Politechnika Opolska

Wydział Elektrotechniki, Automatyki i Informatyki

Instytut Automatyki i Informatyki



PRACA DYPLOMOWA INŻYNIERSKA

Monitoring parkingu z rozpoznawaniem numerów   
rejestracyjnych - implementacja aktywnej wizji

Adam Herman

nr albumu: 84039

kierunek: Informatyka

studia stacjonarne I stopnia

promotor: dr inż. Michał Podpora

Opole, 2016

**MONITORING PARKINGU Z ROZPOZNAWANIEM   
NUMERÓW REJESTRACYJNYCH –   
IMPLEMENTACJA AKTYWNEJ WIZJI**

**S t r e s z c z e n i e**

Celem niniejszej pracy jest opracowanie rozwiązania systemu monitorującego parking z rozpoznawaniem numerów rejestracyjnych, a następnie jego implementacja. Praca składa się z dwóch części: teoretycznej i praktycznej. Część teoretyczna opisuje podstawowe pojęcia związane z tematem, porusza wymagania prawne dotyczące tablic rejestracyjnych oraz prezentuje kilka gotowych rozwiązań. Część praktyczna dotyczy opisu sprzętu użytego podczas realizacji systemu oraz opisuje proces jego powstawania, począwszy od wymagań, następnie tworzenie oprogramowania, a na opisie interfejsu graficznego kończąc. Pracę wykonano pomyślnie, zgodnie z założonymi celami.

**MONITORING OF A PARKING LOT   
WITH RECOGNITION OF CAR PLATES –   
AN IMPLEMENTATION OF ACTIVE VISION**

**S u m m a r y**

The objective of this study is to develop a parking lot monitoring system with recognition of car plates, and then its implementation. The study consists of two parts: theoretical and practical. The theoretical part describes the basic concepts related to the topic, raises the legislative requirements for car plates and presents a few existing solutions. The practical part relates to the description of the hardware used during the realisation of the system and describes the process of its creation, starting with requirements, through software development, and on the description of the GUI (Graphical User Interface) ending. The study is made successfully in accordance with the objectives.

# Spis treści

[Spis treści 3](#_Toc440397591)

[1. Wstęp 5](#_Toc440397592)

[2. Cel i zakres pracy 6](#_Toc440397593)

[3. Systemy rozpoznawania tablic rejestracyjnych 7](#_Toc440397594)

[3.1. Wprowadzenie 7](#_Toc440397595)

[3.2. Wymagania prawne polskich tablic rejestracyjnych 8](#_Toc440397596)

[3.2.1. Ogólna specyfikacja tablic rejestracyjnych 8](#_Toc440397597)

[3.2.2. Numery rejestracyjne 9](#_Toc440397598)

[3.2.3. Lokalizacja tablic rejestracyjnych 9](#_Toc440397599)

[3.3. Zasada działania systemu 10](#_Toc440397600)

[3.3.1. Rozpoznawanie statyczne 11](#_Toc440397601)

[3.3.2. Rozpoznawanie dynamiczne 11](#_Toc440397602)

[3.4. Analiza praktycznych rozwiązań w danej dziedzinie 11](#_Toc440397603)

[3.4.1. CARMEN Parking ANPR 11](#_Toc440397604)

[3.4.2. Siemens Sicore ANPR 12](#_Toc440397605)

[3.4.3. Metasoft ARTR 12](#_Toc440397606)

[3.4.4. PolCam Mobile ANPR 13](#_Toc440397607)

[4. Część praktyczna 14](#_Toc440397608)

[4.1. Przedstawienie systemu 14](#_Toc440397609)

[4.2. Zastosowany sprzęt 16](#_Toc440397610)

[4.2.1. Raspberry Pi B 16](#_Toc440397611)

[4.2.2. Raspberry Pi Camera HD 17](#_Toc440397612)

[4.2.3. Karta Wi-Fi Edimax EW-7811Un USB 17](#_Toc440397613)

[4.2.4. Ultradźwiękowy czujnik odległości HC-SR04 18](#_Toc440397614)

[4.2.5. Silnik krokowy 28BYJ-48 ze sterownikiem ULN2003 18](#_Toc440397615)

[4.2.6. Komputer PC / Laptop 19](#_Toc440397616)

[4.3. Środowisko programistyczne, język i wykorzystane biblioteki 19](#_Toc440397617)

[5. Realizacja systemu monitorującego parking 21](#_Toc440397618)

[5.1. Specyfikacja funkcjonalna 21](#_Toc440397619)

[5.2. Specyfikacja niefunkcjonalna 22](#_Toc440397620)

[5.3. Działanie systemu 23](#_Toc440397621)

[5.3.1. Wykrycie pojazdu i przesłanie obrazu 23](#_Toc440397622)

[5.3.2. Lokalizacja tablicy rejestracyjnej 24](#_Toc440397623)

[5.3.3. Segmentacja znaków 34](#_Toc440397624)

[5.3.4. Optyczne rozpoznawanie znaków 35](#_Toc440397625)

[5.3.5. Operacje na bazie danych 36](#_Toc440397626)

[5.3.6. Sterowanie szlabanem 36](#_Toc440397627)

[5.4. Opis interfejsu użytkownika 37](#_Toc440397628)

[5.5. Podsumowanie działania systemu 40](#_Toc440397629)

[6. Wnioski i uwagi 41](#_Toc440397630)

[Literatura 43](#_Toc440397631)

[Dodatek A. Wybrane fragmenty programu komputerowego 46](#_Toc440397632)

[Dodatek B. Spis zawartości dołączonej płyty CD 54](#_Toc440397633)

# Wstęp

Według raportu Polskiego Związku Przemysłu Motoryzacyjnego, w roku 2013 zarejestrowanych było w Polsce ponad 25 milionów pojazdów samochodowych, natomiast liczba kradzieży samochodów w roku 2014 wyniosła 14124 samochody, z czego, zdaniem policji, złodzieje najczęściej działali na niestrzeżonych oraz osiedlowych parkingach [8].

Automatyzacja ochrony, czy też dostępu do parkingów może zapobiec takiej liczbie kradzieży. Istnieje wiele zabezpieczeń elektronicznych stosowanych na parkingach, są to między innymi:

* systemy kontroli dostępu - składające się z czytników RFID lub umożliwiające dostęp po wprowadzeniu poprawnego kodu PIN na klawiaturze,
* systemy biletowe - służące najczęściej do pobierania opłat za postój, jednakże wymagają wydruku specjalnej przepustki, bez której nie jest możliwe opuszczenie parkingu,
* systemy CCTV - pozwalające na monitoring wizyjny całej przestrzeni parkingu z jednego miejsca za pomocą kamer,
* systemy rozpoznawania tablic rejestracyjnych - korzystające z analizy obrazu przesyłanego z kamery, który pozwala na jednoznaczną identyfikację pojazdu znajdującego się na parkingu na podstawie numeru rejestracyjnego [3].

To właśnie ostatni z powyższych sposobów jest rozwiązaniem najbardziej przyjaznym klientom parkingów, gdyż nie wymagają żadnej interakcji z ich strony. Systemy rozpoznawania tablic rejestracyjnych nie wymagają wprowadzania dodatkowych identyfikatorów do pojazdów, gdyż każdy pojazd, zgodnie z obowiązującym prawem, musi być wyposażony w tablicę rejestracyjną, a ta stanowi jego znak identyfikacyjny. Dzięki zastosowaniu technologii z zakresu przetwarzania obrazu ciągle postępującemu wzrostowi wydajności komputerów oraz zmniejszeniu rozmiaru kamer oraz ich cen, systemy te okazują się dobrą alternatywą, szczególnie w przypadku, gdy jednym z ważniejszych wymagań jest bezobsługowe działanie.

# Cel i zakres pracy

Celem niniejszej pracy jest przegląd literatury, opracowanie oraz implementacja systemu komputerowego umożliwiającego autonomiczny monitoring parkingu z rozpoznawaniem tablic rejestracyjnych.

Niniejsza praca składa się z pięciu rozdziałów.

Rozdział pierwszy stanowi wstęp do tematu monitoringu parkingu.

Rozdział drugi zawiera sformułowany cel pracy dyplomowej oraz jej zakres.

Rozdział trzeci stanowi analizę obecnych rozwiązań w tej dziedzinie oraz wprowadzenie do tematu automatycznego rozpoznawania numerów rejestracyjnych.

Rozdział czwarty poświęcony jest opisowi części praktycznej pracy dyplomowej. W jego skład wchodzi opis użytego sprzętu.

Piąty rozdział opisuje proces tworzenia oprogramowania. Przedstawia kolejne etapy analizy obrazu przez system, zawiera diagramy przedstawiające działanie systemu, opis graficznego interfejsu użytkownika oraz podsumowuje działanie systemu.

Ostatni, szósty rozdział, zawiera podsumowanie pracy.

# Systemy rozpoznawania tablic rejestracyjnych

## Wprowadzenie

Temat automatycznego rozpoznawania tablic rejestracyjnych przez systemy komputerowe znany jest od lat 70. ubiegłego wieku. Pierwszy system ANPR (ang. Automatic Number Plates Recognition) rozpoczęto opracowywać w roku 1976 na potrzeby Oddziału Rozwoju Naukowego Policji w Wielkiej Brytanii. Kilka lat później powstały pierwsze przemysłowe systemy [15]. System ANPR jest sprzętowo-programowym rozwiązaniem, pozwalającym na automatyczne rozpoznawanie tablic rejestracyjnych. Można go obecnie spotkać pod wieloma nazwami:

* Automatic License Plate Recognition (ALPR),
* Automatic Vehicle Identification (AVI),
* Car Plate Recognition (CPR),
* License Plate Recognition (LPR),
* Computerized License Plate Recognition (CLPR).

Również w języku polskim system ten posiada swoje nazewnictwo:

* Automatyczne Rozpoznawanie Tablic Rejestracyjnych (ARTR),
* Automatyczne Rozpoznawanie Numerów Rejestracyjnych (ARNR). [9]

Aktywna wizja jest metodą przetwarzania danych wizualnych. Systemy aktywnej wizji potrafią zarządzać punktem widzenia kamery w celu zbadania środowiska i uzyskania lepszej informacji od niego. Algorytm aktywnej wizji wybiera istotny do dalszej analizy fragment obrazu, podobnie jak narząd wzroku w organizmach biologicznych [16]. Zwiększenie percepcji w wybranej części pola widzenia pozwala na jego szczegółową analizę. Nawiązując do tematu rozpoznawania tablic rejestracyjnych, aktywna wizja umożliwia skrócenie czasu przetwarzania obrazu poprzez analizę jedynie fragmentu zawierającego tablicę rejestracyjną.

## Wymagania prawne polskich tablic rejestracyjnych

### Ogólna specyfikacja tablic rejestracyjnych

Obecnie użytkowane tablice rejestracyjne rozpoczęto wydawać od 1 maja 2000 r.. Poprzednie tablice, z czarnym tłem i białymi literami, okazały się słabo czytelne w nocy, dlatego aktualnie nie stosuje się tej kolorystyki, zastępując ją w większości przypadków następującą: białe tło oraz czarne litery. W maju 2006 roku wprowadzono unijny wzór tablic rejestracyjnych, gdzie 12 gwiazdek Unii Europejskiej zastąpiło polską flagę [11].



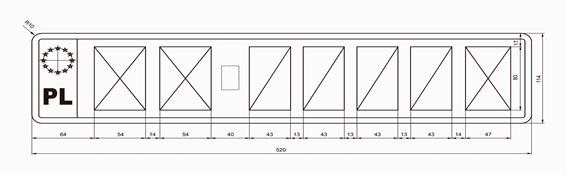
Rys. .. Wzór tablic rejestracyjnych używanych w Polsce, u góry obowiązujący od 1 maja 2006, u dołu obowiązujący od 1 maja 2000 do 30 kwietnia 2006. Źródło: [29]

W Polsce wyróżnia się 6 rodzajów tablic rejestracyjnych:

* Zwyczajne
* Tymczasowe
* Indywidualne
* Zabytkowe
* Dyplomatyczne
* Służb Dyplomatycznych

Ze względu na wielkość wyróżnia się tablice [14]:

* samochodowe (jednorzędowe i dwurzędowe) — do oznaczania wszystkich rodzajów pojazdów z wyjątkiem motocykli, ciągników rolniczych i motorowerów,
* motocyklowe (dwurzędowe) — do oznaczania motocykli i ciągników rolniczych,
* motorowerowe (dwurzędowe) — do oznaczania motorowerów.



