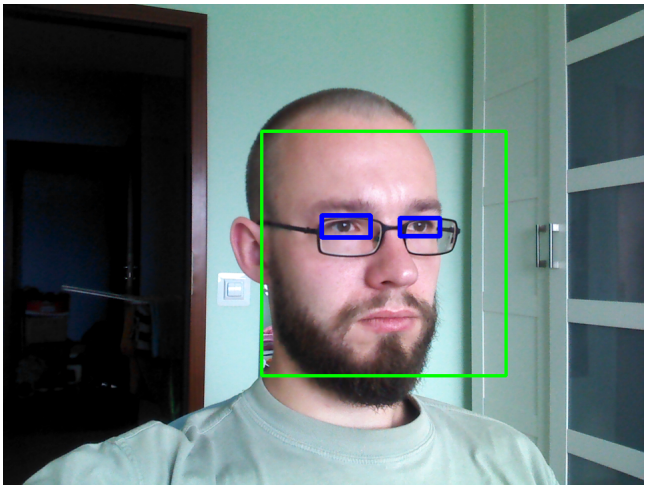
Kolejnym krokiem działania algorytmu użytego do określana senności u kierowcy jest obserwacja oczu. W tym celu musi nastąpić detekcja tych elementów. Jest ona dokonywana na ramce obrazu ograniczonej po wykryciu twarzy obszarem zainteresowania (ROI - ang. Region of Interest). Wysokość otrzymanego prostokąta jest dodatkowo zmniejszana dwukrotnie. Dodatkowymi operacjami, które są wykonywane to wyrównanie histogramu oraz rozmycie gaussowskie. Następnym krokiem jest użycie kaskadowych klasyfikatorów Haara z odpowiednimi parametrami wejściowymi:

– minimalny rozmiar oka: wysokość i szerokość określono jako 0.1 wysokości oryginalnego rozmiaru prostokąta twarzy,

– maksymalny rozmiar oka: wysokość i szerokość określono jako 0.3 wysokości oryginalnego rozmiaru prostokąta twarzy.

Wykryte obszary nie zawsze określają położenie oka, dlatego zastosowano dalsze operacje wspomagające. Jako prawidłowe wybrano te prostokąty, których składowe y punktów określających środek wysokości znajdują się jak najbliżej składowej y punktu określającego środek wysokości prostokąta twarzy. W przypadku braku detekcji oka w obecnej ramce, jako odnaleziony obszar stosowany jest ostatni prawidłowo wykryty prostokąt. Autor zaimplementował również rozwiązanie polegające na zastosowaniu kaskadowych klasyfikatorów Haara nie na całym obszarze twarzy, ale oddzielnie dla prawego i lewego oka (biblioteka OpenCV dostarcza do tego celu osobnych plików konfiguracyjnych). Rozwiązanie to spowodowało jednak obniżenie wydajności oraz nadmierną detekcję obszarów uznanych za oczy i nie zostało ono użyte w ostatecznej wersji systemu.

Działanie algorytmu detekcji oczu z użyciem kaskadowych klasyfikatorów Haara zostało zaprezentowane na rysunku 4.14.



Rysunek 4.14: Wynik działania kaskadowych klasyfikatorów Haara dla twarzy i oczu

4.5. Wykrywanie oznak senności kierowcy

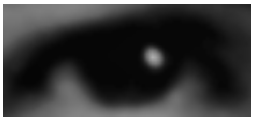
Głównym zadaniem systemu detekcji senności i zapobiegania zaśnięciu jest wykrywanie zdarzeń, które świadczą o tym problemie. W pracy autor wykorzystał jedno z najlepszych i najbardziej miarodajnych sposobów, czyli detekcję czy oko jest zamknięte.

4.5.1. Detekcja zamkniętego oka

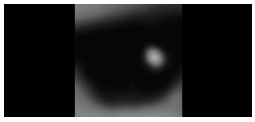
Autor zaimplementował kilka sposobów pozwalających na detekcję zamkniętego oka. Wszystkie te operacje są wykonywane na obrazie z nałożonym obszarem zainteresowania określonym prostokątem otrzymanym po poprzednim etapie pracy systemu, czyli detekcji oka. Dodatkowo wykryty obszar został zmniejszony co pozwoliło na usunięcie brwi, krawędzi okularów i ciemnych kącików oka.

Z użyciem binaryzacji

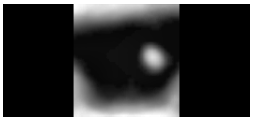
Metoda ta opiera się na różnicy jasności między ramką obrazu z okiem otwartym a zamkniętym. Po przeprowadzeniu operacji wyrównania histogramu stosowana jest binaryzacja jednoprogowa z progiem ustalonym na średnią jasność w przetwarzanej ramce. Kolejnym krokiem jest wyeliminowanie pojedynczych szczegółów dzięki operacji erozji. Następnie obliczany jest stosunek liczby pikseli czarnych do białych (określających obiekt oraz tło) i na tej podstawie następuje decyzja, czy oko jest zamknięte. Zamiast binaryzacji jednoprogowej autor testował również binaryzację adaptacyjną. Poszczególne etapy zostały przedstawione na rysunku 4.15, zaś rysunek 4.16 prezentuje rezultaty działania metody dla oka otwartego i zamkniętego. Porównanie to ma na celu pokazania różnic, które mogą zostać wykorzystane.



(a) Obraz wejściowy



(b) Obraz z ustawionym ROI



(c) Obraz po wyrównaniu histogramu (CLAHE)

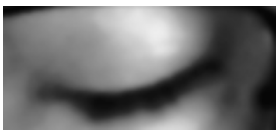
(d) Obraz po operacji binaryzacji jednoprogowej

(e) Obraz zbinaryzowany po operacji erozji - obraz wyjściowy

Rysunek 4.15: Metoda detekcji zamkniętego oka przy pomocy binaryzacji jednoprogowej - etapy działania

(a) Obraz z okiem otwartym

(b) Rezultat metody dla oka otwartego



(c) Obraz z okiem zamkniętym

(d) Rezultat metody dla oka zamkniętego

Rysunek 4.16: Metoda detekcji zamkniętego oka przy pomocy binaryzacji jednoprogowej - rezultaty

Wadą tego rozwiązania okazał się brak odporności na zmieniająca się jasność w ramce obrazu. Czasami jako otwarte oko błędnie były wykrywane cienie na zamkniętej powiece lub oprawki okularów. Metoda ta nie została zatem uwzględniona w ostatecznym rozwiązaniu.

Zastosowanie laplasjanu

Dobre rezultaty dało zastosowanie laplasjanu do wykrywania zamkniętego oka. Algorytm, który wykorzystuje wykrywanie krawędzi na podstawie drugiej pochodnej wygląda następująco:

1. Wyrównanie histogramu z użyciem metody CLAHE.
2. Zastosowanie rozmycia gaussowskiego.
3. Wykrycie krawędzi z zastosowaniem laplasjanu i rozmiarem apertury (kontekstu) równym 3.
4. Zwiększenie kontrastu na obrazie.
5. Progowanie obrazu metodą adaptacyjną - wyróżnienie krawędzi.

6. Zastosowanie operacji morfologicznych (erozji i dylatacji) w celu usunięcia niepotrzebnych szczegółów oraz wygładzenia krawędzi.

7. Ograniczenie obszaru zainteresowania (ROI) do prostokąta opisującego tęczówkę.

8. Wyznaczenie prostych pokrywających się z krawędzią górnej i dolnej powieki. Do tego celu obliczono liczbę białych pikseli w wierszach i jako proste przyjęto te wiersze, które zawierały odpowiednią liczbę tych pikseli (co najmniej $\frac{1}{4}$ maksymalnej liczby) oraz były wierszami granicznymi (pierwszy i ostatni).

9. Obliczenie odległości między otrzymanymi prostymi i na jej podstawie decyzja, czy oko jest zamknięte.

Poszczególne etapy zostały przedstawione na rysunku 4.17, zaś rysunek 4.18 prezentuje rezultaty działania metody dla oka otwartego i zamkniętego.

(a) Obraz wejściowy

(b) Obraz po wyrównaniu histogramu (CLAHE)

(c) Obraz po rozmyciu gaussowskim

(d) Obraz po zastosowaniu wykryciu krawędzi z użyciem laplasjanu

(e) Obraz po operacji binaryzacji

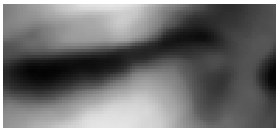
(f) Obraz zbinaryzowany po operacji erozji

(g) Obraz zbinaryzowany z ustawionym ROI po odnalezieniu linii powiek - obraz wyjściowy

Rysunek 4.17: Metoda detekcji zamkniętego oka przy pomocy wykrywania krawędzi z użyciem laplasjanu - etapy działania

(a) Obraz z okiem otwartym

(b) Rezultat metody dla oka otwartego



(c) Obraz z okiem zamkniętym

(d) Rezultat metody dla oka zamkniętego

Rysunek 4.18: Metoda detekcji zamkniętego oka przy pomocy wykrywania krawędzi z użyciem laplasjanu - rezultaty

Projekcja wertykalna (wierszowa)

Metodą, która dała jedne z najlepszych rezultatów wykrywania faktu zamkniętego oka podczas prac implementacyjnych jest projekcja wertykalna średnich wartości jasności. Polega ona na obliczeniu wartości średniej jasności pikseli w każdym wierszu ramki obrazu. Efektem tego jest zdefiniowanie funkcji średniej jasności względem wiersza. Decydowanie o tym, czy oko jest zamknięte następuje na podstawie jej charakterystycznych przebiegów w przypadku oka zamkniętego i otwartego. W celu otrzymania wykresu projekcji wertykalnej opracowano następujący algorytm:

1. Zastosowanie rozmycia gaussowskiego.
2. Ograniczenie obszaru zainteresowania poprzez ograniczenie szerokości ramki.
3. Obliczenie średnich wartości jasności dla każdego wiersza ramki.
4. Zastosowanie filtru medianowego dla otrzymanego ciągu obliczonych wartości. Operacja ta pozwoliła na uzyskanie uśrednionych wartości, pozbycie się wartości znacząco różniących się od pozostałych oraz nie spowodowało utraty informacji.
5. Znormalizowanie wartości ciągu do przedziału $< 0; 255 >$ - przedział wartości jasności w bibliotece OpenCV.
6. Podjęcie decyzji czy oko jest zamknięte na podstawie wartości ciągu.

