Politechnika Warszawska
Wydział Elektroniki i Technik Informacyjnych
Instytut Automatyki i Informatyki Stosowanej

Piotr Sakowicz

numer albumu 192780

Praca dyplomowa inżynierska

Automatyczna kalibracja systemu robot-kamera

Opiekun pracy
prof. nzw. dr hab. inż. Cezary Zieliński

Warszawa, czerwiec 2009

Quidquid agis, prudenter agas et respice finem!

Owidiusz[1]

[1] Cokolwiek robisz, rób roztropnie i patrz końca!

Pragnę podziękować mojemu opiekunowi
prof. nzw. dr. hab. inż. Cezaremu Zielińskiemu
za wsparcie merytoryczne oraz poświęcony mi czas.

Pragnę również podziękować
mgr. inż. Tomaszowi Kornucie
za pomoc i cenne wskazówki rzeczowe,
dzięki którym praca zyskała na jakości.

Streszczenie

Celem pracy dyplomowej jest projekt systemu do automatycznej kalibracji układu robot-kamera oraz jego implementacja, jak również opis zagadnień związanych z procesem kalibracji. Do kalibracji wykorzystywana jest płytka z oznaczonymi czterema kropkami. Przy użyciu kamery wykonywane są zdjęcia płyty, na podstawie tak wykonanych obrazów wyznaczane są współrzędne kropek w układzie kamery oraz dokonywane są obliczenia transformacji układów.

System automatycznej kalibracji obejmuje: moduł analizujący zdjęcia i wyszukujący kropki, moduł przeprowadzający minimalizację zadanej funkcji transformującej współrzędne oraz moduł sterowania i planowania ruchów robota automatyzujący kalibrację. System został zaimplementowany w MRROC++.

Słowa kluczowe: kalibracja, transformacja współrzędnych, kamera, robot, manipulator, rozpoznawanie wzorców, MRROC++.

Automatic calibration of robot - video camera system

Abstract

The aim of the thesis is to design an automatic robot-camera calibration system and implement as well as describe the calibration process. A plate with four marked dots is used for the calibration purposes. Photographs of the plate are taken with a video camera and on the basis of the photographs dots coordinates in the camera coordinate system are determined, then computation of coordinate systems transformation matrix is performed.

The automatic calibration includes: a module analysing photographs and searching for the dots, a module performing minimalistation of a given function transforming coordinates as well as a module controlling and planing movements of the robot, which automates the calibration. The system is implemented with the use of MRROC++.

Keywords: calibration, coordinates transformation, video camera, robot, manipulator, pattern recognition, MRROC++.

Spis treści

1 Wstęp

1.1 Cele i założenia pracy

Kiedy w 1932 roku Karel Čapek po raz pierwszy użył słowa *robot*, mało kto był w stanie przewidzieć, że rozwój techniki spowoduje, że tworzenie „sztucznych ludzi" przestanie być wyłącznie tematem powieści science-fiction. I choć marzenia o robotach sprzątających mieszkania, gotujących i opiekujących się dziećmi są naturalną konsekwencją dążenia do minimalizacji wysiłku, towarzyszącemu ludzkości od dawna, nie każdy zdaje sobie sprawę, jak wielki wpływ mają one już od kilkudziesięciu lat na niemal wszystkie gałęzie przemysłu.

Dziś roboty są wykorzystywane do wielu rzeczy. Oprócz codziennych zastosowań, jak odkurzanie, niezastąpiona jest pomoc tych urządzeń w eksploracji terenów niedostępnych dla człowieka, np. powierzchni Marsa, rozbrajaniu bomb. Jednak największy, choć pośredni, wpływ na nasze życie mają roboty przemysłowe. Znalazły one zastosowanie tam, gdzie powtarzalną pracę ludzką można z łatwością zautomatyzować i dzięki temu wykonywać praktycznie bezbłędnie 24 godziny na dobę.

Manipulatory przemysłowe mają zazwyczaj postać ramienia o kilku stopniach swobody. Do takich robotów należy m.in. IRp-6 wykorzystywany w niniejszej pracy. Wyposażony w dodatkową zewnętrzną kamerę może wykonywać liczne prace, np. segregować przedmioty pod względem kształtów czy kolorów.

Cel i motywacja

Aby manipulator znajdujący się w Pracowni Robotyki mógł wykonywać zlecone mu czynności potrzebna jest niezawodna kalibracja układu robot-kamera. Proces kalibracji polega na wyznaczeniu położenia i orientacji kamery w układzie bazowym robota. Przy każdej zmianie tych parametrów konieczna jest ponowna kalibracja układu. W sytuacji, gdy kamera wykorzystywana jest do różnych zadań, a w związku z tym często przemieszczana, pożądane jest, by jej ponowna kalibracja była zadaniem prostym i możliwie automatycznym. Jednak nawet w systemie z dedykowaną kamerą mogą zdarzyć się nieprzewidziane sytuacje, gdy zostanie ona nieuważnie lub świadomie przesunięta bądź obrócona.

Wyznaczenie wyżej wspomnianych współczynników do tej pory odbywało się w sposób niezautomatyzowany: na podstawie analizy zdjęć planszy odczytywane było jej położenie w układzie kamery, a następnie wyliczano współczynniki transformacji między dwoma układami. Takie rozwiązanie nie było jednak pozbawione wad. Dokładność wyznaczanych

ręcznie współczynników nie była stuprocentowa, a cały proces był skomplikowany.

Nadrzędnym celem pracy było zaprojektowanie i implementacja algorytmu automatycznej kalibracji układu robot-kamera, w taki sposób, aby znacznie uprościć ten proces przy jednoczesnym zwiększeniu jego dokładności.

Założenia

Z uwagi na fakt, że nie można przewidzieć wszystkich scenariuszy zachowań, należało przyjąć kilka ograniczeń w funkcjonowaniu systemu.

- Algorytm generacji trajektorii robota wykorzystuje mechanizm sterowania pozycyjno-siłowego, tzw. „wodzenia za nos", czyli pozostawia operatorowi dowolność ustalania położeń płytki w układzie kamery. Użytkownik musi więc zadbać o to, by położenia te obejmowały jak największy obszar pracy manipulatora.
- Położenie kamery nie będzie się zmieniać w czasie trwania kalibracji.
- System został zaprojektowany przy uwzględnieniu pewnych warunków zewnętrznych, takich jak kolorystyka i oświetlenie laboratorium oraz przy odpowiednich ustawieniach parametrów kamery, w szczególności długości ogniskowej (*zoom*), blokady ostrości, balansu bieli. Zmiana ich w sposób znaczący może spowodować konieczność wprowadzenia poprawek w pliku konfiguracyjnym.
- Po uruchomieniu procesu manipulator ustawia się w pozycji początkowej z odpowiednio rozwartymi szczękami tak, by możliwe było swobodne wsunięcie płytki jednocześnie zapobiegając wysunięciu się jej w trakcie pracy. Płytka musi zostać umieszczona w odpowiedni sposób: czerwona kropka musi być po prawej stronie bliżej mechanizmów ramienia. Dokładna instrukcja postępowania zamieszczona została w załączniku.

Zagadnienia omawiane w pracy

W dalszej części rozdziału 1 opisane zostały biblioteki i narzędzia użyte do implementacji systemu w pracowni oraz sprzęt laboratoryjny wykorzystywany podczas jego tworzenia.

W rozdziale 2 omówiono podstawy dwóch rodzajów transformacji układów współrzędnych mających zastosowanie w kalibracji układu robota z kamerą: transformację między równorzędnymi układami w przestrzeni $\mathbb{R}^3$ oraz tzw. rzut zbieżny. Rozdział ma głównie

charakter teoretyczny, jednakże opisując wspomniane przekształcenia odniesiono je do rzeczywistej sytuacji mającej miejsce przy procesie kalibracji.

W rozdziale 3 opisane zostały wszelkie czynności związane z wyznaczeniem położenia kropek. Omówiono dwie grupy zagadnień: analizę pojedynczych obrazów w celu znalezienia czterech kropek i wyznaczenia ich położenia na zdjęciu oraz obliczanie położenia kropek w układzie kamery przy użyciu autorskiej metody. Dodatkowo wspomniana metoda, bazująca na minimalizacji funkcji, została przetestowana, a wyniki opisane w tym rozdziale.

Rozdział 4 poświęcony został: przeglądowi metod wyznaczana kalibracji na podstawie odpowiednio zadanych danych pomiarowych, wraz ze wskazaniem metody wybranej do realizacji, scenariuszowi działań oraz wpływowi i eliminacji błędów na proces kalibracji. Przedstawiony został również interfejs aplikacji.

W ostatnim rozdziale podsumowano wykonane prace, opisano przykładowy proces kalibracji, refleksje autora powstające w trakcie tworzenia tej pracy oraz końcowe wnioski. Z uwagi na możliwość poprawy działania systemu oraz dostosowania go do nowego sprzętu, z jakim mógłby współpracować, przedstawione zostały propozycje modyfikacji programu celem m.in. zmniejszenia wpływu błędów.

W załącznikach podane zostały informacje dla przyszłych użytkowników systemu, tj. budowa pliku konfiguracyjnego, instrukcja użytkownika oraz zamieszczone zostały tabelki ze współczynnikami, których wyznaczenie było konieczne w trakcie trwania prac.

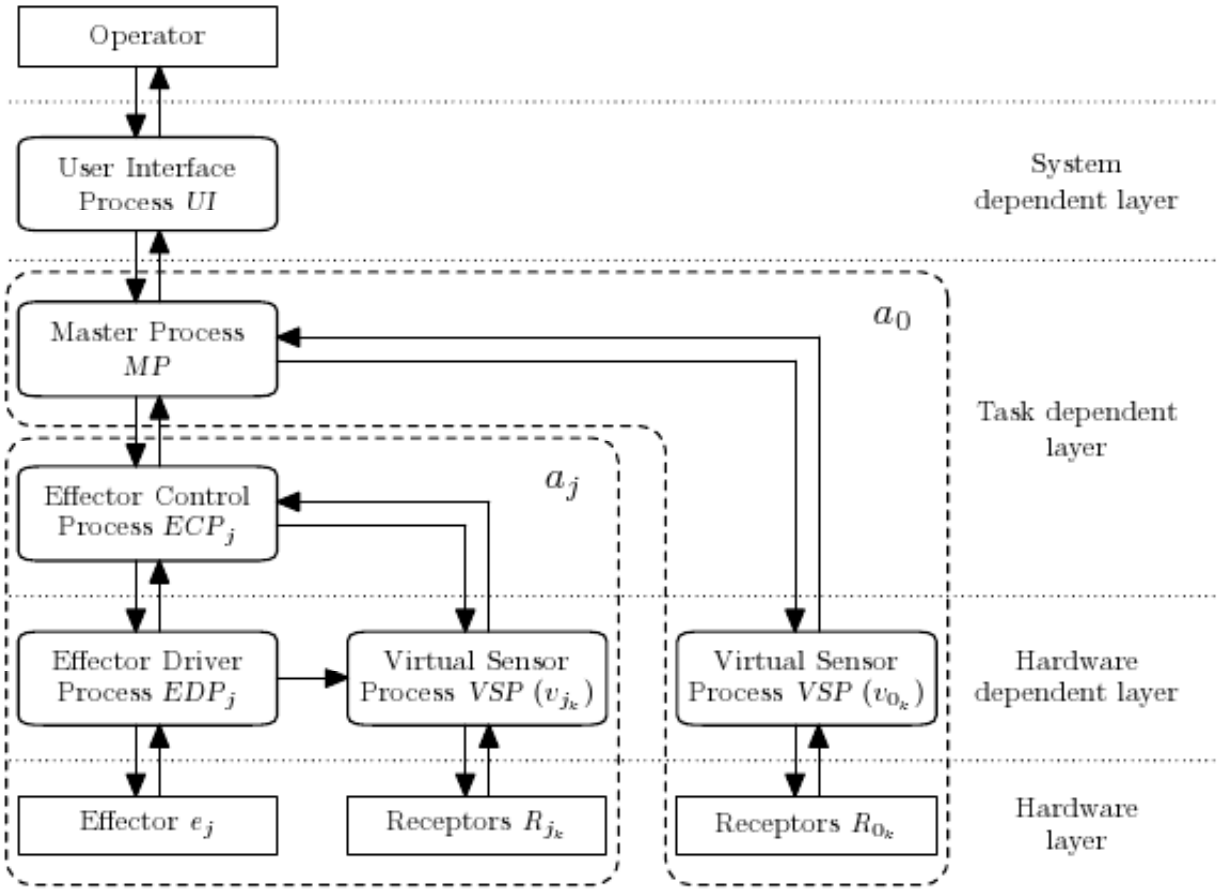
Z uwagi na to, że problem kalibracji układu laboratoryjnego wykorzystuje wiele pojęć z różnych dziedzin i niekiedy konieczne było wprowadzenie własnych oznaczeń i sformułowań, aby ułatwić czytanie pracy na początku niektórych rozdziałów umieszczono sekcje: Oznaczenia i definicje, gdzie przybliżono własne sformułowania wykorzystywane w rozdziale oraz Cele, gdzie krótko uzasadniono, np. sens wykonywania danych obliczeń lub pomiarów. Na końcu niektórych rozdziałów umieszczono sekcję Wnioski, w której podsumowano wyniki przeprowadzonych doświadczeń.

1.2 Biblioteki i narzędzia wykorzystane do napisania pracy

1.2.1 MRROC++ jako sterownik systemu wielorobotowego

MRROC++ (z ang. *Multi-Robot Research Oriented Controller*) jest hierarchiczną, rozproszoną strukturą ramową. Budowa sterownika opartego na MRROC++ została przedstawiona na poniższym rysunku[2].

[2]Rysunek pochodzi z [10]. System-, Task-, Hardware dependent- oraz Hardware Layer oznaczają odpowiednio warstwy zależne od struktury systemu, zadania, sprzętu oraz warstwę sprzętową. Procesy



Rysunek 1: Struktura systemu opartego na MRROC++.

Proces UI odpowiada za komunikację całego systemu z operatorem, podczas gdy pojedynczy proces MP koordynuje pracę pozostałych procesów sterownika. W systemie może współpracować kilka procesów ECP, które realizują sterowanie efektorami wykorzystywanymi w wykonywanym aktualnie zadaniu. Tak więc liczba działających ECP jest ściśle powiązana z liczbą efektorów i jest równa liczbie EDP, z których każdy pełni rolę sterownika efektora (kontroluje sprzęt skojarzony z konkretnym efektorem oraz rozwiązuje proste i odwrotne zadanie kinematyki). Procesy VSP są pośrednikami między rzeczywistym sprzętem powiązanym z danym sensorem (np. kamerą), a ECP lub MP - jego zadaniem jest agregacja i przetwarzanie danych odczytanych z czujników.

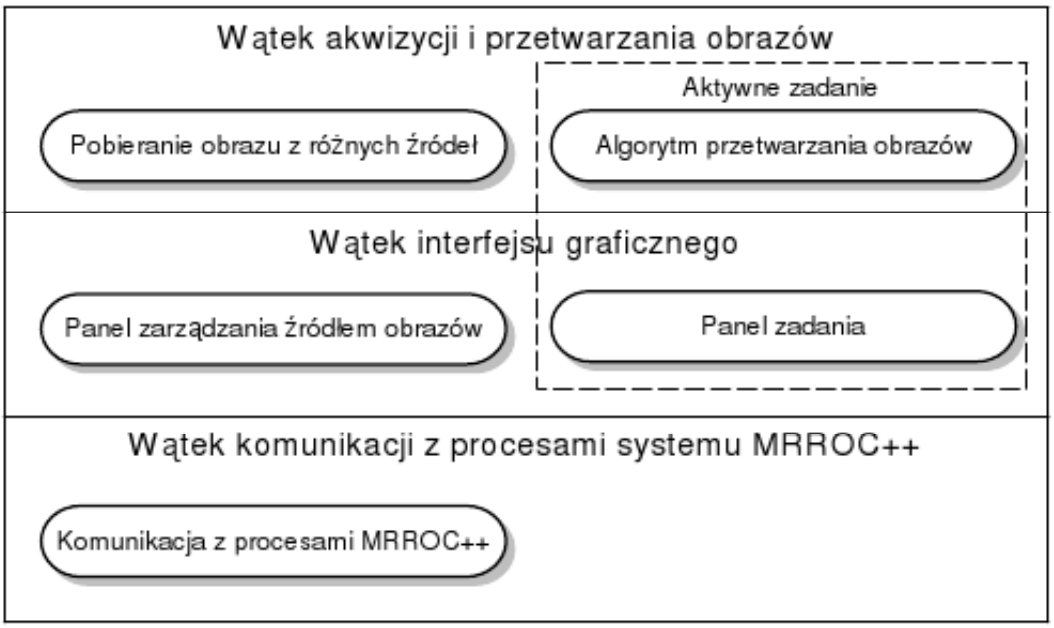
Sterownik stworzony na bazie struktury ramowej MRROC++ pracuje pod kontrolą systemu QNX, systemu operacyjnego czasu rzeczywistego.

$UI, MP, ECP_j, EDP_j, VSP$ oznaczają odpowiednio proces interfejsu użytkownika, koordynatora, kontroli efektora, sterownika efektora oraz proces wirtualnego czujnika.

1.2.2 FraDIA jako struktura ramowa do analizy i przetwarzania obrazów

FraDIA (z ang. Framework for Digital Image Analysis) jest uniwersalną programową strukturą ramową umożliwiającą budowę aplikacji przetwarzania obrazów. Została napisana w C++ i oparta na dwóch bibliotekach: OpenCV oferującą wiele algorytmów przetwarzania cyfrowych obrazów i opisaną dokładniej w kolejnym paragrafie oraz FLTK (ang. Fast, Light Toolkit) - międzyplatformową biblioteką stworzoną z myślą o prostym programowaniu interfejsów graficznych.

Struktura biblioteki FraDIA została przedstawiona na rysunku 2[3].



Rysunek 2: Struktura systemu wykorzystującego bibliotekę FraDIA.

FraDIA działa jako niezależna aplikacja, umożliwiając pracę jednego z zaimplementowanych algorytmów przetwarzania obrazów. W jednym wątku aplikacji zaimplementowany jest mechanizm komunikacji ze strukturą ramową MRROC++, co umożliwia jej pracę jako wirtualnego czujnika związanego z kamerą.

W typowym zadaniu FraDIA można wyróżnić część odpowiedzialną za analizę i przetwarzanie obrazów oraz interfejs graficzny. W procesie automatycznej kalibracji pozyskiwanie danych z kamery oraz przetwarzanie zdjęć realizowane jest przez pierwszą część, natomiast możliwość wykorzystania interfejsu użytkownika miała znaczenie głównie przy tworzeniu i testowaniu algorytmu. Wątek komunikacji z MRROC++ umożliwia odbiór

[3]Rysunek pochodzi z [9].

poleceń oraz przekazywanie wyników, np. położenia znalezionych kropek.

1.2.3 Inne ogólnodostępne biblioteki

W tym paragrafie opisane zostały niezależne biblioteki, które zostały użyte do implementacji systemu opisanego w tej pracy.

GNU Scientific Library

GNU Scientific Library (zwana w skrócie GSL) jest biblioteką do zastosowań numerycznych przeznaczoną dla programistów C i C++. Jest wolnym oprogramowaniem opartym na licencji GNU. W bibliotece zaimplementowano ponad 1000 funkcji, przy pomocy których możliwe jest rozwiązanie różnorakich probelmów matematycznych. Do zagadnień omawianych w pracy wykorzystano metody minimalizacji funkcji i rozwiązywania układów równań oraz struktury umożliwiające operacje na wektorach i macierzach.

OpenCV

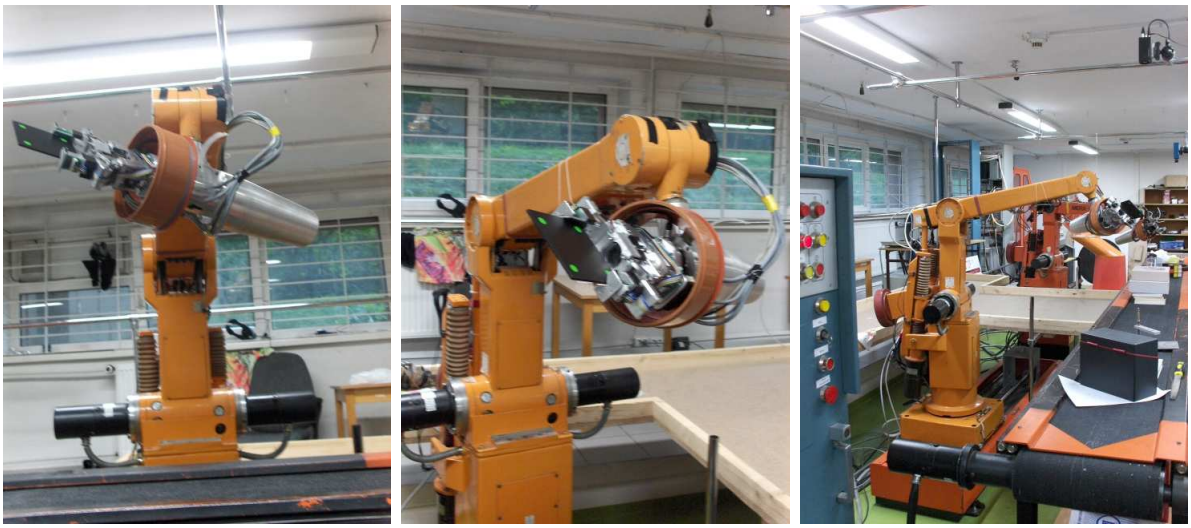
W oparciu o procedury Open Source Computer Vision Library zostały zaprojektowane liczne mechanizmy m.in. akwizycji obrazu z kamery, importu i eksportu obrazów zapisanych w plikach graficznych w strukturze ramowej FraDIA.

OpenCV jest wolnym oprogramowaniem (na licencji BSD), pierwotnie napisanym przez firmę Intel. Głównym celem autorów było stworzenie biblioteki do przetwarzania obrazów tam, gdzie niezbędna jest natychmiastowa analiza i pozyskiwanie informacji m.in. w środowiskach bliskiej współpracy człowieka z maszyną, np. w laboratoriach robotycznych, samochodach, itp.

1.3 Opis sprzętu laboratoryjnego i narzędzi wykorzystywanych w pracy

Manipulator IRp-6

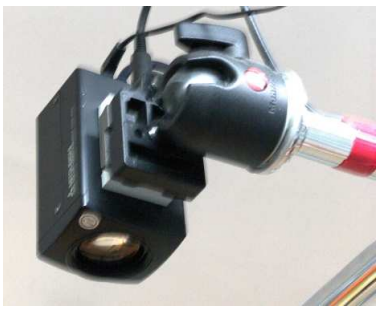
W Laboratorium Robotyki znajdują się dwa roboty IRp-6. Każdy z nich został wyposażony w dodatkowe stopnie swobody oraz specjalne chwytaki. Spośród wielu zadań, z którymi potrafią sobie radzić, warto wyróżnić segregację elementów przesuwających się po taśmociągu, grę w warcaby z człowiekiem oraz układanie kostki Rubika.



Rysunek 3: Manipulator w trakcie kalibracji.

Jeden z robotów (zwany *IRp-6 na torze*), przy użyciu którego tworzony był system, umieszczony jest na torze jezdnym i ma 7 stopni swobody. Drugi robot (zwany *IRp-6 na postumencie*) ma 6 stopni swobody.

Kamera cyfrowa

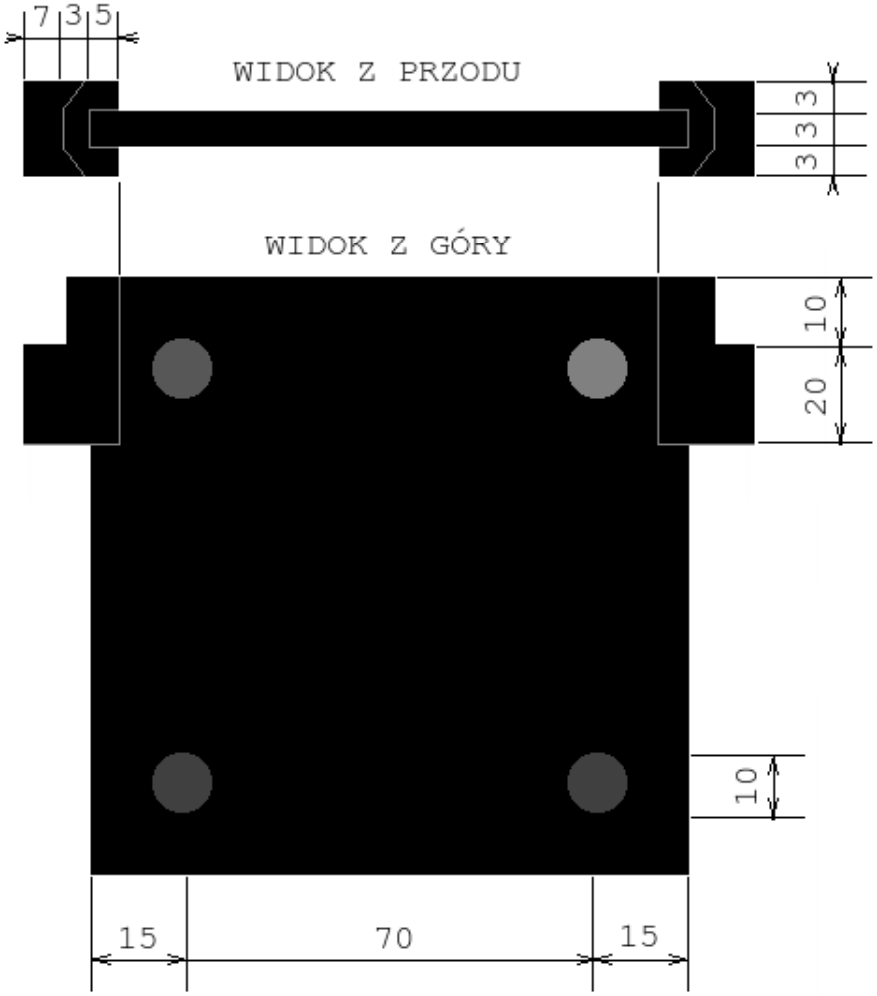


Nad każdym z manipulatorów umieszczone zostały kamery, z których obraz przekazywany jest za pomocą framegrabbera do komputerów. W pracy posłużono się kamerą Mitsubishi CCD-400, mimo to, po odpowiedniej modyfikacji parametrów w pliku konfiguracyjnym aplikacji odpowiadających ogniskowej oraz położeniu środka projekcji, możliwe jest dokonanie kalibracji robota z dowolną kamerą.

Rozmiar matrycy światłoczułej podany w specyfikacji technicznej wynosi 1/4 cala, co odpowiada 470 000 pikselom, z czego 440 000 to piksele użyteczne.