Rys. .. Wymiary tablicy rejestracyjnej samochodowej, jednorzędowej. Źródło: [30]

Wymiary tablic samochodowych jednorzędowych wynoszą 520×114 mm. Z lewej strony na niebieskim pasku znajduje się międzynarodowe oznaczenie Polski (PL) oraz w zależności od daty wydania: flaga Polski lub dwanaście żółtych gwiazdek Unii Europejskiej. Dokładne wymiary przedstawiono na rysunku 3.2.

### Numery rejestracyjne

Na tablicy umieszczony jest numer rejestracyjny pojazdu. Numer ten składa się z dwóch części: wyróżnika województwa i wyróżnika powiatu, stanowiących razem całość oraz wyróżnika pojazdu.

Zbiór znaków stosowanych na tablicach rejestracyjnych określa ustawa o rejestracji pojazdów: „Numery rejestracyjne są tworzone ze zbioru następujących 25 liter: A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, R, S, T, U, V, W, X, Y i Z oraz cyfr od 0 do 9.” [10]. Znaki zapisywane są krojem pisma podobnym do DIN 1451.

Pierwsza litera jest wyróżnikiem województwa, a druga lub trzecia wyróżnikiem powiatu. W przypadku powiatu grodzkiego stosuje się jedną literę, zaś powiaty ziemskie oznacza się dwiema literami. Zdarzają się jednak odstępstwa od tej reguły. Następnie po wyróżniku powiatu znajduje się wyróżnik pojazdu, składający się z kombinacji liter i cyfr. W części będącej wyróżnikiem pojazdu nie można stosować liter: B, D, I, O, Z gdyż są zbyt podobne do cyfr: 8, 0 ,1, 2. Ograniczenie to znajduje również uzasadnienie w systemach ANPR, gdzie bardzo ważna jest precyzja rozpoznawanych znaków.

### Lokalizacja tablic rejestracyjnych

Lokalizację tablic rejestracyjnych, a dokładniej miejsce i sposób mocowania tablic w pojazdach, określa Rozporządzenie Ministra Infrastruktury z dnia 31 grudnia 2002 r. [10]. Zgodnie z punktem 1 paragrafu 2 w/w rozporządzenia, lokalizacja tablic rejestracyjnych w pojeździe jest następująca: „Właściciel pojazdu umieszcza na pojeździe tablice z przodu i z tyłu w miejscach konstrukcyjnie do tego przeznaczonych, z wyjątkiem przyczep, ciągników rolniczych, motocykli i motorowerów, na których tablice umieszcza się tylko z tyłu, z zastrzeżeniem ust. 2.” Bazując na powyższym fragmencie rozporządzenia można stwierdzić, iż możliwe jest tworzenie systemów ANPR odczytujących tablice rejestracyjne zarówno z przodu jak i z tyłu pojazdu. Niestety niektóre systemy rozpoznawania tablic nie będą w stanie rozpoznawać pewnych pojazdów, ponieważ nie wszystkie pojazdy posiadają tablice przednie.

W punkcie 3 wspomnianego wcześniej paragrafu napisano: „Z przodu pojazdu umieszcza się wyłącznie tablicę jednorzędową, z wyjątkiem ciągnika rolniczego, o którym mowa w ust. 2.” Przepis ten powoduje pewne ułatwienia w zakresie tworzonego systemu rozpoznawania tablic rejestracyjnych. Rozpoznawanie tablic jednorzędowych jest procesem mniej skomplikowanym od rozpoznawania tablic dwurzędowych. Tworząc monitoring parkingu nie przewiduje się umieszczania ciągników rolniczych na jego terenie, dlatego fakt posiadania odmiennych tablic przez te pojazdy nie wpłynie na działanie systemu.

Z tyłu pojazdów mogą być umieszczanie zarówno tablice jednorzędowe jak i dwurzędowe. Jako że przewiduje się jedynie monitoring obejmujący pojazdy wjeżdżające na parking, rodzaj tablic tylnych nie ma znaczenia w kontekście tworzonego systemu.

W prawie nie znajdują się zapisy określające dokładnie na jakiej wysokości powinna znajdować się tablica rejestracyjna. Musi być jedynie umieszczona w miejscu konstrukcyjnie do tego przeznaczonym.

## Zasada działania systemu

System ANPR składa się z siedmiu podstawowych algorytmów wymaganych przez oprogramowanie w celu rozpoznania tablicy rejestracyjnej:

1. lokalizacja tablicy rejestracyjnej,
2. dostosowanie orientacji i rozmiaru tablicy,
3. normalizacja,
4. segmentacja znaków,
5. optyczne rozpoznanie znaków,
6. analiza składniowa i geometryczna,
7. uśrednienie rozpoznanej wartości. [5]

### Rozpoznawanie statyczne

Statyczne rozpoznawanie numerów tablic rejestracyjnych wymaga zastosowania nieruchomej kamery oraz pojazdu znajdującego się w bezruchu. Najczęściej rozpoznawanie statyczne stosuje się razem ze szlabanami zezwalającymi na wjazd np. na bramkach wjazdowych lub przejściach granicznych. Rozwiązanie to zakłada, że w danym punkcie pomiaru nie znajduje się więcej niż jeden pojazd. Systemy wykorzystujące rozpoznawanie statyczne wykorzystuje się przy wjazdach na parkingi lub w punktach poboru opłat [9].

### Rozpoznawanie dynamiczne

Rozpoznawanie dynamiczne zakłada możliwość rozpoznawania wielu tablic rejestracyjnych w tym samym czasie. Możliwe jest także rozpoznawanie podczas przemieszczania się kamery, dzięki czemu możliwe jest jej zamontowanie m.in. w samochodach. Zastosowanie tej metody jest o wiele szersze w porównaniu do rozpoznawanie statycznego, od patrolowych służb drogowych, po punkty pomiarowe na autostradach [9].

## Analiza praktycznych rozwiązań w danej dziedzinie

Istnieje wiele komercyjnych rozwiązań w dziedzinie automatycznego rozpoznawania tablic rejestracyjnych. W większości są to systemy bardzo rozbudowane z dużą dokładnością rozpoznawania oraz stosujący rozpoznawanie dynamiczne. Poniżej opisano kilka z nich.

### CARMEN Parking ANPR

Carmen Parking ANPR to rozwiązanie węgierskiej firmy Arh Inc. System został opracowany w związku z rosnącym zapotrzebowaniem na systemy zarządzania parkingami oraz kontroli dostępu. Możliwości jakie oferuje Carmen Parking ANPR są m.in. następujące:

* elastycznie i automatycznie zarządza wjazdami i wyjazdami samochodów,
* dostarcza informacje odnośnie użycia parkingu,
* poprawia bezpieczeństwo zarówno operatorów jak i użytkowników parkingu,
* zwiększa płynność ruchu w godzinach szczytu.

System potrzebuje 50ms na rozpoznanie tablicy rejestracyjnej, jeśli operacja jest wykonywana na komputerze z procesorem o częstotliwości taktowania 2GHz, a obraz ma rozdzielczość 768 x 288 pikseli. Po rozpoznaniu wymagane są 3 sekundy odstępu przed kolejną operacją. [1] Opcjonalną funkcjonalnością jest rozpoznanie identyfikatora kraju znajdującego się na tablicy. System działa również nocą dzięki wykorzystaniu podczerwieni.

### Siemens Sicore ANPR

Siemens Sicore ANPR jest systemem rozpoznawania dynamicznego potrafiącym rozpoznać tablice rejestracyjną pojazdu poruszającego się z prędkością do 250km/h. Sicore monitoruje do dwóch pasów ruchu w obu kierunkach jednocześnie, odczytując zarówno przednie jak i tylne tablice rejestracyjne. Parametry tego systemu są następujące [13]:

* zakres działania: 5 ­– 35m,
* wykrywania pojazdów poruszających się z prędkością do 250km/h,
* rozpoznawanie w nocy dzięki podczerwieni,
* baza mieszcząca dane do 1,000,000 pojazdów,
* komunikacja przewodowa i bezprzewodowa,
* szyfrowanie transmisji SSL,
* rozdzielczość kamery 1280x960 pikseli,
* odczytywanie tablic z odwróconą kolorystyką.

### Metasoft ARTR

Oprogramowanie polskiego producenta, analizujące tablice rejestracyjne pojazdów i zapisujące je w bazie danych. System przeznaczony do kontroli wjazdów przy bramkach wjazdowych. Umożliwia zdalne sterowanie szlabanem na podstawie listy zaufanych numerów tablic. Specyfikacja systemu [4]:

* rozpoznawanie polskich (białych i czarnych) tablic jedno i dwurzędowych,
* automatyczne rozpoznawanie województwa, powiatu dla polskich tablic,
* rozpoznawanie statyczne i dynamiczne,
* czas rozpoznania 1 tablicy poniżej 100ms,
* skuteczność rozpoznawania >96% (rozpoznawanie statyczne),
* zakres kąta pracy kamery do 45°.

### PolCam Mobile ANPR

PolCam Mobile ANPR to pierwszy całkowicie polski system automatycznego rozpoznawania tablic rejestracyjnych. Skuteczność rozpoznawania pojazdów znajdujących się w ruchu wynosi aż do 99%. System ten łatwo integruje się z serią wideorejestratorów marki PolCam, testowany był przez jednostkę policji w Radomiu. Proces rozpoznawania tablic realizowany jest równocześnie z pomiarem prędkości.



Rys. .. System PolCam Mobile ANPR w użyciu. Źródło: [21]

Połączenie z centralną bazą danych umożliwia kontrolę kierowców i samochodów bez ich zatrzymywania, dzięki czemu funkcjonariusze mają natychmiastowy dostęp do takich danych jak termin upływu badań technicznych, ubezpieczenia OC lub informacji o skradzionym pojeździe. [7]

# Część praktyczna

W tym rozdziale opisane są poszczególne etapy wykonania systemu monitorującego parking, począwszy od doboru sprzętu, aż po utworzenie kodu programu w wybranym języku i środowisku programowania. Projekt jest na bieżąco implementowany, w wyniku czego powstaje działający prototyp.

## Przedstawienie systemu

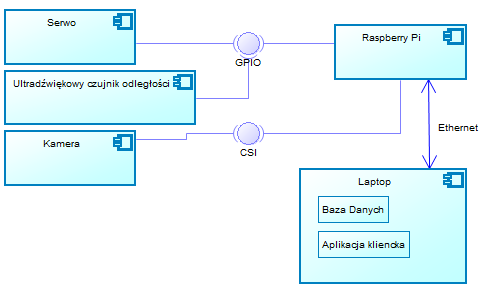
Monitoring parkingu ma na celu obserwację wjeżdżających na jego teren pojazdów. Jednoznaczna identyfikacja właściciela pojazdu możliwa jest na podstawie porównania numeru tablicy rejestracyjnej i danych zapisanych w bazie komputerowej. W celu rozpoznania numeru zapisanego na tablicy rejestracyjnej wykorzystuje się aktywną wizję. Stworzony został więc system, który w oparciu o dane z kamery, przetwarza uchwycony obraz, analizuje go, a następnie wykonuje niezbędne operacje mające na celu autoryzację dostępu do parkingu. System ten wykorzystuje rozpoznawanie statyczne, zakłada on, że dany pojazd podjeżdża pod bramkę wjazdową i zatrzymuje się, po poprawnej identyfikacji pojazdu, szlaban wjazdowy podnosi się, a użytkownik pojazdu może wjechać na parking.



Rys. .. Rysunek poglądowy. Źródło: [24]

Umieszczenie kamery przy bramce wjazdowej niesie ze sobą kilka ułatwień. Pojazd podjeżdżający pod bramkę musi się zatrzymać w pewnej odległości przed nią. W takim przypadku znacznie łatwiej jest wykryć znajdujący się obiekt przed kamerą. Dodatkowo analizowany obraz znajduje się w stałej odległości od obiektywu, przez co rozmiar tablicy rejestracyjnej na zdjęciach jest stały, a dzięki temu łatwiej dostosować algorytm lokalizacji. Możliwość umieszczenia kamery na szlabanie, na wprost nadjeżdżającego pojazdu eliminuje problem oślepienia światłem pochodzącym z reflektorów.

Zastosowana kamera wykonuje zdjęcia w rozdzielczości 640x480 pikseli, działanie nocą wymaga jednak odpowiedniego oświetlenia pochodzącego z zewnątrz. Algorytm rozpoznawania numerów tablicy rejestracyjnej wymaga znacznej mocy obliczeniowej, głównie ze względu na wymagania silnika OCR (optycznego rozpoznawania znaków). Zdecydowano więc o podziale systemu na dwie mniejsze części – jedną składającą się z ultradźwiękowego czujnika odległości, kamery, komputera Raspberry Pi oraz modułu sterującego bramką wjazdową oraz drugą część, która otrzymuje obraz do analizy, przechowuje dane i wysyła odpowiedź o autoryzacji dostępu bądź jej braku (w jej skład wchodzi także komputer PC, a w przypadku prototypu laptop Lenovo Edge E320).