Ostatni, najważniejszy etap został podzielony na kilka części. Wartym nadmienienia jest fakt, iż najmniejsza wartość w otrzymanym ciągu reprezentuje dwie sytuacje:

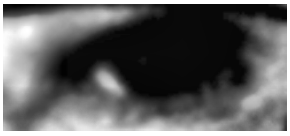
– dla oka otwartego - określa położenie środka źrenicy oka,

– dla oka zamkniętego - określa przybliżone położenie linii powieki.

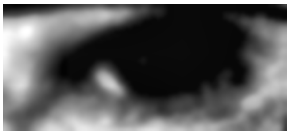
Zauważono również prawidłowość, że dla stałego otoczenia wartości najmniejszej zbiór wartości należących do tego otoczenia jest większy w przypadku oka otwartego niż w przypadku oka zamkniętego. Na

tej podstawie określono próg liczby tych wartości (na 30% liczby wszystkich wartości w ciągu), dzięki któremu podejmowana jest decyzja o tym, czy oko jest zamknięte. Problemem, na który napotkał się autor podczas implementacji tej metody jest nieprawidłowe działanie podczas występowania odblasków w siatkówce oka. W celu eliminacji tego zastosowano obniżanie jasności pikseli o największych wartościach. Poszczególne etapy zostały przedstawione na rysunku 4.19, zaś rysunki 4.20 oraz 4.21 prezentują rezultaty działania metody dla oka otwartego i zamkniętego dla różnego rodzaju oświetlenia.

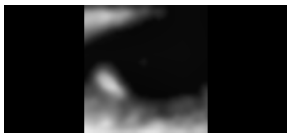
(a) Obraz wejściowy



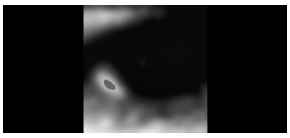
(b) Obraz po wyrównaniu histogramu (CLAHE)



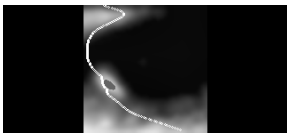
(c) Obraz po rozmyciu gaussowskim



(d) Obraz z ustawionym ROI



(e) Obraz po zapełnieniu jasnych pikseli ciemniejszymi



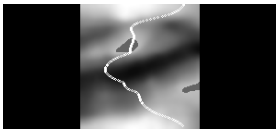
(f) Obraz wyjściowy z nałożonym wykresem projekcji wertykalnej

Rysunek 4.19: Metoda detekcji zamkniętego oka przy pomocy projekcji wertykalnej - etapy działania

(a) Obraz z okiem otwartym

(b) Rezultat metody dla oka otwartego

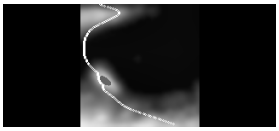
(c) Obraz z okiem zamkniętym



(d) Rezultat metody dla oka zamkniętego

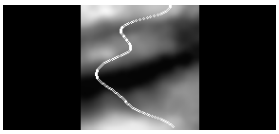
Rysunek 4.20: Metoda detekcji zamkniętego oka przy pomocy projekcji wertykalnej - rezultaty detekcji podczas wysokiego natężenia oświetlenia

(a) Obraz z okiem otwartym



(b) Rezultat metody dla oka otwartego

(c) Obraz z okiem zamkniętym



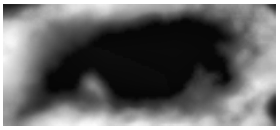
(d) Rezultat metody dla oka zamkniętego

Rysunek 4.21: Metoda detekcji zamkniętego oka przy pomocy projekcji wertykalnej - rezultaty detekcji podczas niskiego natężenia oświetlenia

Projekcja horyzontalna (kolumnowa)

Metoda ta jest identyczna do poprzedniej opiera się jednak na średniej jasności w kolumnach. W związku z tym zmianie uległ również sposób detekcji, czy oko jest zamknięte. Z otrzymanego ciągu (o długości n) wybierane są wartości pierwsza a_0 oraz ostatnia a_n. Następnie na podstawie różnicy między nimi tworzony jest nowy ciąg wartości, który jest ciągiem arytmetycznym o różnicy między elementami równej $\frac{a_0-a_n}{n}$ i pierwszym elemencie równym a_0. Porównując oba ciągi zauważono następującą prawidłowość: w przypadku oka otwartego liczba wartości ciągu oryginalnego większych od odpowiadających im wartości nowo utworzonoego ciągu jest większa niż w przypadku oka zamkniętego. Stanowi to podstawę do decyzji o tym, czy oko jest zamknięte. Poszczególne etapy zostały przedstawione na rysunku 4.22, zaś rysunki 4.23 oraz 4.24 prezentuje rezultaty działania metody dla oka otwartego i zamkniętego dla różnego rodzaju oświetlenia.

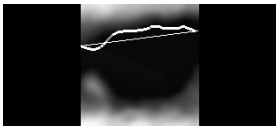
(a) Obraz wejściowy



(b) Obraz po wyrównaniu histogramu (CLAHE)

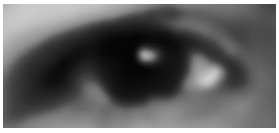
(c) Obraz po rozmyciu gaussowskim

(d) Obraz z ustawionym ROI

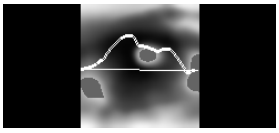


(e) Obraz wyjściowy z nałożonym wykresem projekcji wertykalnej

Rysunek 4.22: Metoda detekcji zamkniętego oka przy pomocy projekcji horyzontalnej - etapy działania

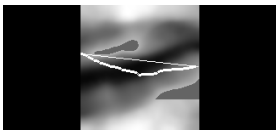


(a) Obraz z okiem otwartym



(b) Rezultat metody dla oka otwartego

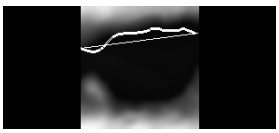
(c) Obraz z okiem zamkniętym



(d) Rezultat metody dla oka zamkniętego

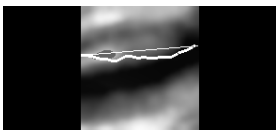
Rysunek 4.23: Metoda detekcji zamkniętego oka przy pomocy projekcji horyzontalnej - rezultaty detekcji podczas wysokiego natężenia oświetlenia

(a) Obraz z okiem otwartym



(b) Rezultat metody dla oka otwartego

(c) Obraz z okiem zamkniętym



(d) Rezultat metody dla oka zamkniętego

Rysunek 4.24: Metoda detekcji zamkniętego oka przy pomocy projekcji horyzontalnej - rezultaty detekcji podczas niskiego natężenia oświetlenia

Do końcowego rozwiązania zostały wybrane metody opierające się na projekcji wartości średniej. Autor umożliwił użytkownikowi wybór metody, domyślne wybrane jest używanie obu metod jednocześnie ze względu na najlepsze rezultaty.

4.5.2. Metody wykrywania senności kierowcy

Ze względu na trudność w prostym określeniu stopnia senności i wykrycia zaśnięcia autor zaimplementował dwie metody:

– PERCLOS (ang. Percentage of Eye Closure) - opierająca się na dłuższej obserwacji obszaru oka,

– bazująca na mierzeniu długości czasu zamknięcia oka - skutki jej działania następują po kilku sekundach od wykrycia zamkniętego oka.

PERCLOS

Metoda PERCLOS określa stosunek liczby ramek obrazu z zamkniętych okiem do liczby wszystkich ramek obrazu w określonym odcinku czasu. Okres, w którym badana jest ta zależność została ustalona na

podstawie testów (opisanych w następnym rozdziale) na 30 sekund. Następnie zostały określone stopnie, dzięki którym można zakwalifikować otrzymaną wartość do stopnia senności:

– niski stopień senności - wartość PERCLOS: 0.2, jako ostrzeżenie wyświetlany jest symbol graficzny na wyświetlaczu urządzenia mobilnego,

– wysoki stopień senności - wartość PERCLOS: 0.3, jako ostrzeżenie uruchamiany jest alarm dźwiękowy.

Metoda ta pozwala na detekcję ospałości związaną z krótkotrwałym, ale częstym zamykaniem oka. Dzięki niej można wykryć symptomy narastającej senności zatem algorytm ten ma charakter zapobiegawczy.

Długi czas zamknięcia oka

Innym sposobem na wykrycie zaśnięcia jest sprawdzanie ramek po kolei i w przypadku wykrycia kilku pod rząd z zamkniętymi oczami następuje reakcja systemu. Dla tej metody zostały określone dwa stopnie senności:

– średni stopień senności - długi czas zamknięcia oczu powoduje uruchomienie krótkiego sygnału dźwiękowego (czas trwania tego stanu został określony na 2 sekundy, użytkownikowi została udostępniona zmiana tej wartości),

– wysoki stopień senności (zaśnięcie) - kilkukrotne powtórzenie faktu długiego czasu zamknięcia oczu (liczba powtórzeń została określona wartością 10) lub zamknięcie oczu na jeszcze dłuższy czas (5 sekund) powoduje uruchomienie alarmu.

