POLITECHNIKA WARSZAWSKA

WYDZIAŁ MECHATRONIKI

Rozprawa doktorska

mgr inż. Bogdan Harasymowicz-Boggio

Rozpoznawanie obiektów w chmurach punktów na potrzeby robotyki mobilnej

Promotor:
dr hab. Barbara Siemiątkowska, prof. PW

WARSZAWA, 2017

Streszczenie

Umiejętność semantycznego rozumienia świata przez komputer warunkuje użyteczność i stosowalność robotów mobilnych w środowisku człowieka. W ostatnich latach obserwujemy intensywny rozwój motorycznych możliwości robotów, mocy obliczeniowej kompaktowych komputerów (w postaci postępu technik masowo-równoległych), a także sensorów, szczególnie szybkich kamer głębi. Mimo to metody wizyjnego rozumienia świata nie są na wystarczającym poziomie, by uczynić pełny użytek z tych możliwości, przez co zastosowania autonomicznych robotów mobilnych są bardzo ograniczone. Powodem takiego stanu rzeczy, zdaniem autora, jest bardzo duża złożoność samej rzeczywistości postrzeganej wizyjnie. Interpretacja sygnałów wzrokowych jest wymagająca również dla organizmów żywych i nawet u człowieka angażuje znaczącą część mózgu.

Algorytmy analizujące obraz z kamery 2D muszą cechować się bardzo dużą zdolnością uogólniania, gdyż rozmiar i proporcje obiektów na obrazie płaskim są uzależnione od położenia kamery. Wygląd obiektów jest także silnie związany z warunkami oświetlenia. Trójwymiarowa chmura punktów pochodząca z kamer głębi, takich jak Kinect, pozwala na zdobycie informacji bardziej jednoznacznych metrycznie, a więc potencjalnie prostszych w interpretacji. Z drugiej strony jednak chmura punktów jest, w porównaniu z obrazem, bardziej złożona i wymagająca obliczeniowo w przetwarzaniu. Kamery głębi są szczególnie użyteczne na pokładzie robotów mobilnych działających w budynku, gdyż wnętrze budynku stwarza optymalne warunki działania dla takich kamer. Informacje trójwymiarowe pozwalają na szybsze uczenie się rozpoznawania nowych obiektów (bez konieczności zdobywania tysięcy widoków).

Autor proponuje w rozprawie szereg założeń dotyczących przydatnych metod rozpoznawania obiektów w mobilnej robotyce usługowej, w świetle których omawia różne konkretne algorytmy wizyjne. Założenia te dotyczą m.in. szybkiego przetwarzania sceny, łatwego uczenia się rozpoznawania nowych obiektów, rozpoznawania częściowo przysłoniętych i stykających się elementów, występowania w pracy robota konkretnych obiektów zainteresowania, a także znanego nachylenia kamery.

Oprócz szerokiego przeglądu znanych w literaturze metod interpretacji obrazu 2D i 3D, w zasadniczej części pracy autor przedstawia własne badania. Prace te stanowią wkład naukowy w dziedzinie rozpoznawania obiektów w chmurach punktów i w większości zostały opisane również w publikacjach o zasięgu międzynarodowym. Dotyczą one:

- Wnioskowania metryczno-semantycznego w celu interpretacji sceny (wykorzystując m.in. relacje semantyczne między hipotezami obiektów).
- Wykorzystania dostępnych informacji o ograniczeniach percepcyjnych systemu wizyjnego (m.in. przysłonięciach i utracie dokładności sensora wraz ze wzrostem odległości).
- Masowo-równoległej, autorskiej metody segmentacji sceny w celu jej interpretacji; Wielostopniowego, heurystycznego i masowo-równoległego ograniczania uwagi w rozpoznawaniu obiektów z użyciem deskryptorów lokalnych.
- Implementacji systemu wizyjnego 3D w rzeczywistym systemie robotycznym.

Omawiając konkretne prace, autor przedstawia zarówno części teoretyczne, implementacyjne, jak i eksperymentalne, co umożliwia wyciąganie wniosków, a także identyfikację trudności i obszarów możliwego rozwoju.

Słowa kluczowe: *rozpoznawanie obiektów, chmury punktów, obraz 3D, robotyka mobilna*

Abstract

Object recognition on point clouds for mobile robotics

The capacity of semantic understanding of the world by a computer determines the usefulness of mobile robots in human environments. In recent years we observe a rapid development of robots' motor capacity, computing power of miniature computers (mainly related to massively parallel techniques) and development of sensors, especially depth cameras. However, the methods of visual understanding of the world are not advanced enough to make full use of the mentioned assets, which results in a very limited applicability of mobile robots. The author believes that the reason for this is the high complexity of the visually perceived reality. The interpretation of visual signals is a demanding task for living beings as well and even for humans it involves a large part of the brain.

2D image processing algorithms are required to have a high generalization capacity, since the size and proportions of objects on a flat image depend on the camera position. The appearance of objects is also strongly dependent on the lighting conditions. 3D point clouds obtained from depth cameras, such as the Kinect, provide more metrically unambiguous information, which is arguably easier to interpret. On the other hand, compared to flat images, point clouds are more complex and their processing is more computationally demanding. Depth cameras are especially useful on board of indoor mobile robots, since indoor environments provide optimal conditions for their function. 3D information allows to quickly learn to recognize new objects (without the need to provide thousands of views).

In this thesis the author proposes several assumptions regarding useful object recognition methods for service mobile robots, in light of which specific computer vision algorithms are discussed. These assumptions are related, a.o. to fast scene processing, easy learning of new objects, recognition of partially occluded and adjacent elements, a limited number of interest objects and a known camera inclination.

In addition to a wide review of 2D and 3D image interpretation methods known in literature, in the main part of the thesis the author presents his own research. These works constitute a

scientific contribution to the field of object recognition using point clouds and have mostly been described in international publications. The discussed works concern:

- Metric-semantic inference for scene interpretation (using, a.o. semantic relations between object hypotheses).
- Using available information about perceptual limitations of the vision system (a.o. regarding occlusions and loss of precision with distance increase).
- A massively-parallel scene segmentation method which aids interpretation; A multi-level, heuristic and massively-parallel algorithm of focus reduction for object recognition based on local descriptors.
- The implementation of a 3D computer vision system on a physical robotic system.

While discussing specific works, the author presents theoretical, implementational and experimental parts, which allow to draw conclusions, as well as to identify difficulties and areas of possible development.

Keywords: *object recognition, point clouds, 3D image, mobile robotics*

Spis treści

Lista symboli

$\hat{\mathbf{g}}$	wektor grawitacji znormalizowany do długości jednostkowej
$\mathbf{n}$	wersor (wektor jednostkowy) normalny do opisywanej powierzchni
$\setminus$	operator różnicy zbiorów
$\subseteq$	operator zawierania się zbiorów
$\vec{u}$	wektor obserwacji losowego pola Markowa
$\vec{x}$	wektor etykiet losowego pola Markowa
$\vec{x}_{opt}$	wektor optymalnych etykiet losowego pola Markowa
$\widehat{\bullet}$	operator normalizacji wektora (skalujący wektor do długości jednostkowej)
$A(\mathbf{p})$	anizotropia wypukłości w punkcie $\mathbf{p}$
bel	miara przekonania teorii Dempstera-Shafera
$C(\mathbf{p})$	wypukłość w otoczeniu punktu $\mathbf{p}$
$C(\mathbf{p}, \mathbf{p_i})$	wypukłość w punkcie $\mathbf{p}$ ze względu na punkt $\mathbf{p_i}$
$corr$	współczynnik korelacji Pearsona
E	zbiór krawędzi grafu losowego pola Markowa
$E1$	całkowity wskaźnik błędów
$E2$	ważony wskaźnik błędów
F	funkcja energii Losowego Pola Markowa
f_s	funkcja składowa energii F_{data} związana z pojedynczym węzłem s

F_{data} składowa energii losowego pola Markowa pochodząca z obserwacji

f_{pq} funkcja składowa energii F_{prior} związana z krawędzią między węzłami p i q

F_{prior} składowa energii losowego pola Markowa pochodząca z ograniczeń kontekstowych (wiedzy systemu)

FN rozpoznania fałszywie ujemne (ang. *false negative*)

FP rozpoznania fałszywie dodatnie (ang. *false positive*)

G graf definiujący losowe pole Markowa

H odcień w przestrzeni barw HSV

$I(\mathbf{p})$ kątowe nachylenie powierzchni w punkcie $\mathbf{p}$

k średnia liczba sąsiadujących punktów w promieniu $1cm$

m masa w teorii Dempstera-Shafera

m_w zmodyfikowana, ważona masa w teorii Dempstera-Shafera

$m_{1,2}(a)$ masa dwóch źródeł po agregacji dla zbioru a

N średnia liczba punktów obiektu w widoku modelowym

$N_{\mathbf{p}}(r)$ sąsiedztwo kuliste punktu $\mathbf{p}$ o promieniu r

N_{TMP} całkowita liczba punktów części modelu obiektu

N_{VMP} liczba punktów części modelu obiektu, które wedle rozpatrywanej transformacji powinny być widoczne dla kamery