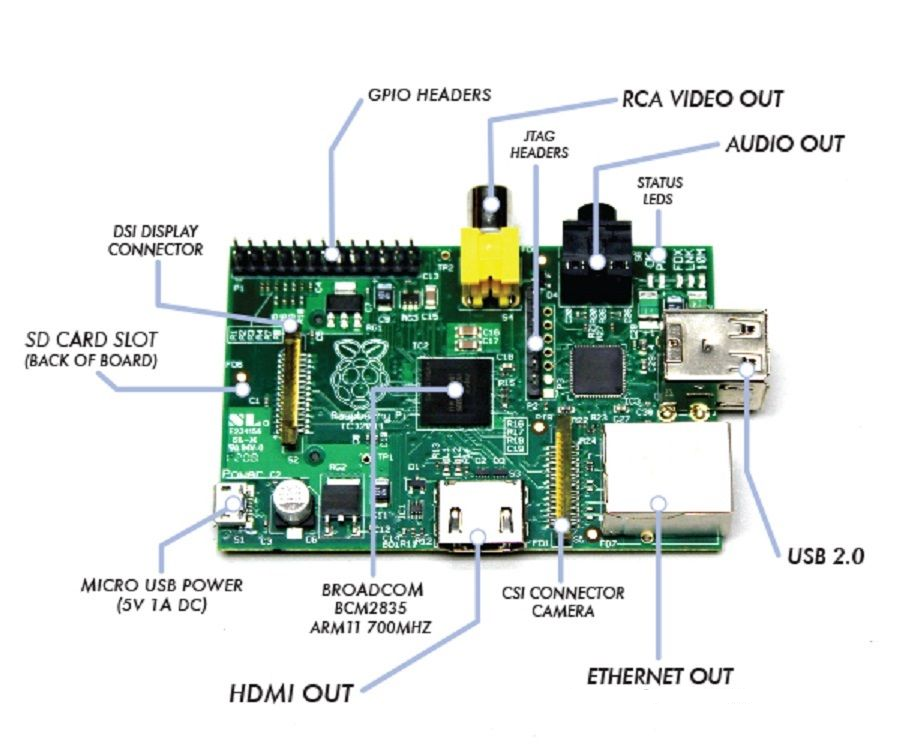
Rys. .. Diagram komponentów realizowanego systemu. Źródło: opracowanie własne

O ile pierwsza część systemu jest niezmienna, o tyle druga część może być uruchomiona na dowolnej platformie obsługującej system Windows 10. Komunikacja między obiema częściami odbywa się za pomocą sieci Ethernet.

## Zastosowany sprzęt

### Raspberry Pi B

Raspberry Pi to platforma komputerowa składająca się z pojedynczego obwodu drukowanego, która doskonale nadaje się do prototypowania prostych systemów wbudowanych. „Urządzenie oparte jest na układzie Broadcom BCM2835 SoC, który składa się z procesora ARM1176JZF-S 700 MHz, VideoCore IV GPU i 512 MB pamięci RAM. Urządzenie nie ma dysku twardego, ale w celu załadowania systemu operacyjnego i przechowywania danych oferuje złącze dla kart SD. Raspberry Pi ma również złącze USB do podłączenia dowolnych urządzeń zewnętrznych.” [17].

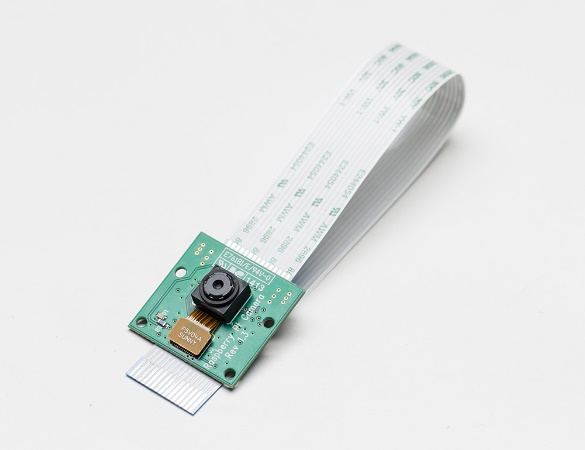


Rys. .. Raspberry Pi Model B. Źródło: [22]

Dużą zaletą Raspberry Pi jest interfejs GPIO znajdujący się na płytce, służy on do komunikacji pomiędzy elementami systemu komputerowego, takimi jak mikroprocesor czy różne urządzenia peryferyjne. Urządzenie pracuje pod kontrolą systemu Raspbian.

### Raspberry Pi Camera HD

Raspberry Pi Camera HD jest modułem kamery, podłączanym do minikomputera Raspberry Pi za pomocą złącza CSI. Rozdzielczość kamery wynosi 5 Mpx, kamera wspiera tryby HD 1080p, 720p oraz 640x480.



Rys. .. Raspberry Pi Camera HD. Źródło: [28]

Raspberry Pi posiada sprzętowe wsparcie do obsługi tej kamery, dzięki czemu nie zużywa mocy obliczeniowej procesora.

### Karta Wi-Fi Edimax EW-7811Un USB

Nano-karta USB firmy Edimax umożliwia łączenie się z sieciami bezprzewodowymi. Zgodna ze standardami: 802.11b/g/n, o przepustowości do 150 Mbps.



Rys. .. Bezprzewodowa karta Wi-Fi Edimax EW-7811Un. Źródło: [22]

Karta ta współpracuje z Raspberry Pi bez konieczności instalacji sterowników do jej obsługi.

### Ultradźwiękowy czujnik odległości HC-SR04

Czujnik ultradźwiękowy działający w zakresie od 2 do 200 cm. Zasilany napięciem 5 V. Czujnik ten dokonuje pomiaru odległości dźwiękiem o częstotliwości 40 kHz. Wyjściem jest sygnał, którego czas trwania jest proporcjonalny do zmierzonej odległości.



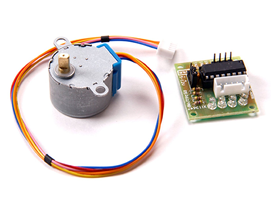
Rys. .. Ultradźwiękowy czujnik odległości HC-SR04. Źródło: [25]

Odległość w centymetrach można obliczyć ze wzoru:

|  |  |
| --- | --- |
|  | (4.1) |

### Silnik krokowy 28BYJ-48 ze sterownikiem ULN2003

Zastosowany silnik krokowy ma na celu symulację pracy prawdziwego szlabanu znajdującego się przy bramce wjazdowej.



Rys. .. Silnik krokowy 28BYJ-48 ze sterownikiem ULN2003. Źródło: [27]

Silnik 28BYJ-48 zasilany jest napięciem 5 V. Posiada zintegrowaną przekładnię 64:1, jej moment trzymający wynosi 0,03 Nm. Użycie tego silnika pozwala na zobrazowanie działania szlabanu, po otrzymaniu sygnału silnik obraca się o zadaną ilość kroków, odczekuje zdeklarowany czas, a następnie wraca do pozycji początkowej.

### Komputer PC / Laptop

W prototypie systemu wykorzystany jest notebook firmy Lenovo wyposażony w procesor Intel Core i3 2310M pracujący z częstotliwością 2,1 GHz. Laptop działa pod kontrolą systemu operacyjnego Windows 10.



Rys. .. Laptop Lenovo Edge E320. Źródło: [26]

Podzespołem wpływającym w największym stopniu na wydajność działania aplikacji jest procesor. Komputer z procesorem klasy Intel Core i3 zapewni sprawną pracę systemu, w przeciwieństwie do minikomputera Raspberry Pi, gdzie moc obliczeniowa nie jest zbyt duża, co wydłuża czas rozpoznania numeru rejestracyjnego znajdującego się na tablicy.

## Środowisko programistyczne, język i wykorzystane biblioteki

Główna część systemu, odpowiadająca za analizę obrazu i rozpoznawanie numerów tablic rejestracyjnych napisana jest w języku C++, przy wykorzystaniu biblioteki OpenCV 3.0.0 oraz silnika OCR Tesseract 3.0.2. Zdecydowano się na wybór tego języka z uwagi na łatwą integrację tego języka z biblioteką OpenCV, która została napisana w języku C. Dane przechowywane są w lokalnej bazie danych opartej o Microsoft SQL Server 2008.

Wspomniana biblioteka OpenCV jest zestawem funkcji wykorzystywanych podczas obróbki obrazu. Oparta jest na otwartym kodzie i pozwala na przetwarzanie obrazu w czasie rzeczywistym. Biblioteka ta czyni implementację aktywnej wizji łatwiejszą, zawiera ponad 2500 zoptymalizowanych algorytmów [2].

Zastosowany w implementacji silnik do optycznego rozpoznawania znaków (OCR) – Tesseract – jest uważany za jeden z najbardziej dokładnych silników o otwartym kodzie źródłowym znajdujących się obecnie na rynku [18]. Wspiera ponad kilkanaście języków, w tym język polski. Wymaga wstępnego przetworzenia obrazu przed poddaniem procesowi rozpoznania znaków, dlatego też użycie w projektowanym systemie OpenCV i silnika Tesseract jest idealnym rozwiązaniem opartym na wolnym, darmowym oprogramowaniu.

Wymienione wyżej biblioteki i język wykorzystano przy użyciu środowiska programistycznego Microsoft Visual Studio 2013, pracującego pod systemem Windows 10. Środowisko Visual Studio 2013 to ogromne narzędzie służące do tworzenia oprogramowania na różne platformy. Dzięki wielkiej popularności posiada szerokie wsparcie. Funkcje takie jak automatyczne uzupełnianie nawiasów, podświetlanie błędnych fragmentów kodu czy też możliwość wykonywania wielu akcji za pomocą skrótów klawiaturowych czynią pisanie aplikacji w tym środowisku przyjemnym i wygodnym. Środowisko to posiada również edytor graficznego interfejsu użytkownika oraz pozwala na integrację z bazą danych Microsoft SQL.

Pozostała część tworzonego oprogramowania, która znajduje się na platformie komputerowej Raspberry Pi działającej pod kontrolą systemu Raspbian, napisana jest w języku Python. W tym przypadku nie było wymagane środowisko programistyczne, posłużono się edytorem tekstu, a napisany kod zapisano w postaci skryptu. Odpowiada on za odbieranie sygnału z ultradźwiękowego czujnika odległości, wykonywanie zdjęć za pomocą modułu kamery, wysyłanie obrazu poprzez protokół Ethernet oraz odbieranie informacji, a następnie sterowanie szlabanem wjazdowym. Gotowy skrypt uruchamiany jest za pomocą konsoli systemowej.

# Realizacja systemu monitorującego parking

System rozpoznaje numery tablic rejestracyjnych w oparciu o szereg algorytmów. Poniżej przedstawiam specyfikację systemu oraz kolejne etapy jego działania.