Płytka kalibracyjna

Jako narzędzie służące do rozpoznawania położenia ramienia manipulatora względem kamery wykorzystuje się kwadratową płytkę przedstawioną schematycznie na poniższym rysunku (wymiary są wyrażone w milimetrah). Na płytce umieszczone są cztery wyróżnione koliste obszary zwane zamiennie punktami charakterystycznymi lub niekiedy



Rysunek 4: Płytka służąca do kalibracji.

kropkami. Rozmiar płytki został dobrany tak, by z łatwością mieściła się w chwytaku manipulatora, jej kolor – by nie wyróżniała się na tle taśmy, która znajduje się w obiektywie kamery, a kolory kropek – by uniknąć zbieżności barw z innym wyposażeniem laboratorium. Trzy kropki są zielone, a jedna czerwona, dzięki temu można wyznaczyć orientację płytki względem kamery bez względu na kąt obrotu względem jej osi optycznej. Dobrana średnica kropek zapewnia ich dobrą widoczność w obszarze pracy robota, co zostało potwierdzone w dalszej części pracy.

2 Teoria transformacji układów współrzędnych

Niniejszy rozdział ma charakter wprowadzający. Zdecydowano się go zamieścić w tej pracy, gdyż niemożliwe jest omówienie procesu kalibracji układu robot-kamera bez przybliżenia kilku najważniejszych pojęć związanych z transformacją układów współrzędnych. Mimo, że pojęcia te mają ścisłe zastosowanie w omawianym środowisku, przestawiono je w sposób możliwie uniwersalny wyraźnie oddzielając ogólne teorie matematyczne od pracy własnej i algorytmów autora.

2.1 Transformacje układów współrzędnych w przestrzeni $\mathbb{R}^3$

W tym rozdziale opisano sposoby zapisu transformacji między układami współrzędnych, czyli układami kartezjańskimi w przestrzeni $\mathbb{R}^3$. Aby poprawnie wyrazić transformację jednego układu kartezjańskiego względem innego, wystarczy wyznaczyć wzajemne położenie ich środków oraz ich wzajemną orientację, przyjmuje się, że układy są w tej samej skali, nie potrzebne jest również nieliniowe przekształcanie współrzędnych.

2.1.1 Pojęcie macierzy przekształcenia jednorodnego

Położenie i orientację sztywnego obiektu względem układu odniesienia można określić za pomocą sześciu niezależnych parametrów. W takim celu można związać z ciałem układ współrzędnych, a następnie opisać położenie i orientację tego układu względem układu odniesienia. Najwygodniejszym do zastosowań technicznych, a dzięki temu najbardziej rozpowszechnionym w robotyce sposobem jest przedstawienie go w postaci tzw. macierzy przekształcenia jednorodnego, której interpretacja oraz zastosowania zostały szczegółowo opisane w pracy [1].

W tym rozdziale opisano wyłącznie teoretyczne rozważania dotyczące transformacji układów współrzędnych. Można zatem przyjąć, bez utraty ogólności rozważań, że układy C, P i G, które się tu pojawiają są w zasadzie dowolne. Jak okaże się w dalszym ciągu pracy, ich wybór nie był przypadkowy - mają one praktyczne zastosowanie do kalibracji i zostaną schrakteryzowane w kolejnym rozdziale przy użyciu opisanych tu pojęć.

Najbardziej intuicyjnym sposobem opisu położenia jednego układu współrzędnych (np. układu płytki P) względem innego (np. kamery C) jest podanie trzech współrzędnych x, y, z początku tego układu w drugim układzie pełniącym rolę układu odniesienia, czyli tzw. wektora położenia (lub inaczej translacji) ${}^{C}\mathbf{t}_{P} = [{}^{C}x_{P}, {}^{C}y_{P}, {}^{C}z_{P}]^{T}$. Jednoznaczny opis wzajemnej orientacji układów można uzyskać wyrażając wersory kierunkowe osi jednego z

nich w drugim układzie stosując zapis ${}^{C}\widehat{X}_P, {}^{C}\widehat{Y}_P, {}^{C}\widehat{Z}_P$[4]. Wygodne jest zestawienie wersorów w kolumnach macierzy 3x3. Tak zdefiniowana macierz nosi nazwę macierzy obrotu (lub inaczej rotacji). Oczywiście nie wszystkie elementy tej macierzy są niezależne.

Należy wspomnieć kilka zależności, które spełniają wektory jednostkowe układu kartezjańskiego, w szczególności te zapisane jako kolumny macierzy rotacji:

1. Każdy z wektorów jest wersorem, więc $||\widehat{x_i}|| = 1$.
2. Iloczyn skalarny dowolnych dwóch spośród trzech wektorów jest równy 0.
3. Jak wspomniano, iloczyn wektorowy dowolnych dwóch spośród trzech wektorów pozwala na wyznaczenie trzeciego z dokładnością do znaku (zwrotu). Zachodzi więc:
$$\widehat{x_i} \times \widehat{x_j} = \epsilon_{ijk}\widehat{x_k},$$
gdzie ϵ_{ijk} jest permutacją wersorów[5].

W trakcie obliczeń istotne będzie zapewnienie spełnienia tych warunków. W przypadku idealnym warunki te powinny być spełnione bezwarunkowo, a wszelkie powstałe w trakcie kalibracji niedokładności można potraktować jako błędy. Dyskusja, w której omówiono sposoby ich eliminacji została przeprowadzona w dalszej części rozdziału.

Warto na tym etapie wspomnieć pewną przydatną w obliczeniach zależność z algebry liniowej:

4. Macierz odwrotna macierzy z kolumnami ortonormalnymi (a taką jest macierz rotacji) jest równa transponowanej.

Do dalszych rozważań znajdujacych się w głównej mierze w dalszej części pracy elementy macierzy rotacji będą oznaczone, jak poniżej:

$$\underset{P}{^{C}}\mathrm{R} = \left[{}^{C}\widehat{X}_P, {}^{C}\widehat{Y}_P, {}^{C}\widehat{Z}_P\right] = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}. \tag{1}$$

Korzystając w wprowadzonych pojęć położenie i orientację wektora $\mathbf{v}$ zdefiniowanego w układzie P względem układu C można zapisać w formie równania:

$${}^{C}\mathbf{v} = \underset{P}{^{C}}\mathrm{R} \cdot {}^{P}\mathbf{v} + {}^{C}\mathbf{t}_P.$$

[4]Zastosowanie trzeciego wersora nie jest konieczne, gdyż można go z łatwością uzyskać z dwóch pozostałych, zachodzi bowiem $\widehat{x} \times \widehat{y} = \widehat{z}$ dla dowolnego układu kartezjańskiego w trójwymiarowej przestrzeni. Operowanie trzema wersorami jest jednak wygodniejsze.

[5]Jeśli wersory są w permutacji parzystej, np. $(\widehat{x}, \widehat{y}, \widehat{z})$, to $\epsilon = 1$. Jeśli permutacja jest nieparzysta, $\epsilon = -1$. W przypadku, gdy któryś z wersorów powtarza się, np. $(\widehat{x}, \widehat{y}, \widehat{x})$, to $\epsilon = 0$.

Taki zapis ma jednak wady - pożądana jest możliwość przedstawienia transformacji między układami w formie operatora macierzowego, co umożliwiłoby np. w prosty sposób składać odwzorowania. Dlatego tworzy się tzw. macierz przekształcenia jednorodnego:

$$ {}^{C}_{P}\mathrm{T} = \left[\begin{array}{ccc|c} & {}^{C}_{P}\mathrm{R} & & {}^{C}\mathbf{t}_{P} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2} $$

oraz czterowektory położenia tak, by spełnione było równanie:

$$ \begin{bmatrix} {}^{C}\mathbf{v} \\ 1 \end{bmatrix} = \left[\begin{array}{ccc|c} & {}^{C}_{P}\mathrm{R} & & {}^{C}\mathbf{t}_{P} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \cdot \begin{bmatrix} {}^{P}\mathbf{v} \\ 1 \end{bmatrix} \tag{3} $$

Własności macierzy przekształcenia jednorodnego

Macierz $\mathbf{T}$ opisana wzorem (2) ma dwie przydatne własności:

1. Przekształcenie złożone można wyrazić za pomocą iloczynu pojedynczych przekształceń. Jeśli rozważymy np. opisane wcześniej układy P, C i G można zauważyć, że:

   $$ {}^{G}\mathbf{v} = {}^{G}_{P}\mathrm{T} \cdot {}^{P}_{C}\mathrm{T} \cdot {}^{C}\mathbf{v}, $$

   czyli

   $$ {}^{G}_{C}\mathrm{T} = {}^{G}_{P}\mathrm{T} \cdot {}^{P}_{C}\mathrm{T}. $$

2. Korzystając z czwartej własności z poprzedniego rozdziału dotyczącej macierzy odwrotnej do macierzy rotacji można udowodnić, że:

   $$ {}^{G}_{C}\mathrm{T} = {}^{C}_{G}\mathrm{T}^{-1}. $$

Powyższe własności pozwalają zapisać równanie:

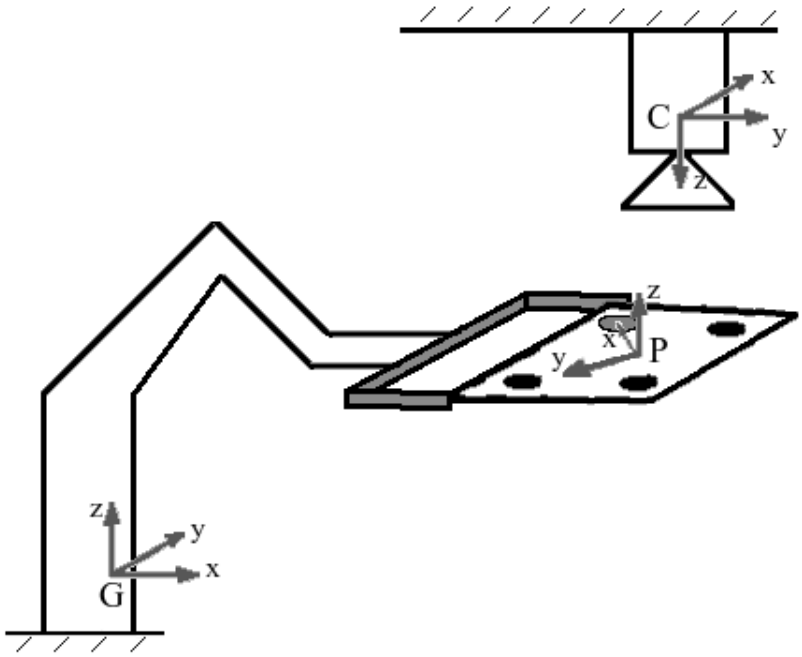
$$ {}^{G}_{C}\mathrm{T} = {}^{G}_{P}\mathrm{T} \cdot {}^{C}_{P}\mathrm{T}^{-1}, \tag{4} $$

co prowadzi do wniosku, że macierzy transformacji z układu kamery do układu bazowego robota, której wyznaczenie jest tożsame z kalibracją układu, można poszukiwać jako iloczynu przekształcenia układu związanego z płytką względem układu bazowego oraz odwrotnego przekształcenia tego układu względem układu kamery.

2.1.2 Charakterystyczne, dla zagadnień kalibracji, układy współrzędnych

W tym rozdziale opisano charakterystyczne układy współrzędnych zdefioniowane dla opisanych powyżej manipulatora, kamery i płytki z wyszczególnieniem odpowiednio dobranych orientacji tych układów.

Układ laboratoryjny przeznaczony do kalibracji znajdujący się w laboratorium został schematycznie przedstawiony na rysunku 5.



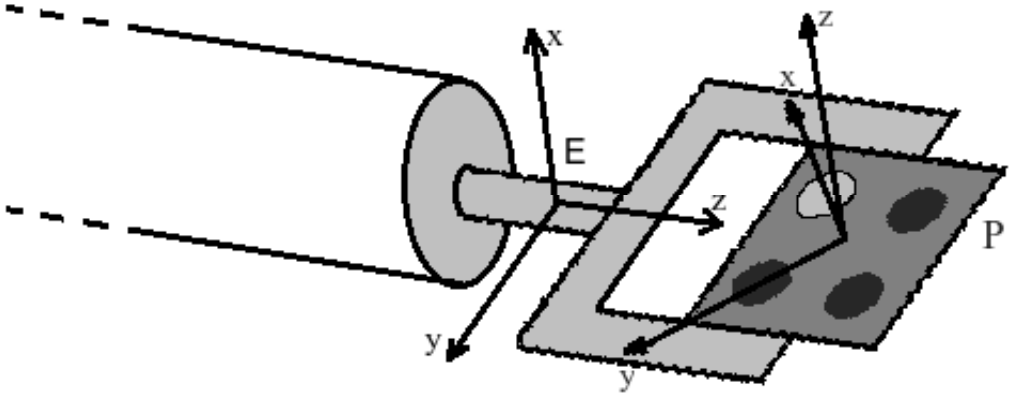
Rysunek 5: Schemat układu do kalibracji.

Jak już wspomniano, manipulator do kalibracji używa prostokątnej płyty z oznaczonymi w widoczny sposób punktami tworzącymi wierzchołki kwadratu o znanej długości boku. Zdjęcia płyty wykonane przy użyciu kamery są poddawane analizie, w celu uzyskania informacji o współrzędnych charakterystycznych punktów. Z każdym z przedstawionych na rysunku urządzeń i przedmiotów można powiązać przynajmniej jeden charakterystyczny układ współrzędnych. Wynikiem kalibracji jest wyznaczenie położenia i orientacji układu charakterystycznego kamery C w układzie globalnym G.

Układ globalny, bazowy i związany z chwytakiem

W każdym systemie wielorobotowym, wyróżnia się podstawowy układ odniesienia, zwany układem globalnym G. Dla każdego robota wchodzącego w skład systemu definiuje się jego układ bazowy, względem którego opisywane jest położenie jego efektora. Aby w prostszy sposób opisać relację między efektorami a otoczeniem, do opisu włącza się również układy współrzędnych związane z każdym efektorem.

W będącym przedmiotem opisu tej pracy środowisku układ bazowy robota biorącego udział w kalibracji pokrywa się z układem globalnym. Układ E związany z chwytakiem trzymającym płytkę zorientowany został tak, jak przedstawiono na rysunku 6.



Rysunek 6: Wzajemna orientacja układów chwytaka i płytki.

Układ związany z kamerą

Układ C związany został z matrycą CCD kamery. Oś **z** układu pokrywała się z osią kamery (prostopadłą do soczewki) i jest zwrócona w kierunku, w który „patrzy" obiektyw, a osie **x** i **y** są równoległe do boków prostokątnej matrycy tak, by na wykonanym przy użyciu kamery zdjęciu, były zgodne z umownie przyjętymi kierunkami **x** i **y** w programach graficznych. Punkt początkowy układu znajduje się na przecięciu osi optycznej soczewki obiektywu i płaszczyzny matrycy.

Warto zauważyć, że środek matrycy nie musi leżeć idealnie na osi optycznej - związane z tym przesunięcia mają wpływ na zależność położenia przedmiotu i obrazu na matrycy, które zostały omówione w kolejnym rozdziale.

Układ związany z płytką

Jak wspomniano wcześniej, na płytce służącej do kalibracji umieszczono cztery punkty charakterystyczne, z czego jeden jest wyróżniony - ma inny kolor. Punkty charakterystyczne stanowią wierzchołki kwadratu.

Układ związany z trzymaną przez robota płytką P zdefiniowany jest tak, że oś **z** skierowana jest prostopadle do płytki, ku górze, a punkt zaczepienia układu znajduje się w środku płytki. Pozostałe osie pokrywają się z przekątnymi płytki, przy czym przyjęto, że oś łącząca środek płytki z wyróżnioną kropką stanowi oś **x** układu, natomiast oś **y** jest do niej prostopadła. Tak zorientowany układ ma co najmniej kilka zalet: z łatwością można wyznaczyć transformację między płytką a kamerą, co zostało omówione w dalszej części pracy, dla każdej kropki przynajmniej jedna współrzędna spośród pary x, y jest równa 0 oraz ich współrzędne można z łatwością przetransformować do układu biegunowego, co ułatwia numerowanie kropek.

Z uwagi na fakt, że orientacja układu płytki ma istotne znaczenie, zaprojektowano algorytm wyznaczania kierunków osi poprzez numerowanie kropek. Można przyjąć, że kropka wyróżniona otrzymuje numer 1, a kolejne są numerowane przeciwnie do kierunku ruchu wskazówek zegara (układ kartezjański jest bowiem prawoskrętny). Wynik jest niezależny od orientacji płytki[6] – na osi **x** leżą kropki 1 i 3, a kropki 2 i 4 na osi **y**.

Algorytm bazuje na idei współrzędnych biegunowych. Współrzędne (u_i, v_i) i-tej kropki (względem środka płytki) można przetransformować na współrzędne biegunowe w poniższy sposób:

$$\begin{aligned} u_i &= \rho_i \sin\hat{\theta}_i, \\ v_i &= \rho_i \cos\hat{\theta}_i. \end{aligned} \tag{5}$$

Wówczas każdej kropce można przypisać jednoznaczny kąt $\hat{\theta}$:

$$\hat{\theta}_i = \mathrm{atan2}(v_i, u_i), \tag{6}$$

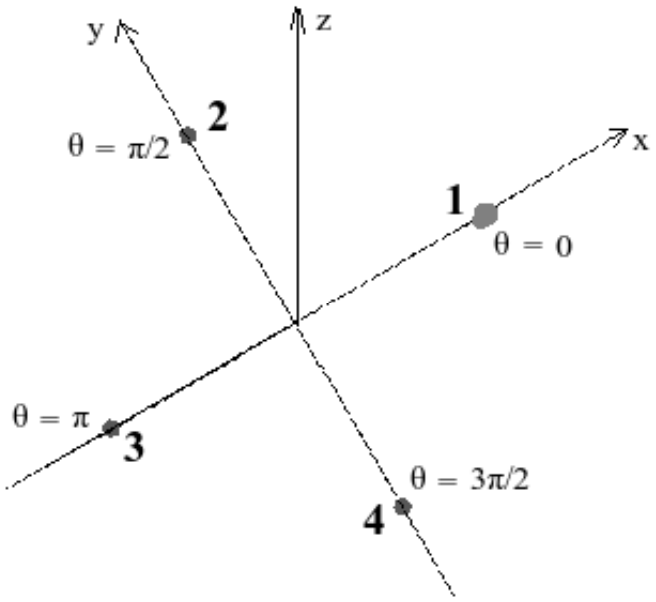
gdzie funkcja atan2 to arcus tangens „ze znakiem”[7]. Odległość kropki od środka układu ρ_i jest dla każdej kropki taka sama i równa połowie długości przekątnej kwadratu. Posiadając informację o położeniu wyróżnionej kropki ($i = 1$) można zdefiniować względne kąty:

$$\theta_i = \hat{\theta}_i - \hat{\theta}_1. \tag{7}$$

[6] pod warunkiem, że oś **z** jest skierowana do góry - płytka nie jest odwrócona spodem do kamery. Takie ustawienie i tak uniemożliwia sfotografowanie jej wierzchniej strony, więc nie stanowi to ograniczenia.

[7] Dla pary współrzędnych v, u funkcja atan2 oblicza arcus tangens ich ilorazu, a znak wyliczonego kąta zależy od ćwiartki układu współrzędnych, w której znajduje się punkt o ww. współrzędnych

Można zaobserwować, że wówczas $\theta_i = (i-1)\frac{\pi}{2}$, czyli osiągnięto założone przyporządkowanie. Całość przedstawiona została na poniższym rysunku:



Rysunek 7: Przykładowy rysunek ilustrujacy ideę numerowania kropek.

2.1.3 Opis transformacji między charakterystycznymi układami współrzędnych

Położenie układu płytki względem chwytaka

Na rysunku 6 zamieszczonym w rozdziale 2.1.2 transformacja między układami płytki i chwytaka została przedstawiona w sposób graficzny. Poniżej opisano wyznaczenie tej macierzy.

Można wprowadzić oznaczenia: a - odległość między kropkami równa 7cm, h - połowa grubości płytki równa 1.5mm oraz d - odległość między początkiem układu współrzędnych płytki i efektora równa 9.7cm. Aby znaleźć macierz ${}^{E}_{P}\mathrm{T}$ należy przeanalizować położenie czterech charakterystycznych niewspółpłaszczyznowych punktów - niech to będą trzy kropki znajdujące się na płytce oraz jeden punkt znajdujący się na pewnej wysokości ϵ nad czwartą z nich. Macierz czterowektorów kolumnowych tych punktów w układzie

efektora w zależności od wcześniej wymienionych parametrów wynosi:

$$ {}^{E}\mathrm{W} = \begin{bmatrix} h+\epsilon & h & h & h \\ -\frac{a}{2} & \frac{a}{2} & \frac{a}{2} & -\frac{a}{2} \\ d-\frac{a}{2} & d-\frac{a}{2} & d+\frac{a}{2} & d+\frac{a}{2} \\ 1 & 1 & 1 & 1 \end{bmatrix}, $$

a analogiczna macierz dla układu płytki wynosi:

$$ {}^{P}\mathrm{W} = \begin{bmatrix} \frac{a\sqrt{2}}{2} & 0 & -\frac{a\sqrt{2}}{2} & 0 \\ 0 & \frac{a\sqrt{2}}{2} & 0 & -\frac{a\sqrt{2}}{2} \\ \epsilon & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}. $$

Wówczas zachodzi:

$$ {}^{E}\mathrm{W} = {}^{E}_{P}\mathrm{T} \cdot {}^{P}\mathrm{W} \Rightarrow {}^{E}_{P}\mathrm{T} = {}^{E}\mathrm{W} \cdot \left({}^{P}\mathrm{W}\right)^{-1}. $$

Rozwiązaniem tego równania jest macierz transformacji:

$$ {}^{E}_{P}\mathrm{T} = \begin{bmatrix} 0 & 0 & 1 & h \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 & 0 \\ -\frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 & d \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0.002 \\ -0.707 & 0.707 & 0 & 0 \\ -0.707 & -0.707 & 0 & 0.097 \\ 0 & 0 & 0 & 1 \end{bmatrix}. $$

Jest ona z dokładnością do niepewności pomiaru ww. parametrów wyznaczona ściśle.

Położenie układu chwytaka względem układu bazowego. Równanie kinematyki robota

W związku ze zmianą położenia płytki zmienia się orientacja układu P z nią związanego względem układu bazowego G. Dla każdego nowego położenia musi być rozwiązane proste zadanie kinematyki manipulatora opisane równaniem:

$$ {}^{G}_{E}\mathrm{T} = {}^{G}_{1}\mathrm{T} \cdot {}^{1}_{2}\mathrm{T} \cdot \ldots \cdot {}^{i-1}_{i}\mathrm{T} \cdot \ldots \cdot {}^{n-1}_{E}\mathrm{T} $$

tak, by:

$$ {}^{G}\mathbf{p}_j = {}^{G}_{E}\mathrm{T} \cdot {}^{E}_{P}\mathrm{T} \cdot {}^{P}\mathbf{p}_j. $$

Iloczyn macierzy ${}^{G}_{1}\mathrm{T} \cdot \ldots \cdot {}^{n-1}_{E}\mathrm{T}$ zależy od konfiguracji członów i może być obliczony raz dla każdego ustawienia manipulatora. W programowej strukturze ramowej MRROC++ zaimplementowane zostały metody pozyskiwania informacji o orientacji niektórych członów w układzie bazowym robota, w szczególności końcówki jego chwytaka (układ E).

Macierz ${}^{E}_{P}\mathrm{T}$ opisuje orientację płyty względem końcówki manipulatora i jest stała w procesie kalibracji. Jej konkretna postać została podana w poprzednim punkcie.

Wektor ${}^{P}\mathbf{p}_j$ to współrzędne j-tej kropki w układzie płyty (nie zmienia się, jeśli płyta nie jest zginana, niszczona, itp.), a wektor ${}^{G}\mathbf{p}_j$ to współrzędne j-tej kropki w układzie podstawy robota.

2.2 Transformacja przy użyciu rzutu zbieżnego

Rzut zbieżny (inaczej perspektywa) to nieliniowe przekształcenie przestrzeni trójwymiarowej na płaszczyznę polegajace na tym, że każdemu punktowi P_P w przestrzeni przypisany jest punkt na płaszczyźnie P_S leżący na prostej łączącej ten punkt P_P ze środkiem projekcji. Taki sposób transformacji jest często wykorzystywany w technice, architekturze i malarstwie, gdyż jest najlepszym przybliżeniem obrazu przestrzeni, jaki tworzy ludzkie oko.

2.2.1 Prosty model kamery. Parametry kamery

Jednym z najważniejszych elementów kamery cyfrowej jest matryca CCD (ang. charge coupled device). Jest to krzemowy układ scalony pokryty siatką elektrod. Każda elektroda odpowiada jednemu pikselowi. Na skutek zjawiska fotoelektrycznego światło padające na matryce wybija z niej elektrony, ktore jednak pozostają uwięzione w izolatorze. Powstałe w ten sposób elektrony generują napięcie, którego pomiar dostarcza informacji o jasności piksela. Informację o kolorze uzyskuje się w wyniku interpolacji - komórki przetwornika AC, do którego trafiają informacje o napięciach od poszczególnych pikseli, pokryte są filtrami, jedna komórka może dostarczać informacji o jednej składowej koloru. Proces demozaikacji polega na łączeniu powstałych w ten sposób niepełnych obrazów monochromatycznych.

Parametry kamery można podzielić na dwie grupy:

- Zewnętrzne parametry kamery stanowią macierz rotacji i wektor translacji dowolnego wektora z układu związanego ze światem (w przypadku omawianego problemu z robotem) do układu związanego z kamerą. Postać i proces wyznaczania tych parametrów opisany został w dalszej części pracy.

- Wewnętrzne parametry kamery stanowi zestaw współczynników f_x, f_y, o_x, o_y[8]. Parametry f_x i f_y opisują stosunek ogniskowej kamery do odpowiednio szerokości i

[8]Do wewnętrznych współczynników zalicza się również zestaw parametrów związany ze zniekształceniem geometrycznym, tzw. efektem beczki, czyli zniekształconym odwzorowaniu linii prostych na matrycy kamery. Można założyć, że wpływ błędów wynikających z zaniedbania tego efektu jest, w porównaniu do

wysokości piksela:

$$f_x = \frac{f}{s_x}, \qquad f_y = \frac{f}{s_y}.$$

gdzie s_x, s_y to odpowiednio szerokość i wysokość piksela mierzona w systemie metrycznym.

Parametry o_x i o_y są to współrzędne przecięcia osi optycznej względem matrycy wyrażone w układzie zdjęcia (w pikselach[9]).

Aby wyznaczyć parametry kamery należy przeprowadzić proces jej kalibracji. W opisywanym w pracy problemie parametry wewnętrzne kamery są znane, celem jest wyznaczenie zewnętrznych parametrów.

