

**POLITECHNIKA ŚLĄSKA**

**WYDZIAŁ AUTOMATYKI, ELEKTRONIKI I INFORMATYKI**

Projekt inżynierski

Integracja oprogramowania wizyjnej detekcji pozycji obiektu manipulacji ze sterownikiem robota adept.

Autor: Artur Meller

Kierujący Pracą: dr hab. inż. Tadeusz Szkodny

Konsultant: mgr inż. Adrian Łęgowski

Gliwice, Luty 2015

Spis treści

[1 Wstęp 3](#_Toc410810662)

[2 Cel i zakres prac 4](#_Toc410810663)

[2.1 Cele projektu 4](#_Toc410810664)

[2.2 Wykorzystane elementy 4](#_Toc410810665)

[3 Stanowiska zrobotyzowane wyposażone w system wizyjny 6](#_Toc410810666)

[3.1 Elementy stanowiska 6](#_Toc410810667)

[3.2 Metody komunikacji 7](#_Toc410810668)

[3.3 Systemy wizyjne 8](#_Toc410810669)

[4 Opis Oprogramowania 9](#_Toc410810670)

[4.1 Program komputerowy 9](#_Toc410810671)

[4.1.1 Graficzny interfejs użytkownika 9](#_Toc410810672)

[4.1.2 Proces przetwarzania obrazu 10](#_Toc410810673)

[4.1.2.1 Pobieranie obrazu z kamery 10](#_Toc410810674)

[4.1.2.2 Binaryzacja obrazu 11](#_Toc410810675)

[4.1.2.3 Erozja/Dylatacja 14](#_Toc410810676)

[4.1.2.4 Rozpoznawanie obiektów 16](#_Toc410810677)

[4.1.3 Komunikacja ze sterownikiem robota 17](#_Toc410810678)

[4.2 Zaprogramowanie Sterownika Robota 18](#_Toc410810679)

[5 Testy Aplikacji 20](#_Toc410810680)

[5.1 Plan testów 20](#_Toc410810681)

[5.2 Przebieg testów 20](#_Toc410810682)

[5.3 Wyniki testów aplikacji 24](#_Toc410810683)

[6 Podsumowanie 25](#_Toc410810684)

[7 Bibliografia 26](#_Toc410810685)

# Wstęp

Rozwój przemysłu i powszechna automatyzacja stawiają coraz większe wyzwania projektantom stanowisk zrobotyzowanych. Coraz popularniejsze staje się przeprowadzanie procesów przemysłowych z minimalnym lub nawet zerowym udziałem człowieka. Autonomiczne stanowiska zrobotyzowane wyposaża się zatem w systemy detekcji wizyjnej, które umożliwiają szeroko rozumianą analizę wizyjną otoczenia i podjęcie decyzji na jej podstawie.

Systemy wizyjne znajdują zastosowanie w coraz liczniejszych aplikacjach zrobotyzowanych. Wykorzystuje się je do określenia położenia i orientacji obiektu manipulacji np. w celu paletyzacji lub pakowania. W zrobotyzowanym spawaniu systemy wizyjne rozpoznają spawany obiekt, opracowują trajektorie palnika robota i dokonują oceny jakości spoiny. Uzupełnianie ubytków w półproduktach odbywa się poprzez dozowanie odpowiedniej ilości masy w miejsce ubytku. W tym przypadku system wizyjny lokalizuje ubytki i oblicza objętość masy uzupełniającej na podstawie wykonanego zdjęcia. Te niektóre z wielu zastosowań dobitnie pokazują, jak szerokie możliwości posiadają zrobotyzowane stanowiska produkcyjne wyposażone w detekcje wizyjną.

Współczesne czujniki wizyjne mają za zadanie nie tylko wykonanie zdjęcia otoczenia. Nowoczesne modele kamer takie jak modele z rodzinny SENSOPART VISOR Color lub OMRON FQ2 poza uchwyceniem obrazu przeprowadzają jego obróbkę, lokalizują charakterystyczne elementy obrazu, pełnią rolę serwera w modelu komunikacji Klient-Serwer~~.~~ Po zaprogramowaniu w przyjaznych użytkownikowi interfejsach graficznych, mogą zostać odłączone od komputera i pracować samodzielnie w stanowisku zrobotyzowanym.

# ****Cel i zakres prac****

Niniejszy rozdział poświęcony jest celom i założeniom, które powinny zostać zrealizowane w ramach niniejszego projektu.

## ****Cele projektu****

Celem niniejszej pracy było opracowanie oprogramowania umożliwiającego komunikację między kamerami, komputerem i sterownikiem robota. Opracowany został algorytm detekcji wizyjnej, w którym zastosowano metody przetwarzania obrazu aby w kolejnym kroku zastosować funkcje lokalizacji punktów charakterystycznych na odpowiednio obrobionym obrazie. Następnie zrealizowano połączenie pomiędzy komputerem a kontrolerem robota i sieciową kamerą przemysłową. Zakres prac zawiera następujące kwestie:

* Nauka programowania robota przemysłowego Adept SIX300 (z pomocą literatury [1,2].
* Nauka obsługi sieciowej kamery przemysłowej.
* Zdobycie wiedzy na temat cyfrowego przetwarzania obrazów cyfrowych.
* Zapoznanie się z tematem sieci przemysłowych i metodami komunikacji.
* Przegląd literatury na temat języka programowania C# [3].
* Napisanie programu pozwalającego na pobranie obrazu z kamery przemysłowej, cyfrową obróbkę zdjęcia, zlokalizowanie elementów charakterystycznych i przesłanie współrzędnych tych elementów do sterownika robota przemysłowego.
* Zaprogramowanie kontrolera robota przemysłowego tak, aby był w stanie odebrać współrzędne punktów charakterystycznych.
* Zaprojektowanie stanowiska i testy aplikacji.

## ****Wykorzystane elementy****

Podczas realizacji projektu wykorzystano:

* Sieciową kamerę przemysłową EDIMAX IC-7100P zilustrowany na rys.1.



Rys. 1: Kamera EDIMAX IC-7100P

* Robota przemysłowego Adept SIX300 wraz z kontrolerem zilustrowany na rys.2.