## Specyfikacja funkcjonalna

Tabela .. Wymagania funkcjonalne systemu

|  |  |  |
| --- | --- | --- |
| Lp. | Nazwa wymagania | Opis wymagania funkcjonalnego |
| 1. | Pobieranie obrazu z kamery | System pobiera obraz z kamery rejestrującej obraz w czasie rzeczywistym. |
| 2. | Analiza obrazu | System realizuje proces rozpoznawania numerów tablicy rejestracyjnej z pobranego obrazu pochodzącego z kamery. |
| 3. | Dostęp do wyników analizy | Wyniki analizy obrazu, czyli numery tablic rejestracyjnych zapisywane są w lokalnej bazie danych i dostępne są dla użytkownika systemu. |
| 4. | Modyfikowalność bazy danych | Baza danych ma możliwość edycji, usuwania wybranych rekordów, a także dodawania zaufanych numerów tablic rejestracyjnych wraz z danymi o właścicielach pojazdów w celu późniejszej autoryzacji dostępu. |
| 5. | Konfiguracja połączenia sieciowego | Aplikacja obsługująca system ma możliwość konfiguracji połączenia sieciowego w celu połączenia z jednej strony części odpowiedzialnej za pracę kamery i sterowanie szlabanem, a z drugiej za część analizującą obraz i przechowującą dane. |
| 6. | Sterowanie szlabanem | System po rozpoznaniu numeru tablicy rejestracyjnej zarządza pracą szlabanu znajdującego się przy bramce wjazdowej. |
| 7. | Detekcja pojazdu za pomocą czujnika | Kamera w momencie wykrycia obiektu przez czujnik dokonuje zapisu obrazu, a następnie przesyła go do części systemu odpowiedzialnej za analizę. |

## Specyfikacja niefunkcjonalna

Tabela .. Wymagania niefunkcjonalne systemu

|  |  |  |
| --- | --- | --- |
| Lp. | Nazwa wymagania | Opis wymagania niefunkcjonalnego |
| 1. | Przechowywanie lokalnej bazy danych na dysku | Baza danych przechowywana jest w postaci pliku na dysku twardym użytkownika systemu. |
| 2. | Bezprzewodowe połączenie sieciowe między komponentami systemu | Obie części systemu połączone są za pomocą sieci bezprzewodowej. |
| 3. | Przechowywanie opcji programu na dysku | Wszelkie opcje konfiguracyjne programu przechowywane są na dysku twardym użytkownika systemu. |
| 4. | Prostota obsługi | System jest prosty w obsłudze, nie wymaga zaawansowanej znajomości obsługi komputera. |
| 5. | Czytelny interfejs graficzny | Graficzny interfejs użytkownika jest skomponowany w sposób czytelny i łatwy w obsłudze. |
| 6. | Przenośność aplikacji | Aplikacja kliencka ma możliwość zastosowania na dowolnej konfiguracji sprzętowej obsługującej systemy podane w punkcie 7 wymagań niefunkcjonalnych. |
| 7. | Obsługa pod systemami Windows 7, Windows 8, 8.1, Windows 10 | Aplikacja kliencka działa pod systemami Windows 7, Windows 8, 8.1 oraz Windows 10. |
| 8. | Aplikacja w języku polskim | Aplikacja przeznaczona jest dla użytkowników posługujących się językiem polskim. |

## Działanie systemu

### Wykrycie pojazdu i przesłanie obrazu

Obraz pobierany z kamery powinien przedstawiać przód samochodu z poziomu wysokości szlabanu, około 50 cm. W celu wykonania zdjęcia w odpowiednim momencie, posłużono się ultradźwiękowym czujnikiem odległości pracującym w zakresie 5 – 200 cm. Przyjęto, że odległość samochodu od obiektywu kamery, wynosząca 150 cm, jest odpowiednia do pobrania obrazu z kamery, dlatego też, napisano skrypt w języku Python, który zarządza pracą czujnika. Skrypt ten w momencie zlokalizowania obiektu znajdującego się przed czujnikiem w odległości mniejszej niż 150 cm wykonuje zdjęcie za pomocą kamery. Następnie obraz ten przesyłany jest za pomocą sieci bezprzewodowej do części systemu odpowiedzialnej za jego analizę. Odczyt odległości z czujnika odbywa się co 300 ms.



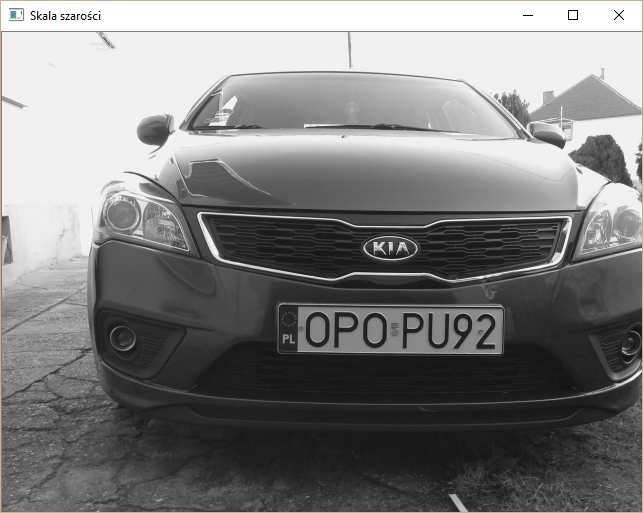
Rys. .. Obraz wykonany w momencie zlokalizowania obiektu w odległości mniejszej niż 150cm. Źródło: opracowanie własne

Rysunek 5.1 przedstawia obraz pobrany z kamery po wykryciu obiektu, do którego odległość nie wynosi więcej niż 150 cm. Rozdzielczość tego obrazu wynosi 640 x 480 pikseli, jak można dostrzec jakość jest bardzo dobra, a im mniejsza rozdzielczość, tym mniejszy rozmiar pliku i czas jego przesyłania przez sieć.

### Lokalizacja tablicy rejestracyjnej

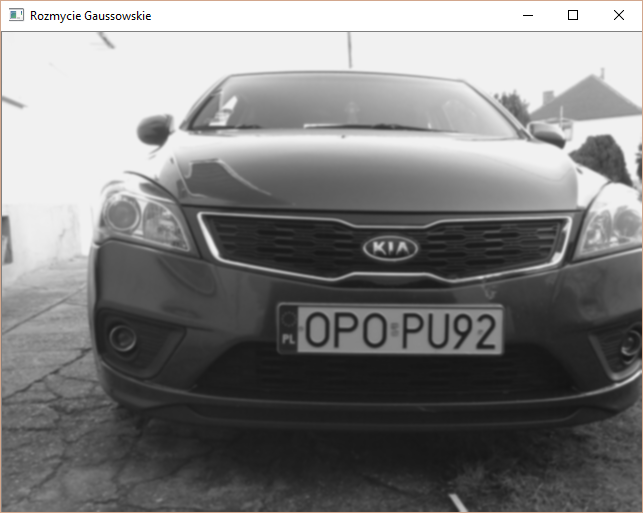
Kluczowym etapem w procesie rozpoznania numeru tablicy rejestracyjnej jest jej precyzyjna lokalizacja. Kod programu odpowiedzialny za lokalizację został napisany w języku C++ przy użyciu biblioteki OpenCV.

System po otrzymaniu obrazu z kamery ładuje zdjęcie do pamięci. Pierwszym krokiem jest konwersja kolorowego obrazu do obrazu w skali szarości (czarno – białego). Biblioteka OpenCV posiada wbudowane funkcje wykonujące tę operację za pomocą jednej linii kodu. Rysunek 5.2 pokazuje wynik konwersji obrazu.



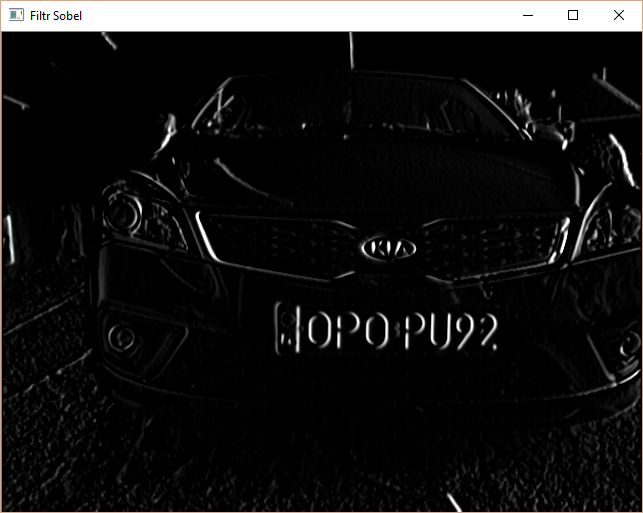
Rys. .. Obraz w skali szarości. Źródło: opracowanie własne

Usunięcie szumu z obrazu ma na celu eliminację niechcianych krawędzi pionowych. Użyte jest w tym wypadku rozmycie gaussowskie, na obraz nakłada się filtr o wymiarach 5x5 pikseli. W przypadku nie zastosowania filtru, obraz mógłby posiadać zakłócenia pochodzące z kamery lub otoczenia.



Rys. .. Zastosowanie filtra gaussowskiego o wymiarach 5x5 pikseli. Źródło: opracowanie własne

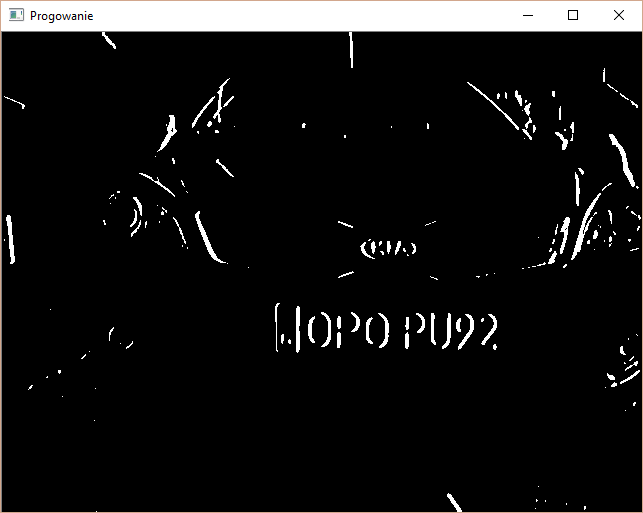
W celu wykrycia pionowych krawędzi wykorzystywany jest filtr Sobel. Uwydatnia on krawędzie na obszarze całego obrazu, a szczególnie w okolicy tablicy rejestracyjnej, gdzie znajduje się ich więcej niż w pozostałych regionach. Filtr ten dokonuje dyskretnego różniczkowania i aproksymacji pochodnych kierunkowych intensywności obrazu.



Rys. .. Zastosowanie filtra Sobel. Źródło: opracowanie własne

Funkcja nakładająca filtr Sobel w bibliotece OpenCV wymaga podania kilku parametrów: głębi obrazu wynikowego, parametrów dx i dy, związanych z kierunkiem wykrywania krawędzi, oraz wymiaru filtra [12]. Jako że kolejna operacja wymaga obrazu 8-bitowego, głębię obrazu wynikowego ustawiono na 8-bitową, parametr dx ustawiono na 1, zaś dy na 0, dzięki temu wykrywane są krawędzie pionowe, względem osi X, filtr ma wymiar 3x3 pikseli.

Jedną z najprostszych metod wydzielania obszarów jest progowanie. Polega ono na porównywaniu kolejnych pikseli z zadaną wartością, a następnie przypisaniu im odpowiednich wartości.



Rys. .. Zastosowanie progowania binarnego połączonego z progowaniem metodą Otsu. Źródło: opracowanie własne

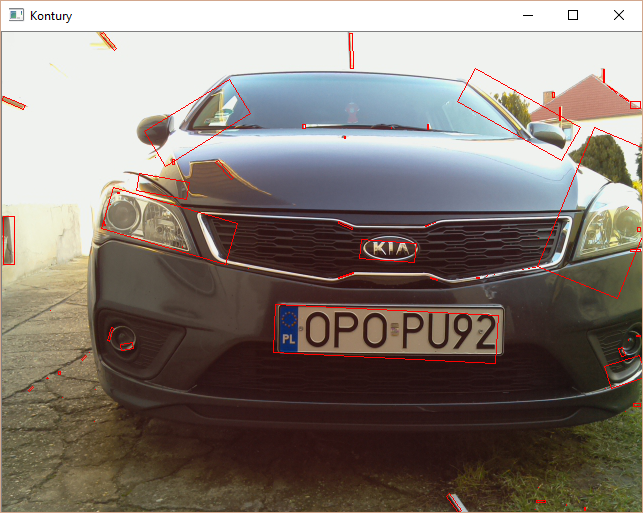
W wykonywanym procesie lokalizacji tablicy rejestracyjnej używane są dwa rodzaje progowania: progowanie binarne, porównujące dany piksel z progiem o wartości 0 oraz przypisujące wartość 255 jeżeli piksel nie jest równy progowi, a także progowanie metodą Otsu, która optymalnie dobiera wartość progowania [19][20]. Drugi z podanych rodzajów progowania wymaga obrazu 8-bitowego, o którym wspomniano w poprzednim akapicie.

Jako ostatnią operację przygotowywania obrazu do lokalizacji tablicy wykonuje się operację morfologiczną usuwającą przestrzeń między pionowymi liniami krawędzi, w wyniku czego otrzymuje się połączone regiony, które mogą być potencjalnymi obszarami obrazu zawierającymi tablicę rejestracyjną. Do wykonania operacji morfologicznej niezbędne jest wykorzystanie elementu strukturyzującego, posłużono się prostokątem o wymiarach 23x2 piksele.