W wyniku analizy zdjęcia otrzymuje się współrzędne czterech znalezionych kropek u_i, v_i (w pikselach). Współrzędne te wyznaczone są względem lewego górnego rogu zdjęcia, dlatego należy również znaleźć współrzędne środka projekcji, który jest zazwyczaj przesunięty względem środka obrazu. Aby wyliczyć współrzędne ${}^A x_i, {}^A y_i$, korzysta się ze wzorów:

$${}^A\widetilde{x_i} = s_x(u_i - o_x), \tag{8}$$

$${}^A\widetilde{y_i} = s_y(v_i - o_y), \tag{9}$$

gdzie A jest dwuwymiarowym układem matrycy.

2.2.2 Zależności geometryczne między przedmiotem a obrazem

W wyniku sporządzenia fotografii płyty otrzymuje się obraz, którego analiza może dostarczyć informacji o współrzędnych punktów charakterystycznych płyty w układzie kamery. Aby wyznaczyć te współrzędne należy wykorzystać zależności wynikające z rzutu zbieżnego. Warto wspomnieć, że transformacja rzutowa nie jest odwracalna, dlatego w dalszej części obliczeń konieczne będzie odwołanie się do pewnych dodatkowych warunków na położenie kropek w przestrzeni.

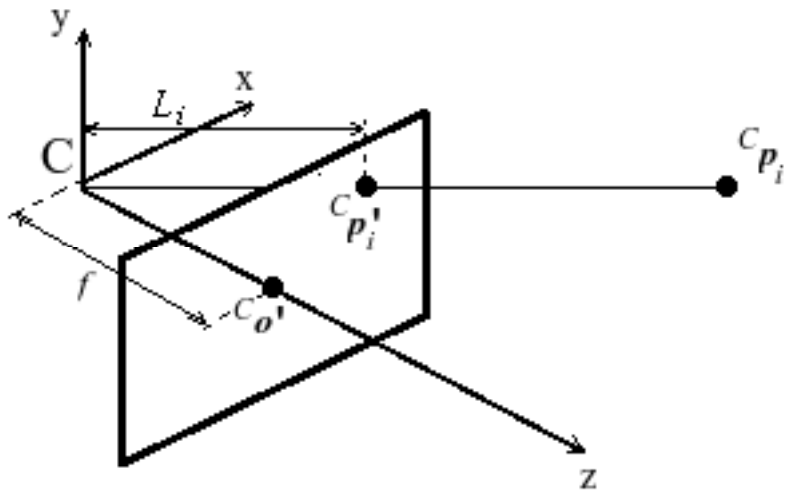
Oznaczenia i definicje

Oznaczenie ${}^C i'$, gdzie $i = \{x, y, z\}$ używane jest do wyszczególnienia współrzędnych w układzie kamery specjalnych punktów znajdujacych się na płaszczyźnie matrycy CCD. Oczywiście ${}^C z' = f$. Analogicznego oznaczenia użyto dla wektora ${}^C\mathbf{p'} = [{}^C x' \quad {}^C y' \quad {}^C z']^T$. Dla odróżnienia, wielkości „z tyldą" określają położenie tych samych punktów we współrzędnych w odpowiednio zdefiniowanym dwuwymiarowym układzie matrycy (A) ${}^A\widetilde{\mathbf{p}} = [{}^A\widetilde{x} \quad {}^A\widetilde{y} \quad {}^A\widetilde{z}]^T = [{}^C x'/f \quad {}^C y'/f \quad 1]^T$.

innych opisanych w pracy błędów, pomijalnie mały.

[9]Słowa piksel w tym kontekście użyto jako jednostki długości na zdjęciu - zwyczajowo przyjmuje się, że poziomy obiekt ma długość n pikseli, jeśli jest ona równa $n \cdot s_x$

Poniższy rysunek pokazuje zależności geometryczne między fotografowanym obiektem, a obrazem:



Rysunek 8: Zależności geometryczne między przedmiotem, a obrazem.

Traktując soczewkę kamery jako sferyczną, f jest jej ogniskową. Wektor ${}^C\mathbf{p}_i$ opisuje rzeczywiste położenie i-tej kropki w układzie kamery C. Współrzędne kropek w tym układzie nie są znane. Wektor ${}^A\mathbf{p}_i$ opisuje położenie rzutu stereograficznego i-tej kropki na matrycę kamery, również w układzie kamery. Współrzędne x, y przez niego zdefiniowane można w pośredni sposób wyznaczyć znając parametry kamery oraz analizując zdjęcia. ${}^A\mathbf{o}_i$ to położenie punktu przecięcia matrycy z osią optyczną kamery w układzie C[10]. Wektory $\mathbf{p}$ można rozpisać na współrzędne w poniższy sposób:

$$ {}^C\mathbf{p}_i = \begin{bmatrix} {}^Cx_i \\ {}^Cy_i \\ {}^Cz_i \end{bmatrix}, \qquad {}^C\mathbf{p'}_i = \begin{bmatrix} {}^Cx'_i \\ {}^Cy'_i \\ f \end{bmatrix}. \tag{10} $$

Z prostych proporcji wynikają następujące równania:

$$ {}^Cx_i = \frac{{}^Cz_i}{f} \cdot {}^Cx'_i, \tag{11} $$

[10]Wykorzystanie w powyższych zależnościach współrzędnych właśnie tego punktu upraszcza analizę w przestrzeni 3W eliminując potrzebę uwzględnienia przesunięcia go względem środka matrycy. Wspomniane przesunięcie widoczne podczas analizy zdjęć uwzględnione zostało właśnie w przestrzeni obrazu. Transformację tę opisują równania 8 - 9 podane w poprzednim paragrafie.

$$ {}^{C}y_i = \frac{{}^{C}z_i}{f} \cdot {}^{C}y'_i \tag{12} $$

lub w postaci wektorowej:

$$ {}^{C}\mathbf{p}_i = \frac{{}^{C}z_i}{f} \begin{bmatrix} {}^{C}x'_i \\ {}^{C}y'_i \\ f \end{bmatrix} = {}^{C}z_i \cdot {}^{A}\widetilde{\mathbf{p}}_i. \tag{13} $$

Każda z zależności (11), (12) opisuje cztery równania z łącznie dwunastoma niewiadomymi (współrzędne x, y, z w układzie C). Do ich rozwiązania potrzebne są dodatkowe informacje. W przypadku płytki z czterema kropkami są to odległości między nimi - na potrzeby weryfikacji kalibracji zaprojektowano metodę ich rozwiązywania, którą opisano w rozdziale 3.2.

2.2.3 Płaska homografia

W komputerowym przetwarzaniu obrazów (ang. *computer vision*) definiuje się pojęcie płaskiej homografii (ang. *planar homography*), czyli rzutu odwzorowującego jedną płaszczyznę w inną. Odwzorowanie charakterystycznych punktów w układzie płytki na płaszczyznę matrycy CCD kamery jest przykładem takiej homografii. Możliwe jest wyrażenie tego odwzorowania w formie operatora macierzowego, jeśli użyje się tzw. współrzędnych jednorodnych[11]. Oznaczając:

$$ \widetilde{{}^{C}\mathbf{p}} = \begin{bmatrix} {}^{C}x \\ {}^{C}y \\ {}^{C}z \\ 1 \end{bmatrix}, \qquad \widetilde{{}^{A}\mathbf{p}} = \begin{bmatrix} \widetilde{{}^{A}x} \\ \widetilde{{}^{A}y} \\ 1 \end{bmatrix}, $$

homografię można wyrazić w postaci równania:

$$ \widetilde{{}^{A}\mathbf{p}} = sH\widetilde{{}^{C}\mathbf{p}}, \tag{14} $$

gdzie s jest pewnym współczynnikiem skalującym, a H - macierzą homografii składającą się z dwóch części: macierzy M związanej z parametrami kamery oraz macierzy W, która opisuje położenie płaszczyzny zawierającej kropki względem płaszczyzny matrycy CCD.

[11] Powyższy fragment jest tłumaczeniem tekstu zaczerpniętego z [7]. Niektóre oznaczenia w dalszej części tekstu oraz komentarze powstały na bazie tej pozycji. Dokładniejszy opis wykorzystania homografii znajduje się w tej pozycji. Tyldy w wyrażeniu na czterowektor $\widetilde{{}^{C}p}$ użyto dla odróżnienia go od wektora trzyelementowego.

Jeśli M zdefiniuje się jako macierz 3x3:

$$M = \begin{bmatrix} f_x & 0 & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix},$$

a W jako dwunastoelementową macierz transformacji między układem związanym z płytką a układem związanym z obrazem:

$$W = \begin{bmatrix} {}^{A}_{P}R & {}^{A}\mathbf{t}_P \end{bmatrix},$$

można zapisać:

$$\widetilde{{}^{A}\mathbf{p}} = sMW\widetilde{{}^{G}\mathbf{p}}.$$

3 Wyznaczanie położenia kropek na podstawie analizy obrazów

Do transformacji z trójwymiarowego układu przestrzeni do dwuwymiarowego układu płaszczyzny wykorzystuje się rzuty. W wyniku zastosowania tego przekształcenia tracona jest pewna informacja o położeniu punktu. Do transformacji odwrotnej, czyli np. z układu „płaskiego" zdjęcia do układu kamery, można wykorzystać odpowiednio zmodyfikowane wzory opisujące rzuty, przy czym, aby dokładnie określić położenie punktu, należy wykorzystać jakieś dodatkowe równania. W wypadku rozważanej płytki z czterema punktami charakterystycznymi są to informacje o odległości między nimi.

W niniejszym rozdziale, oprócz podstaw teoretycznych omawianych zagadnień, opisano metody praktycznej ich realizacji. Z uwagi na to, że w trakcie pisania tej pracy po porównaniu autorskich metod z wbudowanymi w bibliotekę OpenCV zdecydowano, z uwagi na mniejsze błędy, o wykorzystaniu gotowego algorytmu[12], nie wszystkie zamieszczone rozwiązania są wykorzystywane do kalibracji układu. Nie mniej jednak istnieją co najmniej dwa powody, dla których dość szczegółowe opisanie własnej metody jest celowe. Po pierwsze, rozwiązanie to może być wykorzystywane do testowania aplikacji – wyniki dają w miarę dokładną informację o położeniu płytki w układzie kamery, co może być pomocne podczas kalibracji nowej kamery oraz do sprawdzenia poprawności kalibracji układu kamera - robot dla nowego położenia. Po drugie, pozwala ono poznać mechanizmy poszukiwania położenia obiektu we wspomnianym układzie, nawet jeśli w profesjonalnych algorytmach wykorzystywane są bardziej zaawansowane zależności.

Cele

Celem analizy pojedynczego zdjęcia opisanej w rozdziale 3.1 jest uzyskanie informacji o współrzędnych środka płytki wyrażonych w pikselach, na podstawie których kalibrowany jest układ robot-kamera. Celem wyznaczenia położenia punktów charakterystycznych w układzie kamery, opisanego w rozdziale 3.2, jest sprawdzenie poprawności ww. czynności oraz umożliwienie weryfikacji poprawności kalibracji.

Oznaczenia i definicje

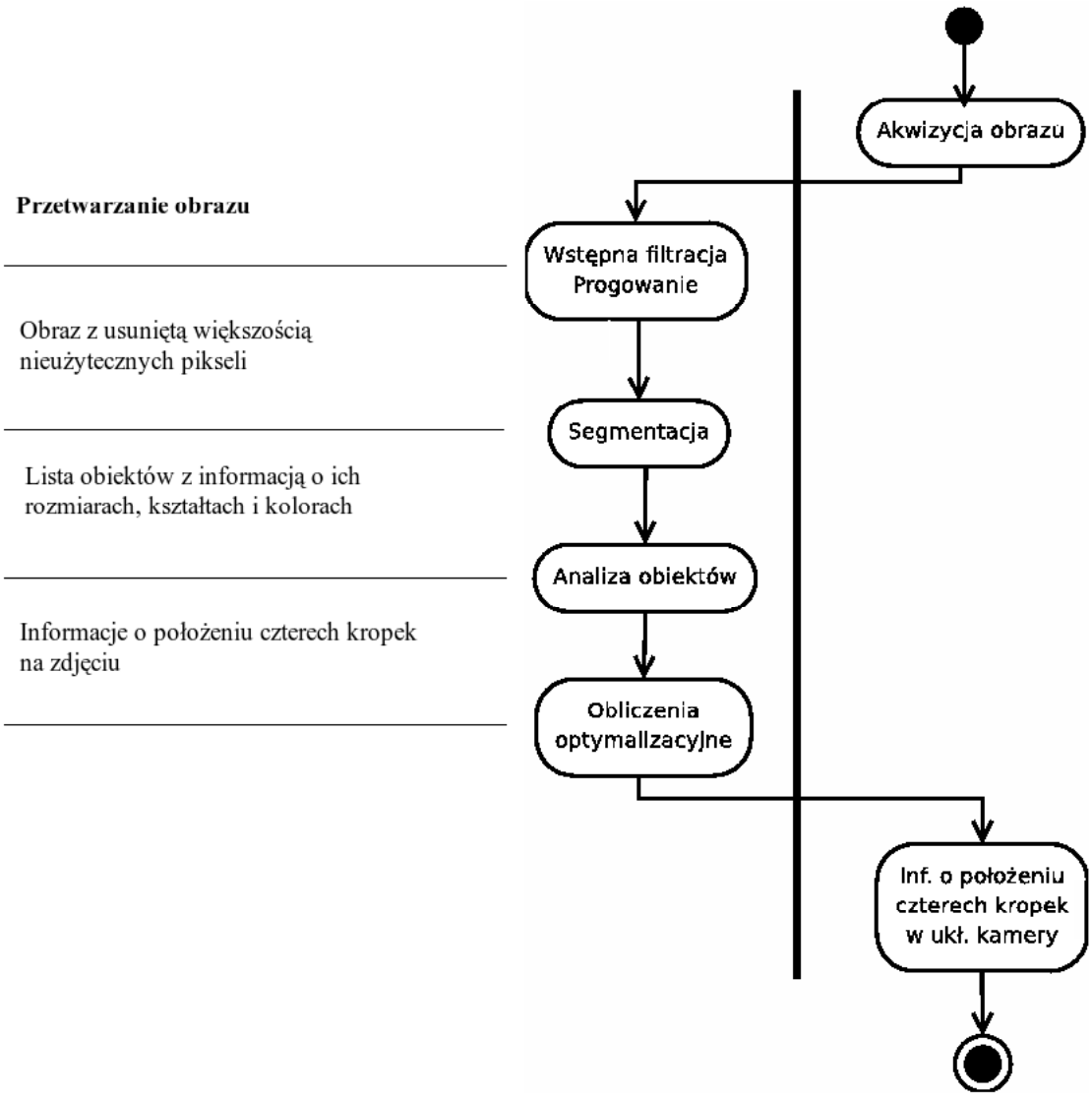
Przez położenie środka płytki należy rozumieć zarówno położenie geometrycznego środka płytki w analizowanym układzie współrzędnych, jak również położenie środka kwadratu, w

[12]Gotowe rozwiązanie polega na wywoływaniu funkcji `FindExtrinsicCameraParams2` ww. biblioteki, która dla podanych położeń punktów w pikselach podaje ich położenie w układzie bazowym robota - przejście do układu kamery jest dla programisty przezroczyste.

którego rogach znajdują się punkty charakterystyczne płytki i który wypada dokładnie w tym samym punkcie.

3.1 Analiza pojedynczego obrazu

Przez analizę pojedynczego obrazu, będącą tematem tego rozdziału, należy rozumieć algorytm postępowania mający na celu wyznaczenie ze zdjęcia płytki współrzędnych kropek wyrażonych w pikselach. Wyznaczone współrzędne pochodzące z wielu takich zdjęć posłużą następnie do kalibracji kamery z robotem. Działania mające prowadzić do tak zdefiniowanego celu zostały przedstawione na rysunku 9. Pionowa linia oddziela zadania algorytmu od pozostałych działań systemu komputerowego.



Rysunek 9: Algorytm analizy pojedynczego obrazu.

Większa część niniejszego rozdziału poświęcona jest opisowi teorii związanej z analizą obrazu oraz metod, które zostały wykorzystane do tego problemu. Analizowanie obrazów jest konieczne do wyodrębnienia informacji użytecznych dla algorytmu kalibracji. Towarzyszy mu radykalna redukcja informacji nieużytecznej. W niniejszym przypadku oczekiwane jest pozostawienie wyłącznie informacji o parametrach czterech kropek na tle zdjęcia z kamery laboratoryjnej.

W kolejnych sekcjach opisane zostały teoretyczne podstawy kolejnych punktów algorytmu analizy, które zostały zaimplementowane w opisywanej aplikacji. Akwizycja obrazu z kamery jest możliwa dzięki strukturze ramowej FraDIA, dlatego zadanie „Spots Recognition" pracuje na już odpowiednio zaalokowanych obrazach.

3.1.1 Wstępna filtracja. Progowanie

W celu ograniczenia kosztownych obliczeń związanych z poszukiwaniem obiektów w obrazie możliwe jest dokonanie wstępnej filtracji obrazu. Jednym z jej etapów jest progowanie, czyli odrzucenie pikseli o kolorach niespełniających danych założeń.

W erze kolorowych kamer cyfrowych istnieją różne standardy zapisu kolorowych obrazów. Na przykład w modelu przestrzeni barw opisanej współrzędnymi RGB każdemu pikselowi odpowiadają trzy liczby odpowiadające natężeniom kolorów odpowiednio czerwonemu, zielonemu i niebieskiemu, dla 24-bitowego zapisu koloru. Każda z tych trzech liczb należy do zbioru

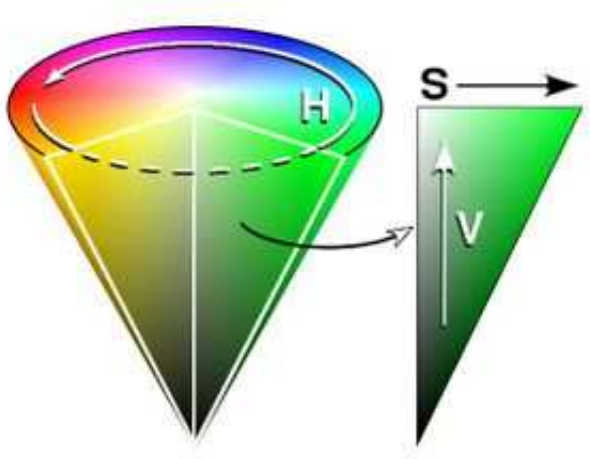
$$\mathbb{Z}_{256} = \{i \in \mathbb{Z} |\ 0 \leq i < 256\},$$

czyli przyjmuje wartość z zakresu 0-255. Istnieją również inne sposoby zapisu kolorów, np. format HSV, który jest uważany za bardziej przydatny w cyfrowym przetwarzaniu obrazów. Zaletami przestrzeni RGB nad HSV są: fakt, że współrzędne kolorów są znormalizowane do 256 oraz, że składowe zielona i czerwona (a taki kolor mają kropki na płytce) są jednymi ze współrzędnych.

W rozpoznawaniu kropek do reprezentacji kolorów użyto przestrzeni HSV. Było to spowodowane tym, że ustalenie wystarczająco szerokiego progu w przestrzeni RGB, aby w wyniku progowania nie odrzucono charakterystycznych kropek w żadnym ujęciu kamery i przy dowolnym oświetleniu, a jednocześnie wystarczająco wąskiego - aby zminimalizować liczbę obiektów przechodzących przez progowanie, okazało się bardzo trudne, podczas gdy w przestrzeni HSV okazało się to znacznie prostsze.

W przestrzeni HSV zdefiniowane są trzy składowe: H - odpowiadająca częstotliwości, S - nasyceniu oraz V - wartości światła białego. Istotę tych składowych ilustruje

rysunek 10[13].



Rysunek 10: Stożek barw w przestrzeni HSV.

We FraDIA zaimplementowane zostały mechanizmy konwersji kolorów między przestrzeniami RGB, a HSV oraz normalizacji barw.

Funkcja progująca

Dla każdego zdjęcia można ustalić pewien próg T, który dzieli piksele pod względem wybranego kryterium na dwie grupy: takie, dla których wartość progowa została przekroczona oraz o nieprzekroczonym progu, przy czym pożądane jest odrzucenie jak największej ilości nieużytecznych pikseli zachowując użyteczne. Do tego celu można zdefiniować ogólną funkcję:

$$I\colon \mathbb{Z}^3 \to \mathbb{R}.$$

Na przykład w słabo zdefiniowanych zadaniach, gdy nieznana jest dokładna kolorystyka poszukiwanych obiektów definiuje się jasność piksela w przestrzeni RGB:

$$I = \frac{c_R + c_G + c_B}{3}, \tag{15}$$

gdzie $c_i \in \{c_R, c_G, c_B\}$ to natężenia wyżej wspomnianych trzech barw. Taki sposób ma jednak dwie podstawowe wady: filtracji nie ulega bardzo dużo niepożądanych informacji, np. mechaniczne stosowanie metody progowania może spowodować pojawienie się wielu izolowanych punktów mających wartość nadprogową oraz tracona jest informacja o kolorystyce pikseli (zamiast informacji o trzech składowych przechowywana jest wyłącznie

[13]Obrazek pochodzi ze strony wikipedia.org.

ich jasność), z czym nierozłącznie wiąże się trudność w wyborze odpowiedniego progu dyskryminacji, w szczególności, gdy wykonujemy serię zdjęć z różnych ujęć i warunki na każdym z nich mogą się nieznacznie różnić.

Wartość progową można wybrać dowolnie w zależności od oczekiwanych cech fotografowanych obiektów, w szczególności wybiera się ją np. na podstawie histogramu każdego zdjęcia. Metoda powyższa jest bardzo prosta w implementacji i wystarczająca do wielu najprostszych zastosowań rozpoznawania obrazów.

Bardziej efektywną metodą jest utworzenie prostopadłościanu w przestrzeni HSV o środku w odpowiednio wybranym punkcie (c_{H0}, c_{S0}, c_{V0}) o bokach $2\Delta c_H, 2\Delta c_S, 2\Delta c_V$ tak, by kolory odpowiadające kolorom kropek znajdowały się wewnątrz prostopadłościanu, a przeważająca większość informacji niepożądanych (np. tła) poza nim.

W przestrzeni HSV podczas wyboru progu, wybór zakresu kolorów, jakie mają przechodzić progowanie ustala się jako zakres składowej H. Składowe S i V mają wartość pomocniczą - pozwalają dobrać odpowiedni odcień i nasycenie. Wybór progu dokonano przy użyciu narzędzi zaimplementowanych w strukturze programowej FraDIA analizując kilka charakterystycznych ujęć z kamery. Z uwagi na wyżej opisane podejście, przez próg dyskryminacji należy rozumieć zestaw sześciu liczb $c_{Hmin}, c_{Smin}, c_{Vmin}, c_{Hmax}, c_{Smax}, c_{Vmax}$, a funkcja progująca jest binarną funkcją, która przyjmuje wartość „1", jeśli kolor danego piksela zawiera się we wszystkich przedziałach wyznaczonych przez odpowiednie pary ww. współczynników oraz „0" w pozostałym przypadku.

3.1.2 Segmentacja obrazu

Celem segmentacji obrazu jest podział obrazu na spójne regiony (dalej zwane obiektami). W jej trakcie do wszystkich pikseli badanych obiektów przypisywane są etykiety w ten sposób, żeby wszystkie piksele konkretnego obiektu były jednakowo zaetykietowane oraz, żeby etykieta była unikalnym identyfikatorem obiektu. Jedną z możliwych metod jest *segmentacja przez progowanie* (ang. segmentation by thresholding, opisana w pozycji [6]).

Istnieje wiele technik przeszukiwania obrazu. Jedną z prostszych w implementacji jest metoda rekurencyjna oparta na algorytmie *Flood fill*. Wadą tej metody jest stosunkowo gorsza szybkość działania, niż np. metod iteracyjnych, pozwala ona jednak przeprowadzić indeksację bez kolizji, dzięki czemu niepotrzebne jest tworzenie tablicy sklejeń.

Ogólny zarys algorytmu

Algorytm ma budowę rekurencyjną. Można go podzielić na trzy fazy, z czego pierwsza inicjalizacyjna wykonywana jest tylko raz, a dwie pozostałe przeplatają się:

1) Faza inicjalizacji odpowiednich parametrów, w tym ustawienie kursora na lewej górnej pozycji, następnie przejście do kroku 2);

2) Faza wywołania pewnej funkcji przeszukującej piksele w obrębie jednego spójnego obszaru. Funkcja ta nadaje pikselom identyfikatory konkretnego obszaru, zlicza ich ilość, itp. Sprawdzenie całego spójnego obszaru skutkuje przejściem do kroku 3).

3) Faza poszukiwania kolejnego obszaru, znalezeinie obszaru powoduje przejście do kroku 2).

Algorytm kończy się, gdy cały obraz zostaje przetworzony.

Na potrzeby działania algorytmu utworzono dwie zmienne: `current_id`, która przechowuje informację o aktualnym numerze id, który będzie przypisany do znajdowanego obiektu oraz `p` przechowującą informację o położeniu piksela. Zmienne te mogą być modyfikowane tylko poza wspomnianą funkcją poszukującą, co jest jednoznaczne z przejściem do kolejnego obszaru spójnego. Obraz w pamięci komputera przechowywany jest jako jednowymiarowa tablica, dlatego istotne jest ustalenie kolejności przeglądania go piksel po pikselu. W bibliotece OpenCV aktualna pozycja w tabeli wyznaczana jest zgodnie ze wzorem (16):

$$p(x, y) = y \cdot n_c \cdot w + x \cdot n_c, \tag{16}$$

gdzie x, y są to współrzędne piksela na obrazie, w jest szerokością obrazu, a n_c jest liczbą kanałów, dla RGB $n_c = 3$. W wyniku przyjęcia takiej konwencji zapis $p = p + 1$ oznacza przejście do kolejnego piksela: jeśli jest to możliwe, przesunięcie się w prawo o jeden piksel ($x = x + 1$), jeśli napotkano koniec wiersza, przejście do pierwszego piksela w kolejnym wierszu ($x = 0, y = y + 1$).

Aby sprawnie zarządzać wywołaniami rekurencyjnymi zaproponowano zmienną `prev_pix`, która dla danego piksela „pamięta", który piksel był przetwarzany ostatnio - dokładny opis znajduje się w dalszej części rozdziału. Jej zasięg ogranicza się wyłącznie do danego wywołania funkcji.

Korzystając z notacji C++ można informacje o danym pikselu przedstawić w formie struktury:

```
struct piksel
{
     long id;          // Numer spójnego obszaru, do którego należy piksel
     int x, y;         // Położenie piksela względem lewego górnego rogu obrazu
     short colour;     // Kolor piksela
};
```

Zmienna `piksel.id` może przyjmować wartości $-1, 0, 1, \ldots, N$, gdzie N jest liczbą znalezionych obszarów. Wartości dodatnie jednoznacznie identyfikują piksel z danym regionem. Wartość 0 oznacza, że piksel nie należy do żadnego obszaru - został odfiltrowany w procesie progowania. Domyślną wartością każdego piksela jest -1, co oznacza, że piksel nie został jeszcze przyporządkowany. Pole `piksel.colour` może przyjmować wartości: 0, jeśli piksel został odfiltrowany, 1 - jeśli jego współrzędne barw znalazły się w prostopadłościanie w przestrzeni barw odpowiadającym kolorom zielonym oraz 2 - jeśli w prostopadłościanie odpowiadającym kolorom czerwonym.