Rys.2: Ramie robota Adept SIX300

* Komputer z systemem operacyjnym Windows 7 wyposażony w oprogramowanie Visual Studio Express
* Router

# ****Stanowiska zrobotyzowane wyposażone w system wizyjny****

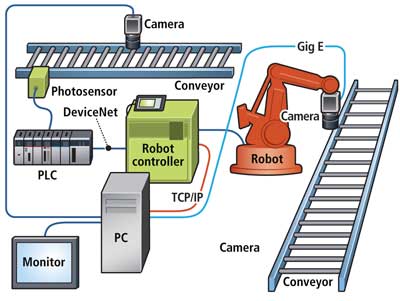
Rozdział został poświęcony opisowi stanowisk zrobotyzowanych wyposażonych w system wizyjny. Opisane zostały metody komunikacji wewnątrz takiego stanowiska. Poruszony został także problem metod przetwarzania obrazów i określania lokalizacji detali w systemach wizyjnych.

## ****Elementy** **stanowiska****

Podstawowymi elementami stanowiska zrobotyzowanego wyposażonego w system wizyjny są[4]:

* kamera,
* komputer,
* kontroler robota,
* robot przemysłowy.

Obraz otoczenia pozyskiwany jest przez kamerę lub czujnik wizyjny, następnie przetwarzany przez procesor lub komputer z odpowiednim oprogramowaniem. Wynik analizy przesyłany jest dalej, do kontrolera robota, który na ich podstawie porusza ramieniem robota.



Rys.3. Przykładowe stanowisko zrobotyzowane wyposażone w system wizyjny

Rys.3 przedstawia elementy przykładowego stanowiska zrobotyzowanego wyposażonego w system wizyjny. Powyższe stanowisko zawiera dodatkowo sterownik PLC, komunikujący się ze sterownikiem robota przy pomocy protokołu Device/Net. W praktyce, wyposażenie analogicznego stanowiska można mnożyć w celu zwiększenia jego możliwości a  w konsekwencji uzyskania niemal całkowicie autonomicznego procesu przemysłowego.

## ****Metody komunikacji****

W celu komunikacji pomiędzy urządzeniami cyfrowymi stosuje się sieci przemysłowe. Wymaga się od nich przede wszystkim dużej niezawodności a efektem pracy urządzeń w sieci przemysłowej jest posiadanie wiarygodnych informacji dotyczących procesu przemysłowego.

Wiadomości w sieciach przemysłowych są krótkie (mają niewielki rozmiar) ale wysyłane są często, w przeciwieństwie do sieci biurowych, ogólnych etc.

W przemyśle do komunikacji urządzeń pomiarowych stosowana jest rodzina protokołów cyfrowych Fieldbus, do której należą protokoły tworzone najczęściej przez producentów sprzętu przemysłowego takie jak Modbus, Profibus, CC-Link etc.

Obecnie jednak coraz bardziej popularne stało się wykorzystywanie w przemyśle rozwiązań typu Ethernet, który stopniowo wypiera z wielu aplikacji sieci własnościowe. Główną zaletą Ethernetu przemysłowego jest jego kompatybilność, efektywność i uniwersalność. Oferuje także stale rosnące prędkości transferu, nie generując przy tym dodatkowych kosztów. Ważną zaletą jest także fakt, że sieć pozwala na pracę z wieloma szybkościami transmisji. Prędkość przesyłu danych nie jest zatem ograniczona przez najwolniejsze z urządzeń. Umożliwia on komunikację przy użyciu różnych protokołów korzystających z tych samych łączy. Nie ma konieczności, aby każde z urządzeń miało możliwość obsługi wszystkich wykorzystywanych formatów danych. Jedynymi protokołami które są konieczne do przesyłania informacji to TCP i UDP.

Jedną z architektur proponowanych do komunikacji jest model TCP/IP, którego nazwa jest połączeniem dwóch protokołów wchodzących w jego skład – Transmission Control Protocol i  Internet Protocol. Powyższy model ma strukturę warstwową[5], jest zorganizowany w cztery koncepcyjne warstwy, które stanowią nadbudowę nad piątą warstwą sprzętu:

* Warstwa Programów Użytkowych – w tej warstwie generowane są dane użytkownika
* Warstwa Transportowa – zapewnia pewność przesyłanych danych dzięki wykorzystaniu portów określonych dla każdego połączenia
* Warstwa Intersieci – w warstwie ustalana jest odpowiednia droga do docelowego urządzenia w sieci
* Warstwa Interfejsu sieciowego – warstwa zajmuje się przekazywaniem danych przez fizyczne połączenia między urządzeniami

Kolejnym ważnym aspektem przy projektowaniu połączenia jest określenie modelu aplikacji, czyli wybór spośród modeli klient/serwer oraz peer-to-peer (każdy z każdym). Podstawową różnicą jest brak głównego serwera w tym drugim modelu.

## ****Systemy wizyjne****

Zaawansowane czujniki wizyjne wyposażone są w szereg detektorów, których zadaniem jest analiza scenerii i przesłanie danych niezbędnych do podjęcia decyzji dotyczącej ruchu ramienia robota. Detekcja cech obrazu jest pierwszą czynnością, która wykonywana jest przez system wizyjny. Podstawą metodą rozpoznawania jest porównywanie fragmentów obrazu ze zdefiniowanym wcześniej i zapisanym w bazie wzorcem.

Analiza obrazu polega na wykrywaniu charakterystycznych konfiguracji pikseli, które układają się w pewne struktury:

* Detektory krawędzi umożliwiają wyodrębnienie fragmentów obrazu, w których gwałtownie zmienia się intensywność składowej koloru,
* Detektory rogów działają analogicznie do detektorów krawędzi, ponadto wykrywają nagłe zmiany kierunku krawędzi,
* Detektory skupisk wyodrębniają jasne/ciemne obszary na ciemnym/jasnym tle

Ponadto, analiza obrazu umożliwia uzyskanie dokładnych informacji dotyczących scenerii takich jak:

* Jasność
* Kontrast
* Poziom Szarości

W celu określenia współrzędnych detalu, stosuje się pierwszą grupę detektorów. Detal kontrastujący z tłem zostaje wykryty dzięki jednej z powyższych metod. Następnie, odpowiednie algorytmy pozwalają na obliczenie środka ciężkości detalu i wysłanie jego współrzędnych do komputera lub do kontrolera robota.

Ważnym czynnikiem połączenia robot-kamera jest odpowiednia kalibracja[6]. Celem kalibracji jest wyznaczenie parametrów określających zależności między układem bazowym robota a układem związanym z kamerą, które występują łącznie z transformacją perspektywy oraz parametrów związanych z kamerą i układem optycznym. Znając te zależności, komputer lub kontroler robota jest w stanie przeliczyć współrzędne w układzie współrzędnych kamery na współrzędne w układzie bazowym robota co umożliwia pobranie detalu.