O funkcja szacowania złożoności obliczeniowej

pl miara możliwości w teorii Dempstera-Shafera

R promień obliczania deskryptorów lokalnych

S saturacja w przestrzeni barw HSV

T operator progowania

TN rozpoznania prawdziwie ujemne (ang. *true negative*)

TP rozpoznania prawdziwie dodatnie (ang. *true positive*)

V zbiór węzłów grafu losowego pola Markowa

1. Wstęp

Niniejsza rozprawa dotyczy ważnego obszaru w robotyce mobilnej, jakim jest rozumienie otoczenia. Zdolności lokomocyjne i manipulacyjne współczesnych robotów dalece wyprzedzają ich zdolności percepcyjne i kognitywne, przez co ich użyteczne wykorzystanie w złożonym środowisku jest ograniczone. W ostatnich latach obserwuje się znaczący rozwój w dziedzinie autonomicznej robotyki mobilnej [68, 82, 51, 69]. W tym samym czasie w wielu rozwiniętych krajach technologia mechatroniczna staje się coraz bardziej dostępna, a koszty ludzkiej pracy rosną. Z tych powodów można by było oczekiwać, że roboty w niedalekiej przyszłości staną się powszechnym narzędziem w życiu codziennym. Robotyzacja pozaprzemysłowa faktycznie postępuje, jednak wkraczanie użytecznych robotów autonomicznych (nie będących jedynie gadżetem rozrywkowym) do skomplikowanego środowiska człowieka jest powolne. Zdaniem autora jednym z głównych powodów takiego stanu rzeczy jest brak możliwości wystarczającego zrozumienia przez roboty tego środowiska. Szczególnie ważny jest brak wystarczającej zdolności interpretacji sygnałów wizyjnych, gdyż wyklucza to celową interakcję z większością elementów otoczenia, które nie są wyposażone w dedykowane dla robota znaczniki (a więc wyklucza większość możliwych zastosowań robotów usługowych).

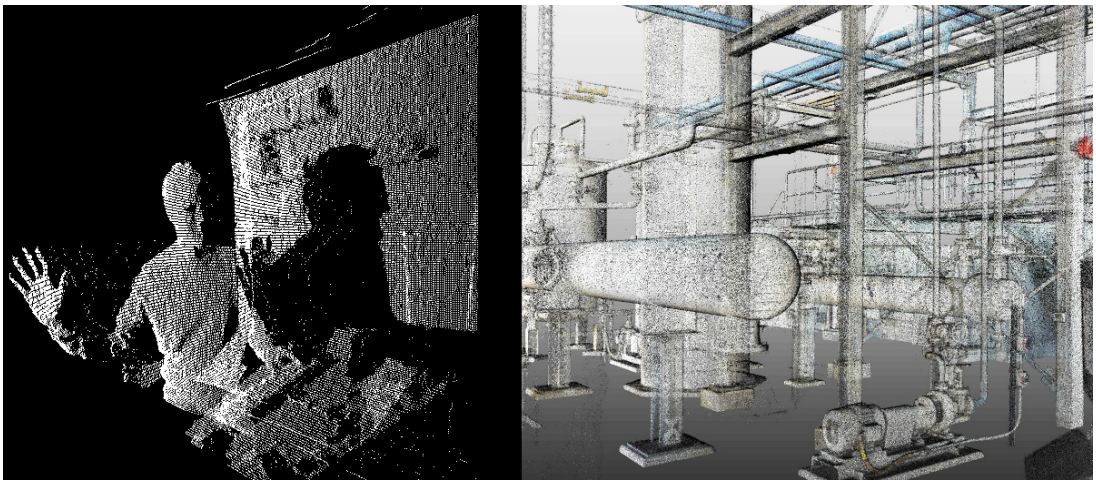
Chociaż w ostatnich dekadach opracowano wiele kamer i innych sensorów wizyjnych, a także wiele algorytmów przetwarzania danych uzyskiwanych z tych sensorów, wciąż brakuje wystarczająco dobrych algorytmów pozwalających na określanie semantycznego znaczenia elementów środowiska. Taki stan rzeczy nie wynika z braku zainteresowania tą problematyką w środowisku naukowym czy w przemyśle. Rozumienie ludzkiego otoczenia przez komputer ma niemalże natychmiastowe zastosowanie chociażby w urządzeniach mobilnych, które dysponują coraz większą mocą obliczeniową i coraz lepszą sensoryką. Skuteczne rozpoznawanie obiektów w środowisku człowieka otwiera nowe możliwości dla rzeczywistości rozszerzonej, systemów eksperckich, grafiki komputerowej i systemów automatycznie nadzorujących miejsca publiczne. Uzasadnione wydaje się stwierdzenie, że ograniczone rozumienie danych wizyjnych przez maszyny wynika głównie z dużej trudności cechującej samo zadanie rozumienia obra-

zu. Prowadzone przez dekady badania neurobiologiczne nad mózgiem ssaków pokazują wielką złożoność funkcji kory wzrokowej i zachodzącego w niej procesu rozpoznawania obiektów [28, 93, 23, 104], w porównaniu z którym algorytmy komputerowe są dalece mniej rozwinięte.

W szczególnym zainteresowaniu autora tej pracy są techniki przetwarzania wizyjnych danych trójwymiarowych. Jak zostanie uzasadnione bardziej szczegółowo w rozdziale 2, w warunkach robotyki mobilnej dane 3D są pozbawione pewnych ważnych wad obrazu 2D. Stereowizyjne techniki obrazowania głębi są znane od lat osiemdziesiątych [5], zaś pierwsze kamery głębi zaczęły się pojawiać w latach dziewięćdziesiątych [102]. Jednak dopiero pojawienie się sensora Microsoft Kinect w 2010 [103], choć dedykowany był do śledzenia ruchów człowieka w grach wideo, przyniosło gwałtowny wzrost zainteresowania trójwymiarowym widzeniem maszynowym. Nowy sensor łączył po raz pierwszy dość dobrej jakości obraz RGB-D (zawierający kolory i głębię), wysoką częstotliwość klatek (do 30 na sekundę), niską cenę oraz łatwość w użyciu. Po kamerze Kinect rozpowszechniły się inne podobne sensory, takie jak Asus Xtion, Intel Realsense, a także później Kinect V2 (sensor opierający swoją zasadę działania o czas przelotu fal – ang. *time of flight*) w 2013. Dostępność nowej generacji sensorów wraz z wysoką mocą obliczeniową komputerów w trwającej dekadzie otworzyła nowe możliwości widzenia 3D w robotyce, dając nadzieję na rozwiązanie wielu problemów obecnych w widzeniu maszynowym 2D. Tego typu problemami są m.in.: niejednoznaczność określania rozmiaru obiektów, zależność obserwowanego kształtu od orientacji względem kamery oraz znaczny wpływ oświetlenia na wygląd obiektu i całej sceny. Literatura naukowa wskazuje wyższą użyteczność informacji RGB-D nad samym obrazem RGB (w warunkach, gdzie taka informacja jest możliwa do zdobycia) [15], lecz w tej młodej dziedzinie brak jest jeszcze tak ugruntowanej bazy metod jak w widzeniu maszynowym 2D. Brakuje także porównywalnej jakości narzędzi programistycznych i liczby osób rozwijających te narzędzia. Łatwo to dostrzec porównując najbardziej spopularyzowane biblioteki wizji 2D i 3D, którymi są OpenCV [13] i Point Cloud Library (PCL) [80]. Druga wymieniona biblioteka, która jest przeznaczona do przetwarzania chmur punktów, ma znacznie mniej metod, słabszą dokumentację, mniejszą społeczność, wolniejszy rozwój i gorszą kompatybilność z innymi popularnymi narzędziami.

Chmura punktów jest główną rozważaną w pracy formą obrazu trójwymiarowego, gdyż łączy w sobie prostotę i uniwersalność. Istotą chmury punktów jest zbiór wektorów wielowymiarowych (punktów), które przechowują informację o współrzędnych 3D oraz (opcjonalnie) o kolorze i dowolnych innych cechach lokalnych (np. obliczonych wektorach normalnych do

powierzchni). Punkty te mogą stanowić elementy macierzy i odpowiadać poszczególnym pikselom obrazu kamery głębi (tzw. „chmura zorganizowana", stanowiąca proste uogólnienie obrazu płaskiego), a także w postaci listy o dowolnej kolejności punktów (tzw. „chmura niezorganizowana"). Chmura niezorganizowana jest postacią bardziej ogólną, nie ograniczoną jednym konkretnym punktem widzenia. Dzięki tej właściwości nie jest obarczona problemem przysłaniania się punktów czy narzuconej lokalnej rozdzielczości. Chmura niezorganizowana, choć zwykle bardziej wymagająca w przetwarzaniu komputerowym, pozwala m.in. na trywialną agregację widoków pochodzących z kilku punktów obserwacji sceny, a nawet z wielu różnych sensorów dokonujących pomiarów głębi (choć zwykle agregowane chmury poddaje się dodatkowej filtracji). Chmura punktów stanowi obecnie powszechnie przyjętą postać danych w widzeniu maszynowym 3D. Na Rys. 1.1 pokazano przykładowe widoki chmury zorganizowanej i niezorganizowanej. Widzimy, ze chmura niezorganizowana pozwala na znacznie pełniejsze przedstawienie obiektów.