Na rysunku 11 umieszczony został algorytm indeksowania wykorzystany dla omawianego problemu.

Zadania funkcji przeszukującej spójny obszar

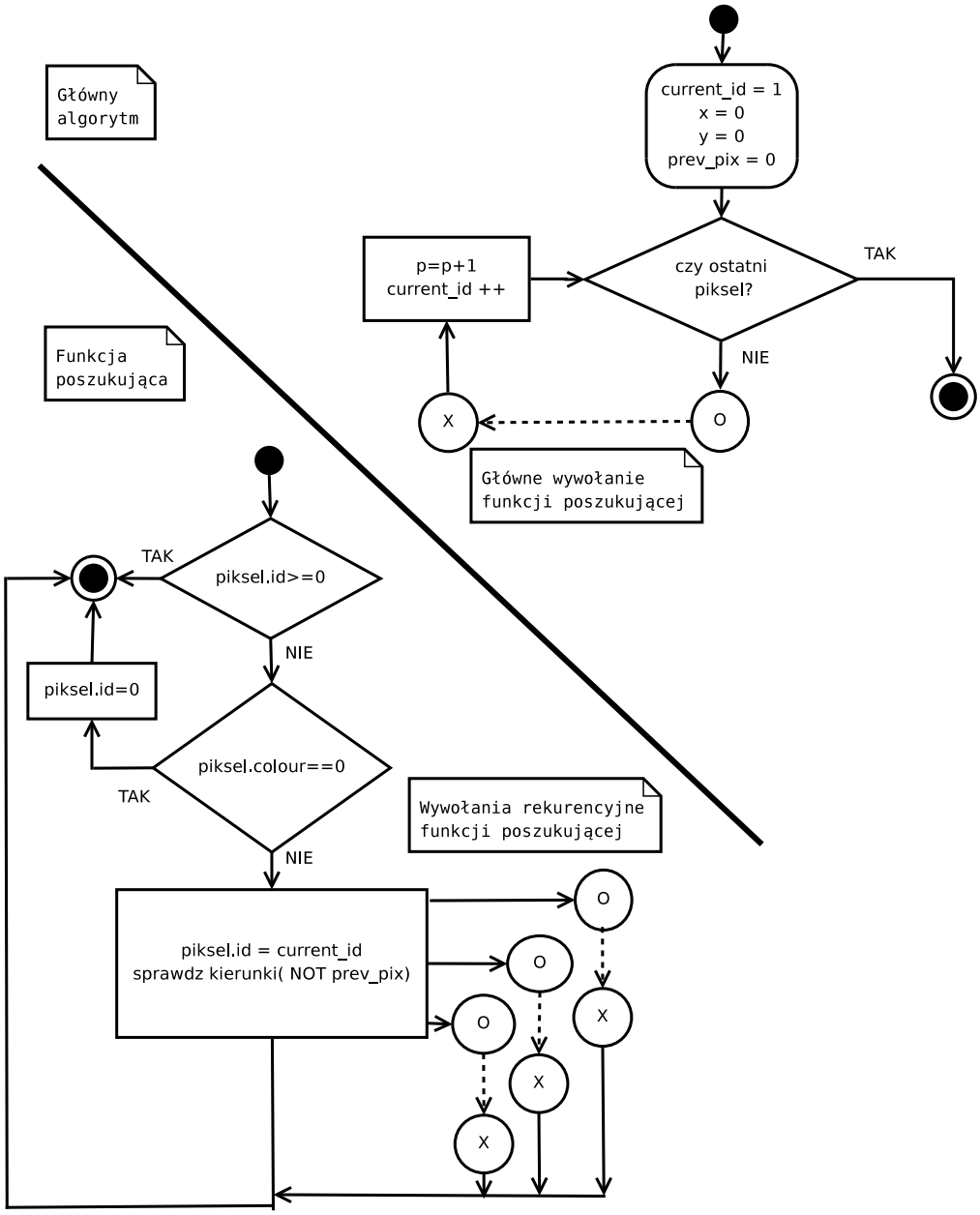
Druga faza algorytmu rozpoczyna się tzw. głównym wywołaniem funkcji przeszukującej (co na rysunku oznaczone zostało przez symbol „O w kółku"; dla odmiany opuszczenie funkcji oznaczono przez „X w kółku"). Funkcja ta może wywoływać sama siebie wielokrotnie. Jako parametr funkcja przyjmuje: pozycję piksela wyliczoną w oparciu o wzór 16, opisaną już zmienną `current_id`, żeby odpowiednio przyporządkować dany analizowany piksel do aktualnie regionu obszaru oraz zmienną `prev_pix`.

Istota działania funkcji przeszukującej opiera się na sprawdzeniu dwóch warunków:

1. Czy `id` piksela jest większe bądź równe 0. Pozytywna odpowiedź oznacza, że piksel został już sprawdzony i powoduje opuszczenie funkcji.

2. Czy `colour` piksela jest równe 0. Parametr `colour` informuje o kolorze niesprawdzonego jeszcze piksela. Wartość zerowa oznacza, że został usunięty w procesie progrowania – wówczas jego `id` jest zerowane i funkcja kończy działanie

Należy zauważyć, że opisane warunki są istotne zarówno dla głównego, jak i rekurencyjnych wywołań funkcji. Jeśli po analizie warunków okazuje się, że piksel nie został nigdy



Rysunek 11: Schemat działania algorytmu segmentacji.

wcześniej sprawdzony, nie został również odfiltrowany konieczna jest jego indeksacja. W tym celu wartości jego `id` przypisywane jest `current_id`. Jak wspomniano, ta ostatnia zmienna modyfikowana jest poza funkcją poszukującą, co powoduje, że bez względu na liczbę wywołań rekurencyjnych wszystkie piksele wchodzące w skład jednego spójnego obszaru otrzymają to samo `id`.

Zmienna `prev_pix`

Aby przeszukiwanie spójnego obszaru było możliwie skuteczne zdecydowano się, po sprawdzeniu piksela, rekurencyjnie wywoływać funkcję dla trzech sąsiadujących pikseli[14]. Niepotrzebne jest bowiem ponowne sprawdzanie piksela, który sprawdzało się przed chwilą. Dlatego wprowadzono zmienną `prev_pix`, która przechowuje informację o ostatnio odwiedzanym pikselu. Z punktu widzenia analizy arbitralnie wybranego piksela nie jest możliwe stwierdzenie, czy jest to główne, czy rekurencyjne wywołanie funkcji przeszukującej, dlatego zmienna ta przyjmuje domyślnie wartość „LEWO" – ma to związek z naturalnym kierunkiem „poruszania się" po zdjęciu między głównymi wywołaniami funkcji przeszukującej od lewej do prawej strony.

Dzięki propagacji wartości zwracanej przez opisywaną funkcję możliwe jest stwierdzenie, czy po zakończeniu głównego jej wywołania przynajmniej jeden piksel został zaetykietowany. Jeśli tak, inkrementowana jest zmienna `current_id`. Zakończenie głównego wywołania powoduje przejście do kolejnej pozycji na zdjęciu ($p = p + 1$).

Eliminacja obiektów na podstawie analizy ich pól powierzchni

Podczas indeksacji obrazu wygodnie jest tworzyć struktury odpowiadające kolejnym znalezionym obiektom. Metoda rekurencyjna segmentacji pozwala na obliczanie rozmiarów obiektów (czyli liczby pikseli wchodzących w ich skład) w czasie trwania ideksacji (nie jest potrzebne późniejsze przeszukiwanie struktury obiektów i obliczanie ich pól powierzchni). Analiza rozmiarów obiektów może dostarczyć pewnych informacji, które mogą posłużyć do odfiltrowania obiektów nieporównywalnie większych niż poszukiwane oraz izolowanych punktów pozostałych po metodzie progowania. Pobieżna analiza obrazów po wstępnej filtracji pokazała bowiem, że oprócz kropek proces filtracji przechodzi niewielka liczba pojedynczych lub zgrupowanych w małe obiekty pikseli. Odfiltrowanie jak największej liczby niepożądanych obiektów zmniejsza nakład obliczeniowy poniesiony w kolejnych etapach segmentacji.

3.1.3 Algorytm rozpoznawania wzorców

Rozpoznawanie kształtu obiektów rozpoczyna się, gdy liczba występujących na zdjęciu obiektów została zredukowana do ok 10, co jest zapewnione przez wcześniejsze operacje. Wówczas dla każdego obiektu znane jest pole powierzchni. Założono, że w otoczeniu

[14]Sąsiedztwo piksela to cztery przylegające do niego krawędziami piksele

badanej płytki nie będą się znajdować zielone eliptyczne przedmioty podobnej wielkości, co badane. Dlatego zazwyczaj już na tym etapie jedyne zielone obszary pochodzą od kropek. Nieco gorsza sytuacja ma miejsce dla kropki czerwonej, której kolor zbliżony jest do koloru manipulatora[15], którego niewielkie fragmenty przechodzą proces progowania.

Celem algorytmu jest znalezienie trzech zielonych kropek i jednej czerwonej o podobnym rozmiarze. Odnalezione zielone obszary są sortowane pod względem rozmiarów i brana jest mediana ich rozmiarów. Jeśli odnaleziono wyłącznie kropki, zapamiętywany jest ich średni rozmiar i rozpoczyna się poszukiwanie czerwonej. Jeśli jednak zdarzy się, że zielonych obszarów jest więcej (np. na skutek przypadkowego umieszczenia zielonego przedmiotu w pobliżu płytki), wybierane są trzy obszary o rozmiarze najbliższym do mediany. Pozwala to odrzucić 1-2 niepożądane obiekty o większym lub mniejszym rozmiarze.

Średni rozmiar trzech najbliższych medianie zielonych obiektów służy do znalezienia czerwonej kropki. Rozmiar obszarów czerwonych, które przeszły proces progowania, jest porównywany z wcześniej wyznaczoną średnią i wybierany jest najbliższy obiekt.

Jeśli mimo działania algorytmu do dalszego etapu wykorzystany zostany nieprawidłowy obszar, łatwo można ten przypadek wychwycić w trakcie wyznaczania położenia kropek w układzie kamery, które zostało opisane w kolejnym rozdziale.

3.2 Wyznaczanie położenia kropek w układzie kamery

Jak już wspomniano, opisana w niniejszym rozdziale metoda wyznaczania położenia kropek w układzie kamery jest metodą autorską i nie bierze ona bezpośrednio udziału w wyznaczaniu kalibracji układu robot-kamera. Znajduje ona natomiast zastosowanie do weryfikacji wyznaczonej kalibracji.

3.2.1 Warunki umożliwiające odwrotną transformację

W rozdziale 2.2.2 wyprowadzono zależności (13) między położeniami punktów w układzie kamery i układzie matrycy CCD. Pokazano również, że z uwagi na to, że rzut zbieżny jest transformacją jednokierunkową, aby wyznaczyć położenie płytki w układzie kamery na podstawie współrzędnych w pikselach potrzebne są dodatkowe informacje. Poniżej opisano metodę, która wykorzystuje informacje o odległościach między sąsiednimi kropkami, czyli odpowiednio zadane równania więzów.

[15]Niestety wybór koloru kropki został podyktowany trudnościami w znalezieniu odpowiedniej naklejki w innym kolorze. Jednak, w przypadku możliwości zmiany koloru kropki na inny, wystarczy drobna modyfikacja pliku konfiguracyjnego.

Równania więzów

Odległości między kolejnymi punktami są jednakowe. Oznaczając te odległości jako a oraz pary indeksów kolejnych punktów, jako $(i,\ j) = (1,\ 2), (2,\ 3), (3,\ 4), (4,\ 1)$, otrzymano:

$$({}^{C}x_i\ -\ {}^{C}x_j)^2 + ({}^{C}y_i\ -\ {}^{C}y_j)^2 + ({}^{C}z_i\ -\ {}^{C}z_j)^2 = a^2. \tag{17}$$

Podstawiając równania (11) i (12) do (17) otrzymano ostateczny układ czterech równań (18):

$$({}^{C}z_i{}^{C}x'_i\ -\ {}^{C}z_j{}^{C}x'_j)^2 + ({}^{C}z_i{}^{C}y'_i\ -\ {}^{C}z_j{}^{C}y'_j)^2 + f^2({}^{C}z_i\ -\ {}^{C}z_j)^2 = f^2a^2. \tag{18}$$

Równania (18) po wymnożeniu przyjmują postać:

$${}^{C}z_i^2({}^{C}x'^2_i + {}^{C}y'^2_i + f^2) + {}^{C}z_j^2({}^{C}x'^2_j + {}^{C}y'^2_j + f^2) - 2{}^{C}z_i{}^{C}z_j({}^{C}x'_i{}^{C}x'_j + {}^{C}y'_i{}^{C}y'_j + f^2) = f^2a^2. \tag{19}$$

Przyjmując, że odległość między środkiem projekcji, a środkiem $i-tej$ kropki wynosi L_i, dokonano oznaczeń poszczególnych członów:

$${}^{C}x'^2_i\ +\ {}^{C}y'^2_i\ +\ f^2 = L_i^2,$$

$${}^{C}x'^2_j\ +\ {}^{C}y'^2_j\ +\ f^2 = L_j^2$$

i wykorzystując zależności (10):

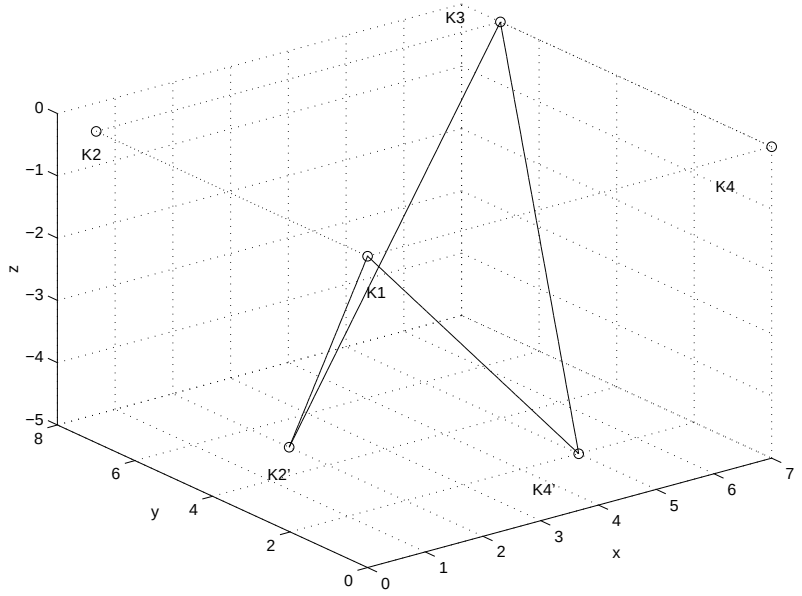
$$\begin{bmatrix} {}^{C}x'_i,\ {}^{C}y'_i,\ f \end{bmatrix} \begin{bmatrix} {}^{C}x'_j \\ {}^{C}y'_j \\ f \end{bmatrix} = {}^{C}\mathbf{p}'_i \cdot {}^{C}\mathbf{p}'_j,$$

otrzymano uproszczony zapis postaci:

$${}^{C}z_i^2L_i^2 + {}^{C}z_j^2L_j^2 - 2{}^{C}z_i{}^{C}z_j\ {}^{A}\widetilde{\mathbf{p}}_i \cdot {}^{A}\widetilde{\mathbf{p}}_j = f^2a^2. \tag{20}$$

Eliminacja niepożądanych rozwiązań

Równania więzów są kwadratowe. Z uwagi na to, każde z nich spełniają współrzędne z-owe dwóch punktów i może się zdarzyć, że w wyniku procedury minimalizacyjnej znalezione zostaną nieodpowiednie współrzędne. Przykładowe rozwiązanie spełniające wspomniane równania kwadratowe, ale nieprawidłowe, gdyż w rzeczywistości wierzchołki nie



Rysunek 12: Nieprawidłowe rozwiązanie spełniające równania więzów.

tworzą kwadratu w przestrzeni trójwymiarowej, przedstawiono na rysunku 12 (punkty K2' i K4' to znalezione rozwiązania, a K1, K2, K3, K4 - prawidłowe)[16].

Istnieją dwa rozwiązania tego problemu: nałożenie dodatkowych warunków poprzez dodanie nowych równań lub poprzez redukcję zmiennych. Pierwsza metoda, choć prostsza w implementacji, zwiększa nakład obliczeń procedury minimalizacyjnej. Przykładowe warunki mogłyby mieć postać równań na długość przekątnych - analogicznie do równań (17) można by wprowadzić zależności na przeciwległe punkty, których odległość również jest znana i wynosi $a\sqrt{2}$.

Druga metoda jest bardziej zaawansowana koncepcyjnie. Należy zauważyć, że płytka, która służy do kalibracji ma ustalony kształt - jej powierzchnia jest kwadratem, a co jest z tym związane, kropki leżą w jednej płaszczyźnie. Płaszczyznę tworzy nieskończenie wiele punktów o współrzędnych $(x, y, z) \in \mathbb{R}^3$ spełniających równanie:

$$\begin{vmatrix} x - x_1 & y - y_1 & z - z_1 \\ x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \end{vmatrix} = 0,$$

gdzie trójki $(x_i, y_i, z_i), i = 1 \ldots 3$ to współrzędne trzech różnych punktów leżących na tej płaszczyźnie. Zawężając rozważania do płaszczyzny płytki w układzie kamery przyjmując za (x_i, y_i, z_i) współrzędne dowolnych trzech z czterech kropek oraz podstawiając

[16] Jest to wyłącznie rysunek przykładowy mający na celu zobrazowanie problemu. Na rysunku warunek nakładany przez równania (11) i (12) uwzględniono z dużym przybliżeniem.

współrzędne czwartej kropki w miejsce x, y, z:

$$\begin{vmatrix} {}^{C}x_4 - {}^{C}x_1 & {}^{C}y_4 - {}^{C}y_1 & {}^{C}z_4 - {}^{C}z_1 \\ {}^{C}x_2 - {}^{C}x_1 & {}^{C}y_2 - {}^{C}y_1 & {}^{C}z_2 - {}^{C}z_1 \\ {}^{C}x_3 - {}^{C}x_1 & {}^{C}y_3 - {}^{C}y_1 & {}^{C}z_3 - {}^{C}z_1 \end{vmatrix} = 0$$

oraz, uwzględniając równania (11) i (12) otrzymano

$$\begin{vmatrix} \frac{{}^{C}z_4}{f} \cdot {}^{C}x'_4 - \frac{{}^{C}z_1}{f} \cdot {}^{C}x'_1 & \frac{{}^{C}z_4}{f} \cdot {}^{C}y'_4 - \frac{{}^{C}z_1}{f} \cdot {}^{C}y'_1 & {}^{C}z_4 - {}^{C}z_1 \\ \frac{{}^{C}z_2}{f} \cdot {}^{C}x'_2 - \frac{{}^{C}z_1}{f} \cdot {}^{C}x'_1 & \frac{{}^{C}z_2}{f} \cdot {}^{C}y'_2 - \frac{{}^{C}z_1}{f} \cdot {}^{C}y'_1 & {}^{C}z_2 - {}^{C}z_1 \\ \frac{{}^{C}z_3}{f} \cdot {}^{C}x'_3 - \frac{{}^{C}z_1}{f} \cdot {}^{C}x'_1 & \frac{{}^{C}z_3}{f} \cdot {}^{C}y'_3 - \frac{{}^{C}z_1}{f} \cdot {}^{C}y'_1 & {}^{C}z_3 - {}^{C}z_1 \end{vmatrix} = 0. \tag{21}$$

Równanie (21) pozwala na jednoznacze wyznaczenie ${}^{C}z_4$ w postaci:

$${}^{C}z_4 = \frac{{}^{C}z_1\alpha + {}^{C}z_1\frac{{}^{C}x'_1}{f}\beta + {}^{C}z_1\frac{{}^{C}y'_1}{f}\gamma}{\alpha + \frac{{}^{C}x'_4}{f}\beta + \frac{{}^{C}y'_4}{f}\gamma}, \tag{22}$$

gdzie wprowadzono pomocnicze oznaczenia:

$$\alpha = (({}^{C}z_2\frac{{}^{C}x'_2}{f} - {}^{C}z_1\frac{{}^{C}x'_1}{f})({}^{C}z_3\frac{{}^{C}y'_3}{f} - {}^{C}z_1\frac{{}^{C}y'_1}{f}) - ({}^{C}z_2\frac{{}^{C}y'_2}{f} - {}^{C}z_1\frac{{}^{C}y'_1}{f})({}^{C}z_3\frac{{}^{C}x'_3}{f} - {}^{C}z_1\frac{{}^{C}x'_1}{f})),$$

$$\beta = (({}^{C}z_2\frac{{}^{C}y'_2}{f} - {}^{C}z_1\frac{{}^{C}y'_1}{f})({}^{C}z_3 - {}^{C}z_1) - ({}^{C}z_2 - {}^{C}z_1)({}^{C}z_3\frac{{}^{C}y'_3}{f} - {}^{C}z_1\frac{{}^{C}y'_1}{f})),$$

$$\gamma = (({}^{C}z_2 - {}^{C}z_1)({}^{C}z_3\frac{{}^{C}x'_3}{f} - {}^{C}z_1\frac{{}^{C}x'_1}{f}) - ({}^{C}z_2\frac{{}^{C}x'_2}{f} - {}^{C}z_1\frac{{}^{C}x'_1}{f})({}^{C}z_3 - {}^{C}z_1)).$$

Można pokazać, że mianowanik wyrażenia na ${}^{C}z_4$ jest różny od 0[17].

Jak można zauważyć, rozwiązanie układu równań (20) przy uwzględnieniu podstawienia zadanego wzorem (22) jest praktycznie niemożliwe do rozwiązania metodą analityczną. Dlatego zdecydowano się na rozwiązanie polegające na minimalizacji funkcji.

3.2.2 Wyznaczenie poszukiwanych współrzędnych metodami minimalizacji funkcji

W dalszej części rozdziału opisana została metoda wyznaczania położenia kropek w układzie kamery w oparciu o minimalizację odpowiednio zdefiniowanej funkcji. Jak już

[17] Istotnie mianownik wyrażenia (22) jest sumą trzech wyrażeń, które, po odpowiednim dodaniu i odjęciu jednego członu, da się zapisać w postaci czterech wyznaczników 3x3. Wówczas, sumę tych wyznaczników można zapisać jako wyznacznik 4x4 umieszczając w dodatkowej kolumnie jedynki. Wyznacznik ten jest równy zero tylko, gdy któryś z wierszy jest liniowo zależny od innych lub wszystkie elementy w danym wierszu lub kolumnie są tożsamościowo równe 0. Sytuacja taka ma miejsce jedynie, gdy spośród współrzędnych czterech punktów, współrzędne dwóch z nich są takie same (rozważany jest dwukrotnie ten sam punkt). W opisywanym przypadku sytuacja taka nie może mieć miejsca.

wspomniano, wchodzi ona w skład własnego algorytmu autora, a więc nie bierze udziału w wyznaczaniu parametrów kalibracji robota z kamerą. Jednak analiza wyników tej metody dostarczyła kilku kluczowych spostrzeżeń odnośnie uzasadnionego wykorzystania zaprojektowanej wcześniej płytki oraz istnienia i wpływu szumów kamery na wyniki.

Minimalizacja funkcji g polega na rozwiązaniu problemu

$$\min_{x \in \mathbb{R}^n} g(\mathbf{x}),$$

czyli znalezienia *optimum*. Według Stachurskiego „o optimum w punkcie $\hat{\mathbf{x}}$ (optymalnych wartościach zmiennych decyzyjnych) można mówić wtedy, gdy wartość funkcji celu w tym punkcie $g(\hat{\mathbf{x}})$ jest najlepsza z możliwych do osiągnięcia" [4].

Do rozwiązania problemu minimalizacji posłużono się bezgradientową[18] metodą sympleksu Neldera-Meada[19]. Istotą tej metody jest uporządkowanie $n+1$ punktów $x^1, x^2, \ldots, x^{n+1}$ tworzących tzw. *sympleks* [20] w taki sposób, że wartości funkcji w tych punktach spełniają nierówność:

$$g^{n+1} \geq g^n \geq \cdots \geq g^1.$$

W każdej iteracji należy wyznaczyć środek ciężkości punktów $x^1 \ldots x^n$ oraz skonstruować punkt próbny, wykonując odbicie punktu x^{n+1} względem hiperpowierzchni rozpiętej na pozostałych punktach. W zależności od uzyskanego wyniku można przejść do kolejnej iteracji, w przypadku znalezienia lepszego punktu od najlepszego można dokonać próby eskpansji (rozciągnąć sympleks w tym kierunku), w skrajnie pesymistycznym przypadku konieczne jest zmniejszenie sympleksu.

Przygotowanie funkcji

Do rozwiązania równań (20) posłużono się algorytmami minimalizacji wielowymiarowej zaimplementowanymi w bibliotece numerycznej GNU Scientific Library [13]. Metoda minimalizująca wymaga podania w postaci jawnej funkcji

$$g\colon \mathbb{R}^n \to \mathbb{R}.$$

[18] Przy okazji wyboru metody należy zauważyć, że metody gradientowe minimalizacji funkcji są zazwyczaj lepsze. Niestety wymagają one podania jawnej postaci gradientu, co w przypadku konieczności różniczkowania równania (22) okazało się zbyt skomplikowane. Funkcja zaproponowana w kolejnym punkcie jest ciągła, daje się więc bez trudu zminimalizować metodą Neldera-Meada.

[19] Oznaczenia i opis algorytmu działania metody pochodzi z pozycji [4]

[20] wielościan o $n+1$ wierzchołkach

Można wprowadzić oznaczenia:

$$\begin{aligned} r_1 &= {}^C z_1^2 L_1^2 + {}^C z_2^2 L_2^2 - 2 {}^C z_1 {}^C z_2 \ {}^A\widetilde{\mathbf{p}}_1 \cdot {}^A\widetilde{\mathbf{p}}_2 - f^2 a^2, \\ r_2 &= {}^C z_2^2 L_2^2 + {}^C z_3^2 L_3^2 - 2 {}^C z_2 {}^C z_3 \ {}^A\widetilde{\mathbf{p}}_2 \cdot {}^A\widetilde{\mathbf{p}}_3 - f^2 a^2, \\ r_3 &= {}^C z_3^2 L_3^2 + {}^C z_4^2 L_4^2 - 2 {}^C z_3 {}^C z_4 \ {}^A\widetilde{\mathbf{p}}_3 \cdot {}^A\widetilde{\mathbf{p}}_4 - f^2 a^2, \\ r_4 &= {}^C z_4^2 L_4^2 + {}^C z_1^2 L_1^2 - 2 {}^C z_4 {}^C z_1 \ {}^A\widetilde{\mathbf{p}}_4 \cdot {}^A\widetilde{\mathbf{p}}_1 - f^2 a^2, \end{aligned} \tag{23}$$

oczekując możliwie dokładnego spełnienia równości (24):

$$\begin{aligned} r_1 &= 0, \\ r_2 &= 0, \\ r_3 &= 0, \\ r_4 &= 0. \end{aligned} \tag{24}$$

Wiedząc, że wektor $\mathbf{r}$ zdefiniowany jest jak w równości (25):

$$\mathbf{r} = \begin{bmatrix} r_1 & r_2 & r_3 & r_4 \end{bmatrix}^T, \tag{25}$$

zapostulowano utworzenie funkcji g zgodnej z założeniami postaci:

$$g({}^C z_1,\ {}^C z_2,\ {}^C z_3, {}^C z_4({}^C z_1,\ {}^C z_2,\ {}^C z_3)) = ||r||^2 = r_1^2 + r_2^2 + r_3^2 + r_4^2. \tag{26}$$

Jest to 2-norma zmodyfikowana poprzez opuszczenie pierwiastka, gdyż każdorazowe pierwiastkowanie zwiększa nakład obliczeniowy. Wielkość $\frac{\sqrt{g}}{f^2}$ opisuje błąd minimalizacji. Wówczas celem jest znalezienie minimum funkcji $g \geq 0$. Należy pamiętać, że współrzędne ${}^C z_4$ są ściśle wyznaczone przez pozostałe trzy.