# ****Opis Oprogramowania****

Rozdział został poświęcony opisowi oprogramowania detekcji wizyjnej. Zostały poruszone tematy przetwarzania obrazów, lokalizacji znaczników na obrazie a także programowe rozwiązania problemów pozyskiwania obrazów z kamery IP a także komunikacji sterownika robota z komputerem na którym został uruchomiony program komputerowy.

## ****Program komputerowy****

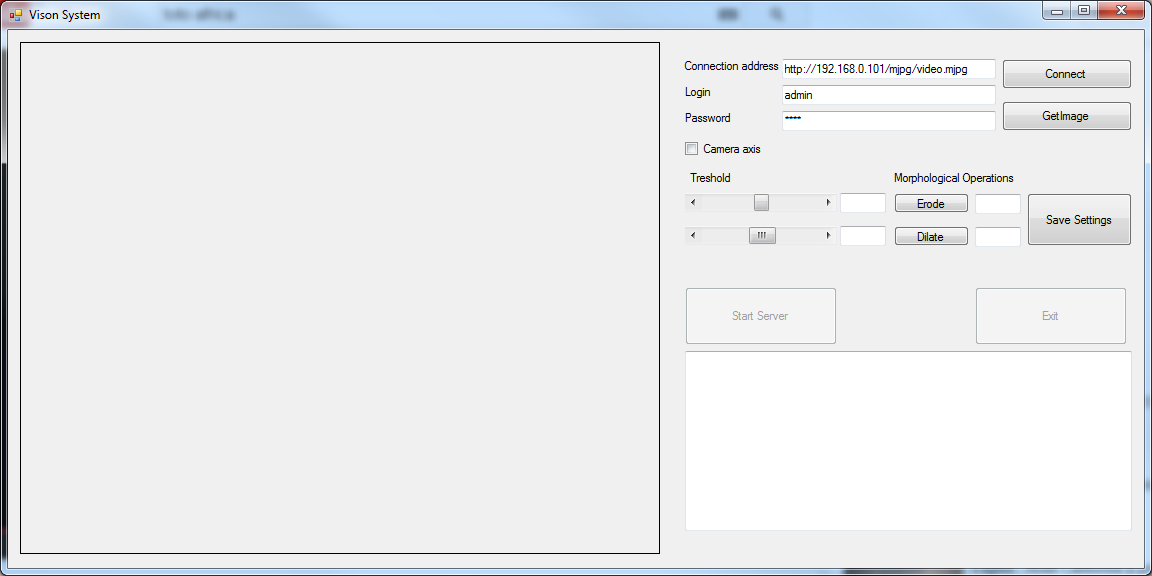
Głównym celem projektu było napisanie programu, który pełniłby następujące funkcje:

* Pozyskiwanie obrazu z kamery
* Przetwarzanie obrazu w celu wyodrębnienia znaczników
* Lokalizacja znaczników w układzie współrzędnych kamery

Ponadto, w modelu komunikacji klient-serwer, zgodnie z którym zostało zrealizowane połączenie, program pełniłby rolę serwera dla klienta ( kontroler robota ). Aplikacja została napisana w języku C# w środowisku Visual Studio Express. Aby uruchomić oprogramowanie, należy otworzyć plik Camera.exe znajdujący się na płycie załączonej do niniejszej pracy.

### ****Graficzny interfejs użytkownika****

Rysunek 4. przedstawia graficzny interfejs użytkownika.



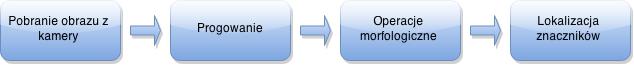
Rys.4: Graficzny interfejs użytkownika

Interfejs można podzielić na 4 segmenty:

* Część przeznaczona na wyświetlanie obrazu z kamery i obrazu przetworzonego
* Część odpowiadająca za komunikację z kamerą i przechwytywanie obrazu
* Ustawienia parametrów przetwarzania obrazu
* Obszar odpowiadający za komunikacje ze sterownikiem robota

### ****Proces przetwarzania obrazu****

W tym podpunkcie opisane zostały metody cyfrowego przetwarzania obrazu w celu uzyskania współrzędnych znaczników. Schemat poniżej przedstawia przebieg procesu[7]:

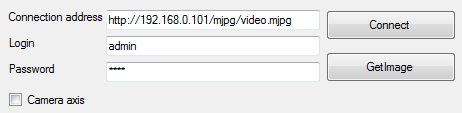


Rys. 5. Schemat blokowy procesu przetwarzania obrazu.

Pierwsze uruchomienie oprogramowania wymusza ręczne przeprowadzenie całego procesu. Stanowi to niejako naukę oprogramowania rozpoznawania obiektu zainteresowania użytkownika. Po przeprowadzeniu testu poprawności, ustawienia powinny być zapisane. Wówczas, kolejne procesy przetwarzania i rozpoznawania detali będą przebiegały automatycznie na każde zapytanie klienta. Kolejne podpunkty stanowią opis kolejnych kroków przetwarzania obrazów na przykładzie kolorowej mapy bitowej, zaprezentowanej na rysunku , załadowanej do programu z dysku komputera. Przykłady zastosowania aplikacji z obrazem przechwyconym z kamery znajdują się w dalszej części pracy.

#### ****Pobieranie obrazu z kamery****

Rysunek 6. przedstawia fragment interfejsu odpowiedzialny za połączenie z kamerą IP. 3 pola tekstowe przeznaczone są na adres obrazu z kamery, login i hasło. Adres obrazu z kamery pozyskuje się bezpośrednio z podglądu widoku kamery w przeglądarce internetowej poprzez kliknięcie prawym przyciskiem myszy na widok kamery oraz wybranie opcji „Kopiuj adres url…”. Login i hasło do kamery, jeśli nie zostało zmienione, znajduje się w dokumentacji kamery. Po uzupełnieniu danych należy wcisnąć przycisk „Connect” aby wyświetlić w obszarze aplikacji obraz z kamery. Aktualna klatka jest przechwytywana i zapisywana do przetwarzania przyciskiem „Get Image”.



Rys.6. Dane kamery sieciowej

Wykorzystana została klasa języka C# MJPGStream, dzięki której obraz z określonego adresu URL zostaje pobrany do strumienia MJPG.