Rysunek 1.1: Przykłady chmury zorganizowanej [110] (po lewej) i niezorganizowanej [111] (po prawej). Chmura zorganizowana ma strukturę dwuwymiarowej mapy, co wiąże się z obecnością charakterystycznych „cieni" obiektów

1.1 Założenia

Rozpoznawanie obiektów w robotyce mobilnej, choć pokrewne, nie jest tym samym zadaniem, co rozpoznawanie kategorii semantycznych dla jak najszerszej grupy obiektów (np. na potrzeby takich konkursów, jak Pascal Visual Object Classes Challenge). Różnica między tymi zadaniami tkwi zarówno w ich celowości, jak i założeniach. Nie ograniczając zadań wykony-

wanych przez robota usługowego, ale uwzględniając dotychczasowe przykłady praktycznego wdrożenia robotów mobilnych [106, 112, 115], a także wyobrażenie autora o pożądanej (lecz realistycznej na dzień dzisiejszy) funkcjonalności użytecznego robota, proponowane są następujące założenia:

- Robot powinien przetwarzać scenę szybko, najwyżej w ciągu kilku sekund.
- Przygotowanie robota do rozpoznawania nowego obiektu, którego nie zna (w tym nietypowego), powinno być łatwe, bez konieczności dostarczenia setek czy tysięcy jego widoków.
- Robot może działać w środowisku mało uporządkowanym, gdzie obiekty się stykają i przysłaniają.
- W większości wypadków robot ma ograniczony zbiór obiektów zainteresowania (związanych z wykonywaną przez niego pracą). Nie musi rozpoznawać wszystkich, ani nawet „jak najwięcej" obiektów w środowisku, tylko skutecznie rozpoznawać te, które są istotne.
- Robot działa w realnym, konkretnym środowisku człowieka (np. w domu, szpitalu, magazynie, itp). To naturalne środowisko charakteryzuje się pewnymi prawidłowościami, które robot może wykorzystać w jego rozumieniu (np. typowe położenia pewnych obiektów w przestrzeni lub w stosunku do innych obiektów).
- Robot może poruszać się w obserwowanym środowisku lub poruszać kamerę, uzyskując wiele widoków sceny.
- Wysokość nad podłożem i nachylenie sensora wizyjnego robota są znane, gdyż robot ma znaną geometrię i jest najczęściej wyposażony w akcelerometry (część sensorów wizyjnych także ma wbudowany akcelerometr).

1.2 Cel, teza i zakres pracy

Celem niniejszej rozprawy jest zbadanie kluczowych aspektów rozpoznawania obiektów w chmurach punktów na potrzeby robotyki mobilnej, w tym przedstawienie autorskich algorytmów rozwiązujących poszczególne związane z tym problemy. Teza stawiana przez autora

jest następująca: **Skuteczne rozpoznawanie obiektów na potrzeby robotyki mobilnej różni się od ogólnego zadania rozpoznawania obiektów i wymaga szybko uczących się i szybko działających algorytmów, a także czerpie korzyści z wykorzystania wiedzy o świecie wykraczającej poza znajomość struktury obiektów: w tym kontekstu semantycznego i ograniczeń percepcyjnych**.

Praca zawiera przegląd wielu współczesnych metod rozpoznawania obiektów 3D oraz szczegółowe omówienie różnych algorytmów i eksperymentów będących dorobkiem naukowym autora. W pracy zawarto teoretyczne i eksperymentalne uzasadnienie wad i zalet poszczególnych podejść, a także wnioski na temat możliwego rozwoju algorytmów.

Istnieją różne systemy o charakterze eksperymentalnym, wyspecjalizowane w maksymalizacji efektów w danym aspekcie rozpoznawania obiektów. Przykładem jest skuteczność rozpoznawania obiektów na scenach dowolnych, pochodzących z kamery głębi według miar sformułowanych w [86], gdzie nie uwzględnia się ograniczeń czasowych lub pracochłonności uczenia systemu [87, 88, 71]. Innym tego typu obszarem jest rozpoznawanie odizolowanych obiektów 3D [99, 43]. W niniejszej pracy autor próbuje szerzej nakreślić problem rozpoznawania obiektów 3D ze szczególną myślą o użyteczności w robotyce mobilnej – a zatem z uwzględnieniem spostrzeżeń sformułowanych powyżej.

Autorskie metody wizyjne cytowane i omawiane w pracy zostały rozwinięte na przestrzeni lat 2012-2017 i stanowią wkład w algorytmy i metodologię rozpoznawania obiektów 3D w robotyce. Ze względu na dynamiczny rozwój badanej dziedziny, niektóre zagadnienia poruszane w pracach autora stanowiły w momencie publikacji nieco większe novum niż w czasie ukończenia niniejszej rozprawy. W późniejszych lub jednoczesnych publikacjach innych autorów pojawiają się np. wspomniane zagadnienia wnioskowania na temat przysłonięć [87], lub wykorzystania kontekstu semantycznego [71] (choć w innej postaci). Stanowi to, zdaniem autora, dodatkowe potwierdzenie istotności omawianych zagadnień.

Praca ma następującą strukturę:

- Rozdział 2 rozprawy zawiera omówienie i klasyfikację różnych rodzajów metod rozpoznawania obiektów.
- Rozdział 3 przedstawia autorskie prace związane z uwzględnieniem kontekstu w rozpoznawaniu obiektów.

- Rozdział 4 prezentuje autorską metodę wnioskowania w rozpoznawaniu obiektów z jawnym uwzględnieniem niepewności i niewiedzy.
- Rozdział 5 omawia prace autora w tematyce algorytmów masowo-równoległych i metod heurystycznego przyspieszania procesu rozpoznawania.
- Rozdział 6 prezentuje wykonaną przez autora implementację masowo-równoległego systemu rozpoznawania obiektów 3D w systemie mobilnego robota autonomicznego.
- Rozdział 7 zawiera ogólne wnioski i podsumowanie pracy.

Ogólny przegląd literatury, na którą autor się powołuje klasyfikując techniki rozpoznawania obiektów znajduje się w rozdziale 2. Kolejne rozdziały natomiast zawierają także krótkie nawiązania do literatury związanej z konkretnym, omawianymi w tych rozdziałach zagadnieniami. Rozdziały 3 - 5 zwierają opisy eksperymentów, które skupiają się na uzasadnieniu postawionych tez użyteczności danej metodologii, a nie na uniwersalnej walidacji. Pełna statystyczna walidacja poszczególnych metod wymagałaby użycia odpowiednich baz testowych. Dostępne duże bazy scen rzeczywistych (takie jak [85, 48, 86]) zawierają jednak tylko etykiety semantyczne (często mało dokładne), bez spójnych identyfikatorów instancji obiektów dla wielu scen. Mylące jest, że autorzy niektórych ogólnodostępnych baz przedstawiają narzędzia do etykietowania instancji (sugerując obecność etykiet w bazie), podczas gdy w rzeczywistości etykiety te nie zostały przygotowane. Brak informacji na temat instancji utrudnia testowanie systemów wizyjnych szybko uczących się rozpoznawania konkretnych obiektów na potrzeby robotyki mobilnej. Wydawałoby się, że dobry system ogólnego przeznaczenia, który jest w stanie rozpoznawać dowolne obiekty z różnych klas powinien się sprawdzić w szczególnym scenariuszu, w którym zbiór obiektów zainteresowania jest ograniczony. Patrząc jednak na wyniki wiodących na tym polu algorytmów (ogólnej klasyfikacji semantycznej) [88, 71] widzimy, że metody te, choć bardzo złożone (stosują metryczne przeszukiwanie sceny w celu dopasowania chmur wektorów cech), nie są w pełni skuteczne. Można zasadnie przypuszczać, że tego typu metody nie poradzą sobie jeśli potrzebne będzie rozpoznawanie konkretnego, nietypowego obiektu (czyli takiego, który nie jest podobny do licznych przykładów uczących). Z tych powodów w eksperymentach przeprowadzonych przez autora tej pracy konieczne było przygotowanie własnych baz uczących i testowych, które, choć ograniczonych rozmiarów, pozwalają na przeprowadzenie testów w pożądany sposób. W niektórych eksperymentach wykorzystano także chmury scen wygładzone i uzupełnione algorytmem Kinect Fusion [47]. Tak podwyższona jakość chmury

punktów sensora Kinect, choć możliwa do uzyskania na pokładzie robota mobilnego, nie jest dostępna w bazach przeznaczonych do ogólnej semantycznej klasyfikacji. Wysoka dokładność chmury punktów jest szczególnie pożądana do celów rozpoznawania przedmiotów niewielkich, które mogą stanowić obiekt manipulacji robota usługowego.

Metody przedstawione w pracy jako autorskie charakteryzują się głównym wkładem metodologicznym, implementacyjnym oraz testowym autora niniejszej rozprawy. W opracowaniu, badaniu tych metod i w przygotowaniu publikacji brały jednak udział także inne osoby – przede wszystkim prof. Barbara Siemiątkowska (której autor zawdzięcza zwrócenie uwagi na bardzo istotne zagadnienia uwzględniania kontekstu semantycznego i niepewności wiedzy) oraz Łukasz Chechliński (który miał wkład w badaniach nad cechami wykorzystanymi w omawianych pracach).