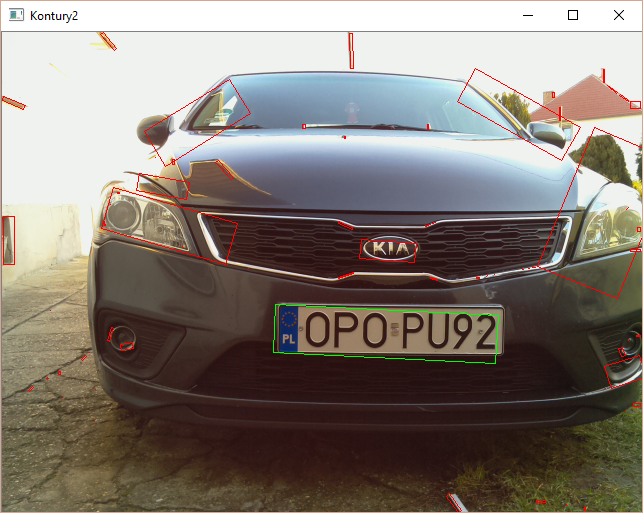
Rys. .. Operacja morfologiczna łącząca obszary mające dużą liczbę krawędzi. Źródło: opracowanie własne

Po wykonaniu powyższych przekształceń i konwersji można przystąpić do wykrycia krawędzi obszarów potencjalnie zawierających tablicę. W tym celu wykorzystuje się funkcję findContours() obrysowującą w/w obszary, jako parametr funkcji przyjęto wykrywanie jedynie zewnętrznych krawędzi, tak aby otrzymany kontur zawierał wszystkie piksele danego obszaru. Dla każdego wykrytego konturu wyodrębnia się minimalny prostokąt znajdujący się w jego wnętrzu. W celu wizualizacji wykrytych konturów obrysowano je kolorem czerwonym, przedstawia je rysunek 5.7.



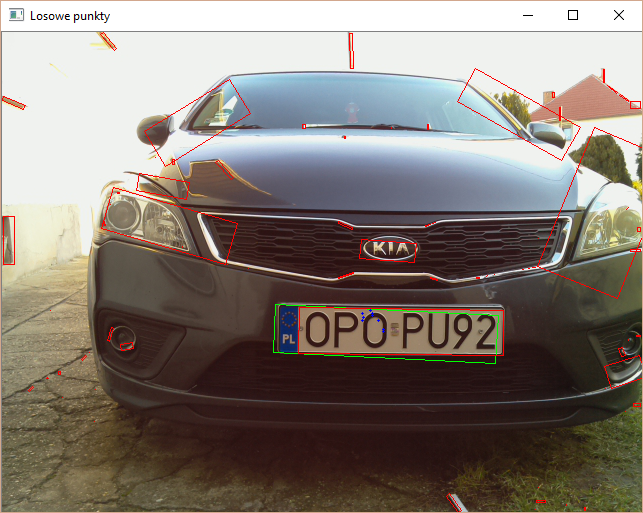
Rys. .. Wizualizacja prostokątnych obszarów mogących zawierać tablicę rejestracyjną. Źródło: opracowanie własne

Jak można zauważyć na rysunku 5.7 wykrywanych jest zbyt wiele fałszywych regionów. Aby się ich pozbyć wykorzystywana jest funkcja sprawdzająca wymiary i powierzchnię kolejnych obszarów. Podczas tworzenia funkcji wykonano szereg obliczeń bazując na rozdzielczości wykonywanych zdjęć oraz wymiarach tablic rejestracyjnych. Zgodnie z polskim prawem obowiązujące tablice mają stosunek szerokości do wysokości równy 4,56, można więc wykorzystać ten fakt i sprawdzić jaki stosunek mają prostokąty wykryte na obrazie. Funkcja ta sprawdza czy stosunek długości boków prostokątów jest większy bądź różny od 4, uwzględnia się więc również margines błędu. Dodatkowo obliczane jest pole prostokątów, po serii prób oszacowano, że pole tablicy rejestracyjnej na zdjęciu powinno mieścić się w przedziale od 8000 do 12000. Takie działanie ogranicza ilość błędnych wskazań, jego wynik przedstawiono na rysunku 5.8, gdzie kolorem zielonym oznaczono prostokąt, który poprawnie spełnił warunki sprawdzone w funkcji.



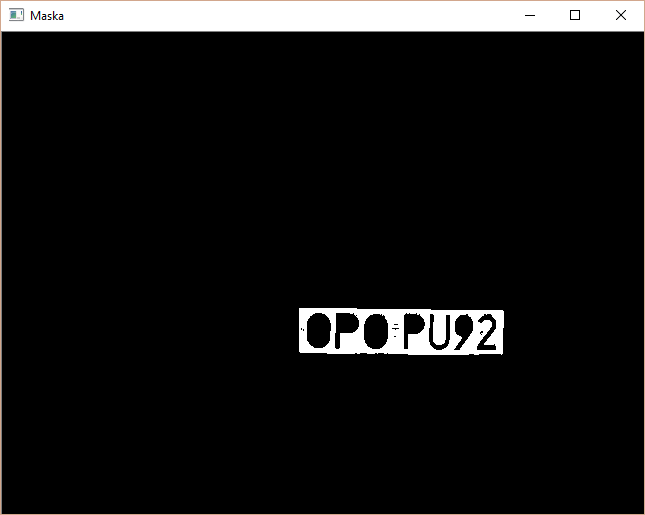
Rys. .. Wynik działania funkcji sprawdzającej wymiary i pole prostokątów (prostokąty o właściwych proporcjach oznaczone kolorem zielonym). Źródło: opracowanie własne

Proces lokalizacji tablic rejestracyjnych można usprawnić jeszcze bardziej. Jak da się zauważyć, tablice rejestracyjne stosowane od maja 2006 r. posiadają białe tło. Stosując algorytm Flood Fill wyodrębnia się precyzyjnie obszar o podobnym kolorze. Pierwszym krokiem przy realizacji tego algorytmu jest wygenerowanie kilku losowych punktów zlokalizowanych wokół środka danego prostokąta. Losowość punktów ma na celu trafienie w obszar będący białym tłem.



Rys. .. Kolorem niebieskim oznaczono losowo wybrane punkty. Źródło: opracowanie własne

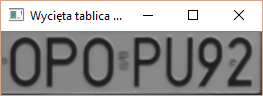
Następnie dla każdego z punktów tworzona jest maska za pomocą algorytmu Flood Fill. Algorytm jest częścią biblioteki OpenCV i wymaga zdefiniowania kilku parametrów, w przypadku realizowanej operacji przyjmuje się, że maksymalna górna i dolna różnica koloru/jasności wynosi 50, a nowa wartość wstawiana w miejsce pikseli wykraczających poza wartość wynosi 255, czyli odpowiada kolorowi białemu. Rozwiązanie te utworzy maskę zawierającą białe tło i czarne litery wypełnione wewnątrz znajdujące się na tablicy rejestracyjnej. Dodatkowo wykorzystuje się dwa parametry: CV\_FLOODFILL\_FIXED\_RANGE, pobierający różnicę między wylosowanym punktem, a obecnie porównywanym pikselem oraz CV\_FLOODFILL\_MASK\_ONLY, który jedynie wypełnia obraz, a nie zastępuje go.



Rys. .. Maska wygenerowana za pomocą algorytmu Flood Fill. Źródło: opracowanie własne

Z racji, iż nowo utworzonych masek jest tyle samo co wygenerowanych losowych punktów potrzeba ponownie skorzystać z wcześniej utworzonej funkcji sprawdzającej wymiary i powierzchnię kolejnych obszarów. Dodatkowo po sprawdzeniu przeprowadzana jest operacja porównywania kolejnych masek w celu wyeliminowania powtarzających się.

Kończąc etap lokalizacji przystępuje się do wycięcia fragmentu zlokalizowanej tablicy rejestracyjnej. Jako że tablica rejestracyjna może nie być ułożona równolegle do kamery, dokonuje się najpierw sprawdzenia kąta pod jakim się ona znajduje, a następnie przy pomocy operacji przekształcenia afinicznego poprawia się jej nachylenie. Tak przekształcona tablica jest gotowa do wycięcia, a następnie powiększenia do rozmiaru 260 x 60 piksele w celu lepszej współpracy z silnikiem optycznego rozpoznawania znaków.

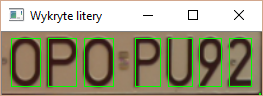


Rys. .. Zlokalizowana i wycięta tablica rejestracyjna. Źródło: opracowanie własne

Przedstawiona na rysunku 5.11 tablica rejestracyjna zlokalizowana jest poprawnie i zawiera wszystkie niezbędne cechy w celu rozpoznania numeru tablicy rejestracyjnej.

### Segmentacja znaków

Każdy znak znajdujący się na tablicy rejestracyjnej musi zostać dostarczony do silnika optycznego rozpoznawania znaków jako osobny element. W celu rozdzielenia znaków wykonuje się podobne operacje jak w przypadku etapu lokalizacji tablicy rejestracyjnej, zaczynając od odpowiednich przekształceń obrazu. Początkowo obraz konwertuje się do obrazu o skali szarości, następnie zaś obraz poddawany jest progowaniu. W tym przypadku jest to progowanie binarne odwrócone, w wyniku czego otrzymuje się obraz z czarnym tłem i białymi literami, jest on wtedy łatwiejszy do analizy przez funkcję wykrywającą krawędzie, gdyż jej algorytm szuka białych pikseli. W następnej kolejności wykorzystywana jest wspomniana funkcja wykrywająca krawędzie, dzięki czemu wyodrębniane są prostokąty zawierające litery. Oczywiście każda ze zlokalizowanych liter lub cyfr poddana jest walidacji, która sprawdza rozmiary i stwierdza czy dany obiekt jest poszukiwanym elementem.



Rys. .. Litery i cyfry zlokalizowane na tablicy rejestracyjnej. Źródło: opracowanie własne

Podczas procesu lokalizowania krawędzi poszczególne litery i cyfry wykrywane są w kolejności wynikającej z działania algorytmu, dlatego też niezbędne jest posortowanie kolejnych elementów. Okazuje się, że wystarczy tu posortowanie prostokątów oznaczonych zielonym kolorem na rysunku 5.12 względem punktów będących ich środkami. Zlokalizowane litery można wyciąć, jak widać na rysunku 5.12 prostokąty nie pokrywają się idealnie z konturami liter, dlatego przyjmuje się margines błędu i wycina obszary powiększone o 6 pikseli zarówno w poziomie jak i w pionie.

Tabela . Litery zlokalizowane na tablicy rejestracyjnej

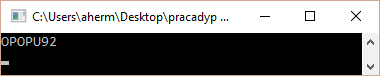
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Kolejne zlokalizowane litery | | | | | | |
|  |  |  |  |  |  |  |

Jak można zauważyć w tabeli 5.1 każdą z fragmentów obrazu zawierający zlokalizowaną literę lub cyfrę poddaje się procesowi progowania w taki sposób, aby kolor tła był biały, a znak znajdujący się na tablicy czarny.

### Optyczne rozpoznawanie znaków

Po zakończonym procesie segmentacji znaków należy przekształcić wyodrębnione fragmenty tablicy rejestracyjnej do postaci tekstowej. Do tego celu wykorzystuje się silnik optycznego rozpoznawania znaków – Tesseract OCR. Podczas konfiguracji silnika ze środowiskiem programistycznym podaje się gotowy zestaw słownikowy zawierający przetrenowane wartości. Przy poszukiwaniach odpowiedniego silnika OCR okazało się, że istnieje gotowy zestaw przetrenowany europejskimi tablicami rejestracyjnymi [6], w związku z tym, że w znacznym stopniu ułatwia to realizację tworzonego systemu, zdecydowano się na jego wykorzystanie.

Jak opisano wcześniej obrazy z białym tłem i czarnymi znakami przekazane zostają kolejno do silnika OCR, taka kolorystyka przyspiesza proces rozpoznawania. Samo rozpoznawanie przeprowadzane jest automatycznie i nie wymaga większej wiedzy użytkownika.

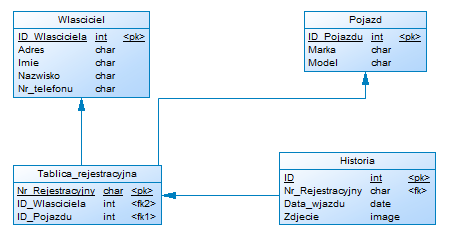


Rys. .. Wynik działania optycznego rozpoznawania znaków w fazie testowej. Źródło: opracowanie własne

Rysunek 5.13 przedstawia wynik optycznego rozpoznawania znaków wykonanego w fazie testów, stąd okno konsoli. Rozpoznane litery odpowiadają znakom znajdującym się na testowanej tablicy rejestracyjnej, ich kolejność jest poprawna.

### Operacje na bazie danych

Baza danych wykorzystana w systemie oparta jest o system zarządzania bazą danych Microsoft SQL Server 2008. Dane, a właściwie numer tablicy rejestracyjnej, po rozpoznaniu porównywany jest ze zbiorem zapisanych w bazie tablic. W przypadku odnalezienia pasującego rekordu, wysyłana jest informacja zwrotna do minikomputera Raspberry Pi, która nakazuje podniesienie szlabanu na bramce wjazdowej. W przypadku gdy dany numer rejestracyjny nie znajduje się w bazie, na ekranie laptopa wyświetla się komunikat z informacją o próbie wjazdu niezidentyfikowanego pojazdu na teren parkingu. Taki pojazd można dodać do bazy danych za pomocą odpowiedniej opcji w aplikacji klienckiej. Wszystkie próby dostępu na parking zapisywane są w systemie.