#### ****Binaryzacja obrazu****

Pierwszym etapem przetwarzania obrazu jest przekonwertowanie obrazu kolorowego do skali szarości. Dla każdego piksela obrazu oblicza się współczynnik iluminacji zgodnie ze wzorem:

*Y’ = 0.2126R’ + 0.7152G’ + 0.0722B’* [8]

Jako ważoną sumę komponentów Red, Green i Blue. Powyższe współczynniki zostały dobrane tak, aby jak najlepiej oddać naturę pracy ludzkiego oka, które jest najbardziej wrażliwe na kolor zielony, najmniej natomiast na niebieski.

Następnie odbywa się progowanie, którego celem jest uzyskanie obrazu o dwustanowych wartościach pikseli gdzie jeden stan reprezentuje detal a drugi tło. Dokonuje się tego zgodnie ze wzorem progowania:

*0 dla f (i, j) < T - t*

*g (i, j) = 1 dla T - t < f (i, j) < T + t*

*0 dla f (i, j) > T + t*

Gdzie:

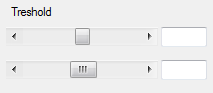
g(i,j) to odpowiednie piksele binarnego obrazu

f(i,j) to piksele obrazu w skali szarości

T – wartość środkowa progu binaryzacji

t – połowa szerokości progu

Rysunek 7 przedstawia fragment interfejsu przeznaczony do konfiguracji parametrów binaryzacji i progowania:



Rys.7: Suwaki do ustawiania parametrów progowania

Górny suwak służy do wyboru wartości progu spośród wartości 0 – 255. Dolny suwak pozwala określić szerokość progu 0 – 40. Wartości parametrów są wyświetlane w polach tekstowych.

Rysunkek 8. przedstawiają kolorowy obraz poddawany procesom przetwarzania. W skład obrazu wchodzą 4 koła na białym tle. Każde koło ma unikalny kolor zgodnie z Tabelą 1.

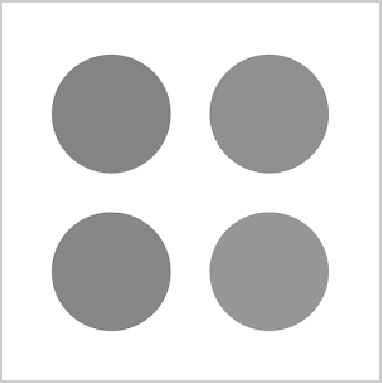


Rys. 8: Obraz przed procesem przetwarzania

|  |  |  |  |
| --- | --- | --- | --- |
| Koło | Składowa R | Składowa G | Składowa B |
| Niebieskie | 26 | 181 | 162 |
| Zielone | 82 | 193 | 58 |
| Różowe | 255 | 82 | 95 |
| Pomaranczowe | 255 | 124 | 6 |

Tabela 1: Składowe wartości barw kół

Rysunek 9. przedstawia obraz przekonwertowany do skali szarości. Tabela 2. Rozszerza tabelę 1. O wartości poziomu szarości obliczonego zgodnie ze wzorem.

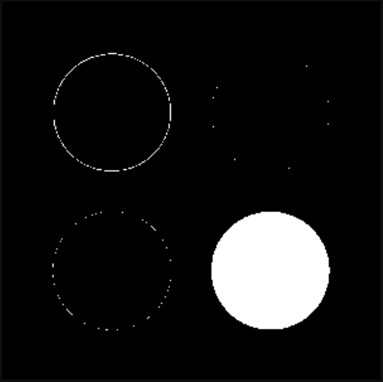


Rys. 9. Obraz w skali szarości

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Koło | Składowa R | Składowa G | Składowa B | Odcień Szarości |
| Niebieskie | 26 | 181 | 162 | 146.6752 |
| Zielone | 82 | 193 | 58 | 159.6544 |
| Różowe | 255 | 82 | 95 | 119.7184 |
| Pomaranczowe | 255 | 124 | 6 | 143.331 |

Tabela 2. Wartość odcienia szarości dla każdego z kół.

Rysunek 10 Przedstawia obraz progowany z parametrami T = 135, t = 10.



Rys.10. Obraz binarny

Powyższe parametry definiują próg o wartości minimalnej równej T-t=135-10=125 natomiast maksymalnej równej T+t=135+10=145. Spośród czterech kół, tylko to kolorze, którego wartość w skali szarości równa jest 143.331 mieści się w progu. Z tego względu, na  rogowanym obrazie, fragment obrazu objętego tymże kołem, został wypełniony pikselami o wartości 1. Barwy fragmentów objętych pozostałymi kołami nie mieszczą się w progu, dla tego ostatecznie przyjmują wartość 0.

#### ****Erozja/Dylatacja****

Mimo przeprowadzenia binaryzacji, obraz powinien być poddany dalszej obróbce. Aby pozbawić obraz niechcianych pojedynczych pikseli należy zastosować operacje morfologiczne[9].

Jednym z określeń towarzyszących operacjom morfologicznym jest sąsiedztwo. Może być ono wykorzystane do określenia, czy dwa punkty obrazu znajdują się obok siebie. Najczęściej wykorzystuje się sąsiedztwa czetrospójne lub ośmiospójne. W sąsiedztwie centralnego piksela definiuje się element strukturalny tzw. maskę, która wykorzystywana jest w operacjach morfologicznych.

Podstawowymi operacjami morfologicznymi są dylatacja (rozszerzanie) i erozja (zwężanie). Są to dualne operacje, które przykładają element strukturalny do każdego piksela na obrazie. W przypadku dylatacji, jeśli choć jeden piksel z sąsiedztwa punktu centralnego ma taką samą wartość równą 1 jak przynależny mu piksel elementu strukturalnego, wówczas wartość piksela centralnego zmienia się na 1. W przypadku erozji, jeśli choć jeden punkt w zadanym sąsiedztwie ma wartość 0, to punkt centralny przyjmuje wartość 0. Operacja dylatacji ma na celu wypełnienie „dziur” w niespójnym obrazie, konsekwencją jej jest zwiększenie obiektu. Erozja, jako operacja odwrotna do dylatacji służy do usunięcia z obrazu małych obiektów lub wąskich gałęzi, w konsekwencji zmniejszając obiekt. Przeprowadzenie procesu dylatacji zaraz po erozji nazywane jest otwarciem. Odwrotny proces natomiast domknięciem. W projekcie zastosowano maskę 3x3 wypełnioną ‘1’ jako element strukturalny.