Metoda ta pozwala na wykrywanie momentu zaśnięcia lub zaawansowanej senności. Sygnały dźwiękowe mają na celu przywrócenie świadomości kierowcy. Ten algorytm ma charakter ostrzegawczy, gdyż wykrywa już ostateczny moment zaśnięcia.

4.6. Mechanizm ostrzegania kierowcy

W przypadku wykrycia określonego poziomu senności system w odpowiedni sposób reaguje na tą sytuację. Autor zdecydował się na 3 rodzaje ostrzeżeń:

– informacja graficzna na wyświetlaczu telefonu komórkowego w przypadku wykrycia senności, której poziom można określić jako niski,

– ostrzeżenie dźwiękowe, czyli pojedynczy sygnał dźwiękowy w przypadku wykrycia senności, której poziom można określić jako średni,

– alarm dźwiękowy, czyli głośniejszy, ciągły sygnał dźwiękowy w przypadku wykrycia wysokiego poziomu senności.

Do alarmów dźwiękowych wykorzystano dostępne na platformie Android sygnały dźwiękowe (dla obiektu `ToneGenerator`) i tak dla:

– ostrzeżenia dźwiękowego użyto sygnału `ToneGenerator.TONE_CDMA_ALERT_CALL_GUARD` trwającego 125ms,

– alarmu dźwiękowego użyto sygnału `ToneGenerator.TONE_SUP_ERROR` możliwego do wyłączenia dopiero po interwencji użytkownika.

5. Wyniki

W tym rozdziale autor podsumowuje wyniki, jakie otrzymywał w trakcie pracy nad programem (są to skutki metod lokalizacji twarzy, oczu oraz detekcji faktu zamkniętych oczu), a także wyniki końcowe dotyczące wykrycia senności i zaśnięcia kierowcy. Uzyskane w procesie implementacji oraz testowania rezultaty miały postać:

- obrazów, które były oceniane wzrokowo przez autora,
- danych liczbowych, które wykorzystano do dalszej obróbki oraz wizualizacji przydatnej do ich analizy.

5.1. Opis przeprowadzonych testów

Testy podzielono na dwie części:

- związane ze skutecznością metod lokalizacji twarzy i oczu - przygotowujące do głównego elementu systemu,
- związane z detekcją zaśnięcia kierowcy - wyniki znalezienia faktu zamkniętego oka oraz rezultaty metod wykrywania senności.

Działanie systemu sprawdzane było zarówno w warunkach stałego, jak i zmiennego oświetlenia. Podobnie było z oświetleniem jednolitym i zróżnicowanym w całej ramce obrazu. Ze względów bezpieczeństwa testy nie były przeprowadzane podczas jazdy samochodem, część z nim została jednak dokonana w trakcie postoju na parkingu. Większość czasu spędzona na analizie zaimplementowanych algorytmów była jednak związana ze środowiskiem zbliżonym do laboratoryjnego. Ponadto autor przeprowadził również test długości pracy systemu na urządzeniu mobilnym.

5.2. Wyniki działania algorytmów do lokalizacji elementów charakterystycznych

W celu wyszukiwania elementów charakterystycznych autor wykorzystał opisane w rozdziale 4 kaskadowe klasyfikatory Haara. Skuteczność ich działania została opisana w następnych podrozdziałach. Opis sposobu uzyskania tych wyników jest jednak wspólny. Uzyskano je badając liczbę ramek, w których nastąpiła prawidłowa detekcja badanych elementów. Eksperyment powtórzono kilkukrotnie, a następnie obliczono średnią z otrzymanych wyników.

5.2.1. Wyniki działania algorytmów wykrywania twarzy

Skuteczność wykrywania twarzy na obrazie bez dodatkowych operacji została określona na poziomie 40%. Zastosowanie wyrównania histogramu na całej ramce obrazu pozwoliła na zwiększenie tej wartości do ok. 75%. W przypadku problemów z detekcją, brak określonego ROI do dalszego przetwarzania został wyeliminowany poprzez zapamiętywanie ostatniego, prawidłowo odnalezionego prostokąta określającego obszar twarzy.

5.2.2. Wyniki działania algorytmów wykrywania oczu

Skuteczność wykrywania oczu na obrazie z ustawionym ROI bez dodatkowych operacji została określona na poziomie 70%, jednak wyrównanie histogramu metodą CLAHE pozwala na zwiększenie tej wartości aż do 85%. W przypadku braku odnalezienia oczu w obecnej ramce do dalszego przetwarzania przekazywany jest ostatni prawidłowo odnaleziony prostokąt.

Do operacji wyszukiwania pojedynczego oka warto jest zawęzić obszar poszukiwań, gdyż zdarza się, że znajdywane są elementy na włosach, powieki oraz usta. Ponadto należy również odpowiednio określić minimalne oraz maksymalne rozmiary prostokąta opisującego obiekt. Kaskadowe klasyfikatory Haara, dostępne z poziomu biblioteki OpenCV, wyszukujące oczy nie są odporne na rotację. Daje jednak możliwość wytrenowania własnych przy użyciu zbioru zawierającego obrócone obiekty.

5.3. Wyniki działania algorytmów wykrywania senności

5.3.1. Wyniki działania algorytmów detekcji zamkniętych oczu

Wpływ na wykrywanie zaśnięcia znacząco zależy od prawidłowego określenia, czy badane oko jest zamknięte. Autor przetestował zaimplementowane algorytmy i do ostatecznego rozwiązania wybrał metody opierające się na:

– analizie projekcji wertykalnej średniej jasności,

– analizie projekcji horyzontalnej średniej jasności.

Metoda, która używała binaryzacji została odrzucona w trakcie badań ze względu na poniższe cechy:

– słaba odporność na zmianę oświetlenia,

– zmienne działanie w trakcie upływu dnia i rodzaju światła,

– niska skuteczność działania podczas słabego oświetlenia.

Metoda z użyciem laplasjanu dawała porównywalne wyniki detekcji jak ta z analizą projekcji. Nie znalazła się ona jednak w końcowym systemie ze względu na występujące błędne wykrywanie powieki górnej. Nieprawidłowość ta wynika z kształtowania się ciemnych fałd skóry, gdy oko jest zamknięte.

Metody wybrane do końcowego rozwiązania zostały przetestowane pod względem skuteczności wykrywania zamkniętego oka po uprzednim, prawidłowym zlokalizowaniu go. Testy zostały podzielone ze

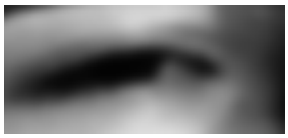
względu na stopień natężenia światła oraz zastosowany algorytm i przeprowadzono je na 300 kolejnych ramkach obrazu. Symulowanie zmęczenia polegało na nieustannym zamykaniu oczu, zaś ruchy głową były ograniczone. Wyniki zaprezentowano w tabelach 5.1 oraz 5.2. Przykłady błędnych detekcji oka zamkniętego zostały zaprezentowane na rysunku 5.1 i zostały one wykonane automatycznie, gdy system wykrył, że oko jest zamknięte.

Warunki / Algorytm	Oczy otwarte, niskie natężenie oświetlenia	Oczy zamknięte, niskie natężenie oświetlenia
Obie metody	288 (96%)	241 (80%)
Projekcja wertykalna	283 (94%)	244 (81%)
Projekcja horyzontalna	271 (90%)	294 (98%)

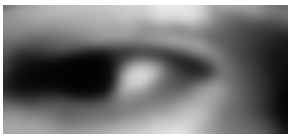
Tablica 5.1: Wyniki skuteczności działania metod detekcji zamkniętego oka dla niskiego natężenia oświetlenia

Warunki / Algorytm	Oczy otwarte, wysokie natężenie oświetlenia	Oczy zamknięte, wysokie natężenie oświetlenia
Obie metody	270 (90%)	118 (40%)
Projekcja wertykalna	260 (87%)	122 (40%)
Projekcja horyzontalna	246 (82%)	291 (97%)

Tablica 5.2: Wyniki skuteczności działania metod detekcji zamkniętego oka dla wysokiego natężenia oświetlenia



(a) Oko przymrużone



(b) Oko otwarte ze skierowanym wzrokiem w bok

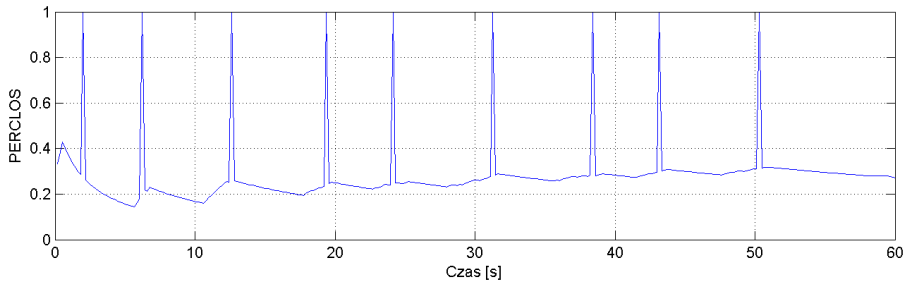
Rysunek 5.1: Przykładowe zrzuty ekranu z działania systemu podczas detekcji zamkniętych oczu.