Rys. .. Model fizyczny bazy danych. Źródło: opracowanie własne

Utworzona baza danych składa się z czterech tabel i zawiera niezbędne informacje do identyfikacji właściciela pojazdu oraz do kontaktu z nim. Każde zdjęcie, które przechodzi proces rozpoznawania jest zapisywane. Baza danych ograniczona jest do absolutnego minimum wystarczającego do sprawnej pracy systemu. Przy tworzeniu systemu zwrócono uwagę na możliwość ewentualnego przeniesienia systemu w inną lokalizację, dlatego baza zapisywana jest lokalnie na dysku użytkownika aplikacji, co pozwala na szybką możliwość wykonania jej kopii. Wszystkie rekordy posiadają możliwość edycji, usunięcia, bądź też dodania nowych informacji.

### Sterowanie szlabanem

Na potrzeby realizowanego systemu utworzono prototypowy szlaban, który jedynie imituje działanie prawdziwych szlabanów znajdujących się przy bramkach wjazdowych. W tym celu wykorzystano silnik krokowy 2BYJ-48, który zarządzany jest z poziomu Raspberry Pi. System po przeanalizowaniu obrazu i porównaniu wyniku z numerami tablic znajdującymi się bazie danych decyduje o otwarciu szlabanu. Informacja przesyłana jest do minikomputera Raspberry, gdzie za pomocą skryptu napisanego w języku Python odbywa się proces sterowania szlabanem. Zostaje on podniesiony, po czym odczekuje 5s i wraca do pozycji początkowej.

## Opis interfejsu użytkownika

W tym podrozdziale znajdują się opisane zrzuty ekranów graficznego interfejsu użytkownika aplikacji obsługującej monitoring parkingu. Przedstawione i wyjaśnione jest również rozwiązanie znajdujące się po stronie minikomputera Raspberry Pi.

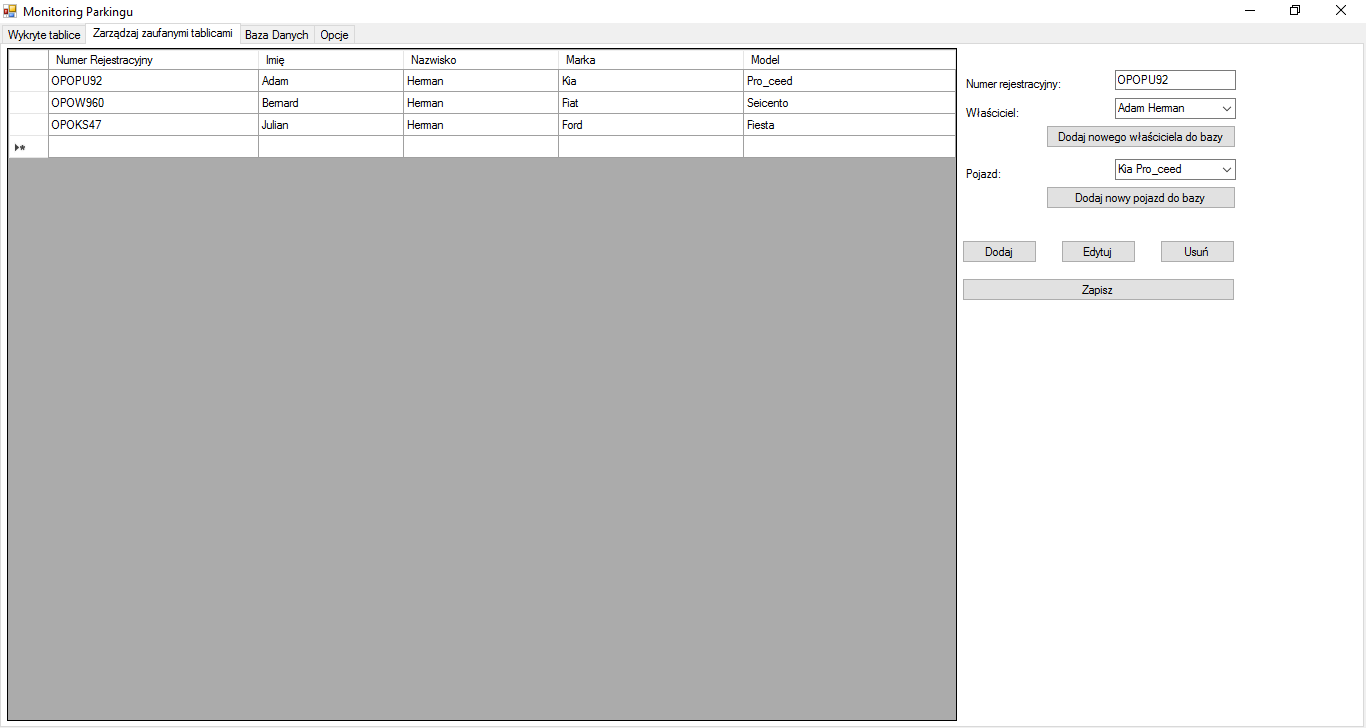
Podstawową kwestią, o której należy wspomnieć, jest odpowiednie skonfigurowanie systemu podczas jego montażu i instalacji. Część systemu odpowiedzialna za wykrywanie pojazdów oraz wykonywanie zdjęć uruchamiana jest automatycznie razem ze startem systemu Raspbian na platformie Raspberry. Niezbędna jest odpowiednia konfiguracja ustawień sieciowych, a także edycja skryptu uruchamianego właśnie ze startem systemu. W owym skrypcie koniecznie należy podać poprawny adres IP komputera z aplikacją kliencką oraz numer portu. Obie części systemu muszą znajdować się w tej samej podsieci.

Ekran startowy wyświetlający się po uruchomieniu aplikacji znajduje się na rysunku 5.15. Przedstawia on najistotniejsze informacje dotyczące pojazdów próbujących wjechać na monitorowany parking. Po przeanalizowaniu obrazu system zapisuje dane do bazy danych, które następnie pojawiają się na ekranie w formie tabeli. Prezentowane są dane odnośnie pojazdu, właściciela pojazdu, numeru rejestracyjnego, a także zdjęcie wykonane przy bramce wjazdowej oraz informacja o tym, czy dana rejestracja została pomyślnie zautoryzowana. Dodatkowo użytkownik ma możliwość wyszukania zadanej wartości takiej jak: numer rejestracyjny, właściciel pojazdu oraz pojazd. Możliwa jest również filtracja wyświetlanych informacji na podstawie daty.



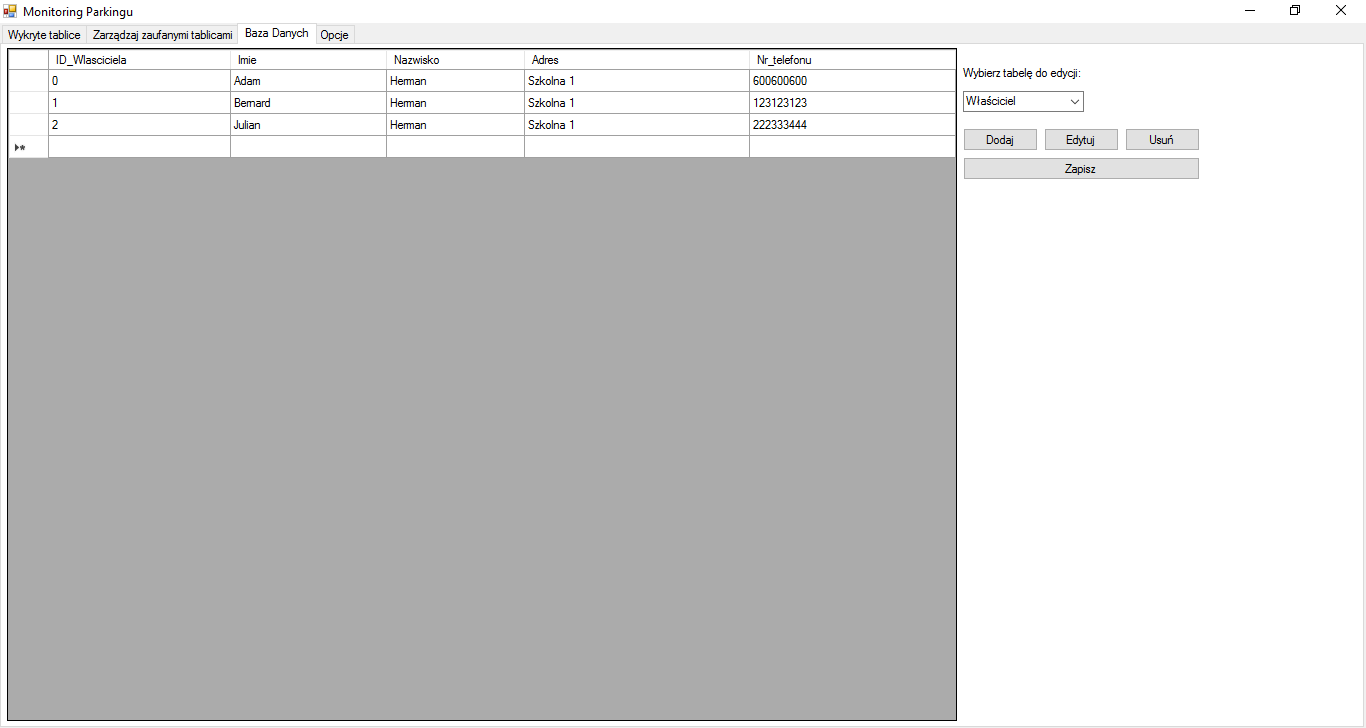
Rys. .. Interfejs graficzny aplikacji - zakładka "Wykryte tablice". Źródło: opracowanie własne

Kolejny ekran przedstawiony na rysunku 5.16 służy zarządzaniem listą, na której znajdują się zaufane numery tablic rejestracyjnych. Użytkownik ma do dyspozycji takie operacje jak dodawanie, edycja bądź usuwanie wymienionych wcześniej numerów tablic. W tej części aplikacji dostępne są również opcje odpowiedzialne za dodawanie nowych właścicieli oraz pojazdów.



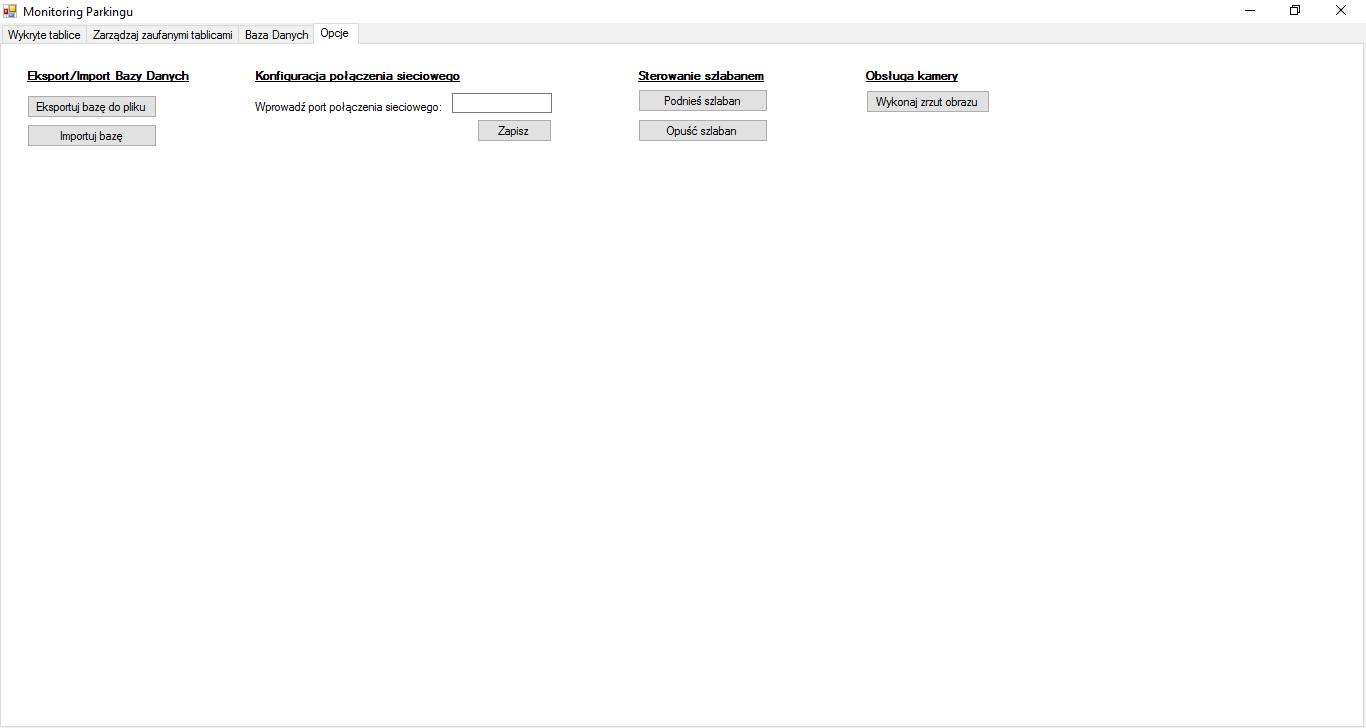
Rys. .. Interfejs graficzny aplikacji - zakładka "Zarządzaj zaufanymi tablicami". Źródło: opracowanie własne

Zakładka „Baza Danych” jak sama nazwa wskazuje pozwala na wgląd do wszystkich informacji znajdujących się w bazie danych, a także na ich modyfikację.