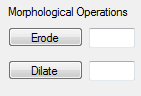
Erozja obrazu A przez element strukturalny B jest opisana przez:

A  \ominus B = \bigcap_{b\in B} A_{-b}

Dylatacja Obrazu A przez element strukturalny B jest opisana przez:

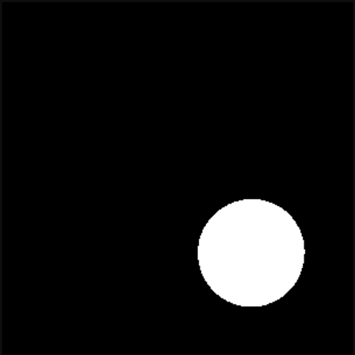
A  \oplus B = \bigcup_{b\in B} A_b.

Rysunek 11 przedstawia fragment interfejsu użytkownika służący do przeprowadzania operacji morfologicznych na binarnym obrazie.



Rys. 11: Przyciski erozji i dylatacji

W polach tekstowych wyświetlana jest ilość przeprowadzonych operacji. Rysunek 12. przedstawia obraz binarny po jednokrotnym przeprowadzeniu erozji.



Rys. 12: Obraz po jednokrotnej erozji

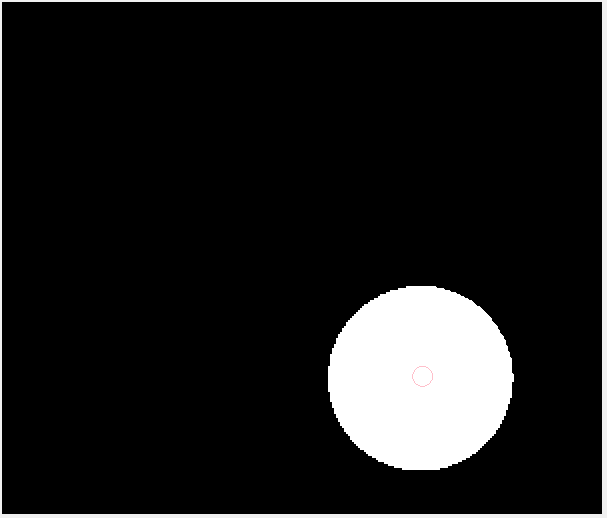
#### ****Rozpoznawanie obiektów****

W tej pracy zdecydowano się na wykorzystanie detekcji skupisk jako metody na  określenie współrzędnych punktów charakterystycznych obrazu. Wykorzystano w tym celu klasę BlobCounter, która zlicza pojedyncze obiekty na obrazie i przechowuje w pamięci ich parametry m.in. współrzędne środka ciężkości. Kliknięcie przycisku „Test” uruchamia funkcje wykrywania skupisk. Rysunek 13 przedstawia fragment obszaru tekstowego w którym wypisane zostają współrzędne środka ciężkości znalezionego elementu.



Rys. 13. Współrzędne znalezionego obiektu

W obszarze obrazu aplikacji, okręgiem zostaje oznaczony środek ciężkości znalezionego obiektu(Rys.14.).

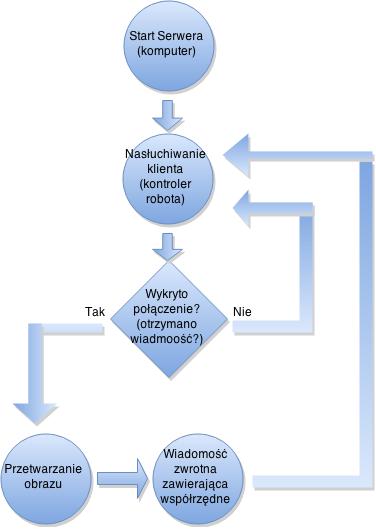


Rys. 14: Zaznaczony środek ciężkości znalezionego elementu

W razie poprawnego odnalezienia obiektu, ustawienia procesów przetwarzania obrazu należy zapisać poprzez kliknięcie w przycisk „Save settings”. Zostanie wówczas odsłonięta możliwość uruchomienia serwera. Od tego momentu, wszystkie procesy przetwarzania obrazu przebiegać będą automatycznie. W przypadku niepoprawnego znalezienia obiektu należy powtórnie przystąpić do doboru parametrów procesów przetwarzania obrazu.

### ****Komunikacja ze sterownikiem robota****

Aplikacja komputerowa, która jest przedmiotem tej pracy, w architekturze Klient-Serwer, w której zostało zrealizowane połączenie pomiędzy sterownika robota a komputerem, pełni rolę nadrzędną wobec sterownika robota. Wykorzystano klasę TcpListener, która nasłuchuje połączeń od klientów sieci TCP. W razie nadejścia połączenia, program czeka na wiadomość, której odebranie rozpoczyna proces przetwarzania obrazów opisany w punkcie 4.1.2. W odpowiedzi wysyłana jest wiadomość zwrotna, zawierająca współrzędne znalezionych obiektów. Proces został zaprezentowany na schemacie blokowym zilustrowanym na rys.15.



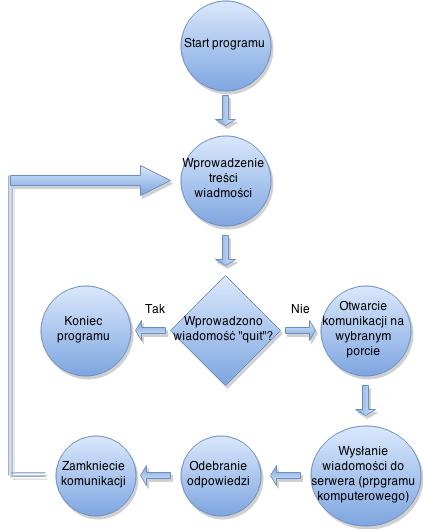
Rys. 15. Schemat blokowy serwera aplikacji

## ****Zaprogramowanie Sterownika Robota****

Sterownik robota w architekturze Klient-Serwer, pełni rolę podrzędną wobec aplikacji komputerowej. W komunikacji jego zadaniem jest połączenie z serwerem, wysłanie zapytania, oczekiwanie na odpowiedź. Następnie wiadomość zwrotna jest analizowana – pozyskiwane są z niej współrzędne wykrytego obiektu w układzie współrzędnym kamery. Przedstawiono fragment kodu w języku V+ odpowiedzialnego za utworzenie komunikacji.

|  |
| --- |
| ATTACH (lun, 4) "TCP"  FSET (lun) "/NODE 'SERVER2' /ADDRESS 172 16 150 1"  FOPEN (lun, 0) "SERVER2/REMOTE\_PORT 1234/BUFFER\_SIZE 1024"  status = IOSTAT(lun)  IF status < 0 THEN  TYPE "Error", $ERROR(status)  END |

Utworzony został obiekt nazwany „SERVER2” o adresie IP karty sieciowej komputera na którym uruchomione zostało oprogramowanie detekcji wizyjnej. Następnie, zostało otworzone połączenie z portem 5004, wewnątrz którego odbywa się komunikacja. Rysunek 16 przedstawia schemat blokowy oprogramowania sterownika robota adept.