Otrzymane wyniki świadczą o dobrym doborze metod opierających się na wykorzystaniu projekcji średnich jasności i użyciu ich w ostatecznym rozwiązaniu. Niska skuteczność dla oka zamkniętego podczas wysokiego natężenia oświetlenia może świadczyć o nieprawidłowym początkowym wykryciu oka. Bardzo dobre wyniki skuteczności w detekcji oka zamkniętego (w przypadku fizycznego zamknięcia oka) posiada metoda projekcji horyzontalnej. Dla oczu otwartych najlepsze rezultaty zostały osiągnięte podczas użycia obu metod projekcji. Jest to ważna informacja, gdyż w ostatecznym rozwiązaniu bardzo niekorzystne jest błędne klasyfikowanie oczu otwartych jako zamkniętych. Z tego powodu w końcowym rozwiązaniu domyślne wybrana metoda w aplikacji to połączenie działania obu algorytmów projekcji.

5.3.2. Wyniki działania metod wykrywania senności

Autor przeprowadził również testy działania metod wykrywania senności i momentu zaśnięcia. Pozwoliły one na dostosowanie wartości progów dla obu metod. Wyniki w postaci wykresów dotyczą głównie metody PERCLOS. Testy zrealizowano dla różnych ustawień czasowych i ich rezultaty zaprezentowano na poniższych wykresach.

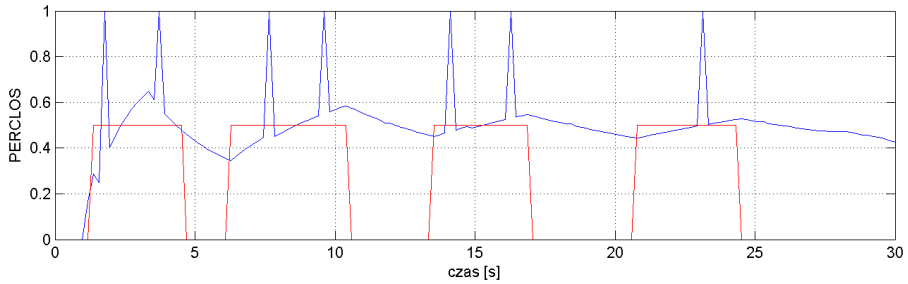
Wyniki metody PERCLOS i metody badającej długość czasu zamknięcia oka dla pomiaru 1 minutowego



Rysunek 5.2: Wynik metody PERCLOS w czasie 1 minuty z zaznaczonymi momentami ostrzeżeń metody wykrywającej długie zamknięcie oka dla użytkownika śpiącego

Test ten był przeprowadzony dla okna czasowego trwającego 30 sekund dla metody PERCLOS i miał na celu detekcję momentów ostrzeżeń dla użytkownika. Ostrzeżenia (symbolizowane przez osiągnięcie najwyższej wartości na wykresie) były generowane przez metodę badającą długość czasu zamknięcia oka. Na chwilę przed ostrzeżeniem widoczne jest zwiększanie się wartości metody PERCLOS, zaś po nim spadek. Po usłyszeniu sygnału autor symulował wybudzanie się.

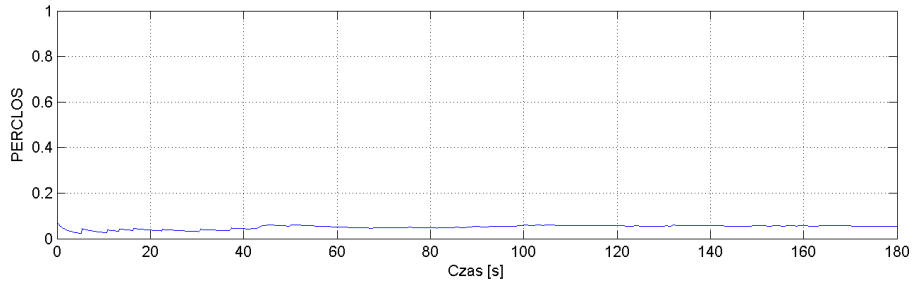
Wyniki metody PERCLOS i metody badającej długość czasu zamknięcia oka dla pomiaru 30 sekundowego



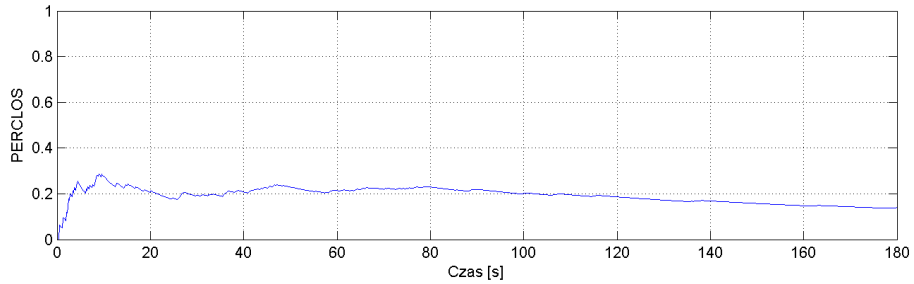
Rysunek 5.3: Wynik metody PERCLOS w czasie 30 sekund z zaznaczonymi momentami ostrzeżeń metody wykrywającej długi czas zamknięcie oka dla użytkownika śpiącego

Rozszerzeniem poprzedniego testu było dodanie ręcznego wskazywania przez użytkownika momentów zamknięcia oczu. Są one zaznaczone na wykresie linią czerwoną. Wartość 0.5 określa moment zamknięcia oczu. Zostało dowiedzione, że automatyczne wykrywanie momentów zamknięcia oczu wraz z punktami ostrzeżenia pokrywają się z naniesionymi manualnymi wskazaniami.

Wyniki metody PERCLOS dla okna czasowego ustawionego na 3 minuty



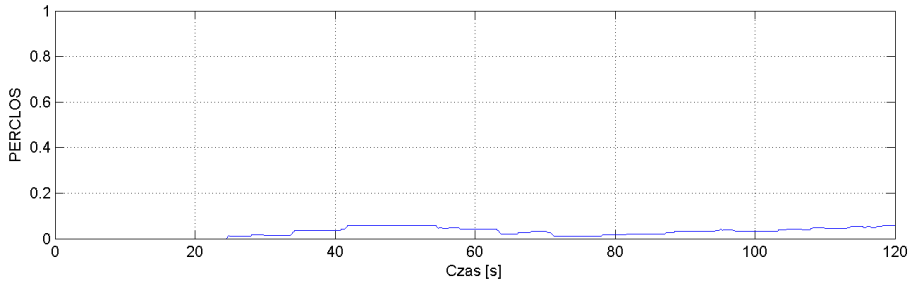
Rysunek 5.4: Wynik metody PERCLOS w czasie 3 minut dla użytkownika aktywnego (wartość średnia ≈ 0.0505, wariancja ≈ 0.00007)



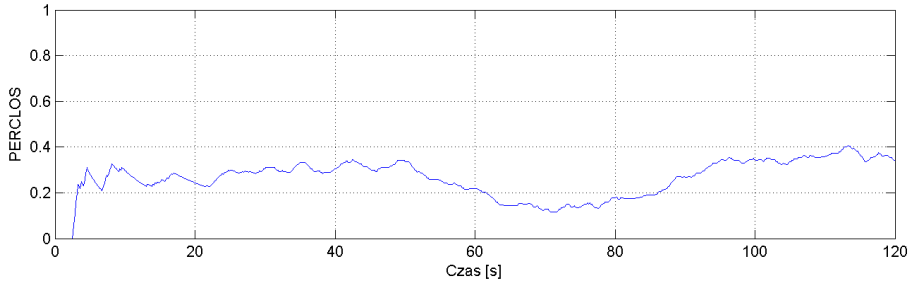
Rysunek 5.5: Wynik metody PERCLOS w czasie 3 minut dla użytkownika śpiącego (wartość średnia ≈ 0.1930, wariancja ≈ 0.0012)

Cechą charakterystyczną wykresów dla 3 minutowego badania podobnie jak w przypadku badania trwającego 30 sekund jest różnica między średnimi wartościami dla użytkownika aktywnego i sennego. Jest ona jednak zbyt mała (podobnie jak wartości wariancji), dlatego autor postanowił, że okno czasowe metody PERCLOS powinno być mniejsze.

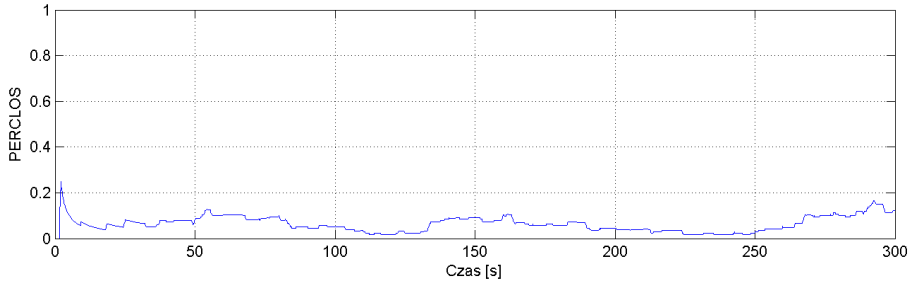
Wyniki metody PERCLOS dla okna czasowego ustawionego na 30 sekund