Wyniki omawianych autorskich prac zostały w większości przedstawione w formie publikacji [41, 42, 39, 38, 37, 66], z czego dwie [42, 39] były wydane w czasopismach znajdujących się w bazie Journal Citation Reports (JCR).

2. Metody rozpoznawania obiektów na obrazie 2D i 3D

Rozpoznawanie obiektów, które jest przedmiotem zainteresowania w tej pracy odbywa się na podstawie danych wizyjnych, które pochodzą z sensorów robota mobilnego. Sensorami wizyjnymi mogą być klasyczne kamery 2D, które przechwytują kolorowy obraz płaski (dwuwymiarowy) lub kamery 3D (kamery głębi), które są w stanie zarejestrować dwuwymiarową mapę odległości obserwowanych powierzchni od kamery (mapę głębi). Współczesne sensory 3D takie, jak znany Microsoft Kinect wyposażone są w obydwa rodzaje kamer, które dzięki wzajemnej kalibracji są wspólnie w stanie dostarczać obraz posiadający zwykłe kanały koloru RGB (czerwony, zielony i niebieski, ang. *red, green, blue*) oraz dopasowany do nich kanał głębi D (ang. *depth*). Taki połączony obraz nazywany jest obrazem RGB-D. Poza tego typu kamerami istnieją inne metody zdobywania obrazu głębi przez robota mobilnego, takie jak ruchome głowice dokonujące pomiarów za pomocą skanujących dalmierzy laserowych (lidarów). Te sensory są jednak z reguły znacznie droższe, posiadają ruchome, skanujące mechanizmy, przez co wykonują pomiary z niższą częstotliwością niż współczesne kamery głębi, a także zwykle z mniejszą gęstością punktów. Dane z kamer głębi stanowią główny temat zainteresowania pracy, gdyż wydają się być najbardziej praktyczne w rozpoznawaniu obiektów na potrzeby robotyki wewnątrz budynku. Część omawianych metod może jednak działać niezależnie od rodzaju sensora, z którego pochodzą dane głębi.

Cyfrowe kamery 2D są szeroko rozpowszechnione i przystępne cenowo. Ich zasada działania opiera się na przepuszczaniu światła przez szereg elementów optycznych, które kierują odpowiednie składowe barwne do matryc elektronicznych elementów światłoczułych. Elementy te zamieniają natężenie światła na sygnał elektryczny, który jest następnie konwertowany z analogowego na cyfrowy, co umożliwia jego przetwarzanie komputerowe. Zaletami tego typu kamer są wysoka dokładność i rozdzielczość, a także pasywny charakter dokonywanych pomiarów (nie wymagają specjalnego oświetlania sceny), dzięki któremu obiekty mogą być widoczne

z dużej odległości. Z pasywnego trybu pracy wynika też niski pobór mocy, który jest zaletą w robotyce mobilnej. Kamerę taką nazywa się „dwuwymiarową", gdyż umiejscowienie dowolnego wybranego punktu na podstawie obrazu jest możliwe tylko w dwóch wymiarach. Odległość danego punktu od kamery (głębia) pozostaje w ogólnym wypadku niemożliwa do określenia.

Kamery RGB-D, choć nieco droższe i mniej spopularyzowane, obecnie są również przystępne cenowo i łatwe w użyciu. Przechwytywanie głębi opiera się na pomiarach aktywnych, czyli wymagających specjalnego oświetlenia sceny i analizy światła odbitego. Widmo emitowanego promieniowania zwykle zawiera się w zakresie podczerwieni, więc jest niewidoczne dla człowieka. Kamery tego typu mają wbudowany oświetlacz, który, zależnie od zasady działania, wyświetla wzorzec używany do pomiarów triangulacyjnych (np. Microsoft Kinect V1, Asus Xtion) lub światło modulowane. W drugim wypadku faza i częstotliwość sygnału po odbiciu pozwala oszacować głębię (w sensorze Kinect V2). Każdy kanał obrazu otrzymanego z takiej kamery, podobnie jak dla kamery tradycyjnej, stanowi dwuwymiarową macierz, której elementy przechowują intensywność danego koloru lub głębię. Jednak w przypadku kamery RGB-D informację zawarta na obrazie można nazwać trójwymiarową, gdyż pozwala jednoznacznie zlokalizować każdy zarejestrowany punkt w przestrzeni 3D – dlatego te kamery nazywa się często kamerami lub sensorami „3D".

Ponieważ pojedyncza mapa głębi dostarcza informacji zorganizowanej dwuwymiarowo, nie może przechować dowolnej sceny 3D, a jedynie informację o pewnej powierzchni. Dla wybranych współrzędnych X, Y mapa głębi może przypisać tylko jedną odległość. Z tego powodu pojedyncza mapa głębi bywa także nazywana obrazem 2,5D. Znając parametry optyczne (uzyskane dzięki kalibracji) można wprost z mapy głębi otrzymać „zorganizowaną" chmurę punktów o identycznym, macierzowym kształcie: każdy punkt chmury odpowiada jednemu pikselowi obrazu. Taka chmura nadal może być rozumiana jako obraz 2,5D, w którym zamiast jednego kanału głębi mamy 3 kanały reprezentujące współrzędne metryczne X, Y, Z oraz opcjonalnie kolejne kanały niosące informację o innych cechach lokalnych. Aby umożliwić uzyskanie pełniejszej trójwymiarowej reprezentacji (potrzebnej np. do agregacji danych z kilku punktów obserwacji sceny), konieczne jest przekształcenie reprezentacji na postać inną niż dwuwymiarowa macierz. Najczęściej stosowaną w w tym celu reprezentacją sceny jest chmura „niezorganizowana", będąca listą punktów nie zakładającą konkretnej ich kolejności.

Ze względu na inny rodzaj przetwarzanej informacji i organizację danych, metodyka przetwarzania niezorganizowanej chmury punktów jest inna niż metodyka przetwarzania obrazu

2D. W przypadku zorganizowanej chmury punktów z kolei często możliwe jest stosowanie zarówno zmodyfikowanych algorytmów przeznaczonych dla obrazu 2D, jak i dla chmur niezorganizowanych. W tym rozdziale zawarto przegląd i klasyfikację znanych metod rozpoznawania obiektów na podstawie danych 2D, jak i 3D. Ponieważ rozpoznawanie trójwymiarowe stanowi jednak główny temat tej rozprawy, przegląd metod 2D ma charakter uzupełniający i hasłowy.

2.1 Techniki rozpoznawania obiektów na obrazie 2D

Badania nad komputerowym przetwarzaniem obrazu w ubiegłych dekadach zaowocowały licznymi metodami rozpoznawania obiektów. Dla obrazów 2-wymiarowych można wymienić kilka różniących się zasadniczo podejść, a także bardzo wiele technik różniących się szczegółowymi metodami wyznaczenia cech, segmentacji, klasyfikacji, itp. Trudno jest o przedstawienie uniwersalnej, kompletnej ich klasyfikacji, lecz między innymi można wyróżnić:

- Segmentację i klasyfikację cech segmentów (cech kształtu, jasności, koloru, połączonych), by rozpoznać w nich obiekty [67].
- Konwolucyjne (splotowe) przeszukiwanie obrazu z klasyfikacją przesuwającego się obszaru zainteresowania – mogą być stosowanie wektory dowolnych cech obszaru nadające się do klasyfikacji klasycznymi klasyfikatorami lub metody porównujące cechy czy wartości pikseli z modelowymi widokami (różnicowo, korelacyjne [90], metodą kernel descriptors [11], lub innymi technikami).
- Metody polegające na nad-segmentacji obrazu (tzn. na podziale na niewielkie, spójne wewnętrznie obszary), a następnie łączeniu segmentów w celu uzyskania obszarów odpowiadających semantycznie sklasyfikowanym obiektom [100].
- Metody porównywania lub dopasowywania obrazu po transformacie. Przekształcenie może się odbywać do przestrzeni częstotliwości, przestrzeni falkowej lub innej [96, 57].
- Metody zliczające (o kumulacyjnej, dyskretnej przestrzeni hipotez) – oparte na uogólnionej transformacie Hougha lub podobnej [6].
- Wykrywanie punktów charakterystycznych i klasyfikacja na podstawie deskryptorów cech lokalnych (SIFT [58], SURF [7], ORB [74], itp). Często rozwiązanie to jest łączone z metrycznym dopasowaniem modelu do grupy sklasyfikowanych deskryptorów [52].

- Metody oparte na głębokich sieciach neuronowych, zwłaszcza posiadających warstwy konwolucyjne (ang. *convolutional neural network, CNN)* [55, 35, 91]. Jest to obecnie grupa metod dających najwyższą jakość rozpoznawania pod warunkiem dostępności bardzo dużej bazy opisanych obrazów uczących.