Funkcja zdefiniowana wzorem (26) posiada jedno minimum globalne. Aby uniezależnić się od konieczności sprawdzania, czy znalezione minimum jest lokalne, czy globalne, można żądać, by znaleziona wartość była mniejsza od pewnej ustalonej granicy $\epsilon \approx 0$.

Szacowanie punktu startowego

Większość metod minimalizacyjnych wymaga wstępnego oszacowania punktu startowego. Jest to szczególnie istotne, jeśli minimalizowana funkcja posiada ekstrema lokalne. Wybór punktu początkowego powinien być podyktowany jego potencjalną bliskością do punktu optymalnego, gdyż znacznie upraszcza to procedurę minimalizacyjną.

W rozważanym w niniejszej pracy przypadku należy oszacować współrzędne z-owe kropek w układzie kamery znając rzeczywiste rozmiary płytki oraz położenie kropek na obrazku. Znane są również parametry wewnętrzne kamery, które zostały opisane dokładniej w rozdziale 2.2.1, co pozwala na oszacowanie położenia punktów w układzie kamery.

Aby uniknąć opisu uwzględniającego parametry kamery dokonano pewnego uproszczenia. Jest to dopuszczalne, gdyż błąd z tym związany jest nieporównywalnie mniejszy niż błąd wynikający z samego szacowania. Jeśli znamy wzajemne odległości kilku punktów l_{ij} znajdujących się blisko siebie (tzn. w porównywalnej odległości od kamery) to można założyć, że liniowa zmiana rzeczywistej odległości odpowiada analogicznej zmianie odległości d_{ij} w pikselach, czyli

$$d_{ij}[pix] \rightarrow l[cm] \Longleftrightarrow k \cdot d_{ij}[pix] \rightarrow k \cdot l[cm],$$

gdzie k jest pewnym współczynnikiem skalującym bliskim 1.

Jedną z najważniejszych odległości badanych na zdjęciach jest odległość między kropkami. Na dwuwymiarowym zdjęciu odległość ta jest największa (i zarazem równa rzeczywistej[21]), gdy łącząca punkty prosta jest prostopadła do osi soczewki kamery (na tym etapie niepotrzebne jest uwzględnianie zniekształcenia przestrzeni wynikającego z „efektu beczki"). Wiedząc o tym można z pewnym przybliżeniem oszacować na potrzeby j-tej kropki „odległość" d_j między dwiema kropkami w pikselach, jako średnią arytmetyczną odległości między kropką j-tą, a dwiema sąsiednimi (i-tą - d_{ij}, k-tą - d_{jk}):

$$d_j[pix] = \frac{1}{2}(d_{ij} + d_{jk}). \tag{27}$$

Ta odległość odpowiada rzeczywistej a, co można zapisać symbolicznie:

$$d_j[pix] \rightarrow a[cm].$$

Wówczas można odczytać przesunięcia każdej z kropek na zdjęciu względem środka zdjęcia i oznaczywszy je przez u_j w kierunku „poziomym[22]" oraz v_j w kierunku do niego prostopadłym, wprowadzić zależności korzystając z liniowości skalowania:

$$u_j[pix] \rightarrow^C x_j[cm] = u_j[pix] \cdot \frac{a[cm]}{d_j[pix]}, \tag{28}$$

$$v_j[pix] \rightarrow^C y_j[cm] = v_j[pix] \cdot \frac{a[cm]}{d_j[pix]}. \tag{29}$$

Jeśli przekształci się wzory (11)-(12) do postaci:

$$^C z_j = f \cdot \frac{^C x_j}{^C x'_j}, \tag{30}$$

[21]Należy rozróżnić rzeczywisty rozmiar płytki, odpowiadający mu „rozmiar rzeczywisty" na obrazku, czyli na matrycy kamery oraz rozmiar w pikselach. W tym porównaniu ważne jest spostrzeżenie, że gdy z-owe współrzędne punktów niewiele się różnią, to każde inne od prostopadłego (do osi optycznej kamery) ustawienie płytki powoduje zmniejszenie rejestrowanych na zdjęciu odległości.

[22]takim, który odpowiada poziomemu na zdjęciu

$$^{C}z_j = f \cdot \frac{^{C}y_j}{^{C}y'_j}, \tag{31}$$

można oszacować $^{C}z_j$ jako średnią arytmetyczną dwóch powyższych, czyli:

$$^{C}z_j = \frac{fa}{2d}\left(\frac{u_j}{^{C}x'_j} + \frac{v_j}{^{C}y'_j}\right). \tag{32}$$

Błędy minimalizacji i sposoby ich eliminacji

Obraz uzyskiwany z kamery nie jest statyczny, szum odczytu jest bez trudu zauważalny podczas obserwacji nieruchomego obiektu w programie komputerowym. Powoduje to chwilowe względne modyfikacje barw obiektów na zdjęciach wykonywanych po sobie, co z kolei powoduje, że w wyniku progowania i segmentacji odczytane położenie kropek może się nieznacznie różnić. Choć różnica w odczytach położenia środka kropki w pikselach dla statycznej płytki jest w zasadzie niewielka (rzędu jednego piksela), tego typu zaburzenie ma widoczny wpływ na przebieg minimalizacji i w efekcie wyniki różnią się nawet o 2-3 cm. Jest to jednak rozrzut przypadkowy, można więc zauważyć, że wykonanie wielu jednakowych zdjęć i odpowiednie uśrednienie wyników pozwala poprawić wyznaczone odległości.

Błąd minimalizacji zdefiniowano jako 2-normę opisaną wzorem (26), funkcja ta obliczana jest oddzielnie dla każdego zdjęcia. Do procesu kalibracji dla jednego położenia robota wykonywanych jest n (10, liczba ta została wyznaczona w drodze eksperymentów, które opisano w dalszej części rozdziału) takich samych zdjęć. Aby na podstawie danych odnośnie n zestawów współrzędnych kropek obliczyć współrzędne najbliższe rzeczywistym można zaproponować następujący sposób.

Niech $x_j^{(i)}, y_j^{(i)}, z_j^{(i)}$ oznaczają odpowiednio x-ową, y-ową i z-ową współrzędną j-tej kropki obliczoną na podstawie i-tego z n zdjęcia. Przyjmując oznaczenie, jak w rozdziale 3.2.2, błędem pojedynczej minimalizacji będzie wartość funkcji $g^{(i)}$, której wartość powinna dążyć do 0. Niech dodatkowo $q^{(i)} = \frac{1}{g^{(i)}}$[23], a

$$Q = \sum_{i=1}^{n} q^{(i)}, \tag{33}$$

można obliczyć współrzędne przekazywane jako ostateczny wynik dla danego położenia

[23]Postać funkcji g zapewnia, że nie przyjmie ona nigdy wartości równej 0, a co za tym idzie q jest dobrze określone. Wiąże się to z tym, że oprócz rzeczywistych warunków, które wyraźnie odbiegają od idealnych, występują również błędy związane z tym, że zadanie jest źle uwarunkowane. W praktyce okazuje się, że wartość funkcji nie jest mniejsza, niż 0.01.

manipulatora jako:

$$\begin{bmatrix} X_j \\ Y_j \\ Z_j \end{bmatrix} = \frac{1}{Q} \sum_{i=1}^{n} \begin{bmatrix} q^{(i)} x_j^{(i)} \\ q^{(i)} y_j^{(i)} \\ q^{(i)} z_j^{(i)} \end{bmatrix}. \tag{34}$$

Jest to średnia ważona, która premiuje zdjęcia o mniejszym błędzie minimalizacji i zapewnia, że ich wkład w ostateczny wynik będzie większy. Za ostateczny błąd minimalizacji można przyjąć $\max(g^{(i)})$. Taki sposób liczenia powoduje tym lepsze wyeliminowanie błędów przypadkowych, im więcej zdjęć jest wykonywanych dla tej samej pozycji. Niestety ta procedura nie umożliwia zminimalizowania błędów systematycznych (np. błędów wyznaczenia wewnętrznych parametrów kamery, błędów algorytmu minimalizującego funkcję, itp.).

3.2.3 Analiza wyników minimalizacji dla różnych położeń płytki

Wpływ szumów kamery na wyniki minimalizacji

Jak już wspomniano w rozdziale 3.2.2, nawet niewielkie zaburzenie odczytu środka kropki w pikselach ma widoczny wpływ na wynik minimalizacji. Może mieć to związek zarówno z budową algorytmu Neldera-Meada, jak i z narzuconymi na niego warunkami: położenie jednej kropki wyznaczane jest na podstawie trzech pozostałych oraz minimializowany jest błąd wyliczonych między nimi odległości. Dlatego zdecydowano się dokonać analizy rozrzutów wyników na podstawie wykonanej serii zdjęć dla niezmieniającego się położenia płytki.

Cele

Celem poniżej opisanych testów jest udowodnienie konieczności uśredniania wyników co najmniej kilku jednakowych ujęć, sprawdzenie zachowania procedury minimalizującej dla zmieniającej się liczby zdjęć w zależności od odległości płytka-kamera oraz wyznaczenie optymalnej liczby zdjęć, dla których wyniki będą wystarczająco wiarygodne.

Test 1 – rozrzut wyznaczonych współrzędnych

Test pierwszy przeprowadzono dla odpowiednio dobranej odległości płytka-kamera: odpowiada ona w przybliżeniu połowie wysokości przestrzeni roboczej manipulatora. Na rysunku 13 przedstawiony został rozrzut (wyznaczonych przy użyciu minimalizacji) współrzędnych poszczególnych kropek, wokół odpowiadających im wartości średniej dla 30

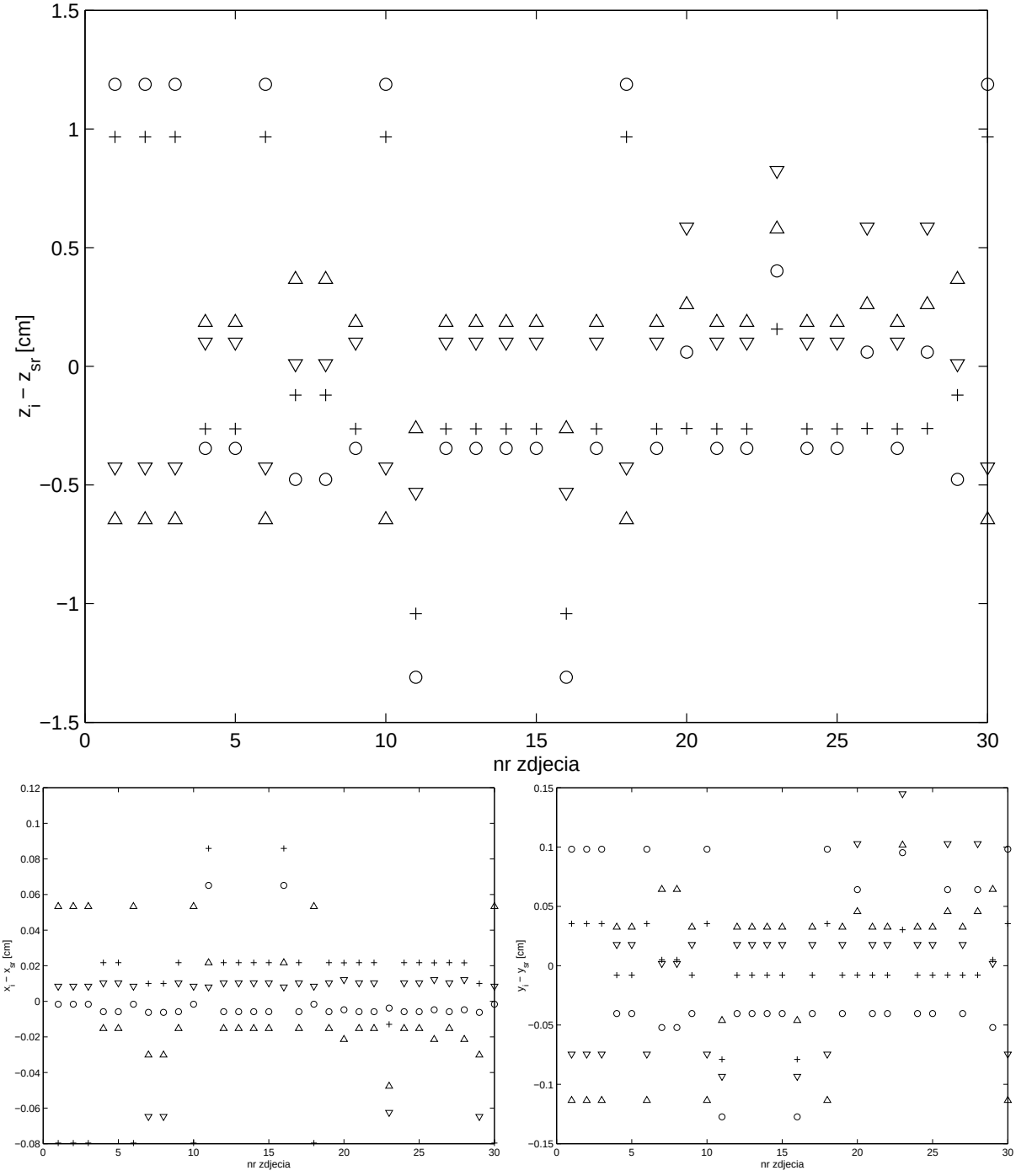
zdjęć nieruchomo położonej płytki. Każdej z kropek odpowiada inny symbol na wykresie. Można zauważyć, że o ile rozrzut współrzędnej z-owej jest zauważalny i wynosi ok. $\pm 1cm$, o tyle rozrzut współrzędnych poziomych jest o rząd wielkości mniejszy.

Test 2 – zależność średniej ważonej od ilości zdjęć

Z uwagi na zmieniające się wyniki minimalizacji dla niezmiennych warunków i ujęć zdecydowano się na uśrednianie wyników zgodnie z założeniami wyprowadzonymi w poprzednim rozdziale. Kluczowym problemem było ustalenie takiej liczy analizowanych zdjęć, aby średnia ważona dla każdego położenia płytki dość dobrze obrazowała rzeczywiste położenie płytki, a jednocześnie analiza jednego ujęcia nie trwała zbyt długo.

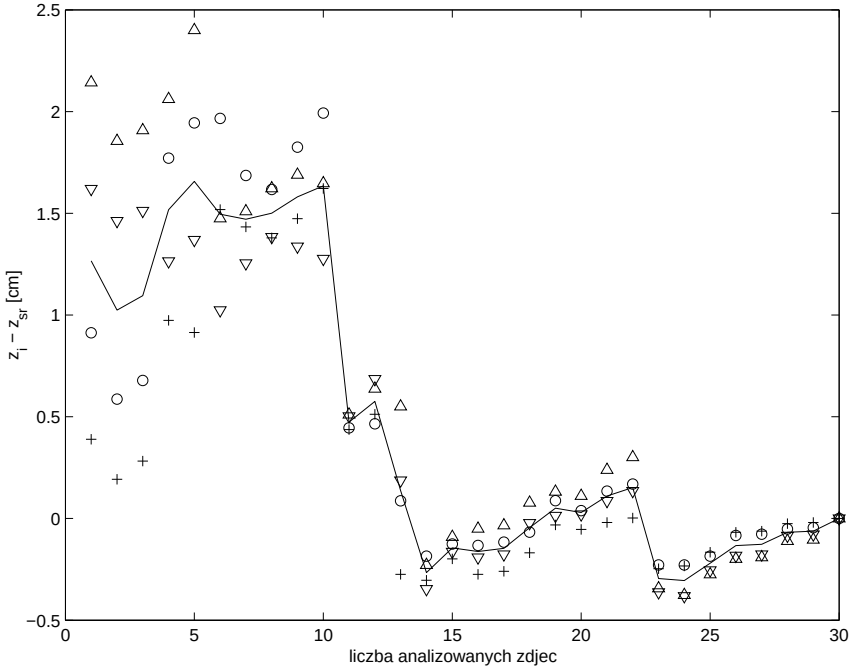
Na trzech kolejnych rysunkach (14 - 16) przedstawiono analizę poprzednio zdiagnozowanego problemu. Na wykresach przedstawiono średnią ważoną współrzędnych z-owych wyznaczonych ze wzoru (34) dla pierwszych i zdjęć, gdzie i zmienia się od 1 do 30 odniesioną do średniej ważonej całego zestawu wyników. Oznacza to, że jeśli $i = 1$, przedstawiony jest po prostu wynik analizy pierwszego zdjęcia, gdy $i = 2$ - średnia ważona pierwszych dwóch, aż do $i = 30$, czyli średnia ważona wszystkich 30 wyników. Pozwala to na obserwację, jak zmienia się średnie położenie i dla ilu zdjęć zmiana jest na tyle nieznaczna, że można już zakończyć uśrednianie. Symbolami graficznymi oznaczono wyniki każdej z kropek, linią ciągłą - środek płytki[24]. Badanie zostało wykonane dla trzech odległości płytka-kamera: ok $110cm, 80cm$ i $40cm$, które to obejmują swoim zakresem niemal cały obszar pracy manipulatora.

[24]czyli średnią arytmetyczną z położeń czterech kropek. Później do kalibracji brany jest właśnie ten wynik, więc jest on kluczowy z punktu widzenia przeprowadzanej analizy.

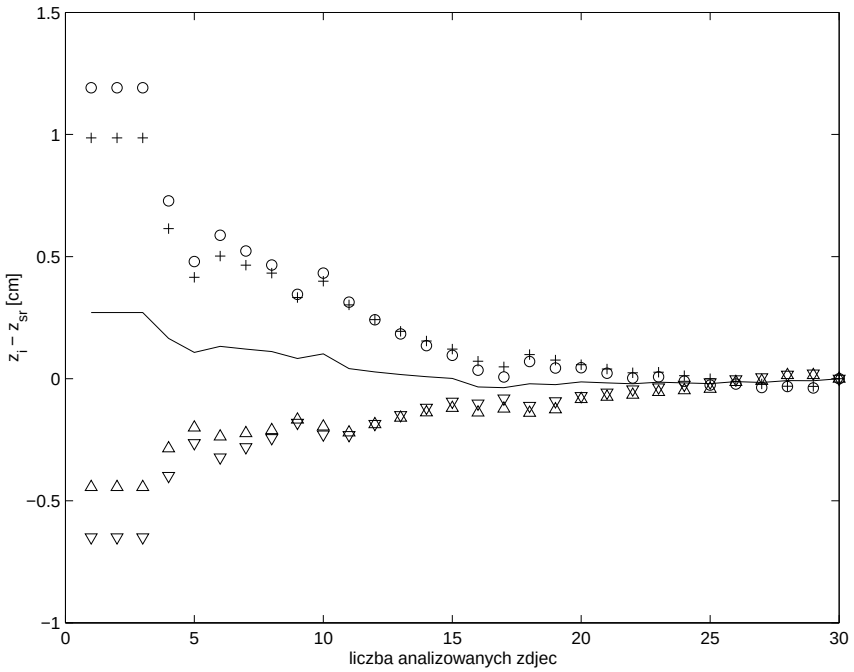


Rysunek 13: Test 1 - rozrzut wyników minimalizacji dla jednakowych ujęć.

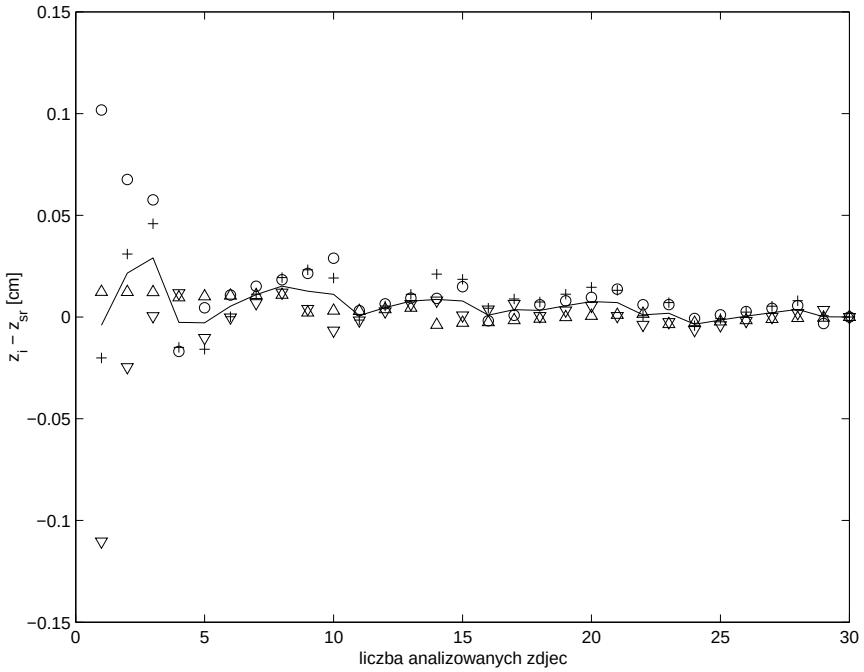
Na wykresach znajduje się rozrzut z- (na górze) oraz x- (z lewej) i y-owej (z prawej) współrzędnej wokół wartości średniej każdej z kropek. Wykres dla współrzędnej z jest większy, gdyż otrzymany rozrzut przyjmuje większe wartości, a zatem ma większy wpływ na ostateczne wyniki. Minimalizacja była przeprowadzana dla 30 jednakowych ujęć nieruchomej leżącej płasko płytki, a więc rozrzut jest wynikiem jedynie wpływu szumu odczytu na procedurę minimalizacyjną. Wyniki dla każdej z kropek zaznaczono oddzielnym symbolem graficznym.



Rysunek 14: Test 2 - analiza średnich ważonych, odległość płytka-kamera ok. 110cm.



Rysunek 15: Test 2 - analiza średnich ważonych, odległość płytka-kamera ok. 80cm.



Rysunek 16: Test 2 - analiza średnich ważonych, odległość płytka-kamera ok. $40cm$.

Na powyższych wykresach znajduje się różnica między średnią ważoną z-owej współrzędnej dla pierwszych $i = 1 \ldots 30$ zdjęć, a średnią za cały zestaw danych. Wszystkie zdjęcia wykonywane były dla płasko leżącej nieruchomej płytki. Wyniki dla poszczególnych kropek zaznaczono symbolami graficznymi, linią ciągłą oznaczono zachowanie wcześniej zdefiniowanych średnich dla środka płytki czyli dla średniej arytmetycznej wyników czterech kropek.

Wnioski

Można zauważyć, że mimo występującej zależności w funkcji odległości płytka-kamera między rozrzutem średniej ważonej wraz z zwiększającą się liczbą jednakowych ujęć, już dla ok. 10-15 zdjęć zmiany średniej ważonej środka płytki mogą już być zaniedbywane. Z uwagi na fakt, że do alogrytmu kalibracji podawane są współrzędne kropek w pikselach, a nie wyniki minimalizacji, co znacznie redukuje wahania, 10 jednakowych zdjęć w pełni wystarcza do uniezależnienia się od chwilowych szumów.

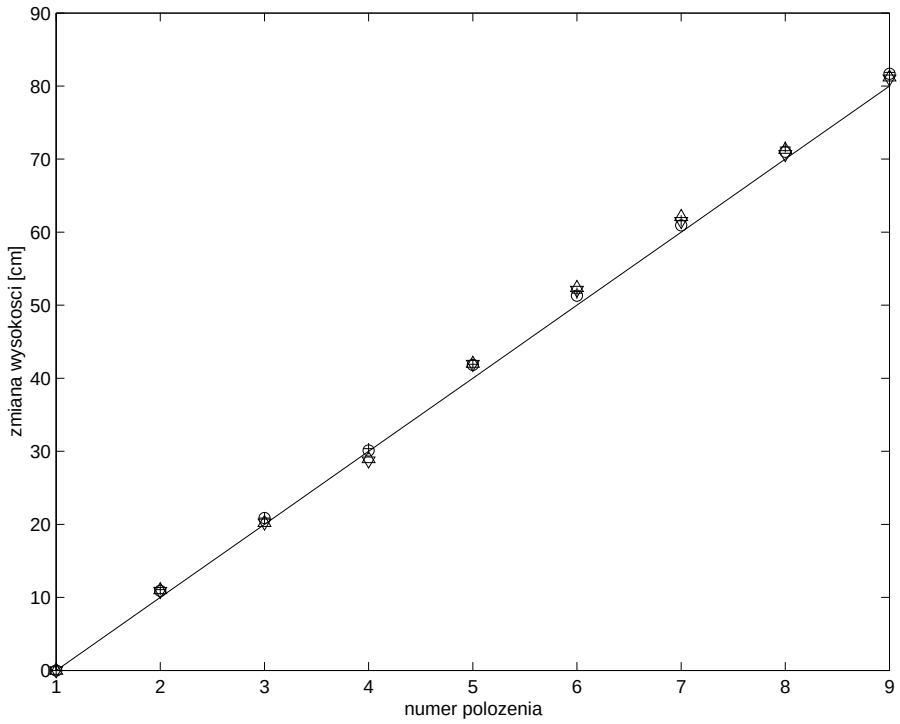
Czułość na zmiany położenia płytki

W niniejszym rozdziale przedstawione zostały dwa testy mające sprawdzić czułość algorytmu minimalizującego na zmiany położeń płytki, a przez to dowieść, że jej budowa, rozmiary, itp. w pełni uzasadnia jej wykorzystanie do kalibracji.

Dla danego położenia płytki wykonywano 10 jednakowych zdjęć, które poddawane były działaniu opisywanego wcześniej algorytmu. Otrzymane w ten sposób wyniki były (w ramach jednego położenia) uśredniane. Regulacja wysokości płytki (czyli odległości płytka-kamera) oraz jej kąt nachylenia modyfikowane były przy użyciu klocków DUPLO - wysokość jednego klocka (po połączeniu, bez uwzględniania wypustek) wynosi ok. 2 cm.