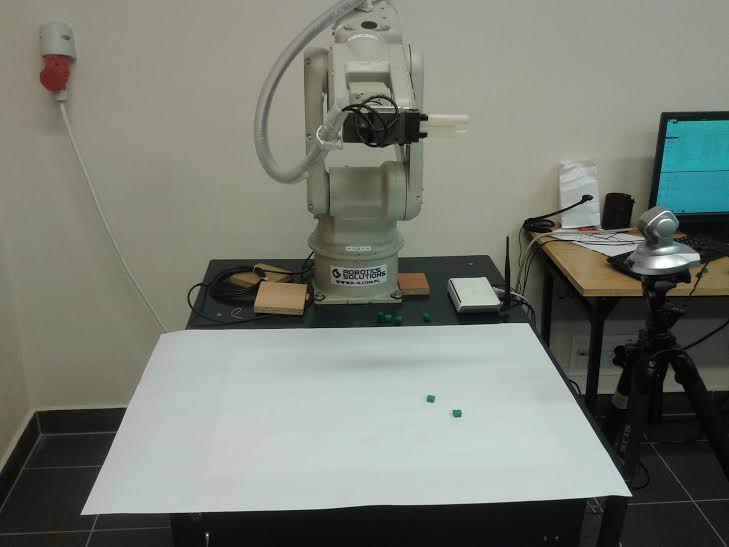
Rys. 16: Schemat blokowy programu robota

# ****Testy Aplikacji****

W tym rozdziale zostały opisane testy oprogramowania, które jest przedmiotem tej pracy.

## Plan testów

Stacja testowa została zilustrowana na rysunku 17. Sterownik robota został podłączony do komputera. Kamera została podłączony przez router do innej karty sieciowej w komputerze.

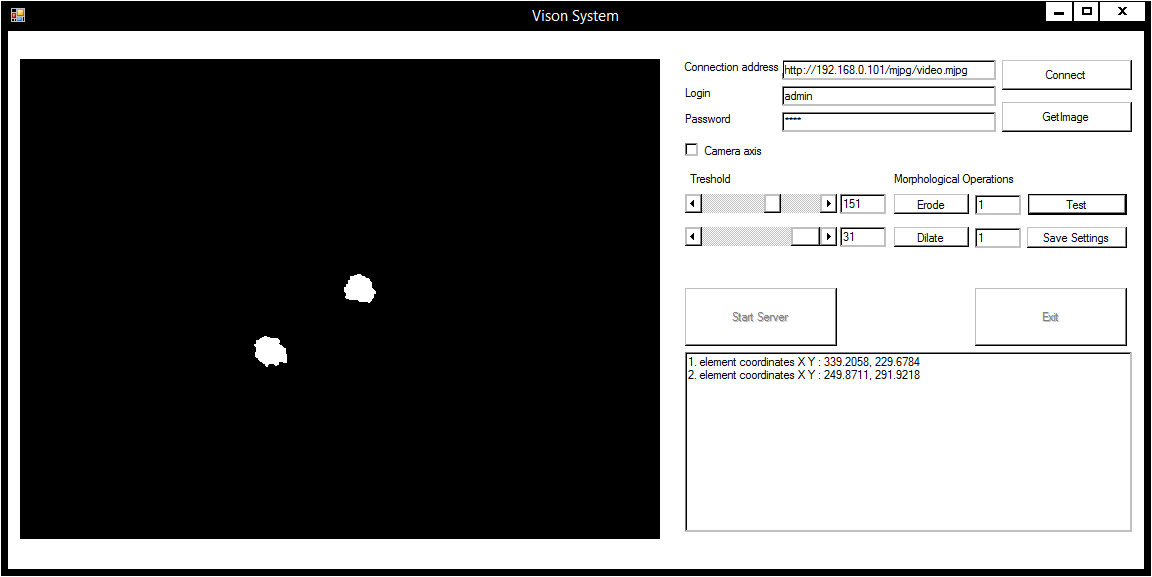


Rys. 17: Stanowisko wyposażone w kamerę

Przebieg testów zakłada wykonanie procedury nauczania, a następnie uruchomienie serwera systemu wizyjnego i przeprowadzenie kilku zapytań o współrzędne. Zapytania można wysyłać poprzez monitor oprogramowania robota przez użytkownika lub w określonych odstępach czasu. W obszarze robota ruchu, dwa charakterystyczne elementy zostały wprowadzone. Później ilość punktów charakterystycznych zostanie zwiększona.

## Przebieg testów

Po pierwsze, została nawiązana komunikacja z kamerą, następnie przeprowadzono procedurę nauczania. Rysunek 18 przedstawia interfejs użytkownika po procedurze nauczania.



Rys. 18: Interfes użytkownika po procedurze nauczania

Jak widać na rysunku 18, parametry przetwarzania obrazu wybrano w następujący sposób:

• T = 151,

• t = 31,

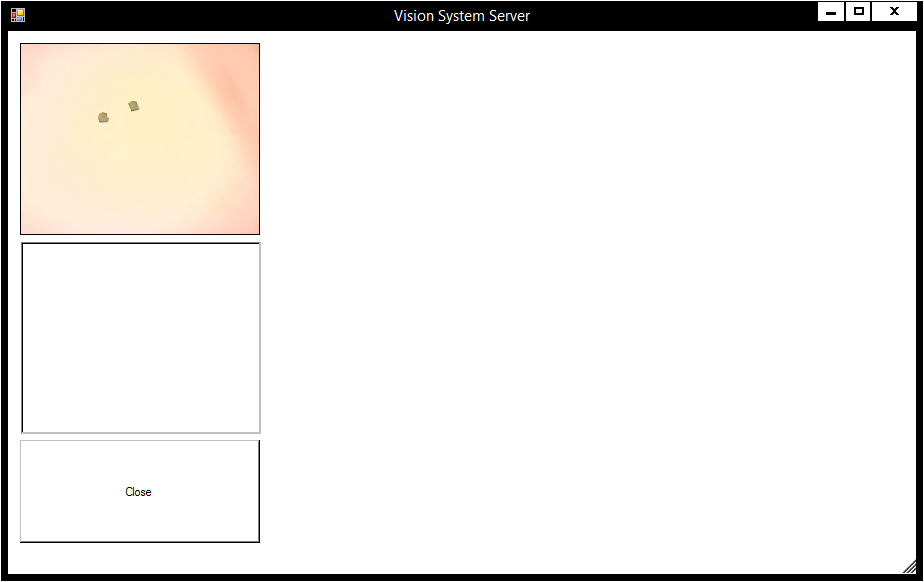
• Liczba erozji - 1,

• Liczba dylatacji - 1.