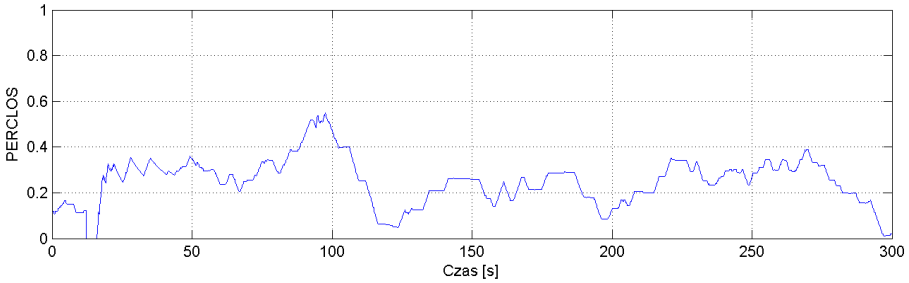
Rysunek 5.6: Wynik metody PERCLOS w czasie 2 minut dla użytkownika aktywnego (wartość średnia ≈ 0.0274, wariancja ≈ 0.0004)



Rysunek 5.7: Wynik metody PERCLOS w czasie 2 minut dla użytkownika śpiącego (wartość średnia ≈ 0.2626, wariancja ≈ 0.067)



Rysunek 5.8: Wynik metody PERCLOS w czasie 5 minut dla użytkownika aktywnego (wartość średnia ≈ 0.0634, wariancja ≈ 0.0011)



Rysunek 5.9: Wynik metody PERCLOS w czasie 5 minut dla użytkownika śpiącego (wartość średnia ≈ 0.2499, wariancja ≈ 0.0104)

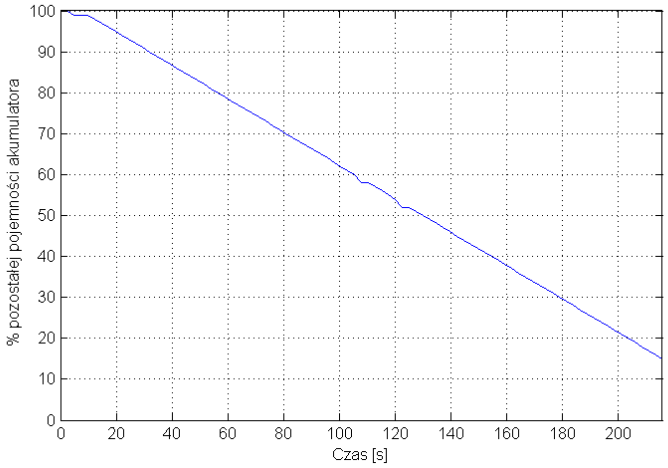
Otrzymane wyniki pozwalają na następujące wnioski:

- wartości metody PERCLOS dla użytkownika sennego oscylują w zakresie $< 0.2; 04 >$, zaś dla użytkownika aktywnego w zakresie $< 0.0; 0.1 >$,
- średnia wartość dla użytkownika sennego i śpiącego jest wyższa niż dla użytkownika aktywnego i wynoszą one:
  - dla użytkownika sennego - ok. 0.25;
  - dla użytkownika aktywnego - ok. 0.05;
- wartości dla użytkownika śpiącego cechuje duża zmienność, ich wariancja jest dużo wyższa niż dla użytkownika aktywnego.

Użycie okna czasowego trwającego 30 sekund pozwala na wniosek, iż wyniki dla badania trwającego 2 minuty i 5 minut są zbliżone jeżeli chodzi o średnią wartość. Dzięki temu autor wybrał to okno czasowe jako wartość stałą dla metody PERCLOS. Mniejsze okno czasowe pozwala również na zwiększenie wariancji wyników. Test przeprowadzany w czasie 5 minut pozwala również na wyciągnięcie dodatkowych wniosków. Na chwilę przed 100 sekundą eksperymentu wartość metody PERCLOS osiągnęła wartość maksymalną, po czym znacząco zmalała. W trakcie tego momentu autor długo nie otwierał oczu, po czym, słysząc alarm, zasymulował przebudzenie się.

5.4. Wyniki testu zużycia baterii

Autor przeprowadził również test zużycia baterii podczas pracy systemu. Oprócz działającej aplikacji włączony był również moduł Wi-Fi. Wykres zużycia pojemności akumulatora w czasie przedstawiony jest na wykresie 5.10. Wbudowany czujnik zużycia baterii w telefonie wskazywał, że system zużywał ok. 43% pojemności akumulatora w trakcie swojego działania. Pozwoliło to na ok. 4 godzinną ciągłą pracę aplikacji.



Rysunek 5.10: Zużycie pojemności baterii w czasie

Z wykresu wynika, że aplikacja jest w stanie działać przez około 3,5 godziny dla w pełni naładowanej baterii telefonu. Pozwala to na użycie systemu w czasie trwania jazdy, po którym zalecany jest odpoczynek.

5.5. Wnioski

Powtarzalność oraz weryfikowalność otrzymanych wyników określają jakość rozwiązania, które zostało opracowane przez autora oraz umożliwiają konkretną interpretację. Na postać wyników metod wykrywania senności największy wpływ ma skuteczność wykrywania zamkniętego oka, które to oko powinno być odnalezione prawidłowo. Zauważalne i zasadnicze oddziaływanie na końcowe rezultaty ma zmienna jasność ramki obrazu podczas pracy systemu. Opracowane rozwiązanie ostateczne wykrywa senność i zaśnięcie u użytkownika. Dzięki wynikom otrzymanym podczas pracy aplikacji, system może jednoznacznie stwierdzić stan kierowcy i zareagować w odpowiedni sposób. O prawidłowości działania świadczy również fakt, iż wyniki zachowują spodziewane cechy charakterystyczne w zależności od stanu kierowcy (aktywność, senność):

– średnia wartość dla metody PERCLOS

– zmienność wyników, której miarą jest wariancja.

– wykrywalne momenty zasypiania i budzenia się.

6. Podsumowanie

Rozdział ten stanowi podsumowanie całej pracy, zawiera wnioski ogólne oraz próbę oceny proponowanego rozwiązania. Przedstawione są również możliwe koncepcje rozwoju powstałego systemu.

6.1. Wnioski ogólne oraz ocena proponowanego rozwiązania

Powstanie propozycji systemu do ostrzegania kierowcy przez zaśnięciem na podstawie obserwacji twarzy oraz oczu jest skutkiem prac badawczych oraz implementacyjnych. Zadanie to wymagało zatem podzielenia pracy na dwie części. Etap związany z badaniami literaturowymi pozwolił na dokładniejsze zapoznanie się z rozwiązaniami, które zostały już zaproponowane przez naukowców. Część analityczno - implementacyjna wymagała zebrania posiadanych informacji oraz stworzenia rozwiązań pośrednich, które stanowiły podstawę do osiągnięcia celu postawionego na początku pracy. Kolejnym krokiem było opracowanie końcowego rozwiązania na podstawie uzyskanych wniosków.

Wybrane zostały najlepiej działające, według autora, metody do lokalizacji twarzy i oka, algorytmy do detekcji oka zamkniętego oraz sposoby wykrywania senności. Wybór ten poparty był analizą rozwiązań pośrednich. W trakcie pracy zostały ponadto określone warunki i ograniczenia, które zapewniają prawidłowe funkcjonowanie aplikacji. Proponowane rozwiązanie opiera się na wykorzystaniu tylko kamery telefonu komórkowego ze względu na wygodę działania oraz niskie koszty używania. Cechy te stanowią przewagę nad rozwiązaniami komercyjnymi dostępnymi tylko w najnowszych modelach samochodów. System, aby stał się w pełni funkcjonalną i niezawodną aplikacją zwiększającą bezpieczeństwo użytkowników wymaga dopracowania zaimplementowanych rozwiązań.

Zagadnienie wykrywania zaśnięcia u kierowcy oraz poziomu senności jest wieloaspektowym problemem. Pomimo dużej liczby istniejących realizacji, jest on stale analizowany, badany, testowany i ulepszany. Metody wizyjne wykorzystujące kamerę telefonu komórkowego stanowią obszar, który ciągle nie jest całkowicie poznany. Autor zaimplementował system, który opierając się na obserwacji oka, wyznacza przybliżony moment zaśnięcia kierowcy i reaguje na ten fakt w odpowiedni sposób. Dokonał tego bazując na dotychczas opracowanych rozwiązaniach. System stanowi solidną podstawę do opracowania kompleksowego rozwiązania, które może przyczynić się do poprawy bezpieczeństwa jego użytkowników.

6.2. Kierunki rozwoju

Elementem, który został wspomniany przez autora jest analiza ruchów głowy do wykrywania senności. Należało stworzyć rozwiązanie, które pozwalałoby to wykorzystać. Innymi dodatkowymi elementami, które przyniosłyby dobre wyniki byłoby użycie dodatkowych informacji dotyczących zachowania kierowcy. Oprócz wykrywania zamkniętego oka możnaby określać stopień otwarcia powiek, analizować częstość ziewania oraz mrugania. Kolejnym usprawnieniem byłoby wykorzystanie drugiej kamery w telefonie i analizowanie obrazu drogi znajdującej się przed maską samochodu, czyli informacji o np. szybkości zmiany kierunku jazdy lub utrzymywania pojazdu między pasami ruchu. W celu poprawy efektywności pracy systemu możnaby zastosować również inne sensory i czujniki znajdujące się w smartfonach. Przykładowo GPS pozwalałby dostosowywać czułość algorytmu wykrywania senności w zależności od prędkości i rodzaju drogi. Akcelerometr mógłby posłużyć do badania nagłych przyspieszeń, hamowań oraz zmian kierunku ruchu. Usprawnieniu i dokładniejszemu badaniu w warunkach zbliżonych do rzeczywistych mogłyby zostać poddane inne metody wykrywania zaśnięcia na podstawie już zebranych danych, np. inne funkcje senności.

Bibliografia

[1] Active Shape Models with Stasm. http://www.milbo.users.sonic.net/stasm/. Dostęp: 1.09.2015 r.