Test 1 – zmiana odległości płytka-kamera

Pierwszy test (rysunek 17) polegał na fotografowaniu leżącej płasko płytki w zależności od odległości kamera - płytka. Celem był pomiar zmiany wysokości płytki (na rysunku aproksymowany linią prostą) oraz jednocześnie zmiany współrzędnych ${}^{C}z$ kropek otrzymanych z wykorzystaniem aplikacji (oznaczone na wykresie odpowiednimi symbolami). Oczywiście pożądana byłaby doskonała zgodność zmian tych dwóch wielkości.

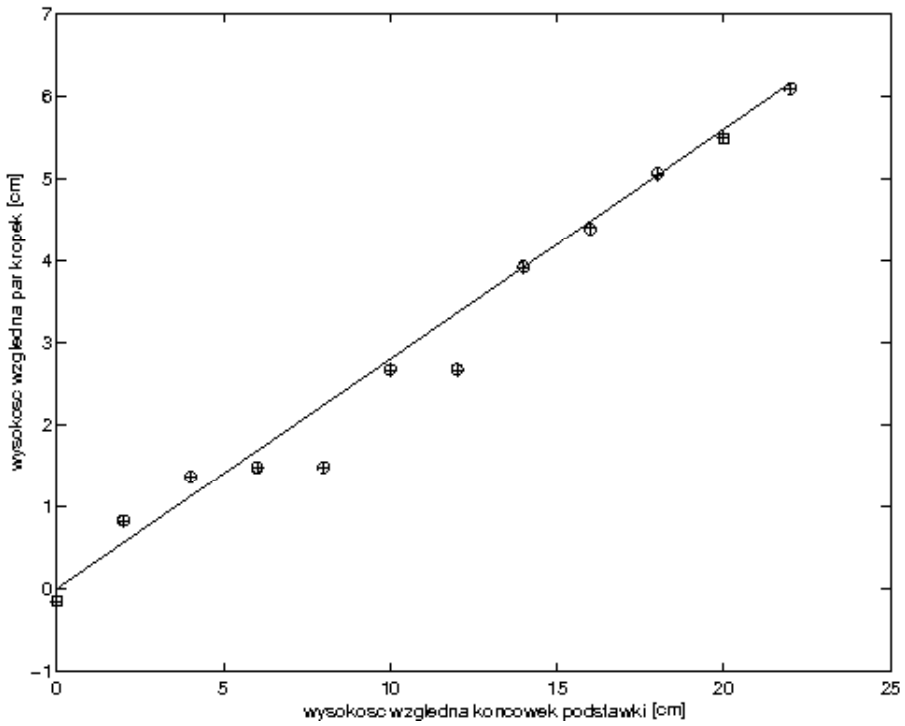


Rysunek 17: Test 1.

Test 2 – zmiana kąta nachylenia płytki

Drugi test (rysunek 18) polegał na umieszczeniu płasko leżącej płytki w odległości ok. 80 cm od kamery na długiej (ok. 30 cm) podstawce i modyfikację kąta ustawienia płytki

względem osi optycznej kamery poprzez dostawianie klocków z jednej strony tacki, a zatem zmianę względnej wysokości odpowiednich par kropek. Na bazie pomiaru wysokości klocków przy użyciu odpowiedniej proporcji wyliczana była różnica wysokości przeciwległych kropek (estymowana na wykresie za pomocą ciągłej linii prostej) oraz dokonwywany był pomiar współrzędnych ${}^{C}z$ kropek (oznaczony na rysunku plusami i kółkami). Pozwoliło to zmierzyć czułość algorytmu na zmianę kąta ustawienia płytki względem kamery.



Rysunek 18: Test 2.

Warto zauważyć, że o ile w pierwszym teście zmiana wysokości o pewną wysokość skutkowała niemal identyczną zmianą wyznaczonych współrzędnych w aplikacji, wystąpiły pewne niezgodnosci, szczególnie widoczne w drugim teście. Mogły one być spowodowane nieregularną powierzchnią klocków – różnica spowodowana wypustkami wynosiła ok 0,5 cm i miała największy wpływ dla małych kątów, co znalazło odzwierciedlenie na wykresie.

Wnioski

Przeprowadzone doświadczenia wykazały, że mimo niewielkich rozmiarów płytki, zmiana odległości płytki od kamery jest przy użyciu algorytmu wyznaczania położenia kropek w pełni widoczna i z dużą dokładnością zgodna z faktycznym przemieszczeniem płytki. Mimo dobrej zgodności wyników doświadczalnych z faktyczną zmianą nachylenia płyki, nie można jednoznacznie stwierdzić, czy

wyznaczona na ich podstawie orientacja między układem płytki, a kamery w jednym ujęciu będzie wystarczająco dokładna oraz czy możliwe będzie jej wykorzystanie.[25]

[25]Istotnie, z pozoru niewielkie błędy wyznaczenia położenia kropek w połączeniu z nieproporcjonalną długością wektorów w układzie kamery do długości wektorów w układzie płytki powodują, że odwrócenie macierzy transformacji z układu płytki do kamery jest obarczone tak dużym błędem, że uniemożliwia to wykorzystanie otrzymanego w ten sposób wyniku. Aby się o tym przekonać zaimplementowano własną metodę obliczania transformacji między tymi dwoma układami oraz wykorzystano funkcję wbudowaną biblioteki OpenCV, co pozwoliło potwierdzić przypuszczenia. Metoda autorska z uwagi na dyskwalifikujące ją błędy opisana została w załącznikach. Do obliczenia kalibracji układu robot-kamera znaleziono alternatywne rozwiązanie również wykorzystujące wspomnianą funkcję biblioteki OpenCV. Dokładny opis tego problemu znajduje się w następnym rozdziale.

4 Procedura automatycznej kalibracji

4.1 Rozwiązanie problemu kalibracji robot-kamera

Istnieje kilka sposobów wyznaczenia transformacji między układem kamery a robota, w wyniku badań prowadzonych nad tym problemem udało się znaleźć wady i zalety niektórych z nich. Z uwagi na fakt, że aby nieznana macierz transformacji o sześciu niezależnych parametrach była dość dokładna, potrzebny jest zbiór co najmniej kilkunastu wektorów położeń kropek w układzie bazowym robota oraz w odpowiadających im położeń układzie kamery bądź na zdjęciu. Oznacza to, że bez względu na wybór ostatecznej metody kalibracji potrzebna będzie optymalizacja wyniku poprzez minimalizację współczynnika błędu. Poniżej opisano trzy metody, z czego drugą i trzecią zaimplementowano i po ich porównaniu zdecydowano się wdrożyć metodę trzecią.

Oznaczenia i definicje

W dalszej części pracy funkcja sum(A), gdzie A jest dowolną macierzą, oznacza sumę wszystkich elementów macierzy A, a funkcja sum2(A) - sumę kwadratów wszystkich elementów. Parą wektorów nazwano zbiór dwóch czterowektorów: wektor położenia dowolnego punktu w układzie kamery oraz wektor położenia tego samego punktu w układzie robota uzupełnione jedynkami w miejscu czwartej współrzędnej.

Metoda działająca w oparciu o położenia i orientacje układów płytki, kamery i bazowego

Opisywana metoda została zaprojektowana na bazie metody kalibracji systemu wielorobotowego [11]. Polega ona na wyznaczeniu w każdym kolejnym i-tym położeniu robota: macierzy transformacji ${}^{G}_{P}T^{(i)}$ między układem płytki, a układem bazowym (realizowane w oparciu o odczyt ze struktury ramowej MRROC++) oraz macierzy transformacji ${}^{C}_{P}T^{(i)}$ między układem płytki, a układem kamery (realizowane w oparciu o wbudowaną metodą `FindExtrinsicCameraParams2` biblioteki OpenCV). Funkcja celu w tej metodzie ma postać:

$$\min_{{}^{G}_{C}T} \sum_{i=1}^{N} \text{sum2}\left({}^{G}_{P}T^{(i)} - {}^{G}_{C}T \cdot {}^{C}_{P}T^{(i)}\right).$$

Zaletą tej metody jest fakt, że wykorzystuje ona nie tylko informację o położeniu, ale i o orientacji wzajemnej układów. Jej wadą jest fakt, że optymalizacja może zostać przeprowadzona dopiero dla kompletu danych pomiarowych, z uwagi na przewidywany czas jej trwania. Funkcja `FindExtrinsicCameraParams2` jako argument przyjmuje minimum cztery wektory położenia punktów charakterystycznych w pewnym układzie sceny

oraz odpowiadające mu pary współrzędnych x, y tych punktów na zdjęciu wyrażone w pikselach. W rezultacie funkcja zwraca transformację między układem sceny a układem kamery[26]. Niestety z uwagi na m.in. rozmiary płytki, błędy wyznaczenia parametrów wewnętrznych kamery itp., wyznaczone w ten sposób macierze nie są poprawne, co uniemożliwia stosowanie tej metody.

Metoda działająca w oparciu o zestaw N par wektorów

Opisywana metoda zaimplementowana i przetestowana w pracy polega na minimalizacji funkcji celu w oparciu o pary wektorów, czyli wyłącznie w oparciu o położenia np. środka płytki dla różnych ustawień robota. Jeśli kolumnowe wektory w układach kamery i robota zapisze się w postaci dwóch macierzy Nx4, można zaproponować funkcję celu:

$$\min_{{}^{G}_{C}T} \sum_{i=1}^{N} \mathrm{sum2}\left(\left[{}^{G}\mathbf{p}_1 \quad {}^{G}\mathbf{p}_2 \ldots {}^{G}\mathbf{p}_N\right] - {}^{G}_{C}T \cdot \left[{}^{C}\mathbf{p}_1 \quad {}^{C}\mathbf{p}_2 \ldots {}^{C}\mathbf{p}_n\right]\right).$$

Również obliczenia w tej metodzie, z uwagi na czas trwania optymalizacji[27], mogą być przeprowadzane po zebraniu kompletu danych pomiarowych. Dodatkowo metodę odróżnia od poprzedniej fakt, że wykorzystuje ona tylko położenia środka płytki - tracona jest część informacji o orientacji. Wykorzystanie jednak informacji o środku płytki, a nie o czterech kropkach zmniejsza błędy argumentów wejściowych procedury.

Po kalibracji systemu przy użyciu tej metody na etapie weryfikacji okazało się, że błędy położeń znajdowanych przez robota w oparciu o dane z kamery dochodzą do 3-4 cm, co przekraczało dopuszczalne wartości.

Metoda działająca w oparciu o zestaw N wektorów w układzie bazowym i odpowiadających im par współrzędnych w pikselach

Metoda, która ostatecznie została zaimplementowana w systemie, również wykorzysuje wbudowaną funkcję `FindExtrinsicCameraParams2` biblioteki OpenCV [7]. Dzięki temu, że jako argument wejściowy przyjmuje ona N wektorów położeń środka płytki w układzie bazowym robota oraz współrzędne tych punktów na zdjęciach, w wyniku otrzymuje

[26] Funkcja ta zwraca trzyelementowy wektor rotacji oraz wektor translacji, które można przekształcić do postaci macierzy jednorodnej korzystając z przekształcenia Rodriguesa [7]

[27] Czas trwania metody związany jest z faktem, że wyniki optymalizacji w dużym stopniu zależą od zadanych punktów początkowych, dlatego przeprowadzano ją dla bardzo wielu ich kombinacji. Uzasadnieniem tego było również zabezpieczenie, by znaleziony wynik był prawidłowy niezależenie od ustawienia kamery.

się od razu macierz transformacji ${}^{G}_{C}T$ między układem globalnym, a układem kamery, a zatem niewymagana jest żadna dodatkowa optymalizacja oprócz wbudowanej, będącej częścią opisywanej funkcji. Skutkuje to niewielkim czasem wymaganym na obliczenia i, w związku z tym, mogą one być wykonywane bezpośrednio po każdym nowym pomiarze, co umożliwia porównywanie wyników na bieżąco.

Działanie funkcji `FindExtrinsicCameraParams2` polega najpierw na znalezieniu homografii w oparciu o zestaw danych pomiarowych, a później na jej bazie, na wyznaczeniu macierzy transformacji między układem kamery, a układem zadanym (w rozważanym przypadku bazowym układem robota). Korzystając z równania (14) i oznaczeń z rozdziału 2.2.3 oraz zapisując 9-elementową macierz homografii w postaci

$$H = [h_1, h_2, h_3] = sM[r_1, r_2, \mathbf{t}]$$

można zdefiniować funkcję celu:

$$\min_{H} \sum_{i=1}^{N} \text{norm} \left(\begin{bmatrix} \widetilde{{}^{A}x_i} \\ \widetilde{{}^{A}y_i} \\ 1 \end{bmatrix} - H \cdot {}^{G}\mathbf{p}_i \right),$$

zapewniając spełnienie równań więzów:

$$h_1^T M^{-T} M^{-1} h_2 = 0$$

oraz

$$h_1^T M^{-T} M^{-1} h_1 = h_2^T M^{-T} M^{-1} h_2.$$

będących konsekwencją zależności [7]:

$$r_1^T r_2 = 0,$$

$$||r_1|| = ||r_2|| \iff r_1^T r_1 = r_2^T r_2.$$

Postać macierzy transformacji można bezpośrednio wyznaczyć ze znalezionej macierzy H oraz parametrów wewnętrznych kamery. Współczynnik skalujący wyznacza się korzystając z warunku:

$$s = ||M^{-1} h_1||,$$

wówczas:

$$r_1 = \frac{1}{s} M^{-1} h_1,$$

$$r_2 = \frac{1}{s} M^{-1} h_2,$$

$$r_3 = r_1 \times r_2,$$

oraz

$$\mathbf{t} = \frac{1}{s} M^{-1} h_3,$$

stąd:

$$^{G}_{C}T = \begin{bmatrix} r_1 & r_2 & r_3 & \mathbf{t} \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Opis przykładowej kalibracji z wykorzystaniem tej metody oraz analiza błędów zostały przedstawione w rozdziale 5.

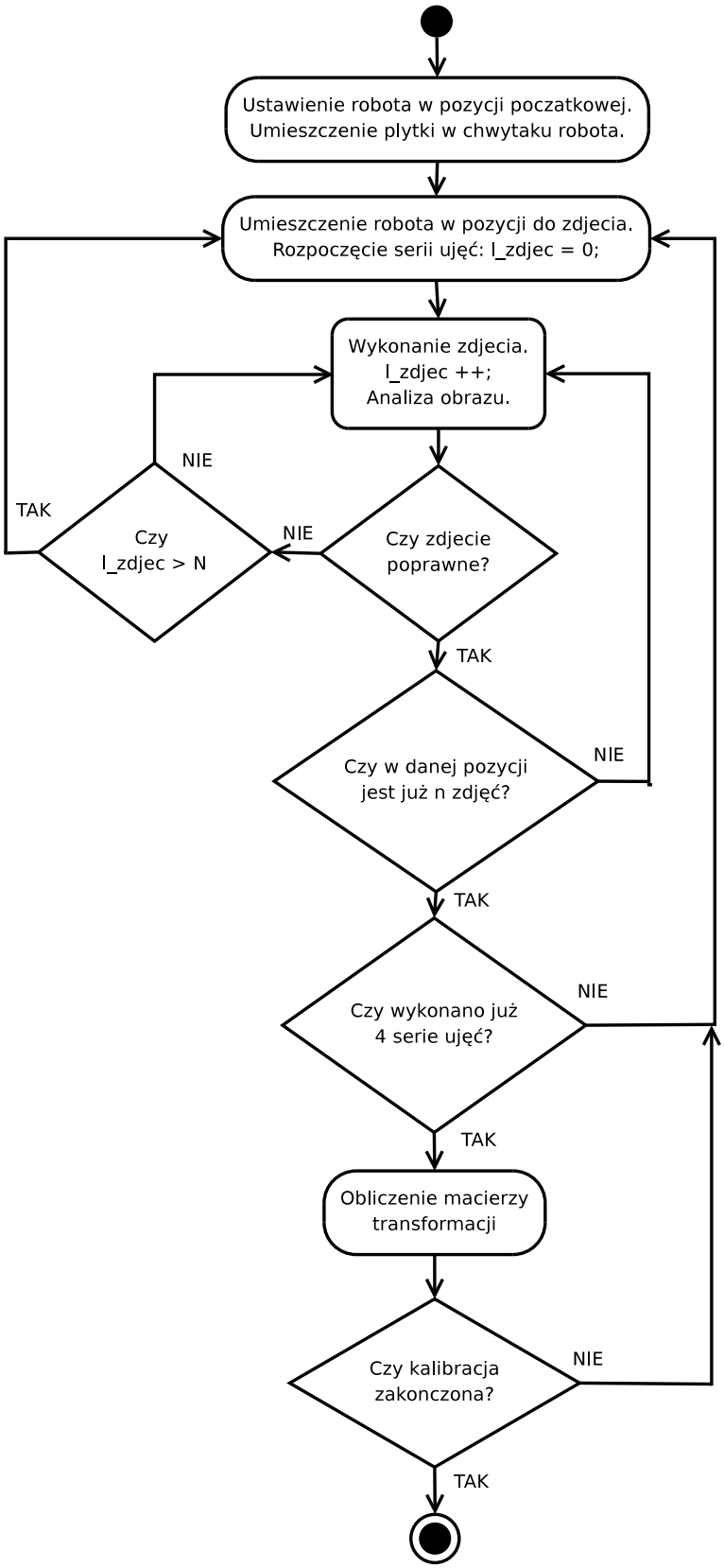
4.2 Scenariusz działana

Główny algorytm programu do automatycznej kalibracji został przedstawiony na schemacie 19.

Po ręcznym umieszczeniu płytki w chwytaku robota i uruchomieniu programu rozpoczyna się powtarzalny kilkukrotnie proces. Wykorzystywany generator „wodzenia za nos" umożliwia interakcję z operatorem za pomocą przycisku *Trigger*, co znalazło zastosowanie w opisywanym algorytmie. Zadaniem użytkownika jest ustawienie manipulatora w pożądanej pozycji poprzez mechaniczne przeciągnięcie go we wskazane miejsce. Następnie za pomocą wspomnianego przycisku zleca się wykonanie serii zdjęć dla danego ujęcia. Każde zdjęcie jest wstępnie analizowane. Wynik może być dwojaki:

1) Jeśli okaże się, że zdjęcie jest niepoprawne, np. rozmazane i w procesie segmentacji nie udało się wyznaczyć położenia czterech kropek, płytka w tym samym położeniu fotografowana jest ponownie. Może się jednak zdarzyć, że na zdjęciu znajdują się tylko trzy kropki albo w miejscu jednej z nich powstaje refleks od oświetlenia i wówczas, przy przekroczeniu pewnej granicznej liczby N wykonanych w ten sposób fotografii, manipulator powinien zostać przesunięty w inne miejsce. Taki wynik nie jest zapamiętywany.

2) Gdy natomiast analiza zdjęcia zakończy się sukcesem następuje wykonanie jeszcze $n - 1$ takich samych zdjęć, żeby uniezależnić wynik od chwilowych szumów, czy przypadkowego poruszenia płytki. Następnie pomiar jest zapisywany i, jeśli jest to co najmniej czwarta pozycja, obliczana jest aktualna macierz transformacji między układami współrzędnych.



Rysunek 19: Główny algorytm programu.

Macierz transformacji obliczana jest po każdym pomiarze począwszy od czwartego, gdyż wiąże się to z warunkiem na minimum danych wymaganych do kalibracji. Takie podejście ma dwie zalety: po pierwsze obserwacja na bieżąco zmian macierzy po każdym kolejnym ujęciu daje operatorowi informację, kiedy dalsze pomiary nie przynoszą już żadnych zmian, po drugie, w wypadku celowego lub przypadkowego przerwania pomiarów, możliwe jest zapisanie najlepszego dotychczasowego wyniku.

Aby zakończyć proces kalibracji można użyć przycisku *Trigger* ponownie, bezpośrednio po obliczeniach. Ustawi to ramię manipulatora w pozycji końcowej z rozwartymi szczękami.

4.2.1 Implementacja algorytmu w systemie MRROC++

Instrukcja `Move`

Instrukcja `Move` jest główną instrukcją ruchu. Przemieszcza ona efektor, jak również dokonuje transferu danych z czujników. W zadaniu kalibracji funkcja ta działa w *ECP*. Można podzielić ją na dwie główne części - w początkowej fazie wykonywana jest instrukcja „pierwszy krok" generatora trajektorii (opisanego w kolejnym punkcie), a następnie w pętli wykonywany jest ciąg kolejnych kroków. Po każdym kroku podejmowana jest decyzja odnośnie dalszego ruchu robota i wywoływana jest komenda „wykonaj ruch".

Generator trajektorii

Generatory trajektorii (ruchu) są obiekatmi, których zadaniem jest nie tylko obliczanie położenia robotów, do których mają się przesunąć w kolejnym makrokroku, ale również przygotowanie nowego rozkazu dla efektorów, które potem zostaną wykorzystane przez instrukcję `Move`. W generatorze muszą zostać zaimplementowane metody: `first_step`, której wykonanie ogranicza się zazwyczaj do odczytania położeń efektorów oraz instrukcje `execute_motion`, której zadaniem jest wykonanie kroku i `next_step` - do obliczenia parametrów (np. współrzędnych wewnętrznych) kolejnego makrokroku.

Do kalibracji utworzono zadanie, który wykorzystuje generatory: „smooth generator" do ustawiania manipulatora w pozycji początkowej i końcowej, własny generator napisany w oparciu „tff nose run generator" do sterowania siłowego robotem oraz oddzielny generator do komunikacji z FraDIA. Uzasadnieniem tego, jest fakt, że ruch robota i analiza obrazów wykonywane są niezależnie i sekwencyjnie.

4.2.2 Planowanie ruchów robota w celu otrzymania różnorodnych i poprawnych zdjęć

Aby kalibracja mogła zostać wykonana poprawnie, należy zaproponować sposób planowania ruchów robota. W początkowych etapach pracy zdefiniowano kilkanaście odpowiednio dobranych pozycji. Umożliwiało to poruszanie manipulatorem bez ingerencji użytkownika. Jednakże takie rozwiązanie miało sens wyłącznie dla bardzo ograniczonych zmian położenia kamery. Montaż kamery w innym miejscu wiązałby się z koniecznością doboru zupełnie innych położeń[28] oraz wprowadzenie ich do systemu. Idealnym rozwiązaniem byłaby implementacja algorytmu planowania ruchów manipulatora na podstawie analizy obrazu z kamery, ale w sytuacji, gdy robot działa w nieskalibrowanym układzie z kamerą, zdecydowano, że jest to rozwiązanie zbyt pracochłonne.

Wykorzystanie pozycyjno-siłowego sterowania robotem przerzuca na operatora obowiązek wyznaczania kolejnych pozycji tak, by w pełni dostosować się do aktualnego ustawienia kamery. W końcowej części pracy zamieszczono analizę wyników kalibracji. Dzięki niej można ustalić optymalną liczbę pozycji, które musi wyznaczyć użytkownik.

4.3 Budowa i interfejs aplikacji

4.3.1 Podział na zadania

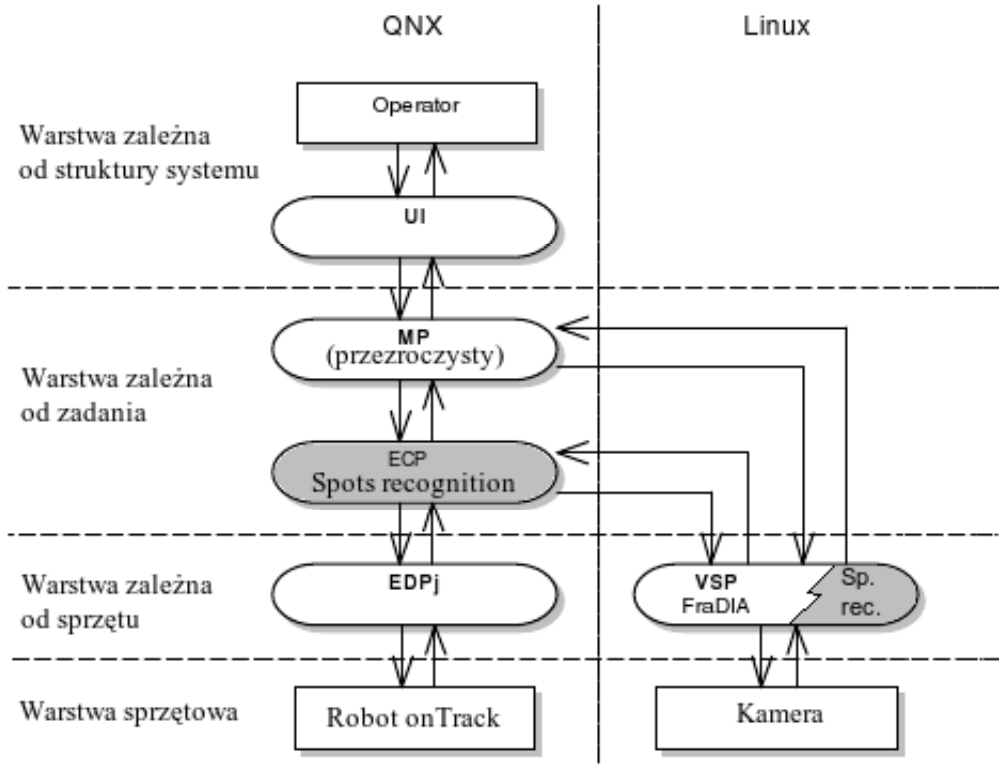
Program do automatycznej kalibracji został napisany jako dwa komunikujące się ze sobą zadania:

- zadanie programowej struktury FraDIA, które służy do akwizycji danych z obrazu otrzymywanego z kamery, rozpoznawania i wyszukiwania kropek oraz minimalizacji i obliczania macierzy transformacji układów współrzędnych,
- zadanie wykonywane przez *ECP* systemu MRROC++, którego celem jest nadzór nad sterowaniem robotem, zlecanie rozpoczęcia obliczeń oraz przekazywanie informacji o położeniu płytki w układzie bazowym.

Schemat systemu z wyszczególnieniem ww. zadań został umieszczony na rysunku 20.