Parametry te pozwalają na uzyskanie obrazu binarnego z jedną parą punktów charakterystycznych o następujących współrzędnych:

* Współrzędne XY pierwszego punktu charakterystycznego: 339.2085, 229.6784,
* Współrzędne XY drugiego punktu charakterystycznego: 249.8711, 291.9218.

które zostały wypisane w polu tekstowym po naciśnięciu przycisku "Test". To koniec procesu nauczania. Po tym, serwer systemu wizyjnego może być uruchomiony poprzez przycisk "Start Server". Rysunek 19 przedstawia zrzut ekranu interfejsu serwera systemu wizyjnego po inicjalizacji.



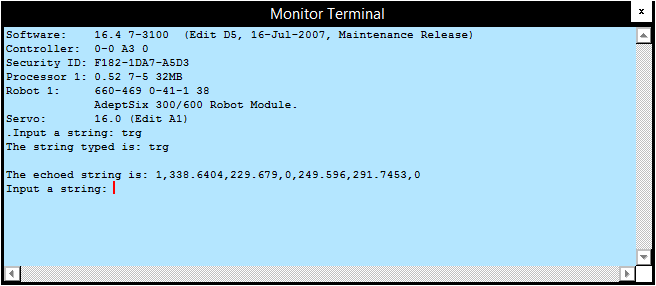
Rys. 19: Serwer systemu wizyjnego

W lewym górnym rogu, wyświetlany jest bieżący widok z kamery. Poniżej znajduje się pole tekstowe poświęcone współrzędnym elementów charakterystycznych. Ponieważ serwer jest uruchomiony, czeka na zapytania klientów. Rysunek 20 przedstawia Monitor Terminal oprogramowania związanego z oprogramowaniem Adept Desktop połączonego do kontrolera robota. Funkcja communication() została uruchomiona. Na monitorze zostaje wyświetlona prośba o treść wiadomości i ponieważ nie ma komunikatu dotyczącego błędu, oznacza to, że komunikacja została prawidłowo osiągnięta. Wiadomość "Trg" (jak trigger - wyzwalacz) została wysłana do serwera systemu wizyjnego. Została odebrana wiadomość zawierająca współrzędne punktów charakterystycznych. Program komputerowy jest przystosowany do wysyłania współrzędych pary elementów charakterystycznych. Dlatego wiadomości są przesyłane w postaci przedstawionej wzorem:

1, x1, y1,z1, x2, y2, z2 … N/2, x(N-1), y(N-1), z(N-1), xN, yN, zN

Gdzie N oznacza liczbę znalezionych elementów.

Teraz monitor czeka na inną wiadomość od użytkownika, co zilustrowano na rysunku 20.



Rys. 20: Monitor Terminal

Server systemu wizyjnego, po otrzymaniu zapytania, wyświetla bieżącą klatkę w postaci obrazu binarnego, w polu tekstowym współrzędne charakterystycznych punktów zostały wypisane, zilustrowano na rysunku 21.

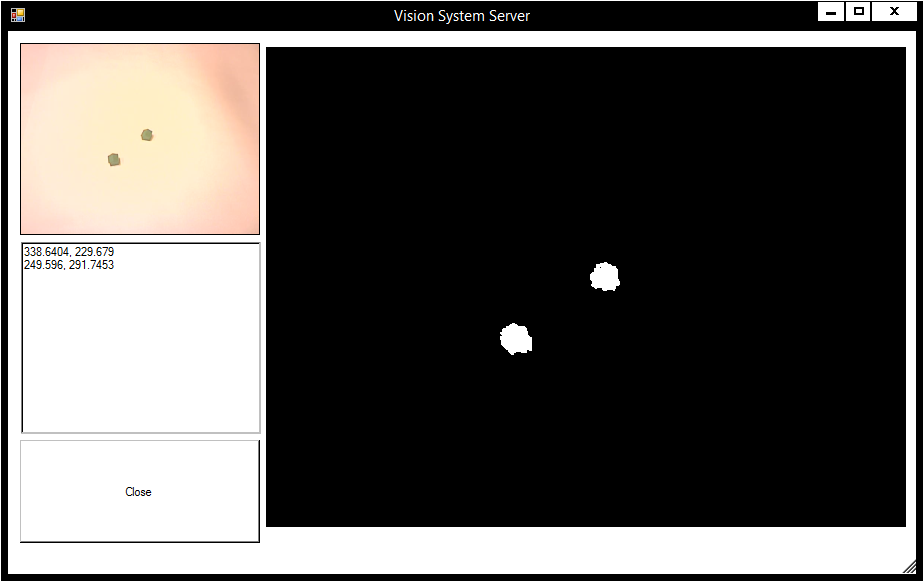


Fig. 21: Zrzut ekranu z serwera systemu wizyjnego z bieżącą klatką, współrzędnymi punktów charakterystycznych i obrazem binarnym.

Zaobserwowano różnice w wartościach współrzędnych znalezionych obiektów pomiędzy (rysunki 18 i 21). Różnice wynoszą od -0.006 do 0.5681 piksela. Prawdopodobnie zostało to spowodowane niezauważalną zmianą pozycji kamery. Nie mniej jednak, dokładność algorytmu lokalizacji znaczników powinna zostać zbadana.

Przeprowadzono kolejny test. Wyzwalanie zdjęcia nie zostało wywołane przez użytkownika, ale zapytania były wysyłane do serwera w 5-sekundowych odstępach. Rysunek 22 pokazuje Monitor Terminal natomiast rysunek 23 przedstawia serwer systemu wizyjnego po kilku wymianach wiadomości.