Techniki rozpoznawania 2D, choć mają szerszy zakres zastosowań, często nie radzą sobie w słabych warunkach oświetlenia (gdzie dane o jasności i kolorze są bardzo zaszumione lub niedostępne). Obiekty o nietypowo zabarwionych powierzchniach także mogą być problematyczne dla wielu metod, gdyż dwuwymiarowy obraz nie pozwala jednoznacznie odróżnić krawędzi obiektu od krawędzi kolorowych obszarów na jego powierzchni. Taki wpływ kolorystyki dotyczy nie tylko segmentacji, lecz także cech lokalnych, jak i właściwości obrazu w przestrzeni częstotliwości. Rozpoznawanie obiektów na obrazie jest szczególnie utrudnione we wnętrzach budynku, gdyż występuje wiele przedmiotów ubogich w charakterystyczne cechy wizyjne (np. stoły, szafy, ściany, itp). Interpretacja sceny jako całości jest w budynku także utrudniona, gdyż bliskość obiektów do kamery skutkuje dużą zmiennością ich wyglądu ze względu na perspektywę. W literaturze spotykane są porównania skuteczności rozpoznawania lub segmentacji obiektów stosując wybrane algorytmy dla danych RGB (obrazu) oraz RGB-D (z dodaną głębią), takie jak [70], gdzie autorzy wykazują użyteczność wprowadzenia danych głębi. Z tych powodów autor niniejszej pracy wnioskuje, że choć widzenie maszynowe 2D należy uznać za bardziej uniwersalne, warto stosować sensory głębi i techniki rozpoznawania 3D tam, gdzie to możliwe. Pozwala to na uzyskiwanie większej ilości użytecznej informacji i w efekcie otrzymywanie lepszych wyników. Ważną zaletą posiadania danych 3D jest możliwość prawie natychmiastowego uchwycenia kształtu obiektu (który w przypadku obiektów sztywnych jest niezmienny) w celach uczenia algorytmu rozpoznawania. Przy technikach 2D często jest konieczne przygotowanie setek lub tysięcy widoków, aby nauczyć algorytm wyglądu danego obiektu. Z punktu widzenia założeń dotyczących użytecznego robota usługowego sformułowanych we wstępie, taka zaleta danych 3D jest kluczowa.

2.2 Techniki rozpoznawania obiektów w chmurach punktów i na obrazach RGB-D

2.2.1 Wykorzystanie segmentacji

Wiele współczesnych metod, opiera się na segmentacji sceny dokonanej przed właściwym rozpoznawaniem [56, 85, 78, 77, 70, 50]. Takie podejście jest możliwe zarówno na obrazach RGB-D, w chmurach zorganizowanych, jak i niezorganizowanych. Działanie tych metod polega na założeniu możliwości podziału sceny na segmenty w sposób ogólny, gdzie segmenty odzwierciedlają obiekty lub ich znaczące części. Dla tych segmentów obliczane są pewne cechy (od prostych cech liczbowych do histogramów lub innych wielowymiarowych deskryptorów), które mają znaną wymiarowość. Tak sformułowane wektory cech są możliwe do klasyfikacji klasycznymi technikami sztucznej inteligencji. Często po tej segmentacji i prostej klasyfikacji dokonywane jest dodatkowe wnioskowanie, które poprawia jakość rozpoznań [85]. W niniejszej pracy kilka autorskich metod, które zostaną przedstawione w kolejnych rozdziałach bazuje właśnie na takim podejściu. Jego zaletą jest możliwość sprowadzenia problemu rozpoznawania do prostego problemu klasyfikacji. Właściwością tego typu technik jest łatwość stosowania uczących się klasyfikatorów, metod wyznaczania odległości wektorów cech czy metod optymalizacji rozpoznań w kontekście całej sceny.

Metody rozpoznawania obiektów 3D oparte na segmentacji sprawdzają się dobrze w uporządkowanych, stosunkowo prostych środowiskach. Choć same techniki segmentacji 2D i 3D są rozwijane i udoskonalane, należy zaznaczyć, że te metody są z założenia w pewien sposób ograniczone. Wstępna segmentacja narzuca stałą strukturę sceny przed jej semantycznym zrozumieniem, co w wielu sytuacjach może prowadzić do błędów. W złożonych środowiskach ze stykającymi się i częściowo przysłoniętymi obiektami, zadanie segmentacji bez wiedzy dotyczącej konkretnych, występujących na scenie obiektów (tj. przed klasyfikacją) staje się często niemożliwe. Wspominają o tym sami autorzy systemów rozpoznawania obiektów opartych na segmentacji, np. [56]. Ograniczenia te dotyczą również rozpoznawania na obrazie 2D – wiodące na tym polu głębokie sieci konwolucyjne wypierają metody oparte na segmentacji „a priori".

2.2.2 Wykorzystanie deskryptorów lokalnych

W minionych dekadach opisano także wiele technik opartych na lokalnych deskryptorach 3D powierzchni odzwierciedlonych w chmurach punktów. Deskryptorami lokalnymi nazywa się wektory cech, które są przypisane do konkretnego miejsca w chmurze punktów. Cechy te niosą informację na temat geometrii i/lub kolorów obszaru będącego w metrycznym sąsiedztwie punktu ich przypisania. Ważnym aspektem deskryptorów lokalnych jest ich zdolność do zakodowania istotnych właściwości powierzchni (pozwalających na ich rozróżnianie) w jak najmniejszej liczbie wymiarów (często nazywa się to „mocą opisową" deskryptorów). Innym ważnym aspektem jest niezmienniczość deskryptorów, czyli odporność ich wartości na typowe zmiany warunków – np. punktu obserwacji czy oświetlenia sceny. Znanymi przykładami deskryptorów lokalnych są *spin-images* [49] PFH [79], FPFH [75], PFHRGB [3], CVFH [1] oraz niektóre deskryptory obliczane na mapie RGB-D [11]. Pojawiły się także ostatnio prace opisujące deskryptory lokalne 3D oparte na algorytmach genetycznych [98], głębokich sieciach neuronowych [59] czy piramidach przestrzennych [73]. Zastosowania deskryptorów lokalnych są różne – od dopasowywania chmur punktów do rozpoznawania części obiektów i szybkiego określenia orientacji obiektu. W miarę wzrostu dokładności tych deskryptorów oraz mocy obliczeniowej komputerów, pozwalającej na ich szybkie wyznaczanie dla wielu punktów, otwierają się także możliwość ich wykorzystania do bezsegmentacyjnego rozpoznawania obiektów.

Użycie deskryptorów opartych na sąsiedztwach wymaga wyznaczenia takiego sąsiedztwa dla każdego rozpatrywanego punktu. Dla każdego z tych punktów konieczne jest określenie, które z punktów chmury znajdują się w odległości nie większej niż wybrany promień sąsiedztwa (na ogół promień jest rozumiany metrycznie). Zbiór rozpatrywanych punktów (dla których wyznaczane są deskryptory) nie musi się pokrywać ze zbiorem punktów chmury wejściowej. Zadanie określania sąsiedztwa na mapie 2,5 D (obrazie z kamery z kanałem głębi) jest dość proste ze względu na regularne, dwuwymiarowe rozmieszczenie pikseli. Na dowolnej chmurze punktów (tzw. chmurze „niezorganizowanej") jest to czasochłonne i do wydajnego rozwiązania wymaga użycia zaawansowanych technik przeszukiwania, takich jak drzewa binarne lub ósemkowe (używane m.in. w bibliotece PCL), a także, dla dużej liczby punktów, obliczeń masowo-równoległych.

Obliczenie złożonych deskryptorów lokalnych jest z reguły jeszcze bardziej czasochłonne i z tego powodu ich użycie wiąże się prawie zawsze z jakąś metodą ograniczania ich liczebności. Oprócz prostych technik przerzedzania chmury punktów, najczęściej wykorzystywaną

metodą ograniczania liczby deskryptorów jest wyznaczanie punktów charakterystycznych, dla których deskryptory będą obliczane. Podejście to polega to na zastosowaniu uniwersalnego algorytmu heurystycznego, który jest w stanie niskim kosztem wyłonić punkty sceny, które niosą najwięcej użytecznej informacji. Tak wybrane punkty zwykle wyróżniają się ze swojego otoczenia (mogą nimi być np. narożniki), dzięki czemu nadają się do powtarzalnego i skutecznego rozpoznawania [30, 34].

Jak pokazano w badaniu na dużą skalę [29], wybór punktów charakterystycznych wpływa na dokładność klasyfikacji, choć nie w tak dużym stopniu jak wybór deskryptorów. Wahania pola pod krzywą klasyfikacji ROC (ang. *receiver operating curve*) są rzędu 5% dla różnych detektorów punktów charakterystycznych przy stałym deskryptorze lokalnym. Wahania dla różnych deskryptorów przy jednakowym wyborze punktów charakterystycznych sięgają natomiast 25%. Autorzy zacytowanego artykułu wnioskują, że deskryptor PFHRGB (Point Feature Histogram RGB) jest najdokładniejszym z badanych deskryptorów, ale także jednym z najbardziej czasochłonnych. Z kolei autorzy [2] dokonują porównania dokładności rozpoznawania klasy i instancji obiektów opartego na deskryptorach dla systemu z użyciem punktów charakterystycznych i dla systemu, gdzie deskryptory są obliczane na całej scenie (przetworzonej filtrem wokselowym). Z badania wynika, że metoda wykorzystująca całą scenę ma wyższą dokładność, lecz ceną znacznie dłuższego czasu obliczeń (ok. 5-6 razy). Wynika to najprawdopodobniej z ograniczonej powtarzalności punktów charakterystycznych wskazywanych przez detektory [34]. Wspomniane badania dotyczą widoków pojedynczych obiektów.