Sterownik zbudowany na bazie MRROC++ działa pod kontrolą systemu QNX, a FraDIA - pod kontrolą Linuksa, dlatego program do kalibracji jest w zasadzie dwiema oddzielnymi wieloprocesowymi aplikacjami. Moduł komunikacji między MRROC++, a

[28] Pod pojęciem położenia należy rozumieć współrzędne końcówki chwytaka w układzie robota oraz jej orientację

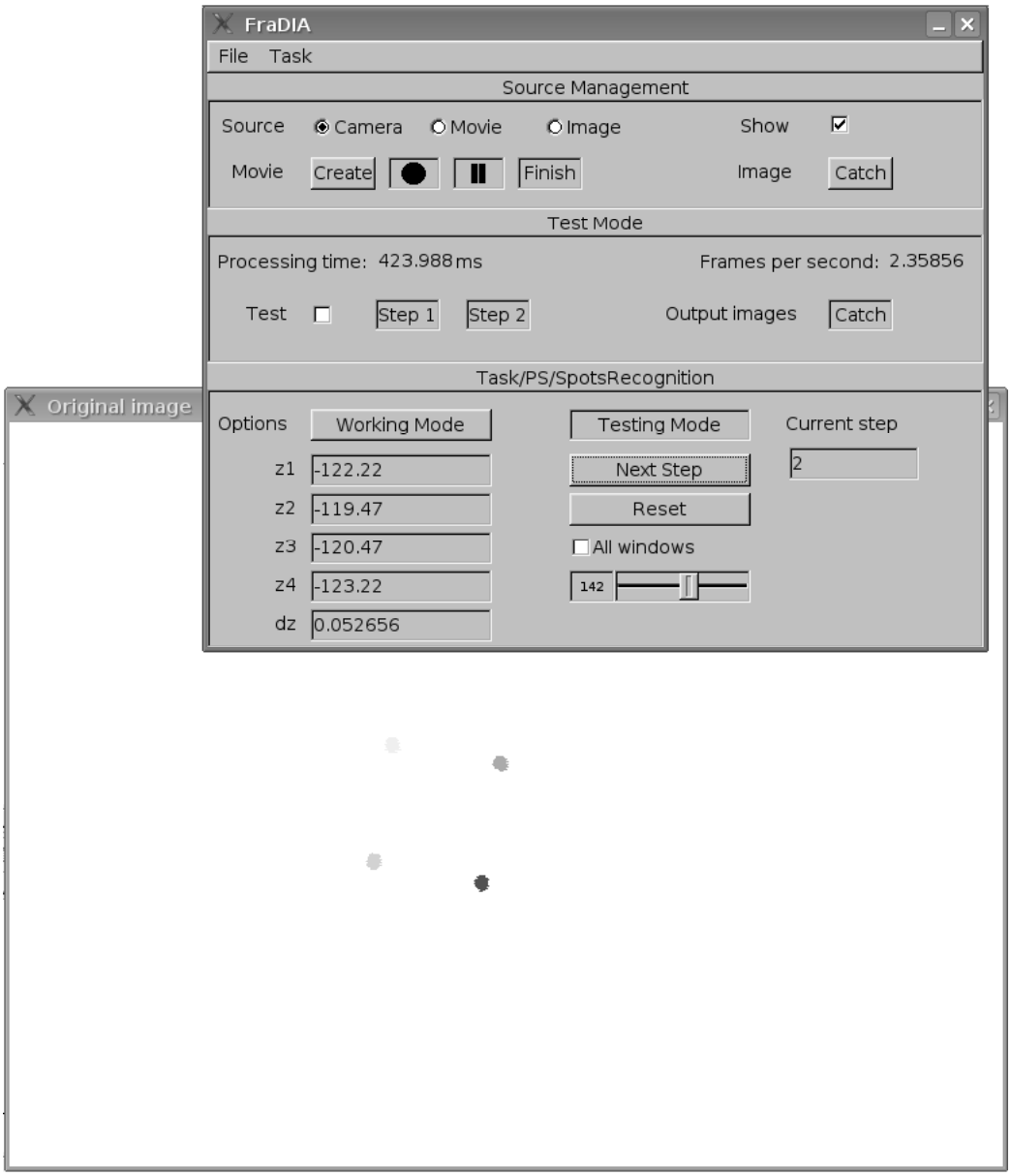


Rysunek 20: Budowa systemu MRROC++ z uwzględnieniem zadań do kalibracji.

FraDIA opisany jest w raporcie [9]. W MRROC++ do komunikacji między procesami wykorzystywane są tzw. bufory danych.

4.3.2 Interfejs aplikacji

Z uwagi na fakt, że sterowanie kalibracją odbywa się poprzez proces *UI* systemu MRROC++, interfejs aplikacji ogranicza się w zasadzie do udostępnionego domyślnego interfejsu zadania. Wciśnięcie przycisku START w panelu sterowania *UI* ustawia manipulator w pozycji początkowej oraz odpowiedni rozstaw szczęk, by możliwe było wsunięcie płytki. Następnie robot może być ustawiony w dowolnym położeniu. Przechodzenie od fazy przesuwania ramieniem do fazy robienia zdjęć odbywa się samoczynnie, jeśli do manipulatora nie jest przykładana siła przez okres ok. 2 sekund. Taka sytuacja jest sygnalizowana pojedynczym dźwiękiem i robot zatrzymywany jest na kolejne dwie sekundy. Pozwala to zniwelować wpływ „pływania" robota, czyli ruchu mimo braku siły. Przejście od fazy zdjęć do fazy ruchu również odbywa się automatycznie i jest sygnalizowane dźwiękiem: podwójnym, jeśli pozycja została zapisana lub wielkokrotnym, jeśli okazała się nieprawidłowa (niemożliwe było poprawne sfotografowanie płytki w tej pozycji).



Rysunek 21: Panel aplikacji w trybie podglądu.

Sterowanie akwizycją danych z obrazów odbywa się w ramach komunikacji MRROC++ - FraDIA, ta ostatnia pełni rolę serwera nasłuchującego poleceń, otrzymuje informacje o położeniu środka płytki w układzie bazowym robota oraz przekazuje wyniki otrzymane w procesie minimalizacji.

Mimo faktu, że w trakcie kalibracji interakcja użytkownika z FraDIA nie jest potrzebna, wykorzystując możliwość tworzenia w prosty sposób interfejsów, zaprojektowano panel, wykorzystywany m.in. do testowania aplikacji. Zrzut ekranu panelu umożliwiającego pracę w „trybie podglądu" znajduje się na rysunku 21.

Jako moduł FraDIA, program może pracować w dwóch trybach:

1.) „working mode" czyli standardowy tryb pracy, który jest normalnie wywoływany przez MRROC++. Nie wymaga on żadnej interakcji ze strony użytkownika, umożliwia jedynie podgląd „on-line" obrazu kamery, co ułatwia planowanie ruchów manipulatora.

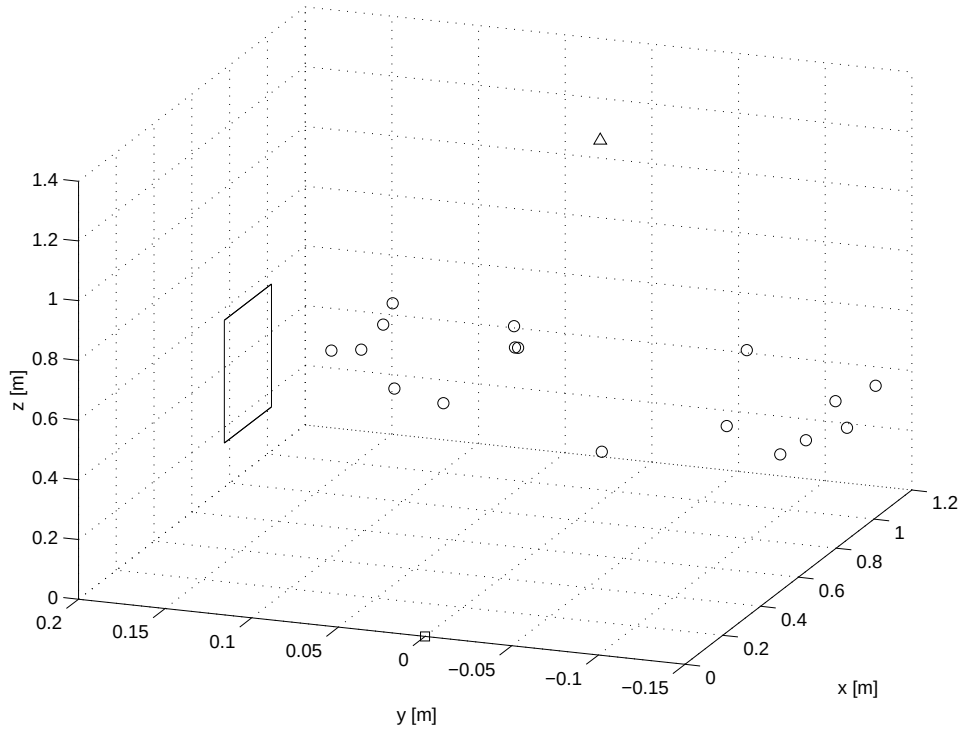
2.) „testing mode" czyli tryb podglądu, gdzie możliwa jest kontrola wszystkich kroków progowania i segmentacji. W tym trybie poprzez przycisk *Next Step* realizowane jest stopniowe przechodzenie przez proces progowania, segmentacji, indeksowania itp. oraz obserwacja wyników. Możliwa jest również ręczna modyfikacja progu w pewnym zakresie przy pomocy paska przewijania.

5 Analiza wyników i wnioski

W tym rozdziale opisany został przykładowy proces kalibracji wraz z analizą wyników i błędów, sprawdzenie poprawności wyznaczonej macierzy, jak również sformuowane zostały wnioski. W końcowej części rozdziału zamieszczono opis problemów, których można było uniknąć oraz propozycje dalszych prac.

5.1 Przykładowy proces kalibracji

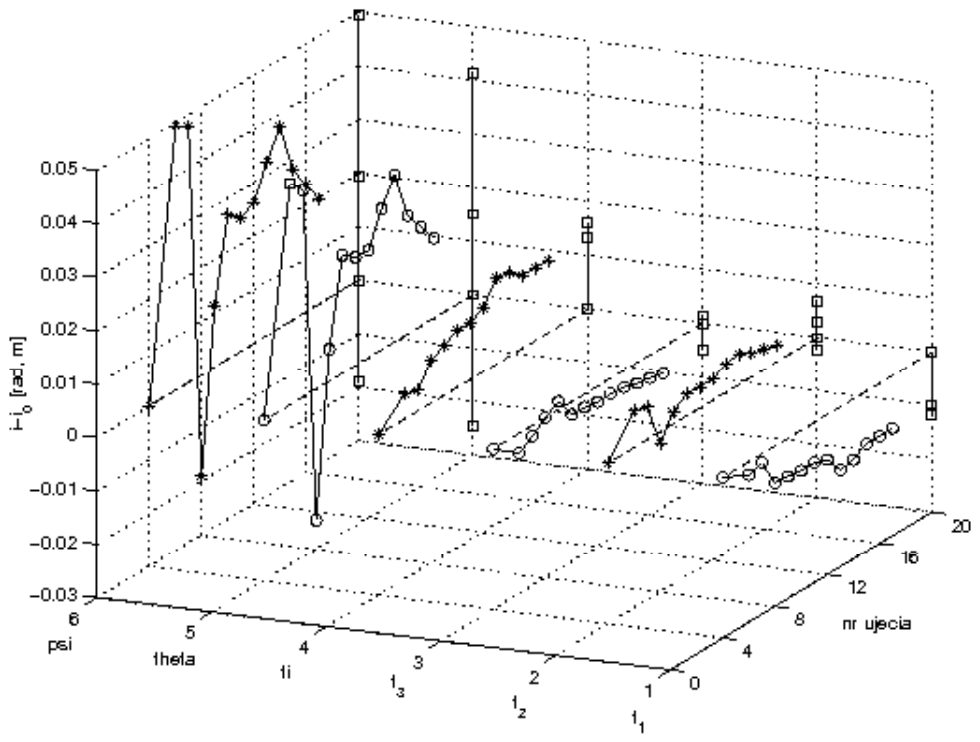
Niemożliwe byłoby ukończenie niniejszej pracy bez przeprowadzenia przykładowej kalibracji sprawdzającej poprawność użytych metod oraz dogłębnej analizy wyników. Korzystając z wdrażanego oprogramowania wykonano serię 17 pomiarów starając się, by obejmowały one jak największy obszar pracy manipulatora widziany przez kamerę. Położenie punktów w przestrzeni trójwymiarowej w układzie robota przedstawiono na rysunku 22.



Rysunek 22: Położenie wyników pomiarów w przestrzeni 3W.

Na wykresie znajduje się zestaw punktów pomiarowych (oznaczonych kółkami) przedstawiony w układzie bazowym robota. Położenie robota (punkt (0,0,0)) oznaczono niewielkim kwadratem, a wyliczone położenie kamery - trójkątem. Za pomocą prostokąta przedstawiono zakres fotografowanego obszaru w płaszczyźnie XZ - dość dobrze jest on widoczny wzdłuż osi Y.

Jak już wspomniano algorytm został zaprojektowany tak, że po minimum czterech pomiarach oraz z każdym dodatkowym położeniem obliczana jest macierz transformacji układów. Aby wyznaczyć, np. optymalną liczbę zdjeć, jakie trzeba wykonać, żeby wynik kalibracji był najlepszy, dobrze jest przedstawić zmiany elementów tej macierzy w zależności od numeru pomiaru. Z uwagi na fakt, że elementy macierzy rotacji są zależne i nie ma sensu analiza ich wszystkich na raz, zdecydowano się zamieścić na wykresie kąty Eulera (ϕ, θ, ψ), które obliczono na jej podstawie. Dodatkowo umieszczono elementy wektora translacji (t_1, t_2, t_3). Aby móc porównać skalę zmian tych sześciu parametrów na rysunku (23) umieszczono wyłącznie moduł różnicy aktualnej wartości parametru oraz wartości początkowej ($|i - i_0|$, gdzie $i \in t_1, t_2, t_3, \phi, \theta, \psi$).



Rysunek 23: Zmiana parametrów kalibracji w trakcie procesu.

Na wykresie przedstawiono zmiany parametrów kalibracji w trakcie jej trwania względem wartości wyznaczonych w pierwszym kroku (czyli $i - i_0$, ozn. na zmianę kółkami i gwiazdkami, linia ciągła dla poprowadzenia wzroku) w funkcji numeru położenia płytki względem kamery. Pionowymi liniami oznaczono dla porównania zakres tych zmian, a kwadratami: wartość minimalną ($min(i - i_0)$), punkt 0, wartość końcową ($i_n - i_0$) oraz wartość maksymalną ($max(i - i_0)$). Przerywane linie poziome oznaczają poziom referencyjny (i_0).

Pobieżna analiza wykresu pozwala stwierdzić, że zmiany elementów wektora translacji są bardzo niewielkie, ledwie przekraczają $1cm$ przez cały czas trwania kalibracji. Mają

również charakter monotoniczny oraz skok zmian maleje wraz z liczbą ujęć. Oznacza to, że położenie kamery w układzie robota wyznaczane jest niemal z tą samą dokładnością od samego początku. Trochę inny charakter zmian wykazują kąty Eulera - ich zmiana jest periodyczna z tendencją gasnącą. Oznacza to, że wyznaczenie orientacji kamery w układzie bazowym robota jest trudniejsze, a amplituda zmian pokazuje, że i mniej dokładne. Maksymalna rozbieżność wartości kątów wynosi ok 3°.

5.2 Algorytmy sprawdzania poprawności otrzymanych wyników

Kalibracja układu kamera-robot polega na wyznaczeniu macierzy transformacji ich układów, czyli położenia układu kamery w układzie bazowym manipulatora. Przed kalibracją położenie jest nieznane lub wyznaczone w sposób niedokładny, dlatego porównanie otrzymanych liczb z „prawidłowym wynikiem" jest w zasadzie niemożliwe. Dlatego należało opracować metodę pozwalającą sprawdzić wyznaczone współczynniki pod kątem ich dokładności.

Jak już pokazano zachodzi:

$$^{G}p = {}^{G}_{C}T \cdot {}^{C}p,$$

gdzie ww. wartości są ścisłe. Niestety otrzymywana w wyniku kalibracji macierz transformacji nie jest wyznaczona idealnie, dlatego równanie to nie jest spełnione dokładnie. Można zdefiniować wektor błędu:

$$\delta p = \widetilde{{}^{G}_{C}T} \cdot {}^{C}p - {}^{G}p, \tag{35}$$

gdzie $\widetilde{{}^{G}_{C}T}$ jest wynikiem kalibracji. Fizyczną interpretacja wektora błędu jest różnica między punktem zadanym (czyli wyznaczonym na podstawie informacji o położeniu w układzie kamery, na zdjęciu, itp.) a pożądanym (czyli takim, do którego manipulator powinien przenieść przedmiot, przesunąć się, itd.). Oczywiście nie uwzględnia on uchybów wyznaczenia pozycji końcówki ramienia.

Istnieją dwa sposoby sprawdzenia poprawności otrzymanego wyniku: pierwszy polega na analizie danych zebranych podczas pomiarów (wadą tego sposobu jest operowanie na danych, które posłużyły do kalibracji, a więc są w pewnym stopniu charakterystyczne, zaletą natomiast łatwość ich analizy), drugi to przeprowadzenie niezależnego eksperymentu sprawdzającego i analiza uzyskanych w ten sposób nowych danych (w przeciwieństwie do poprzedniej metody, ta operuje na niezależnych nowych wektorach, jednak uzyskanych w wyniku pomiarów o ograniczonej dokładności).

5.2.1 Analiza danych pomiarowych

Zmiany błędów w kolejnych krokach optymalizacji

Jak już wspomniano macierz $\widetilde{{}^{G}_{C}T}$ wyznaczana jest po raz pierwszy po uzyskaniu czterech par wektorów położeń w układzie bazowym i kamery[29], a później z każdą dodatkową parą[30] aktualizowana[31].

Oznaczenia i definicje

Niech, zgodnie z wcześniejszą konwencją oznaczeń, δp_i^j oznacza wektor błędów obliczony w j-tym kroku optymalizacji dla i-tej pary wektorów. Niech Δp_i^j równa jest 2-normie tego wektora, czyli jest pewną uniwersalną miarą błędu macierzy $\widetilde{{}^{G}_{C}T}^{j}$ wyznaczonej w j-tym kroku w działaniu na i-tą parę wektorów. Z uwagi na wykorzystywaną metodę, parą wektorów oznaczono wektor położenia środka płytki w układzie bazowym robota oraz współrzędne położenia tego punktu na zdjęciu wyrażone w pikselach.

Dla macierzy obliczonej w j-tym kroku można zdefiniować dwa rodzaje błędów:

- błąd aktualny:

$$err_{curr}^{(j)} = \frac{1}{j}\sum_{k=1}^{j}\Delta p_k^{(j)},$$

  czyli uśredniony błąd dla j wektorów błędów pochodzących od par wektorów znanych w j-tym kroku, czyli tych, na podstawie których wyznaczana jest macierz transformacji; błąd ten ma bardziej charakter informacyjny niż porównawczy,

- błąd całkowity:

$$err_{tot}^{(j)} = \frac{1}{N}\sum_{k=1}^{N}\Delta p_k^{(j)},$$

  czyli uśredniony błąd dla wszystkich N wektorów błędów pochodzących od wszystkich par wektorów zapisanych w trakcie pomiarów; błąd ma charakter porównawczy

[29] Użyto tu pewnego skrótu: procedura wyznaczająca macierz ${}^{G}_{C}T$ jako danych wejściowych wymaga n wektorów $\mathbb{R}^3$ położeń punktów w układzie robota oraz tyle samo wektorów $\mathbb{R}^2$ analogicznych punktów na zdjęciu. Jednak do sprawdzenia poprawności wyniku należy wykorzystać wektory $\mathbb{R}^3$ punktów w układzie kamery, które należy wyznaczyć na podstawie zdjęć w inny sposób - w tym wypadku zastosowano autorską metodę minimalizacji, której poświęcono znaczną część pracy.

[30] w tym wypadku oprócz piątej pary - wyznaczona w tym kroku macierz znacznie się różniła od wszystkich pozostałych i została pominięta. Oczywiście należy zaznaczyć, że pominięto tylko cząstkowy wynik, a nie dane pomiarowe, także czynność ta nie miała żadnego wpływu na przebieg, ani wynik kalibracji.

[31] Aktualizacja macierzy wyznaczonej w j-tym kroku polega oczywiście na obliczeniu nowej macierzy od podstaw dla wszystkich $j+1$ par wektorów, wynik zastępuje starszy.

- można przy jego pomocy jednoznacznie zdecydować, która macierz jest lepsza dla danego zestawu danych pomiarowych.

Uśrednione błędy zostały przedstawione w tabeli 1 oraz zobrazowane na rysunku 24.

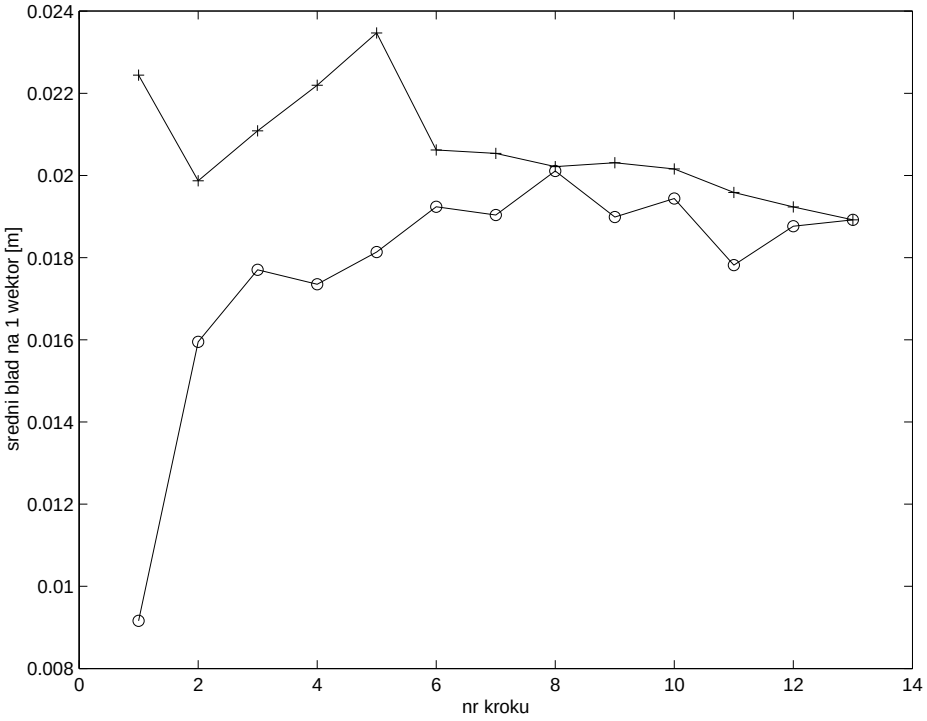
Krok (j)	L. par wektorów	$err_{curr}^{(j)}$ $[m]$	$err_{tot}^{(j)}$ $[m]$
1	4	0.0092	0.0224
2	6	0.0160	0.0199
3	7	0.0177	0.0211
4	8	0.0174	0.0222
5	9	0.0181	0.0235
6	10	0.0192	0.0206
7	11	0.0190	0.0205
8	12	0.0201	0.0202
9	13	0.0190	0.0203
10	14	0.0194	0.0202
11	15	0.0178	0.0196
12	16	0.0188	0.0192
13	17	0.0189	0.0189

Tablica 1: Zmiana błędów w trakcie trwania kalibracji.

Dane można zinterpretować następująco. Dla czterech pierwszych par wektorów (czyli w pierwszym kroku) bardzo łatwo jest znaleźć macierz transformacji o praktycznie zerowym błędzie aktualnym. Z każdą kolejną parą wektorów błąd aktualny rośnie aż do pewnego momentu, kiedy zaczyna się stabilizować - wówczas można uznać, że macierz jest dość dokładnie dopasowana do pomiarów z całego zakresu, z którego zbierane są pomiary. Wówczas błąd całkowity zaczyna powoli maleć. W omawianym przypadku taka sytuacja występuje od 4 kroku (od 8 par wektorów). Można przypuszczać, że gdyby zebrać bardzo dużo danych pomiarowych w pewnym momencie błąd całkowity również ustabilizowałby się na poziomie stabilizacji błędu aktualnego.

Ostateczna wartość błędu dla wszystkich zebranych danych wynosi ok. 1.89 cm, co oznacza, że tyle wynosi średnia odległość między położeniem pożądanym, a osiągniętym. Z uwagi na fakt, że poziom ten jest bardzo blisko poziomu stabilizacji błędu aktualnego, można oczekiwać, że nie da się znacząco zmniejszyć tego błędu przy kalibracji tą metodą.

Jak już zostało napisane, wektory w układzie kamery używane w powyższych testach wyznaczone były przy użyciu autorskiej metody, a kalibracja wykonywana była w oparciu



Rysunek 24: Zmiana błędów w trakcie trwania kalibracji.

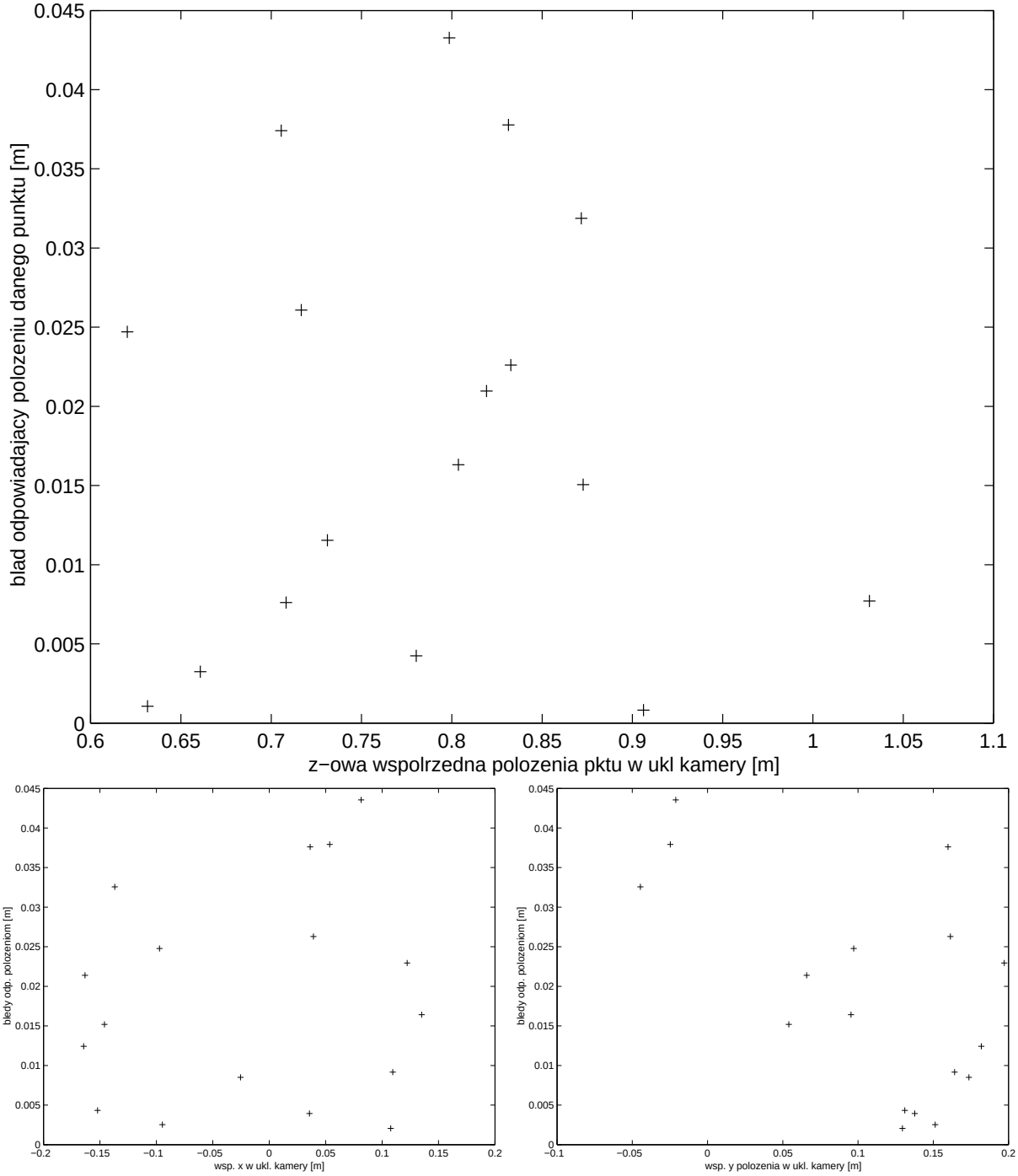
Na wykresie przedstawione zostały zmiany wcześniej zdefiniowanych błędów wraz z kolejną parą wektorów (kolejnym krokiem optymalizacji). Kółkami oznaczono wartości błędów aktualnych (err_{curr}), a krzyżykami - wartości błędów całkowitych (err_{tot}). Linię ciągłą umieszczono dla poprowadzenia oka.

o współrzędne punktów na obrazku z wykorzystaniem funkcji z biblioteki OpenCV. Pierwsza wartość błędu aktualnego na poziomie 0.9 cm pokazuje więc, że położenie w układzie kamery obliczone gotową metodą jest bardzo zbliżone do położenia wyznaczonego metodą autorską, aczkolwiek oznacza to również, że niezgodność metod stanowi ok. 50% wpływ na wartości błędów. Niestety pokazanie tego jest niezwykle trudne, konieczna byłaby bowiem implementacja doskonałej metody wyznaczania położenia punktów w układzie kamery na podstawie informacji ze zdjęć przy dokładnej znajomości położenia tego punktu względem końcówki manipulatora.