[2] Android Developer Guide. http://developer.android.com/guide/index.html. Dostęp: 22.08.2015 r.

[3] Android JNI Interface. http://developer.android.com/training/articles/perf-jni.html. Dostęp: 22.08.2015 r.

[4] Dokumentacja OpenCV C++. http://docs.opencv.org/modules/refman.html. Dostęp: 22.08.2015 r.

[5] Dokumentacja OpenCV Java. http://docs.opencv.org/java/. Dostęp: 22.08.2015 r.

[6] Drivers Beware: Getting Enough Sleep Can Save Your Life this Memorial Day. http://us1.campaign-archive.com/?u=72c7dac36ef8bcb0852893d7c&id=56d88442b5. Dostęp: 25.08.2015 r.

[7] Electric Potential Integrated Circuits. http://www.autoevolution.com/news/epic-car-seats-are-designed-to-save-sleepy-drivers-lives-1-83941.html. Dostęp: 31.08.2015 r,.

[8] Lexus Driver Monitoring System. https://en.wikipedia.org/wiki/Driver_Monitoring_System. Dostęp: 31.08.2015 r.

[9] Motion Detection. http://web.archive.org/web/20080522171806/http:/www.ansatt.hig.no/erikh/papers/hig98_6/node2.html. Dostęp: 26.08.2015 r.

[10] National Advanced Driving Simulator (NADS). http://www.engineering.uiowa.edu/news/ui-develops-simulated-driving-platform-teen-new-drivers. Dostęp: 31.08.2015 r.

[11] OpenCV4Android. http://opencv.org/platforms/android.html. Dostęp: 22.08.2015 r.

[12] Smartphone OS Market Share, Q1 2015. `http://www.idc.com/prodserv/smartphone-os-market-share.jsp`. Dostęp: 25.08.2015 r.

[13] Volvo XC60: Driver Alert Control. `https://www.youtube.com/watch?v=Gb9G9vZ0Tyg`. Dostęp: 31.08.2015 r.

[14] Meng-Che Chuang, R. Bala, E.A. Bernal, P. Paul, and A. Burry. Estimating gaze direction of vehicle drivers using a smartphone camera. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, pages 165–170, June 2014.

[15] Michał Cieśla and Przemysław Kozioł. Eye Pupil Location Using Webcam.

[16] Celine Craye, Abdullah Rashwan, Mohamed S. Kamel, and Fakhri Karray. A Multi-Modal Driver Fatigue and Distraction Assessment System. *International Journal of Intelligent Transportation Systems Research*, 2015.

[17] Taner Danisman, Ian Marius Bilasco, Chabane Djeraba, and Nacim Ihaddadene. Drowsy Driver Detection System Using Eye Blink Patterns. *Machine and Web Intelligence*, pages 230–233, 2010.

[18] Szymon Deja. *System rozpoznawania i aktywnego śledzenia oczy użytkownika komputera za pośrednictwem kamery w czasie rzeczywistym. Praca magisterska*. Akademia Górniczo-Hutnicza, 2010.

[19] Hai Dinh, Emil Jovanov, and Reza Adhami. Eye blink detection using Intensity Vertical Projection.

[20] M. ElSabrouty, A. Hamdy, A. Fawky, and S. Khalil. Drowsy Driver Assistant System. *IEEE International Symposium on Signal Processing and Information Technology*, pages 176–180, 2007.

[21] Marco Javier Flores, José María Armingol, and Arturo de la Escalera. Real-Time Warning System for Driver Drowsiness Detection Using Visual Information. *Journal of Intelligent and Robotic Systems*, pages 103–125, 2010.

[22] AAA Foundation for Traffic Safety, editor. *Prevalence of Motor Vehicle Crashes Involving Drowsy Drivers*, 2014.

[23] Rajat Garg, Vikrant Gupta, and Vineet Agrawal. A Drowsy Driver Detection and Security System.

[24] D.W. Hansen and Qiang Ji. In the Eye of the Beholder: A Survey of Models for Eyes and Gaze. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32:478–500, 2010.

[25] J A Home and L A Reyner. Sleep related vehicle accidents. *British Medical Journal*, 310:565–567, 1995.

[26] He J, Robertson S, Fields B, Peng J, Cielocha S, and Coltea J. Fatigue Detection using Smartphones. *Ergonomics*, 2013.

[27] Liling Li, Mei Xie, and Huazhi Dong. A Method of Driving Fatigue Detection Based on Eye Location. *Communication Software and Networks*, pages 480 – 484, 2011.

[28] James B. Maas. *Power Sleep : The Revolutionary Program That Prepares Your Mind for Peak Performance*.

[29] Teresa Makowiec-Dąbrowska, Alicja Bortkiewicz, Jadwiga Siedlecka, and Elżbieta Gadzicka. Wpływ zmęczenia na zdolność prowadzenia pojazdów. *Medycyna Pracy*, pages 281–290, 2011.

[30] B. Menser and F. Muller. Face Detection in Color Images Using Principal Component Analysis.

[31] S. Milborrow and F. Nicolls. Active Shape Models with SIFT Descriptors and MARS. *VISAPP*, 2014.

[32] Michał Panek and Jakub Sieradzki. *Śledzenie ruchu gałki ocznej w sterowaniu komputerem. Praca inżynierska*. Akademia Górniczo-Hutnicza, 2014.

[33] Komenda Główna Policji, editor. *Wypadki drogowe w Polsce w 2014 roku*, 2015.

[34] Marcin Pomarański. *System aktywnego śledzenia oczu kierowcy celem uniknięcia kolizji na wypadek zaśnięcia kierowcy samochodu. Praca magisterska*. Akademia Górniczo-Hutnicza, 2007.

[35] Arun Sahayadhas, Kenneth Sundaraj, and Murugappan Murugappan. Detecting Driver Drowsiness Based on Sensors: A Review. *Sensors*, pages 16937–16953, 2012.

[36] Adam Schmidt and Andrzej Kasiński. The Performance of the Haar Cascade Classifiers Applied to the Face and Eyes Detection. In Marek Kurzynski, Edward Puchala, Michal Wozniak, and Andrzej Zolnierek, editors, *Computer Recognition Systems 2*, volume 45 of *Advances in Soft Computing*, pages 816–823. Springer Berlin Heidelberg, 2007.

[37] Ryszard Tadeusiewicz and Przemysław Korohoda. *Komputerowa analiza i przetwarzanie obrazów*. Wydawnictwo Fundacji Postępu Telekomunikacji, 1997.

[38] Ines Teyeb, Olfa Jemai, Mourad Zaied, and Chokri Ben Amar. A Novel Approach for Drowsy Driver Detection Using Head Posture Estimation and Eyes Recognition System Based on Wavelet Network.

[39] Anna Łuczak and Krystyna Zużewicz. Zmęczenie kierowców a bezpieczeństwo pracy. *Bezpieczeństwo Pracy*, pages 20–23, 2006.

[40] Vladimir Vezhnevets, Vassili Sazonov, and Alla Andreeva. A Survey on Pixel-Based Skin Color Detection Techniques.

[41] Paul Viola and Michael J. Jones. Robust real-time face detection. *Int. J. Comput. Vision*, 57(2):137–154, May 2004.

[42] Cui Xu, Ying Zheng, and Zengfu Wang. Efficient Eye States Detection in Real-Time for Drowsy Driving Monitoring System. *IEEE International Conference on Information and Automation*, pages 171–174, 2008.

[43] Lunbo Xu, Shunyang Li, Kaigui Bian, Tong Zhao, and Wei Yan. Sober-Drive: A Smartphone-assisted Drowsy Driving Detection System. *International Conference on Computing, Networking and Communications, Mobile Computing & Vehicle Communications Symposium*, pages 398–402, 2014.

[44] Chuang-Wen You, Nicholas D. Lane, Fanglin Chen, Rui Wang, Zhenyu Chen, Thomas J. Bao, Martha Montes-de Oca, Yuting Cheng, Mu Lin, Lorenzo Torresani, and Andrew T. Campbell. CarSafe App: Alerting Drowsy and Distracted Drivers Using Dual Cameras on Smartphones. In *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '13, pages 13–26, 2013.

[45] Wei Zhang, Bo Cheng, and Yingzi Lin. Driver Drowsiness Recognition Based on Computer Vision Technology. *Tsinghua Science and Technology*, pages 354 – 362.

[46] Zutao ZHANG and Jiashu ZHANG. A new real-time eye tracking based on nonlinear unscented Kalman filter for monitoring driver fatigue. *Control Theory And Applications*, pages 181–188, 2010.

[47] Zhi-Hua Zhou and Xin Geng. Projection Functions for Eye Detection.