Rys. .. Interfejs graficzny aplikacji - zakładka "Baza Danych". Źródło: opracowanie własne

Ostatnia z kart o nazwie „Opcje” umożliwia użytkownikowi eksport bądź import bazy danych, a także konfigurację numeru portu, przez który nawiązywane jest połączenie z minikomputerem Raspberry Pi. Użytkownik ma również możliwość ręcznego podniesienia lub opuszczenia szlabanu i wykonania zrzutu obrazu z kamery.



Rys. .. Interfejs graficzny aplikacji - zakładka "Opcje". Źródło: opracowanie własne

Interfejs graficzny został zaprojektowany w sposób czytelny dla użytkownika, a dzięki swojej prostocie jest łatwy w obsłudze.

## Podsumowanie działania systemu

System zrealizowany na potrzeby niniejszej pracy spełnia swoje zadanie. Zastosowanie aktywnej wizji umożliwia analizę obrazu zawierającego tablicę rejestracyjną pojazdu wjeżdżającego na parking. Założone na etapie planowania systemu wymagania funkcjonalne i niefunkcjonalne zostały spełnione w całości. Utworzony system działa stabilnie, zdarzają się jednak błędy w procesie rozpoznania numeru tablicy rejestracyjnej, ich przyczyny opisano poniżej.

Utworzony system został przetestowany w celu sprawdzenia jego niedoskonałości. Jedną z podstawowych wad systemów ANPR jest fakt, iż analizowane tablice rejestracyjne muszą cechować się dużą czystością. Jakiekolwiek większe zabrudzenia wprowadzają przekłamania w obrazie. Dokładnie ten sam problem pojawił się w trakcie prac nad projektem. Wpływ na poprawne działanie systemu ma również oświetlenie, jego niedostateczna ilość lub brak powoduje, że wykonane zdjęcie jest zbyt ciemne. Z kolei zbyt duża ilość światła trafiającego na powierzchnię tablicy rejestracyjnej doprowadza do odbić na jej powierzchni. Głównym problemem zrealizowanego systemu jest błędne rozpoznawanie podobnych do siebie znaków takich jak np. ‘0’ i ‘O’, ‘8’ i ‘B’, ‘1’ i ‘I’. Wiadomo dzięki temu dlaczego w polskim prawie dotyczącym tablic rejestracyjnych ograniczono zbiór liter występujących w części będącej wyróżnikiem pojazdu. Okazało się również, iż wykorzystany silnik OCR, a dokładniej zestaw, którym przetrenowano jego bazę wiedzy zawierał zbyt małą ilość danych o polskich tablicach rejestracyjnych.

Jednym z kluczowych aspektów działania systemu ANPR jest jego szybkość. Z uwagi, iż zdecydowano się rozdzielić system na dwie części, gdzie analiza obrazu odbywa się na sprzęcie o większej mocy obliczeniowej, udało się uzyskać akceptowalny czas obsługi jednego pojazdu. Czas przesłania zdjęcia z jednej części do drugiej wynosi około 3 s, zaś sama analiza zajmuje 1 s. Doliczając do tego pozostałe czynniki takie jak, czas wykonywania zdjęcia, przesłania sygnału do szlabanu oraz porównanie numeru wykrytej tablicy z listą tablic zapisanych w bazie, czas końcowy obsługi wynosi średnio 6 s. Zwracając uwagę na fakt, że system tworzyła jedna osoba, można przyjąć ten wynik jako zadowalający.

# Wnioski i uwagi

Celem niniejszej pracy było zrealizowanie systemu obsługującego monitoring parkingu z rozpoznawaniem numerów rejestracyjnych wraz z implementacją aktywnej wizji. Z uwagi, iż na rynku istnieje wiele gotowych rozwiązań, zdecydowano się utworzyć system, który wyróżnia prostota oraz niski koszt wykonania, a jednocześnie nieustępujący komercjalnym systemom w swojej funkcjonalności.

Jedną z cech odróżniających utworzony system od innych jest bezprzewodowa współpraca sterownika szlabanu, kamery, czujnika odległości z częścią zarządzającą parkingiem i analizującą obraz. W stosunku do rozwiązań z zakresu monitoringu parkingów, wykonany system nie wymaga jakiejkolwiek interakcji osób wjeżdżających pojazdem na parking, cały proces realizowany jest automatycznie po wcześniejszym wprowadzeniu danych do bazy.

Z pewnością wykonany system posiada niedoskonałości, jednak jest to temat do dalszej pracy. Wykorzystane algorytmy można udoskonalić lub napisać własne, innowacyjne. Dobrym rozwiązaniem byłoby wykorzystanie sieci neuronowej do rozpoznawania tablic rejestracyjnych, w wyniku czego wzrosłaby precyzja i szybkość działania systemu. Nowatorskim rozwiązaniem okazałoby się zamknięcie całego systemu w niewielkiej obudowie znajdującej się na szlabanie, a następnie zarządzanie nim poprzez stronę WWW.

Patrząc na rozwiązania komercyjne niemożliwe okazuje się uzyskanie 100% dokładności rozpoznawanych znaków. Istnieje wiele czynników wpływających na wyniki działania systemów automatycznego rozpoznawania numerów rejestracyjnych, których nie da się przewidzieć, ani też zlikwidować. Niekorzystne warunki atmosferyczne, zabrudzenia tablicy, nieodpowiednie oświetlenie to tylko niektóre z nich.

Temat rozpoznawania numerów tablic rejestracyjnych jest stosunkowo młodą dziedziną technologii. Obecny postęp technologiczny pozwala na stosowanie coraz bardziej innowacyjnych rozwiązań, pozwala na łączenie kilku dziedzin razem. Przyszłością systemów ANPR jest ich połączenie ze sztuczną inteligencją. W czasach wszechobecnego monitoringu takie połączenie daje ogromne możliwości, dzięki którym znacznie może poprawić się bezpieczeństwo obywateli.

Omawiany system został pomyślnie wykonany, a następnie zaimplementowany. Wszystkie cele pracy, która stanowi dokumentację wykonanego projektu zostały osiągnięte.

# Literatura

[1] Adaptive Recognition, *CARMEN® Systems* [on-line][dostęp 20.12.2015]. Dostępny   
w World Wide Web: http://www.platerecognition.info/1201.htm.

[2] G. Bradski, A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*, Wyd. 1, Sebastopol 2008.

[3] D. Kamiński, *Jak chronić parkingi?.* Czasopismo „Zabezpieczenia” [on-line] 2013, nr 4 [dostęp 05.11.2015]. Dostępny w World Wide Web: http://zabezpieczenia.com.pl/kontrola-dost%C4%99pu/jak-chroni%C4%87-parkingi.

[4] Metasoft, *ARTR* [on-line] [dostęp 20.12.2015]. Dostępny w World Wide Web: http://www.metasoft.pl/pl/artr.html.

[5] M. Miklasz, A. Nowosielski, G. Kawka, *Automatyczne rozpoznawanie tablic rejestracyjnych w nadzorze ruchu drogowego.* Magazyn Autostrady [on-line] 2010,   
nr 10, s. 64 [dostęp 06.11.2105]. Dostępny w World Wide Web: http://www.zut.edu.pl/fileadmin/pliki/rzecznik/15112010/Rozpoznawanie%20tablic%20rejestracyjnych.pdf.

[6] OpenALPR [on-line] [dostęp 14.12.2015]. Dostępny w World Wide Web: http://doc.openalpr.com/.

[7] PolCam, *PolCam Mobile ANPR* [on-line] [dostęp 20.12.2015]. Dostępny w World Wide Web: http://polcam.eu/pl/products/wideorejestratory/polcam-mobile-anpr.

[8] Polski Związek Przemysłu Motoryzacyjnego, *Raport 2015* [on-line]. 2015, s. 63 [dostęp 16.09.2015]. Dostępny w World Wide Web: http://www.pzpm.org.pl/Rynek-motoryzacyjny/Roczniki-i-raporty/Raport-branzy-motoryzacyjnej-2015.

[9] RoboticVision, *Co to jest ANPR?.* [on-line] [dostęp 05.11.2015]. Dostępny w World Wide Web:https://roboticvision.wordpress.com/2011/02/19/co-to-jest-anpr/.

[10] *Rozporządzenie Ministra Infrastruktury z w sprawie rejestracji i oznaczania pojazdów*, Dz. U. Nr 133, poz. 1123, z dnia 22 lipca 2002 r.

[11] *Rozporządzenie Ministra Transportu i Budownictwa z dnia 20 kwietnia 2006 r. zmieniające rozporządzenie w sprawie rejestracji i oznaczania pojazdów*, [Dz. U. Nr 70, poz. 489](http://isap.sejm.gov.pl/DetailsServlet?id=WDU20060700489) §1 pkt 13 oraz zał. nr 5, z dnia 20 kwietnia 2006 r.

[12] G. Sarbicki*, OpenCV w Pythonie* [on-line] Rozdział 3.8. *Znajdowanie krawędzi* [dostęp 19.12.2015]. Dostępny w World Wide Web: http://www.fizyka.umk.pl/~gniewko/python/OpenCV\_przetwarzanie/.

[13] Siemens, *Sicore* [on-line][dostęp 20.12.2015]. Dostępny w World Wide Web: http://www.siemens.co.uk/traffic/pool/documents/brochure/62163\_sie\_sicore\_brochure\_4pp.pdf.

[14] M. Stec, *Tablice rejestracyjne*. Gazeta Podatnika [on-line] 1 czerwca 2007 r. [dostęp 25 października 2015]. Dostępny w World Wide Web: http://www.gazetapodatnika.pl/artykuly/tablice\_rejestracyjne-a\_3235.htm.

[15] Wikipedia, *Automatic number plate recognition* [on-line] [dostęp 11.10.2015]. Dostępny w World Wide Web:https://en.wikipedia.org/wiki/Automatic\_number\_plate\_recognition#Development\_history.

[16] Wikipedia, *Active Vision* [on-line] [dostęp 11.01.2016]. Dostępny w World Wide Web: https://en.wikipedia.org/wiki/Active\_vision.

[17] Wikipedia, *Raspberry Pi* [on-line] [dostęp 02.01.2016]. Dostępny w World Wide Web: https://pl.wikipedia.org/wiki/Raspberry\_Pi.

[18] Wikipedia, *Tesseract (software)* [on-line] [dostęp 11.01.2016]. Dostępny w World Wide Web: https://en.wikipedia.org/wiki/Tesseract\_(software).

[19] M. Wilczewski, *Metody binaryzacji obrazów* [on-line] [dostęp 29.12.2015]. Dostępny w World Wide Web: http://www.mif.pg.gda.pl/homepages/marcin/Wyklad3.pdf.

[20] S. Wojas, *Metody przetwarzania obrazów z wykorzystaniem biblioteki OpenCV* [on-line]. Kraków 2010. Rozdział 3.1. *Progowanie* [dostęp 28.12.2015]. Dostępny   
w World Wide Web: http://www.focus.agh.edu.pl/theses/MGR04.pdf.

**Źródła rysunków**

[21] Dziennik Gazeta Prawna, *Oto nowy wynalazek dla drogówki. Nie ma zmiłuj, wpadnie każdy* [on-line]. 1 czerwca 2011 r. [dostęp 27.12.2015]. Dostępny w World Wide Web: http://1.s.dziennik.pl/pliki/2130000/2130122-polcam.jpg.

[22] Edimax, *EW-7811Un* [on-line] [dostęp 13.12.2015]. Dostępny w World Wide Web: http://www.edimax.pl/images/Image/ProductImage/WNIC/217x205/EW-7811Un\_217x205.jpg.