Analiza wpływu geometrii układu do kalibracji na błędy

W tym paragrafie przedstawiono zależności norm wektorów błędów $\Delta p_k^{(N)}$ w oparciu o całkowity wynik kalibracji, czyli w N-tym kroku, od rzeczywistych położeń punktów w układzie kamery $({}^C x_k, {}^C y_k, {}^C z_k)$, dla których zostały wyznaczone. Wykresy wspomnianych

zależności znajdują się na poniższych rysunkach.



Rysunek 25: Wartości błędów $\Delta p_k^{(N)}$ w funkcji ${}^C x_k, {}^C y_k, {}^C z_k$.

Na wykresach znajdują się wartości norm wektorów błędów obliczone dla zmierzonych położeń w układzie kamery w funkcji trzech współrzędnych tych położeń: z- (na górze) oraz x- (z lewej) i y-owej (z prawej).

We wcześniejszych pracach, które wykorzystywały autorską metodę znajdowania współrzędnych kropek w układzie kamery można było zauważyć silną korelację błędów od położenia w tym układzie, szczególnie od z-owej składowej. Oznaczało to, że jakość tej metody była uzależniona w dużej mierze od odległości płytki od kamery. Analizując powyższe wykresy można zauważyć brak korelacji błędów. Rozwiązanie pozbawione korelacji przestrzennej błędów ma pewną zaletę – znaleziona macierz transformacji układów powinna być „jednakowo dobra" w całym zakresie pracy manipulatora.

5.2.2 Doświadczenie polegające na poszukiwaniu kropek

Aby sprawdzić poprwność znalezionej macierzy zaproponowano doświadczenie, w którym po kalibracji, manipulator powinien odnaleźć zadany punkt. Doświadczenie polegało na pomiarze dwóch punktów w przestrzeni roboczej manipulatora. Współrzędne każdego z punktów w układzie kamery (${}^{C}p_1, {}^{C}p_2$) wyznaczono przy pomocy zaimplementowanej autorskiej procedury rozpoznawania kropek na płytce, po czym obliczono współrzędne w układzie robota ${}^{G}p_{obl:1}, {}^{G}p_{obl:2}$ przy użyciu wyznaczonej macierzy transformacji. Zadaniem robota było wskazanie każdego z tych punktów, a wynikiem powodzenia procesu kalibracji była różnica między wskazaniem manipulatora, a wcześniej otrzymanymi wektorami:

$$\delta p_i = {}^{G}p_{zm:i} - {}^{G}_{C}T \cdot {}^{C}p_i = {}^{G}p_{zm:i} - {}^{G}p_{obl:i}$$

Pomiarów wskazań robota ${}^{G}p_{zm:i}$ dokonywano przy użyciu linijki. Błąd związany z podziałką wynosił $1mm$, jednak błąd pomiaru oszacowano na $0.5cm$ z uwagi na użycie długiego (ok $14cm$) wskaźnika umieszczonego w szczypcach manipulatora. Nawet niewielkie pochylenie tak zamontowanego narzędzia powodują kilkumilimetrowe różnice, a dodatkowo trudność sprawiało wyznaczenie jego odchylenia od początku układu związanego z efektorem.

${}^{C}p_i$	${}^{G}p_{zm:i}$	${}^{G}p_{obl:i}$	δp_i	Δp_i
0.221	0.999	1.0053	−0.006	
0.233	0.243	0.2519	−0.008	0.013
1.171	0.020	0.0136	0.006	
−0.269	0.976	0.981	−0.005	
0.194	−0.259	−0.235	−0.024	0.025
1.091	0.092	0.088	0.004	

Tablica 2: Błędy położeń kropek.

W powyższej tabeli umieszczono w kolejnych wierszach odczytane położenia kropek w układzie kamery ${}^{C}p_i$ i układzie robota ${}^{G}p_{zm:i}$, odpowiadajace im położenia wyznaczone za pomocą macierzy transformacji ${}^{G}p_{obl:i}$ i błędy będące różnicą tych wielkości δp_i. Miarę błędu, tak jak w poprzednim paragrafie, zdefiniowano jako 2-normę wektora błędu $\Delta p_i = \text{norm}(\delta p_i)$. Wszystkie wartości podane są w metrach.

Mimo niewielkich błędów, można uznać, że metoda dała dobre wyniki. 1-2 centymetrowe błędy mogły zostać spowodowane przez niezgodność metody kalibracji i testowania, uchyby w przegubach manipulatora oraz niedość dokładne pomiary linijką w trakcie testu.

5.3 Źródła błędów oraz sposoby ich eliminacji

Sytuacja laboratoryjna jest znacząco odmienna od idealnej: urządzenia nie są doskonałe, w trakcie dnia zmienia się oświetlenie, itp. Dlatego nieuniknione jest powstawanie rozmaitych błędów na niemal wszystkich etapach procesu kalibracji. W poniższych punktach omówione zostały najważniejsze rodzaje błędów, źródła ich powstawania oraz, jeśli to możliwe, przedyskutowano sposoby ich eliminacji.

Błąd wyznaczania położenia kropek w układzie kamery

Mimo iż do wyznaczania położenia kropek w układzie kamery na podstawie danych ze zdjęcia wykorzystano profesjonalną funkcję wbudowaną w OpenCV, nie można uznać błędów tych położeń za zaniedbywalne. Można scharakteryzować co najmniej dwie przyczyny ich powstawania:

1. Niedoskonały jest proces segmentacji - w wyniku szumów progowanie daje różne wyniki dla jednakowych ujęć. Aby zminimalizować wpływ szumu wykonywana jest seria zdjęć dla jednego położenia płytki, a wynikiem jest położenie środka płytki (czyli średnia arytmetyczna położeń czterech kropek).

2. Niedoskonale wyznaczone parametry wewnętrzne kamery - z uwagi na to, że wewnętrzna kalibracja kamery przeprowadzana była ręcznie, a wyniki zapisane w pliku konfiguracyjnym można przypuszczać, że wynik nie jest bezbłędny. Ten błąd można wyeliminować implementując dodatkowo metodę kalibracji wewnętrznych parametrów kamery, ale nie jest to tematem niniejszej pracy.

Błędy wynikające z dopasowania macierzy transformacji do punktów pomiarowych

Na podstawie n punktów pomiarowych wyznaczane jest 6 niezależnych parametrów transformacji. Aby było to możliwe musi zostać zaimplementowana optymalizacja dopasowująca te parametry, co przekreśla możliwość otrzymania niezwykle dokładnego wyniku. Dokładna analiza błędów tego rodzaju została przeprowadzona w poprzednim rozdziale. Źródłem błędów, co jest widoczne na rysunku 22, jest np. bardzo ograniczony obszar pracy manipulatora widziany przez kamerę. Dodatkowo można zauważyć, że o ile wzdłuż osi Y rozpościera się on dość symetrycznie zarówno względem robota, jak i kamery, o tyle wzdłuż pozostałych osi jest silnie zlokalizowany. Kalibracja w takim obszarze nie jest uniwersalna, prawdopodobnie wyznaczona macierz nie byłaby poprawna np. po drugiej stronie manipulatora. Jednakże obszar za robotem nie jest widziany przez kamerę w analizowanym położeniu, więc można ten problem zaniedbać.

Błędy wynikające z nieskalibrowania kinematyki robota

Z uwagi na fakt, że manipulatory pracujące w laboratorium są stare i zostały w pewnym stopniu przebudowane, kluczowy wpływ na wyniki kalibracji mogą mieć błędy związane z ich funkcjonowaniem. Nie można oczekiwać, że dla każdego zestawu współrzędnych końcówka chwytaka znajdzie się w oczekiwanym położeniu z dowolną dokładnością. Dlatego powstające niedokładności mogą nie być wynikiem błędów w procedurze kalibracji, a niedokładnym skalibrowaniem kinematyki robota oraz luzów w przegubach.

5.4 Problemy, których można było uniknąć

Mimo, iż autor dołożył wszelkich starań, aby zarówno praca, jak i program realizujący kalibrację zostały napisane jak najlepiej, nie udało się uniknąć pewnych ograniczeń i przybliżeń z uwagi na podejmowane w trakcie pisania pracy decyzje.

Poniższe zestawienie opisuje założenia, których zmiana mogła skutkować lepszą wydajnością algorytmu:

- Płytka została wykonana w profesjonalnym zakładzie, na czarnym tle umieszczone zostały zielone kropki. W trakcie prac zauważono, że śliska powierzchnia płytki odbija światło, co zmienia lokalnie jej „kolor emisyjny" odbierany przez kamerę.
- Oczekiwano, że zielone kropki będą charakteryzowały się bardzo wysoką zieloną składową w przestrzeni barw RGB i niskimi pozostałymi. Niestety ta przestrzeń

barw nie dawała zbyt dobrych rezultatów i dlatego zdecydowano się analizować i progować zdjęcia w przestrzeni HSV. Rejestrowane kolory zmieniały się jednak znacząco wraz z odległością płytki i kamery, przez co niemożliwe było postawienie bardzo ostrych warunków na przedziały progowania. Można było tego uniknąć stosując np. diody LED. Problemy z filtracją można by było zmniejszyć stosując inną płytkę. Być może większa, pozwoliłaby zasłonić znaczny obszar nieużyteczny, jednak wiązałoby się to z innym zamocowaniem jej w chwytaku.

5.5 Propozycje dalszych prac

Z uwagi na fakt, że model matematyczny automatycznej kalibracji opiera się na zmieniających się położeniach środka płytki, nie da się osiągnąć stuprocentowej dokładności, mimo że większość opisanych w literaturze przykładów również wykorzystuje pojedyncze punkty. Na przykładzie innych możliwych wzorców kalibracyjnych (np. szachownicy) można zaprojektować płytkę, gdzie umieszczone zostanie więcej kropek w mniejszych odległościach lub możliwe będzie wykorzystanie np. zbieżności linii prostych umieszczonych na wzorcu. Mogłoby zwiększyć to dokładność obliczeń, jednak rozpoznawanie przylegających kropek, ustawianie ich w odpowiedniej kolejności, itp. powinno zostać wykonane w inny i prawdopodobie bardziej skomplikowany sposób.

Zgodnie z początkowymi założeniami automatyczna kalibracja rozpoczyna się, gdy płytka umieszczona jest w szczypcach robota w odpowiedni sposób. Aby tego uniknąć można by przymocować płytkę na stałe do ramienia manipulatora lub zaprojektować algorytm automatycznego pobierania płytki z odpowiednio przygotowanej półki. Należałoby przewidzieć fakt, że tak leżąca płytka mogłaby się przesuwać i robot powinien rozpoznawać jej położenie np. przy użyciu kamery wbudowanej w szczypce.

Można również zaprojektować algorytm „inteligentnej" generacji trajektroii na podstawie, np. obrazu z kamery, aby uniknąć konieczności ręcznego ich tworzenia dla każdego robota i dla innych, znacząco odmiennych, położeń kamer.

5.6 Podsumowanie pracy

Celem niniejszej pracy było zaprojektowanie i implementacja algorytmu automatycznej kalibracji systemu robot-kamera. System został opracowany przy użyciu manipulatora IRp-6 znajdującego się w Pracowni Robotyki i zaimplementowany w strukturze ramowej MRROC++. Do operacji na zdjęciach napisano moduł, który pracuje jako zadanie struktury FraDIA.

Załączniki

Jak wspomniano we wstępie, w załącznikach opisane zostały informacje dla przyszłych użytkowników systemu, tj. budowa pliku konfiguracyjnego, instrukcja użytkownika oraz zamieszczone zostały tabelki ze współczynnikami, których wyznaczenie było konieczne w trakcie trwania prac.

A Informacje dla użytkowników programu

A.1 Opis pliku konfiguracyjnego

Przykładowy plik konfiguracyjny do zadania biblioteki FraDIA z domyślnymi wartościami znajduje się poniżej. Przy każdym parametrze podano jego zastosowanie.

```
#thresholding
#kropki zielone
hz_min=70
hz_max=112
sz_min=50
sz_max=210
vz_min=80
vz_max=220
```

Zmienne *hz_min, hz_max, sz_min, sz_max, vz_min, vz_max* odpowiadają za domyślne minimalne i maksymalne poziomy progu w procesie progowania dla wszystkich składowych HSV odpowiadających kropkom zielonym;

```
#kropka czerwona
hc_min=20
hc_max=350
sc_min=110
sc_max=230
vc_min=90
vc_max=170
```

Zmienne *hc_min, hc_max, sc_min, sc_max, vc_min, vc_max* odpowiadają za domyślne minimalne i maksymalne poziomy progu w procesie progowania dla wszystkich składowych HSV odpowiadających kropce czerwonej;

```
max_count=100
```

Zmienna *max_count* opisuje maksymalną liczbę obiektów, jaka może znajdować się na zdjęciu po segmentacji, aby zostało ono potraktowane jako poprawne;

```
#segmentation
min_size=35
max_size=2000
```

Parametry *min_size, max_size* obejmują zakres rozmiarów wyrażony w pikselach, jakie mogą mieć kropki;

`#minimalisation`
`a=7.0`
`fx=1090`
`fy=1129`
`dx=416`
`dy=207`
	Parametr a oznacza odległość między sąsiednimi kropkami na płytce wyrażoną w centymetrach, fx i fy to stosunki ogniskowej kamery do odpowiednio szerokości i wysokości piksela, dy i dy to położenie środka projekcji [pix] wyznaczone dzięki kalibracji wewnętrznych parametrów kamery;

`#control`
`pic_count=10`
`max_error=3.9`
`max_pic=100`
	Parametry *pic_ count, max_ error, max_ pic* definiują odpowiednio liczbę poprawnych zdjęć dla jednego położenia manipulatora, które muszą być wykonane, maksymalny błąd minimalizacji uznający zdjęcie za poprawne oraz liczbę zdjęć, po których wykonaniu mimo niespełnienia warunku kalibracji, robot zmienia pozycję.

Tablica 3: Budowa pliku konfiguracyjnego.

Każda linia rozpoczynająca się znakiem '#' traktowana jest jako komentarz i jest ignorowana. Podobnie ignorowane są linie puste. Parametry i ich wartości oddzielone są znakiem równości. Podział na sekcje nie jest istotny. Parametry liczbowe są rozpoznawane przy użyciu wbudowanych metod języka C, dlatego występujący po liczbie przypadkowy tekst nie powinien mieć wpływu na działanie programu.

A.2 Instrukcja obsługi programu

Wstęp

Aby kalibracja była możliwa, potrzebne są dwa komputery działające w sieci lokalnej laboratorium, jeden pod kontrolą QNX, a drugi systemu Linux. W pliku konfiguracyjnym zadania MRROC++ należy zdefiniować nazwy komputerów, przy czym komputer wybrany do uruchomienia aplikacji linuksowej musi mieć tzw. *framegrabber*, czyli interfejs zapewniający współpracę z kamerą.

Na komputerze pracującym pod Linuksem należy uruchomić strukturę FraDIA i wybrać zadanie *PS_ SpotsRecognition*. Po przygotowaniu robota, należy pod systemem QNX uruchomić strukturę MRROC++ i wybrać zadanie *spots_ recognition*. Aby rozpocząć kalibrację należy nacisnąć przycisk *START*.

Scenariusz działania

Gdy robot ustawi się w pozycji początkowej należy umieścić płytkę w szczypcach tak, by czerwona kropka znajdowała się po prawej stronie od strony manipulatora. Następnie należy ustawić ramię robota w pożądanym położeniu. Po puszczeniu ramienia i odczekaniu kilku sekund rozpocznie się zbieranie pomiarów, których przebieg widać w oknie modułu FraDIA. Rozpoczęcie i zakończenie zdjęć sygnalizowane jest sygnałem dźwiękowym. Należy zadbać, by wykonanych zostało co najmniej kilkanaście serii zdjeciowych w różnych położeniach płytki. Aby zakończyć kalibrację i zapisać wyniki należy ponownie nacisnąć przycisk *Trigger*. Można również nacisnąć *STOP*, lecz nie spowoduje to ustawienia manipulatora w pozycji końcowej.

B Ścisłe wyznaczenie położenia płytki względem kamery

W tym rozdziale opisano ścisłe rozwiązanie problemu transformacji układu płytki do układu kamery. Niestety z uwagi na bardzo duże błędy nie znalazło ono zastosowania do kalibracji układu robot-kamera. Niepowodzenie to zdecydowało o implementacji metody wyznaczającej, na podstawie szeregu położeń płytki, bezpośrednią transformację z układu bazowego do układu kamery. Próba implementacji tej metody uzasadnia przyjętą orientację układu płytki.

Macierz ${}^{C}_{P}\mathrm{T}$ wiąże dokładnie wyznaczone położenia kropek w układzie płytki z położeniami ich w układzie kamery, które są obarczone pewnym błędem. Dlatego zamiast informacji o pojedynczych kropkach lepiej wykorzystać położenie środka, przynajmniej tam, gdzie jest to możliwe. Pozwala to na redukcję błędów położeń w układzie kamery.

Wiadomo, że dla par wektorów ${}^{C}\mathbf{v}, {}^{P}\mathbf{v}$ zachodzi:

$$ {}^{C}\mathbf{v} = {}^{C}_{P}\mathrm{T} \cdot {}^{P}\mathbf{v}. $$

Można uogólnić to równanie dla położenia środka płytki oraz dwóch par wektorów dla przeciwległych kropek.

Oznaczając położenie i-tej kropki w układzie kamery jako:

$$ \widetilde{{}^{C}\mathbf{p}_i} = \begin{bmatrix} p_{ix} \\ p_{iy} \\ p_{iz} \\ 1 \end{bmatrix}, $$

a położenie środka płytki jako:

$$\widetilde{{}^{C}\mathbf{o}_i} = \frac{1}{4}\left(\widetilde{{}^{C}\mathbf{p}_1} + \widetilde{{}^{C}\mathbf{p}_2} + \widetilde{{}^{C}\mathbf{p}_3} + \widetilde{{}^{C}\mathbf{p}_4}\right) = \begin{bmatrix} o_x \\ o_y \\ o_z \\ 1 \end{bmatrix}$$

można wyznaczyć macierz transformacji rozwiązując kilka charakterystycznych równań.

Można zauważyć, że w szczególności:

$${}^{C}_{P}\mathrm{T} \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} o_x \\ o_y \\ o_z \\ 1 \end{bmatrix}, \tag{36}$$

$${}^{C}_{P}\mathrm{T} \cdot \begin{bmatrix} \frac{a\sqrt{2}}{2} \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} p_{1x} \\ p_{1y} \\ p_{1z} \\ 1 \end{bmatrix}, \qquad {}^{C}_{P}\mathrm{T} \cdot \begin{bmatrix} -\frac{a\sqrt{2}}{2} \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} p_{3x} \\ p_{3y} \\ p_{3z} \\ 1 \end{bmatrix} \tag{37}$$

oraz

$${}^{C}_{P}\mathrm{T} \cdot \begin{bmatrix} 0 \\ \frac{a\sqrt{2}}{2} \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} p_{2x} \\ p_{2y} \\ p_{2z} \\ 1 \end{bmatrix}, \qquad {}^{C}_{P}\mathrm{T} \cdot \begin{bmatrix} 0 \\ -\frac{a\sqrt{2}}{2} \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} p_{4x} \\ p_{4y} \\ p_{4z} \\ 1 \end{bmatrix}. \tag{38}$$

Rozwiązaniami ww. równań są poszczególne kolumny macierzy transformacji. Wykorzystując wprowadzone wcześniej (2) oznaczenia elementów macierzy T, można pokazać, że z równania (36) wynika[32]:

$$t_i = o_i = \frac{1}{4}(p_{1i} + p_{2i} + p_{3i} + p_{4i}),$$

z równań (37):

$$\begin{array}{l} \frac{a\sqrt{2}}{2} r_{i1} + o_i = p_{1i} \\ -\frac{a\sqrt{2}}{2} r_{i1} + o_i = p_{3i} \end{array} \quad \Rightarrow \quad r_{i1} = \frac{1}{a\sqrt{2}}(p_{1i} - p_{3i})$$

oraz podobnie z równań (38):

$$\begin{array}{l} \frac{a\sqrt{2}}{2} r_{i2} + o_i = p_{2i} \\ -\frac{a\sqrt{2}}{2} r_{i2} + o_i = p_{4i} \end{array} \quad \Rightarrow \quad r_{i2} = \frac{1}{a\sqrt{2}}(p_{2i} - p_{4i}).$$

[32]W poniższych wzorach, z uwagi na wymienne stosowanie wskaźników numerycznych i literowych na oznaczenie podobnych wielkości (*vide* o_x a t_1 zamiast t_x), indeks i pełni rolę zamiennie wskaźnika numerycznego lub elementu z trójki (x, y, z) tam gdzie oznacza tę samą współrzędną, w zależności od określanej wielkości w zgodzie ze zdefiniowaną powyżej konwencją oznaczeń.

Brakującą kolumnę macierzy T można wyznaczyć korzystając z iloczynu wektorowego (jest to trzecia własność macierzy rotacji opisana w rozdziale 2.1.1):

$$\begin{bmatrix} r_{13} \\ r_{23} \\ r_{33} \end{bmatrix} = \begin{bmatrix} r_{21}r_{32} - r_{22}r_{31} \\ r_{31}r_{12} - r_{11}r_{32} \\ r_{11}r_{22} - r_{12}r_{21} \end{bmatrix}. \tag{39}$$

Powyższe obliczenia pokazały, że macierz ${}^{C}_{P}\mathrm{T}$ jest zadana jako zbiór ścisłych zależności między obarczonymi błędami wektorami położeń kropek w układzie kamery.

Niestety macierz ${}^{C}_{P}\mathrm{T}$ okazała się bardzo „dopasowana" do konkretnych wielkości, w oparciu o które została wyliczona i dalsze przekształcenia skutkowały złymi wynikami. Metoda dawała prawidłowe wyniki jedynie dla bardzo skomplikowanych położeń płytki względem układu bazowego i kamery (oddalonych od osi układów oraz dla dużych, różnych od $n\pi$, dla $n = 0, 1, ...$ kątów).

C Wyznaczone parametry wewnętrzne kamery

Aby możliwa była kalibracja układu robot-kamera należało wyznaczyć wewnętrzne parametry kamery przeprowadzając jej kalibrację. Wyznaczone wyniki przedstawiono w notacji Calib Toolbox programu MATLAB. Parametry odpowiadają ustawieniom zapisanym na pozycji 6 w pamięci kamery.

Focal Length:	$fc = [1090.50896, 1128.92073] \pm [15.40452, 17.58021]$,
Principal point:	$cc = [416.23124, 207.41472] \pm [24.88048, 29.21469]$,
Distortion:	$kc = [-0.12960, 0.35864, -0.02006, -0.01716, 0.00000]$ $\pm[0.08671, 0.64117, 0.00789, 0.00550, 0.00000]$,
Pixel error:	$err = [0.44534, 0.59589]$.

Tablica 4: Wewnętrzne parametry kamery.

Literatura

[1] John J. Craig, *Wprowadzenie do robotyki: mechanika i sterowanie*, Wydawnictwa Naukowo-Techniczne, Warszawa, 1993.

[2] David F. Rogers, Alan J. Adams, *Mathematical elements for computer graphics*, MacGraw-Hill, New York, 1990.

[3] Mark W. Spong, Seth Hutchinson, M. Vidyasagar, *Robot Modeling and Control*, John Wiley & Sons, Inc., 2006.

[4] Andrzej Stachurski, Andrzej P. Wierzbicki, *Podstawy optymalizacji*, Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa, 2001.

[5] William K. Pratt, *Digital Image Processing* (3rd Ed.), Wiley-Interscience, 2001.

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

[7] Gary Bradski, Adrian Kaehler, *Learning OpenCV*, O'Reilly Media, Inc., 2008

[8] Piotr Tatjewski *Wybrane metody numeryczne*. Skrypt do przedmiotu Metody Numeryczne.

[9] Tomasz Kornuta, *FraDIA. Zadaniowo zorientowana struktura ramowa do analizy i przetwarzania obrazów*. Raport IAiIS nr 08-08, Warszawa, 2008.

[10] Cezary Zieliński, Wojciech Szynkiewicz, Tomasz Winiarski, Tomasz Kornuta, *MRROC++ Based System Description*. Raport IAiIS nr 06-09, Warszawa, 2007.

[11] Tomasz Kornuta i in., *Kalibracja systemu wielorobotowego*, IX Krajowa Konferencja Robotyki - Postępy Robotyki: Systemy i współdziałanie robotów, vol. 2, Wydawnictwa Komunikacji i Łączności, Warszawa, 2006.

[12] Ming-Kuei Hu, *Visual Pattern Recognition by Moment Invariants*, IEEE Transactions on Information Theory, Volume 8, Issue 2, 1962, strony 179-187.

[13] M. Galassi et al, *GNU Scientific Library Reference Manual* (2nd Ed.), ISBN 0954161734, http://www.gnu.org/software/gsl/, wybrane rozdziały.

[14] *Open Source Computer Vision Library. Reference Manual*, Intel Corporation, Order Number: 123456-001.