Dodatek A. Instrukcja instalacji oraz użytkowania systemu

A.1. Instalacja aplikacji

Przed właściwą instalacją systemu do poprawnego działania należy dodatkowo pobrać i zainstalować moduł OpenCV Manager ze sklepu Google Play. Następnym krokiem jest zezwolenie na telefonie możliwość instalacji aplikacji spoza. W tym celu w ustawieniach należy przejść do opcji **Zabezpieczenia** i zaznaczyć pole przy opcji **Nieznane źródła**. Kolejnym etapem jest skopiowanie pliku apk z aplikacją do pamięci urządzenia mobilnego i uruchomienie go. Po pomyślnym procesie instalacji program jest gotowy do działania

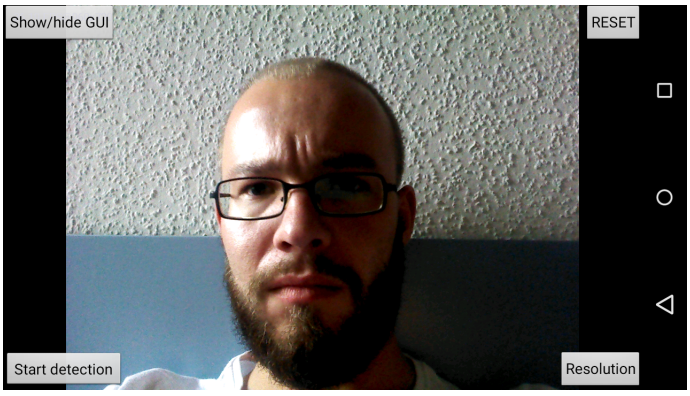
A.2. Użytkowanie systemu

Po uruchomieniu aplikacji użytkownik ma do wyboru dwa tryby działania:

– Developer mode - tryb developerski z możliwością podglądu różnych operacji przetwarzania obrazu, uruchomienia niepełnej wersji systemu i zmiany wartości różnych parametrów. Tryb ten nie zostanie w tym dodatku opisany.

– Normal mode - tryb normalnej pracy aplikacji. Służy do uruchomienia systemu detekcji senności i zaśnięcia, pozwala również na podgląd widoku z kamery.

Okno widoku normalnego

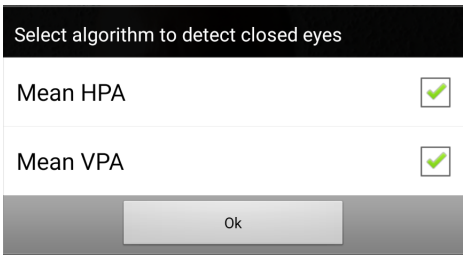


Rysunek A.1: Wygląd głównego ekranu po uruchomieniu aplikacji w trybie normalnym

Przyciski (rys. A.1):

- Start detection - uruchomienie algorytmu detekcji senności i zaśnięcia, skład się z kilku kroków,
- Resolution - wybór rozdzielczości ramki obrazu,
- RESET - zresetowanie działania aplikacji,
- Show/hide GUI - ukrycie przycisków RESET i Resolution.

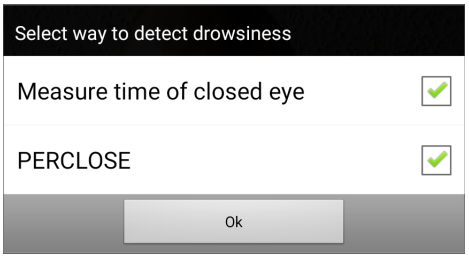
Kroki uruchomienia algorytmu wykrywania senności i zaśnięcia



Rysunek A.2: Okno wyboru algorytmu do wykrywania zamkniętego oka

Algorytmy detekcji zamkniętego oka do wyboru (domyślne zaznaczone obie metody) (rys. A.2):

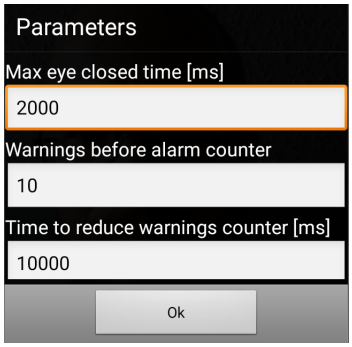
- Mean HPA - algorytm detekcji przy użyciu projekcji horyzontalnej średniej jasności pikseli,
- Mean VPA - algorytm detekcji przy użyciu projekcji wertykalnej średniej jasności pikseli.



Rysunek A.3: Okno wyboru algorytmu do wykrywania senności

Algorytmy detekcji senności do wyboru (domyślne zaznaczone obie metody) (rys. A.3):

- Measure time of closed eye - algorytm mierzący długość czasu zamknięcia oka,
- PERCLOS - algorytm sprawdzający stosunek liczby ramek obrazu z zamkniętymi okiem do całkowitej liczby ramek.

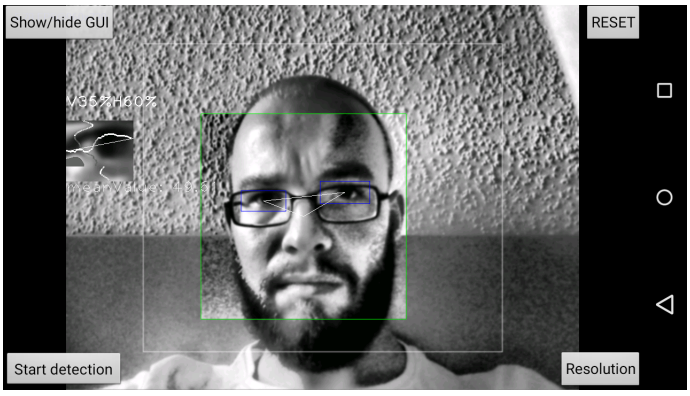


Rysunek A.4: Okno do określenia wartości parametrów algorytmów wykrywania senności

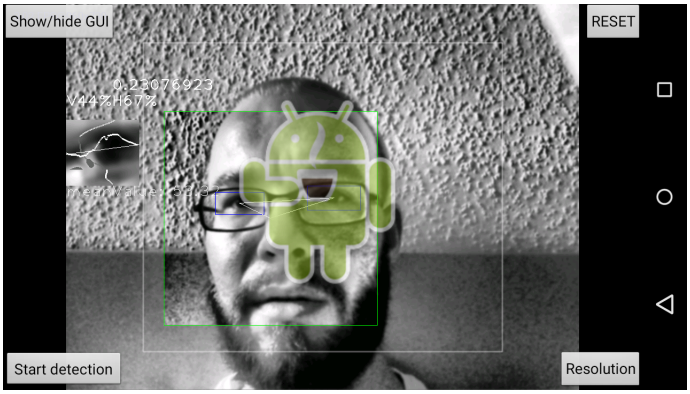
Parametry (rys. A.4):

- Max eye closed time - czas w milisekundach, po którym system ostrzega kierowcę o zamkniętych oczach,
- Warnings before alarm counter - liczba ostrzeżeń, po których włączany jest alarm,
- Time to reduce warnings counter - czas w milisekundach, po którym zmniejszana jest liczba ostrzeżeń przyczyniających się do uruchomienia alarmu,
- Time to start algorithm - czas w minutach po którym algorytm detekcji senności zaczyna działanie

Okna aplikacji w trakcie pracy systemu

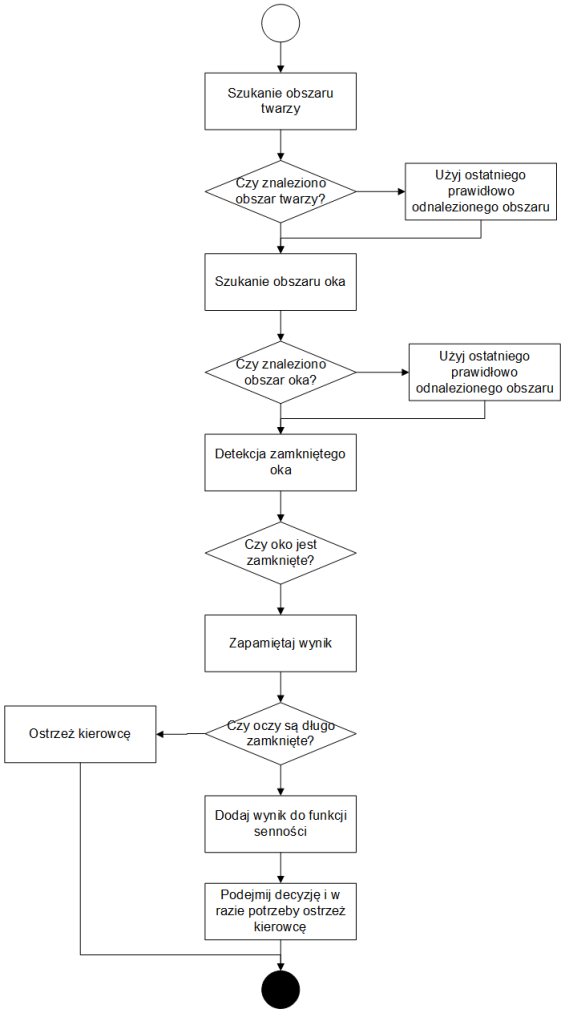


Rysunek A.5: Wygląd okna aplikacji w trakcie pracy algorytmu detekcji senności



Rysunek A.6: Wygląd okna aplikacji po wykryciu niskiej stopnia senności - ostrzeżenie graficzne