W realistycznym środowisku robotyki mobilnej problem ograniczenia obszaru poszukiwań jest tym bardziej istotny, gdyż czas przetwarzania jest kluczowym czynnikiem. Jak uzasadniono we wstępie, interesujące robota obiekty mogą stanowić tylko niewielką część całej sceny i nie ma potrzeby klasyfikacji wszystkich elementów obserwowanego świata. W jednym z późniejszych rozdziałów tej pracy zostanie przedstawiona autorska metoda ograniczania uwagi przy wyliczaniu i klasyfikacji różnych deskryptorów sceny, która uwzględnia powyższe obserwacje i założenia.

2.2.3 Dopasowanie modelu

Szczególnie użytecznym podejściem do rozpoznawania obiektów o znanym, niezmiennym kształcie na scenie 3D jest dopasowywanie modelu do sceny. Dopasowanie to może odbywać się na różne sposoby – metodami heurystycznymi i iteracyjnymi typu RANSAC (ang.

random sample consensus) [31]) lub poprzez konwolucyjne przeszukiwanie przestrzeni możliwych pozycji i orientacji obiektu (6-wymiarowej dla obiektów sztywnych). Podejścia tego typu w naturalny sposób mogą się łączyć z użyciem deskryptorów cech lokalnych. Deskryptory są użyteczne jako heurystyka dopasowań chmury punktów, a także do porównania przestrzennego podczas obliczania funkcji jakości dopasowania. Zaletą tego typu metod jest możliwość porównywania kształtów obiektów na wybranym poziomie szczegółowości bez konieczności ich parametrycznego opisu (bez opracowywania globalnych cech kształtu obiektu). Także, ze względu na brak segmentacji, metody te sprawdzają się przy dużym stopniu przysłonięcia lub stykania się obiektów oraz w obecności szumów pomiarowych. Techniki te w kontekście rozpoznawania obiektów charakteryzują się jednak bardzo wysokim kosztem obliczeniowym. Jest to spowodowane dużą, ciągłą przestrzenią rozpatrywanych pozycji obiektu, możliwością szukania tylko jednego obiektu na raz i koniecznością transformacji wielu punktów dla każdego sprawdzanego położenia i orientacji.

Dopasowywanie chmur punktów metodą RANSAC stosowane jest często w dopasowywaniu scen przechwyconych w niewielkim odstępie czasu na potrzeby jednoczesnej lokalizacji i tworzenia mapy (ang. *SLAM - simultaneous localization and mapping*) [24] oraz odometrii wizyjnej [54]. Dopasowanie scen w takich zastosowaniach umożliwia znalezienie wspólnego układu współrzędnych, niezbędnego do agregacji danych z wielu pomiarów. Dla metod działających na całej chmurze zwykle wykorzystuje się stosunkowo niewielki zbiór punktów charakterystycznych (których dla typowych scen jest dużo). Podejście takie w wykrywaniu obiektów jest w ogólnym wypadku mało skuteczne, gdyż wiele obiektów posiada zbyt mało punktów charakterystycznych. Jednocześnie różnica między widokiem obiektu na scenie a jego modelem (zarejestrowanym zwykle w innych warunkach) może być zbyt duża do stabilnego wyznaczenia takich punktów. Dlatego w wykrywaniu obiektów techniki RANSAC najczęściej wykorzystuje się na „gęstych” (nie ograniczonych do punktów charakterystycznych) fragmentach chmur. Detekcja obiektów w taki sposób jest skuteczna tylko w szczególnych warunkach:

- znając przybliżoną pozycję obiektu na podstawie innych algorytmów [45],
- przetwarzając odsegmentowane wcześniej obszary,
- stosując geometryczne uproszczenia modelu [81] (dobrze się to sprawdza np. w przypadku płaskich ścian),

- znacząco ograniczając stopnie swobody dopasowywanego modelu (np. gdy znana jest orientacja obiektu).

Oprócz detekcji, innym ważnym dla robotyki zastosowaniem metod RANSAC w chmurach punktów jest określanie orientacji obiektu na potrzeby manipulacji.

Współczesne metody o najwyższej wykazanej skuteczności wykrywania klasy obiektu na dużych zbiorach różnorodnych scen [87, 88, 71] stosują konwolucyjne przeszukiwanie przestrzeni możliwych pozycji i orientacji obiektu. W celu obniżenia złożoności algorytmów, ogranicza się stopnie swobody związane z obrotem do jednego (wokół osi pionowej). Niestety tego typu metody wiążą się z ogromnym kosztem obliczeniowym. W artykule [71] podano, że czas wyszukiwania jednej klasy obiektu na pojedynczej scenie wynosi 10-30 minut. Z kolei autorka [87] na wystąpieniu [113] wskazała czasy od kilku godzin do całego dnia dla wyszukiwania krzesła na jednej scenie 3D. Metoda ta (*sliding shapes*) została później rozwinięta do formy używającej głębokich sieci neuronowych, a także znacząco przyspieszona za pomocą technik heurystycznych i obliczeń masowo-równoległych [88]. Choć ostateczny czas przetwarzania sceny nie jest podany wprost w artykule, na podstawie dostarczonych informacji (na temat czasochłonności poszczególnych kroków i ogólnego osiągniętego przyspieszenia), można wywnioskować, że czas wyszukiwania jednego obiektu na jednej scenie wynosi od pół minuty do kilku minut. W eksperymentach użyto wydajnej, dedykowanej do obliczeń naukowych karty graficznej NVIDIA K40. Widzimy więc, że mimo pojawienia się w ostatnich czasach coraz lepszych i bardziej ogólnych metod rozpoznawania obiektów i dokonania możliwie szybkiej, równoległej implementacji, do osiągnięcia praktycznej użyteczności potrzebny jest znaczący rozwój ich wydajności. Ze względu na szczególną istotność zagadnienia szybkiego przetwarzania danych w robotyce, część niniejszej pracy jest poświęcona właśnie technikom heurystycznym w wyszukiwaniu obiektów oraz implementacji algorytmów masowo-równoległych.

3. Kontekst obiektów

3.1 Wprowadzenie

Większość metod rozpoznawania obiektów wykorzystuje jedynie wiedzę na temat cech wizyjnych i metrycznych. Jednak, jak uzasadniono w [32], taki rodzaj informacji jest w stanie uchwycić różnice między klasami obiektów tylko w ograniczonym stopniu. Badania neuropsychologiczne pokazują, że kontekst wizyjny odgrywa kluczową rolę w przypadku percepcji ludzkiej [62, 10, 20]. W pracy [9] cechy kontekstu zostały podzielone na 3 klasy: kontekst przestrzenny, kontekst skali oraz kontekst semantyczny.

Kontekst przestrzenny dotyczy prawdopodobieństwa znalezienia obiektu w danej pozycji względem innych obiektów. Dla przykładu, stół zwykle stoi na podłodze, a kubek na stole. Kontekst skali odnosi się do wielkości obiektu względem innych obiektów w pobliżu. Kontekst semantyczny natomiast jest określany na podstawie współwystępowania obiektów na danej scenie. Zdaniem autora tej pracy wiedza o kontekście jest cennym źródłem informacji w widzeniu maszynowym ze względu na powszechne występowanie różnych prawidłowości obserwowanego świata. Te prawidłowości, choć nie zawsze łatwe do opisu, są obecne w większości scen rzeczywistych (w tym takich, które dotyczą robota usługowego) i mogą zostać wykorzystane do uproszczonego modelowania środowiska.

Istnieją różne prace pokazujące wartość tego rodzaju informacji dla celów rozpoznawania obiektów. Metoda przedstawiona w [22] korzysta z układów 2-wymiarowych, co pozwala na odróżnienie różnych kierunków i odległości. Te typowe układy są uczone statystycznie na podstawie dużych baz obrazów. W [85] autorzy używają relacji „wspierających" lub „podpierających" (ang. *support*) w celu osiągnięcia lepszego zrozumienia sceny trójwymiarowej. Po opublikowaniu przez autora metody, która zostanie omówiona w tym rozdziale [41], ukazały się także inne artykuły czyniące skuteczny użytek z informacji o relacjach semantycznych obiektów, takie jak [71].

Ten rozdział przedstawia badania wykonane przez autora rozprawy nad wykorzystaniem kontekstu przestrzennego i semantycznego do rozpoznawania obiektów 3D. Badania opisano również w publikacji [41]. Przedstawiona metoda rozpoznawania bazuje na połączeniu klasyfikacji opartej na cechach odizolowanych segmentów, jak i informacji kontekstowej dwojakiego rodzaju. Pierwszym z nich są relacje metryczne, które są użyteczne do łączenia kilku części złożonych, sztywnych obiektów. Drugim są relacje semantyczne, które pozwalają na odróżnienie obiektów nie posiadających (wykrywanych) rozstrzygających cech (czyli obiektów będących bardzo podobnych do innych lub takich, które ze względu na ograniczoną widoczność nie dostarczają wystarczającej informacji o cechach własnych). Aby przeprowadzić wnioskowanie kontekstowe, opracowano metodę korzystającą z autorskiego uogólnienia losowych pól Markowa, przedstawionego w podrozdziale 3.5.