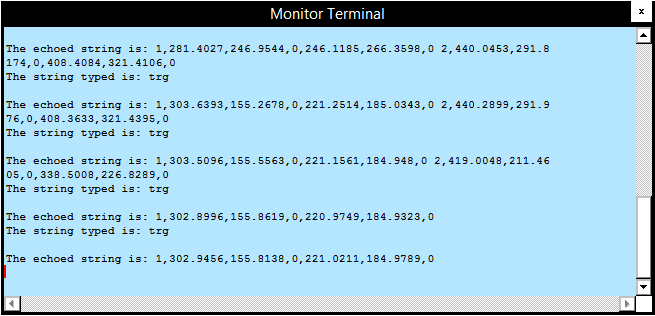


Fig. 22: Monitor Terminal po kilku wymianach wiadomości.

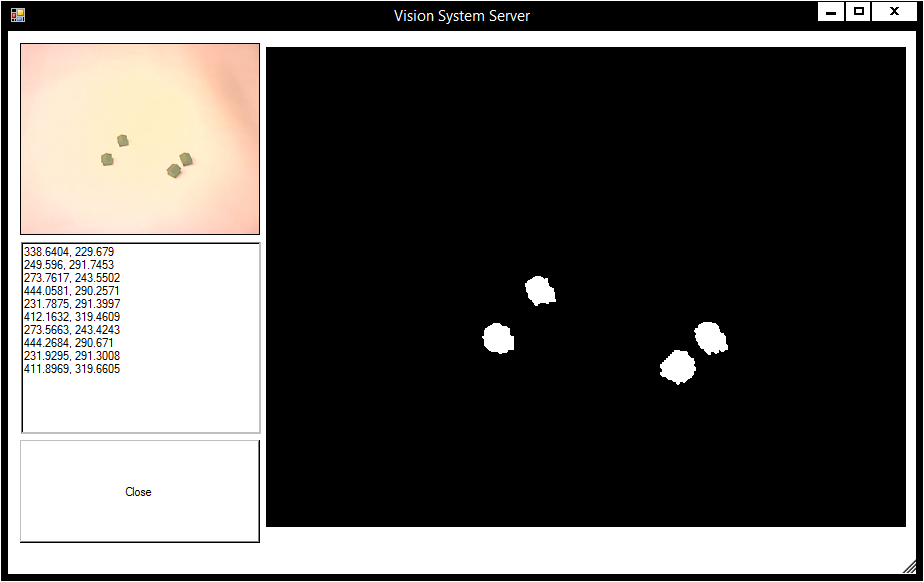


Fig. 23: Serwer systemu wizyjnego po kilku wymianach wiadomości

## Wyniki testów aplikacji

Podczas testów aplikacji, po procesie nauczania, kilka wiadomości zostało wymienionych pomiędzy sterownikiem robota i serwerem systemu wizyjnego. Przetwarzanie obrazu i lokalizacja elementu zostały wywoływane zarówno przez użytkownika za pośrednictwem Monitor Treminala oraz automatycznie, w 5-sekundowych odstępach. Ponadto, różne liczby elementów charakterystycznych umieszczono w obszarze ruchu robota. Okazało się, że maksymalna ilość pary znaczników jest równa 3. Jest to jest ograniczone z góry określonym rozmiaru komunikatu, który może być powiększony w kodzie programu. Wiadomości były wymieniane poprawnie, bez zauważalnej straty czasu. Nie było "rozdzielonych" elementów ani niepożądane pikseli co oznacza, że proces nauczania przeprowadzono w odpowiedni sposób. Dokładność algorytmu lokalizacji znaczników powinna zostać zbadana.

# ****Podsumowanie****

Systemy wizyjne są bardzo przydatne i stanowią coraz większą rolę w nowoczesnych procesach przemysłowych. Programowalne kamery to, w większości, bardzo zaawansowane systemy, które mogą kosztować znaczną ilość pieniędzy. Przedstawione rozwiązanie jest dostosowane do każdej kamery IP. Ponadto, może być stosowany nie tylko do robotów Adept, ale jest kompatybilne z wszystkimi robotami z Ethernetem jako opcję komunikacji.

Celem końcowym projektu było zaprojektowanie serwera systemu wizyjnego, który pozwoliłby na nabycie obrazu z kamery przemysłowej, wykonanie niezbędnego przetwarzania obrazu i wysłanie wyników do innego urządzenia. W tym samym czasie, sterownik robota powinien być zaprogramowany w celu wypytywania o współrzędne oraz ich zapisywanie.

Zadanie zostało zrealizowane z wykorzystaniem połączenia Ethernet w modelu architektury klient-serwer. Zostały użyte metody przetwarzania obrazów takie jak progowanie lub operacje morfologiczne. Funkcje języka C # były wykorzystywane do lokalizacji obiektów charakterystycznych.

Praca nad programem nie została jednak zakończona. Bez odpowiedniej kalibracji, nie ma żadnego związku między współrzędnymi w układzie współrzędnych kamery i bazowym układem współrzędnych robota. W obecnym rozwiązaniu, kalibracja powinna być wykonywana jako kolejny krok w programie kontrolera robota. Ostatecznie proces kalibracji może się odbywać wewnątrz oprogramowania wizyjnej detekcji i wysłane wiadomości będą nie tylko zawierały współrzędnych obiektu, ale będą stanowić pośrednie punkty trajektorii ruchu ramienia robota. Kalibracja nie jest przedmiotem tej pracy.

# Bibliografia

[1] *AdeptSix 300 Robot, Instruction Handbook 2005*

[2] P. Rotyński *Adept Robot Programming. Master Thesis. Institute of Automatic Control. Silesian University of Technology. Gliwice 2011.*

[3]L. O’Brien, B. Eckel *Thinking in C#. Prentice Hall PTR. Upper Saddle River, NJ 2002.*

[4] Zens, Richard *Vision-guided robot system trims parts. Vision Systems Design. PennWell Corporation. Tulsa, OK 2006.*

[5] R. Braden (ed.) *Requirements for Internet Hosts – Communication Layers, October 1989.*

Available on the internet: https://tools.ietf.org/html/rfc1122

[6] T. Szkodny, A.Meller, K.Palka *Accuracy of Determining the Coordinates of Points Observed by Camera. International Conference on Intelligence Robotics and Applications. Guangzhou, China 2014.*

[7] Davies, E.R. *Machine Vision - Theory Algorithms Practicalities (2nd ed.). Harcourt & Company, San Diego, Ca 1996*

[8] M. Stokes, M. Anderson, S. Chandrasekar, R. Motta *A Standard Default Color Space for the Internet – sRGB*

Available on the internet: http://www.w3.org/Graphics/Color/sRGB.html

[9] P. Soille *Morphological Image Analysis; Principles and Application. 2nd edition. Springer Verlag. Heidelberg, Berlin 2002.*