[23] Future electronics, *Raspberry Pi – Model B (Mini Linux/Arm PC)* [on-line] [dostęp 29.12.2015]. Dostępny w World Wide Web: http://www.fut-electronics.com/wp-content/uploads/2013/01/raspberry-in-out.jpg.

[24] ImageAnalytics Pacific, *ANPR/LPR BASED SOLUTIONS FOR ACCESS CONTROL AND PARKING MANAGEMENT* [on-line] [dostęp 29.12.2015]. Dostępny   
w World Wide Web: http://static.wixstatic.com/media/f47fca\_fa2d17cdfdf7496baa6e391cee43a45c.png\_srz\_500\_305\_85\_22\_0.50\_1.20\_0.00\_png\_srz.

[25] MicroControllerElectronics, *HC-SR04* [on-line] [dostęp 29.12.2015]. Dostępny   
w World Wide Web: http://microcontrollerelectronics.com/wp-content/uploads/2014/10/HCSR04.jpg.

[26] Notebookcheck, *Lenovo ThinkPad Edge E320 NWY3RGE* [on-line] [dostęp 03.01.2016]. Dostępny w World Wide Web: http://www.notebookcheck.net/uploads/tx\_nbc2/Lenovo\_E320.jpg.

[27] Random snippets of knwoledge, *Controlling motors on the Raspberry Pi using C#* [on-line] [dostęp 13.12.2015]. Dostępny w World Wide Web: http://blog.turbine51.net/assets/img/blog/28byj-48.jpg.

[28] SB Components, *HD Camera Module for Raspberry Pi Model A and Model B* [on-line] [dostęp 20.12.2015]. Dostępny w World Wide Web: http://www.sbcshop.co.uk/ekmps/shops/sbcshop/images/hd-camera-module-for-raspberry-pi-model-a-and-model-b-[4]-137-p.jpg.

[29] Wikipedia, *Polskie tablice rejestracyjne* [on-line] [dostęp 11.12.2015]. Dostępny   
w World Wide Web:https://pl.wikipedia.org/wiki/Polskie\_tablice\_rejestracyjne#/media/File:Pltableseries2006.jpg.

[30] *Rozporządzenie Ministra Transportu i Gospodarki Morskiej w sprawie rejestracji i oznaczania pojazdów*, Dz. U. 1999.59.632[on-line] [dostęp 02.11.2015]. Dostępny   
w World Wide Web: http://www.lex.pl/image/image\_gallery?img\_id=77339269.

# Dodatek A. Wybrane fragmenty programu komputerowego

Fragment skryptu obsługującego ultradźwiękowy czujnik odległości oraz kamerę:

|  |
| --- |
| nazwa=time.strftime("%Y%m%d-%H%M%S")  def odczytuj(sensor):    GPIO.setwarnings(False)  GPIO.setmode(GPIO.BCM)    if sensor == 0:    GPIO.setup(23,GPIO.OUT)  GPIO.setup(24,GPIO.IN)  GPIO.output(23, GPIO.LOW)  time.sleep(0.3)  GPIO.output(23, True)  time.sleep(0.00001)  GPIO.output(23, False)    while GPIO.input(24) == 0:  signaloff = time.time()    while GPIO.input(24) == 1:  signalon = time.time()    roznicaczasu = signalon - signaloff  odleglosc = roznicaczasu \* 17000  return odleglosc  GPIO.cleanup()  else:  print "Niepoprawna zmienna funkcji."  if odczytuj(0) < 50:  npliku=str(nazwa)+".jpg"  os.system("raspistill -t 100 -n -w 640 -h 480 -o img/"+npliku) |

Fragment skryptu odpowiedzialnego za przesłanie obrazu z kamery do części systemu odpowiedzialnej za analizę:

|  |
| --- |
| HOST = '192.168.1.11' # adres laptopa  PORT = 5005  def wyslij(sciezka):  socket1 = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)  socket1.connect((HOST, PORT))  plik = sciezka  with open(plik, 'rb') as plikwysylany:  for data in plikwysylany:  socket1.sendall(data)  print 'Wyslano pomyslnie'  socket1.close()  return |

Własne funkcje wykorzystane w programie:

|  |
| --- |
| void sortowanie(vector<Rect> boundRectz, vector<Point2f> center, int n, vector<Rect> \*b, vector<Point2f> \*c)  {  Point2f pom;  Rect pom2;  for (int i = 0; i < n; i++){  for (int j = 0; j < n - i - 1; j++)  if (center[j].x > center[j + 1].x)  {  pom = center[j];  center[j] = center[j + 1];  center[j + 1] = pom;  pom2 = boundRectz[j];  boundRectz[j] = boundRectz[j + 1];  boundRectz[j + 1] = pom2;  }  }  \*b = boundRectz;  \*c = center;  }  bool sprawdz(RotatedRect boundRect){  bool wyjscie = false;  float szerokosc = boundRect.size.width;  float wysokosc = boundRect.size.height;  if ((szerokosc != 0) & (wysokosc != 0)){  if (((wysokosc / szerokosc >= 4) && (wysokosc > szerokosc)) || ((szerokosc / wysokosc >= 4) && (szerokosc > wysokosc))){  if ((wysokosc\*szerokosc < 12000) && (wysokosc\*szerokosc>8000)){  wyjscie = true;  }  }  }  return wyjscie;  }  vector<Point> generuj\_punkty(Point2f centre, int szerokosc, int wysokosc){  int minimum = int(min(szerokosc, wysokosc));  vector<Point> punkt(10);  for (int i = 0; i < 10; i++){  int rand1 = rand() % 1000;  int rand2 = rand() % 1000;  int a = ceil(minimum / 2);  if (a == 0) a = 1;  punkt[i] = Point(centre.x + rand1%a - a, centre.y + rand2%a - a);  }  return punkt;  }  Mat generuj\_maske(Mat image, Point punkt){  Mat mask = Mat::zeros(image.rows + 2, image.cols + 2, CV\_8U);  int dolr = 50;  int gorr = 50;  int iloscpix = 4;  int nowawartosc = 255;  int flagi = iloscpix + (nowawartosc << 8) + FLOODFILL\_FIXED\_RANGE + FLOODFILL\_MASK\_ONLY;  floodFill(image, mask, punkt, Scalar(255, 0, 0), (cv::Rect\*)0, Scalar(dolr, dolr, dolr), Scalar(gorr, gorr, gorr), flagi);  return mask;  }  bool roznica(Mat image, Mat image2){  Mat roznica = image - image2;  bool wyjscie = false;  float a = sum(abs(roznica))[0];  float b = sum(image)[0];  float c = sum(image2)[0];  if (a / (min(b,c)) < 0.01){  wyjscie = true;  }  return wyjscie;  } |

Fragment kodu odpowiedzialnego za lokalizację tablicy rejestracyjnej

|  |
| --- |
| cvtColor(image, image2, COLOR\_BGR2GRAY);    Mat imgblur;  GaussianBlur(image2, imgblur, Size(5, 5), 0);  Mat imgsobel;  Sobel(imgblur, imgsobel, CV\_8U, 1, 0, 3);  Mat imgthre;  threshold(imgsobel, imgthre, 0, 255, CV\_THRESH\_BINARY + CV\_THRESH\_OTSU);  Mat imgmorph;  Mat se = getStructuringElement(MORPH\_RECT, Size(23, 2));  morphologyEx(imgthre, imgmorph, MORPH\_CLOSE, se);  vector<vector<Point> > contours;  findContours(imgmorph, contours, RETR\_EXTERNAL, CHAIN\_APPROX\_NONE);  vector< RotatedRect> prostokat(contours.size());  CvPoint2D32f punkty\_prost[4];  Point boxPoints2[4];  for (int i = 0; i < contours.size(); i++)  {  prostokat[i] = minAreaRect(contours[i]);  cvBoxPoints(prostokat[i], punkty\_prost);  for (int j = 0; j < 4; j++){  line(image, punkty\_prost[j], punkty\_prost[(j + 1) % 4], CV\_RGB(255, 0, 0), 1, 8);  }  }  vector<RotatedRect> prostokaty;  vector<Point2f>centery;  for (int i = 0; i < contours.size(); i++)  {  if (sprawdz(prostokat[i]) == true){  cvBoxPoints(prostokat[i], punkty\_prost);  for (int j = 0; j < 4; j++){  line(image, punkty\_prost[j], punkty\_prost[(j + 1) % 4], CV\_RGB(0, 255, 0), 1, 8);  }  prostokaty.push\_back(prostokat[i]);  centery.push\_back(prostokat[i].center);  }  }  Mat image\_mask = oryginal;    vector <Mat> maski;  for (int i = 0; i < centery.size(); i++){  Point2f centre = centery[i];  int width = prostokaty[i].size.width;  int height = prostokaty[i].size.height;  vector<Point> seeds = generuj\_punkty(centre, width, height);  for (int j = 0; j < seeds.size(); j++){  circle(image, seeds[j], 1, (0, 0, 255), -1);  maski.push\_back(generuj\_maske(image\_mask, seeds[j]));  }  }  vector <Mat> sprawdzone\_maski;  Mat temp;  vector<vector<Point> > contours2;  vector< RotatedRect> prostokat3;  for (int i = 0; i < maski.size(); i++){  temp = maski[i].clone();  findContours(maski[i], contours2, RETR\_EXTERNAL, CHAIN\_APPROX\_NONE);  vector<vector<Point> > contours\_poly2(contours2.size());  vector< RotatedRect> prostokat2(contours2.size());  for (int i = 0; i < contours2.size(); i++)  {  prostokat2[i] = minAreaRect(contours2[i]);  if (sprawdz(prostokat2[i]) == true){  cvBoxPoints(prostokat2[i], punkty\_prost);  for (int j = 0; j < 4; j++){  line(image, punkty\_prost[j], punkty\_prost[(j + 1) % 4], CV\_RGB(255, 0, 0), 1, 8);  }  prostokat3.push\_back(prostokat2[i]);  sprawdzone\_maski.push\_back(temp);  }  }  }  vector <Mat> wyciete\_obrazy;  for (int i = 0; i < sprawdzone\_maski.size(); i++){  int width = prostokat3[i].size.width;  int height = prostokat3[i].size.height;  float angle = prostokat3[i].angle;  Size rozmiar\_prost = prostokat3[i].size;  Point2f centre = prostokat3[i].center;  Mat M, obrocony;  if (prostokat3[i].angle < -45.) {  angle += 90.0;  swap(rozmiar\_prost.width, rozmiar\_prost.height);  }  M = getRotationMatrix2D(prostokat3[i].center, angle, 1.0);  warpAffine(oryginal, obrocony, M, oryginal.size(), INTER\_CUBIC);  Mat wycinek;  getRectSubPix(obrocony, rozmiar\_prost, prostokat3[i].center, wycinek);  resize(wycinek, wycinek, Size(260, 60));  wyciete\_obrazy.push\_back(wycinek);  } |

# Dodatek B. Spis zawartości dołączonej płyty CD

* praca\_dyplomowa\_inzynierska.docx – niniejszy dokument,
* dodatkowe\_info\_do\_suplementu.docx – dodatkowe informacje do suplementu,
* suplement\_info\_english.docx – dodatkowe informacje do suplementu w języku angielskim.

Opole, dnia ………...2016 r.

Adam Herman

*(imi*ę *i nazwisko)*

Informatyka

*(kierunek studiów)*

Wydział Elektrotechniki, Automatyki i Informatyki

*(Wydział Politechniki Opolskiej)*

**OŚWIADCZENIE**

Oświadczam, że złożona praca ~~magisterska~~ /inżynierska /~~licencjacka~~\*)

pt.

„Monitoring parkingu z rozpoznawaniem numerów rejestracyjnych -   
implementacja aktywnej wizji”

została napisana przeze mnie samodzielnie, pod kierunkiem promotora:

dr inż. Michał Podpora

(*tytuł/stopie*ń *naukowy, imi*ę *i nazwisko promotora*)

a jej wersja drukowana (forma papierowa) jest w swej treści zgodna z przedłożoną wersją elektroniczną pracy.

Równocześnie informuję, że praca nie narusza praw autorskich osób trzecich w rozumieniu ustawy z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych (tekst jednolity: Dz. U. Nr 90 z 2006 r., poz. 631 z późn. zm.).

Praca nie zawiera informacji i danych uzyskanych w sposób nielegalny i nie była wcześniej przedmiotem innych procedur związanych z uzyskaniem dyplomów lub tytułów zawodowych wyższych uczelni.

………...........………………

*(podpis)*

\_\_\_\_\_\_\_\_\_\_\_\_\_

\* niepotrzebne skreślić