Proces klasyfikacji składa się z 5 etapów, które zostaną omówione szczegółowo w dalszych podrozdziałach:

- wydobycia cech,
- segmentacji,
- formułowania hipotez,
- wnioskowania niskopoziomowego,
- wnioskowania wysokopoziomowego.

Wydobycie cech (na potrzeby przeprowadzonych eksperymentów) polega na obliczeniu zestawu trzech stosunkowo prostych cech lokalnych powierzchni opartych na sąsiedztwach punktów. Aby otrzymać zbiór oddzielnych, prostych (i ciągłych) fragmentów obiektów, obliczone cechy są przedstawiane w postaci mapy dwuwymiarowej. Segmentacja odbywa się na mapie wyznaczonej w poprzednim etapie metodami typowymi dla widzenia maszynowego 2D.

W etapie formułowania hipotez, dla każdego segmentu sceny zostaje obliczony histogram cech, który następnie jest dopasowany korelacyjnie do histogramów wzorcowych (przygotowanych wcześniej widoków fragmentów znanych obiektów). Jeżeli podczas porównania z daną częścią znanego obiektu korelacja histogramów przekracza pewien ustalony próg (właściwy dla obiektu) oraz segment spełnia kilka dodatkowych (opisanych później) warunków przybliżonego podobieństwa, zostaje sformułowana „hipoteza segmentu”. Taka hipoteza stanowi o

przynależności rozważanego segmentu do określonego obiektu. Wiele hipotez może zostać sformułowanych dla każdego segmentu. Dodatkowo, dla każdego segmentu jest zakładana hipoteza „zerowa", oznaczająca, że obiekt nie jest znany.

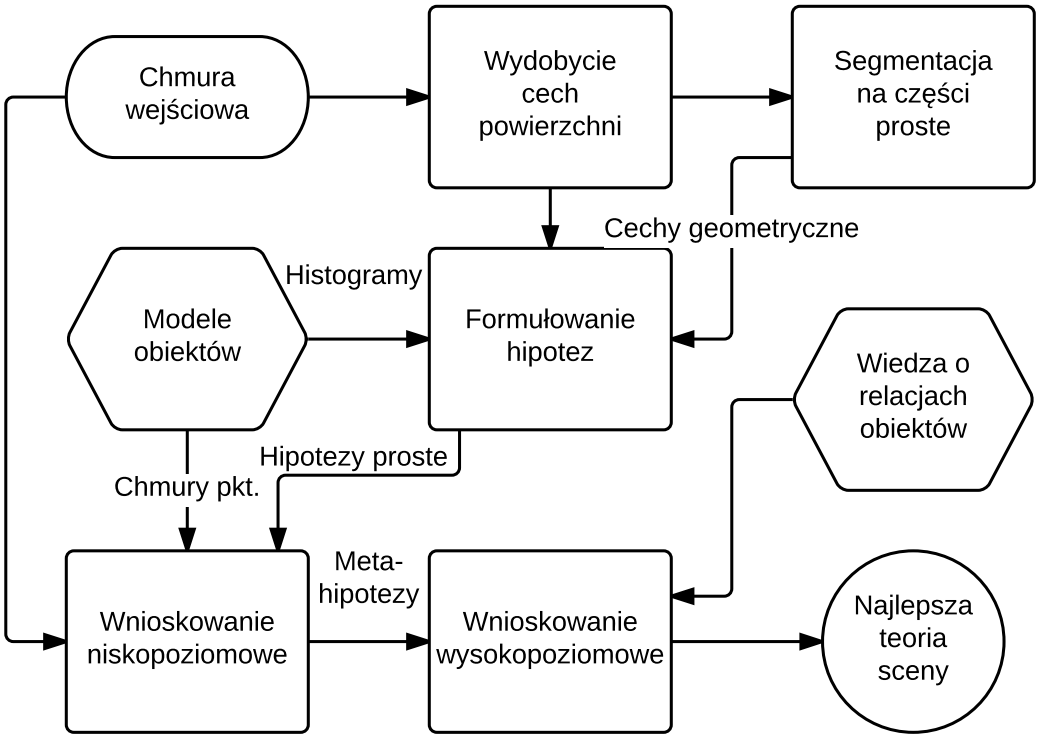
Podczas wnioskowania niskopoziomowego dla otrzymanych hipotez obiektów złożonych (tzn. obiektów składających się z wielu części, rozumianych jako możliwe do odsegmentowania fragmenty), przeprowadzane jest dopasowanie metryczne chmury punktów. Jego celem jest ustalenie prawdopodobnych ułożeń 3D obiektów i powiązania segmentów sceny, które mogą należeć do tego samego obiektu złożonego. Te wynikowe hipotezy – łączące kilka hipotez segmentów w hipotezy obiektów nazywane będą „metahipotezami".

Wnioskowanie wysokopoziomowe polega na kilku krokach. Są nimi: tworzenie połączeń między hipotezami segmentów, grupowanie segmentów oraz właściwe wnioskowanie (optymalizacja). Połączenia między segmentami są tworzone zgodnie ze znanymi relacjami semantycznymi. Grupowanie segmentów polega na znalezieniu podziałów sceny na rozłączne zbiory możliwie powiązanych ze sobą segmentów (w celu ograniczenia złożoności obliczeniowej). Wnioskowanie właściwe na każdym ze zbiorów wynikowych (czyli przypisanie ostatecznych etykiet do segmentów) jest wykonywane za pomocą uogólnionych losowych pól Markowa oraz algorytmu genetycznego. Celem tego etapu jest znalezienie najlepszej teorii sceny, czyli kombinacji hipotez segmentów, która maksymalizuje funkcję energii przy danej strukturze topologicznej powiązanych segmentów.

Diagram systemu został przedstawiony na rysunku 3.1

3.2 Cechy lokalne powierzchni

Pierwszym etapem opisywanego systemu jest obliczenie lokalnych cech powierzchni trójwymiarowych. Cechy te są obliczane w pierwszej kolejności, gdyż wykorzystano je nie tylko do wyliczania histogramów celem klasyfikacji segmentów, ale także do samej segmentacji. Użytymi cechami są nachylenie, wypukłość oraz anizotropia wypukłości – cechy te łączą istotną informację o powierzchniach oraz prostotę pozwalającą na wydajne budowanie i porównywanie histogramów [39].



Rysunek 3.1: Diagram systemu. Ośmiokąty przedstawiają bazy wiedzy systemu, natomiast prostokąty stanowią etapy przetwarzania. Zaokrąglonymi polami oznaczono wejście i wyjście systemu

3.2.1 Wypukłość

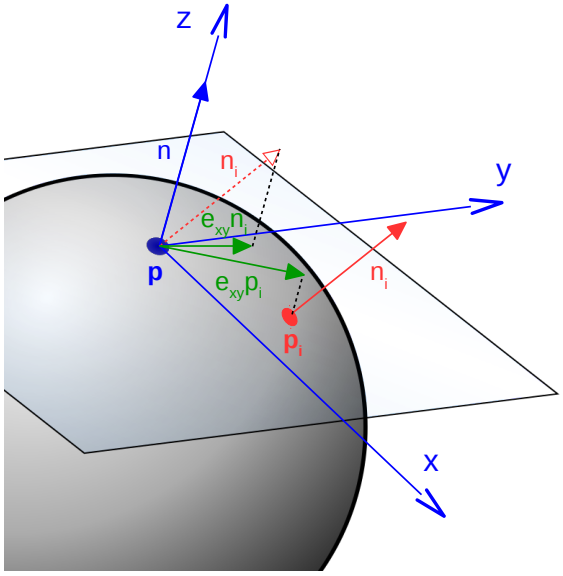
Pierwszą wykorzystaną cechą geometryczną jest *wypukłość*, która stanowi miarę zakrzywienia powierzchni w stronę zewnętrzną (z której jest obserwowana) w danym punkcie. Jeśli powierzchnia jest lokalnie płaska, wartość wypukłości jest zerowa, natomiast dla powierzchni wklęsłych, wypukłość jest ujemna. Zdefiniujmy pomocnicze pojęcie *wypukłości w punkcie* $\mathbf{p}$ *ze względu na* $\mathbf{p_i}$ w lokalnym układzie współrzędnych punktu $\mathbf{p}$, w którym $\mathbf{p} = \begin{bmatrix}0 & 0 & 0\end{bmatrix}^T$ oraz $\mathbf{n} = \begin{bmatrix}0 & 0 & 1\end{bmatrix}^T$ stanowi wersor normalny do powierzchni w $\mathbf{p}$ (Rys. 3.2).