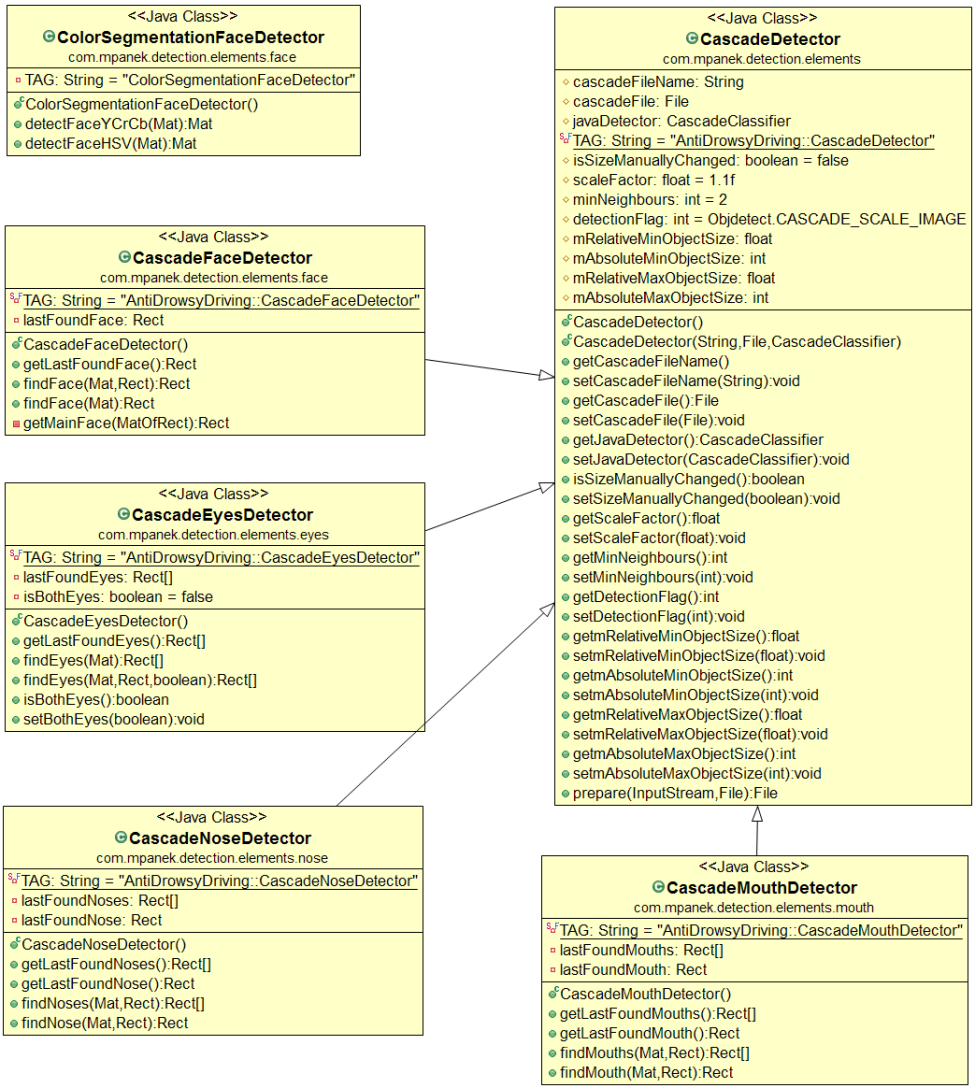
Dodatek B. Diagram czynności przedstawiający etapy działania algorytmu



Rysunek B.1: Diagram czynności przedstawiający kolejne etapy algorytmu do wykrywania zaśnięcia.

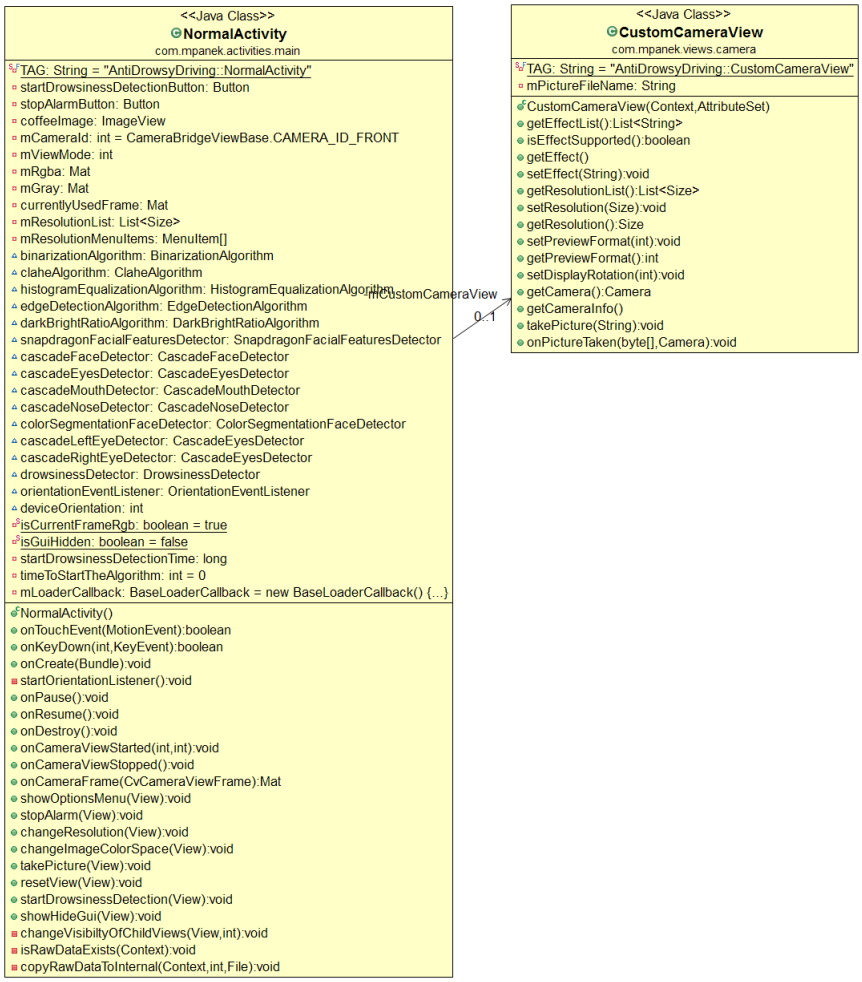
Dodatek C. Diagramy klas

Rysunek C.1: Diagram klas przedstawiający klasę odpowiedzialną za detekcję senności.

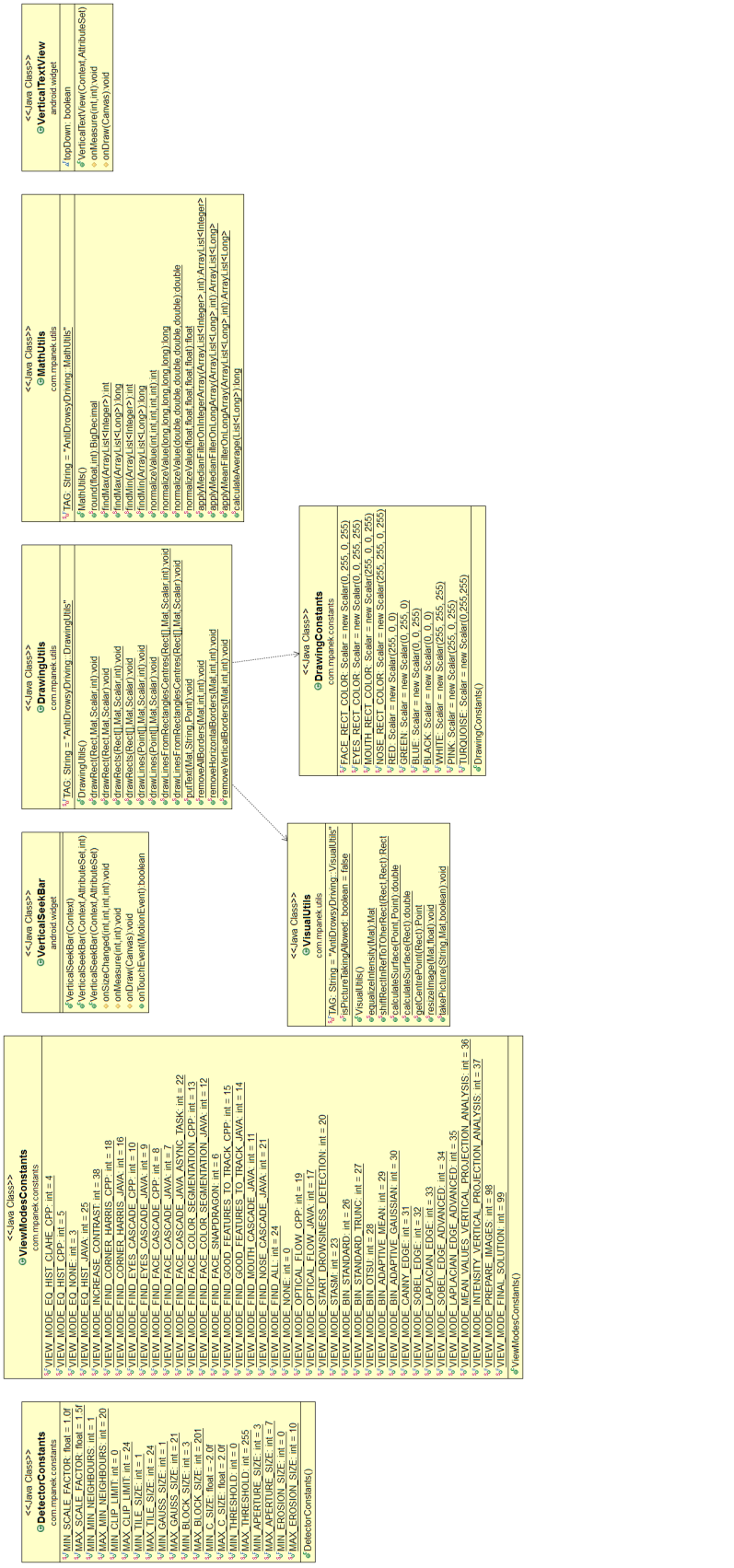


Rysunek C.2: Diagram klas przedstawiający klasy odpowiedzialną za detekcję elementów charakterystycznych.

Rysunek C.3: Diagram klas przedstawiający klasy odpowiedzialną za algorytmy użyte do przetwarzania ramek obrazu.



Rysunek C.4: Diagram klas przedstawiający klasę będąca klasą główną aplikacji na platformie Android (tzw. Activity).



Rysunek C.5: Diagram klas przedstawiający klasy pomocnicze.

Dodatek D. Spis rysunków

Spis rysunków