Wypukłości $(C(\mathbf{p}, \mathbf{p_i}))$ w punkcie $\mathbf{p}$ ze względu na $\mathbf{p_i}$ definiuje wzór (3.1).

$$C(\mathbf{p}, \mathbf{p_i}) = (\widehat{\mathbf{e_{xy}}\mathbf{p_i}}) \cdot (\widehat{\mathbf{e_{xy}}\mathbf{n_i}}) \cdot \frac{\arccos(\mathbf{e_z}\mathbf{n_i})}{\pi/2}, \tag{3.1}$$

gdzie: $\mathbf{e_{xy}} = \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 0\end{bmatrix}, \mathbf{e_z} = \begin{bmatrix}0 & 0 & 1\end{bmatrix}$, zaś $\mathbf{n_i}$ to wersor normalny do powierzchni w punkcie $\mathbf{p_i}$. *Wypukłość w* $\mathbf{p}$ możemy zdefiniować wzorem: (3.2).

$$C(\mathbf{p}) = \frac{1}{K} \sum_{\mathbf{p_i} \in N_{\mathbf{p}}(r_1) \setminus N_{\mathbf{p}}(r_2)} C(\mathbf{p}, \mathbf{p_i}), \tag{3.2}$$



Rysunek 3.2: Ilustracja układu współrzędnych, w którym są zdefiniowane cechy lokalne punktu p (na niebiesko)

gdzie $N_{\mathbf{p}}(r_1) \setminus N_{\mathbf{p}}(r_2)$ jest różnicą sąsiedztw punktu $\mathbf{p}$ o wybranych promieniach r_1 i r_2. Założono, że $r_1 > r_2$.

3.2.2 Anizotropia wypukłości

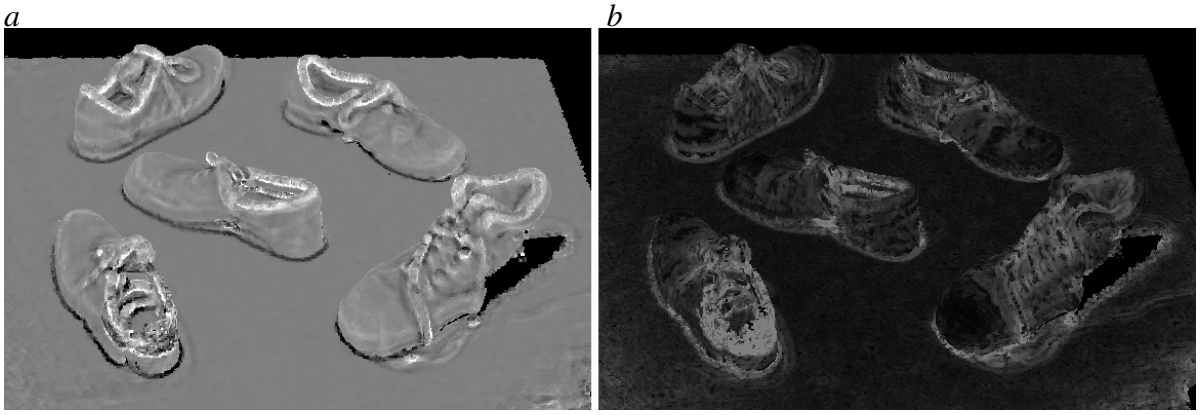
Aby uchwycić więcej informacji na temat kształtu powierzchni niż daje wypukłość, wprowadzono cechę *anizotropii wypukłości* $(A(\mathbf{p}))$. Jest to wielkość określająca jak duża jest zmienność wypukłości w różnych kierunkach w otoczeniu danego punktu powierzchni, zdefiniowana za pomocą (3.3).

$$A(\mathbf{p}) = \max_{\mathbf{p_i} \in N_{\mathbf{p}}(r_1) \setminus N_{\mathbf{p}}(r_2)} [C(\mathbf{p}, \mathbf{p_i})] - \min_{\mathbf{p_i} \in N_{\mathbf{p}}(r_1) \setminus N_{\mathbf{p}}(r_2)} [C(\mathbf{p}, \mathbf{p_i})] - 1. \tag{3.3}$$

Anizotropia wypukłości jest więc cechą czułą na kierunkowość wypukłości – jej wartość będzie minimalna zarówno dla powierzchni płaskiej, jak i dla kulistej, lecz większa dla powierzchni walcowych, stożkowych czy dla krawędzi podłużnych. Przykładowe mapy wypukłości i anizotropii obliczonych na chmurach punktów przedstawiono na rys. 3.3.

3.2.3 Kątowe nachylenie powierzchni

Kolejną wykorzystaną cechą jest *kątowe nachylenie* powierzchni $(I(\mathbf{p}))$– jego wartość jest proporcjonalna do kąta utworzonego między wektorem normalnym do powierzchni, a wekto-



Rysunek 3.3: Przykładowa wizualizacja cech: a – wypukłość oraz b – anizotropia wypukłości. Jasność na obrazie jest proporcjonalna do wartości cech (w zakresie $[-1;1]$). Czarny kolor odpowiada minimalnej, zaś biały maksymalnej wartości cechy (poza obszarami pustymi, które są czarne)

rem przyciągania ziemskiego (grawitację mierzono wykorzystując akcelerometr wbudowany w sensor Kinect). Definicję tej cechy przedstawia wzór (3.4).

$$I(\mathbf{p}) = \frac{2 \cdot \arccos(\mathbf{n} \cdot \hat{\mathbf{g}})}{\pi} - 1, \tag{3.4}$$

gdzie: $\mathbf{n}$ jest wersorem (wektorem jednostkowym) normalnym do powierzchni w $\mathbf{p}$, a $\hat{\mathbf{g}}$ znormalizowanym do jednostkowej długości wektorem grawitacji, o zwrocie zgodnym z siłą przyciągania ziemskiego. Cecha ta jest niezmiennicza względem przemieszczeń liniowych oraz względem obrotu wokół osi pionowej. Mimo braku niezmienniczości na inne obroty, cecha ta jest wysoce użyteczna dla celów robotyki usługowej, gdyż wiele obiektów spotykanych wewnątrz budynku ma określoną podstawę (powierzchnię styku z podłożem).

3.3 Segmentacja i formułowanie hipotez

Celem kolejnego kroku przetwarzania jest otrzymanie listy segmentów chmury punktów, które reprezentują nie więcej niż jeden obiekt i jednocześnie stanowią przynajmniej znaczącą jego część. W środowiskach wnętrza budynku spotykamy wiele przedmiotów mających jedną dominującą, gładką powierzchnię, której odsegmentowanie jest stosunkowo proste (np. kubek, czajnik, mysz komputerowa, zlew, piłka, itd.). Jednakże zadanie segmentacji jest znacznie trudniejsze w przypadku obiektów posiadających wiele istotnych powierzchni, w szczególności powierzchni płaskich (duża część mebli, pudełka, książki, itd.). W tych wypadkach segmentacja sceny na pełne obiekty bez wcześniejszej ich znajomości jest trudna, a czasami wręcz niemożliwa.

Prostym przykładem trudnej sytuacji odsegmentowania całego przedmiotu jest krzesło, którego noga na pojedynczym widoku kamery jest często oddzielonym od reszty klastrem punktów (z powodu częściowego przysłonięcia przez siedzisko). Wielo-kolorowe powierzchnie, a także częściowe przysłonięcia przez inne elementy sceny jeszcze bardziej utrudniają zadanie kojarzenia ze sobą różnych części tego samego obiektu. Innym problemem jest trudność określenia granic między złożonymi obiektami w sytuacji, gdy te się stykają. Aby prawidłowo rozwiązać te problemy, konieczne jest pewne wnioskowanie na podstawie wcześniej wprowadzonej wiedzy o obserwowanych obiektach. System opisywany w tym rozdziale rozwiązuje to zadanie poprzez początkową segmentację (z dopuszczeniem pewnej nad-segmentacji) sceny na proste, gładkie fragmenty, a następnie zastosowanie wnioskowania przestrzennego.

Aby otrzymać proste segmenty (które będą traktowane jako części obiektów), tworzona jest dwuwymiarowa mapa poprzednio wyliczonych cech prostych, która jest następnie wygładzana filtrowaniem *mean-shift* [18]. Właściwa segmentacja jest przeprowadzana algorytmem wykrywania krawędzi *Canny'ego* [14]. Ta procedura skutkuje segmentacją sceny na trójwymiarowe (właściwie 2,5-wymiarowe) gładkie obszary, gdyż nieciągłości i załamania na powierzchniach odzwierciedlone są na mapie cech jako krawędzie.

Po utworzeniu listy segmentów, kolejnym krokiem jest przyporządkowanie do każdego z nich zbioru hipotez. Odbywa się to za pomocą porównania znormalizowanego histogramu cech powierzchni danego segmentu do histogramu obliczonego dla części modelowych widoków obiektu. Miarą odległości histogramów jest współczynnik korelacji Pearsona [109]:

$$corr(X,Y) = \frac{cov(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}, \tag{3.5}$$

gdzie: X i Y są porównywanymi histogramami (traktowanymi statystycznie jak wektory kolejnych prób zmiennej losowej), σ_X i σ_Y to odpowiadające im (estymowane) odchylenia standardowe, zaś μ_X i μ_Y to wartości średnie tych wektorów.

Jeżeli korelacja przekracza określony próg, zostaje sformułowana prosta hipoteza:

„Segment X jest częścią Y obiektu klasy Z".

Korelacja histogramów między obserwowanymi i modelowymi częściami może być różna zależnie od rodzaju powierzchni (korelacja jest zwykle większa przy jednolitych cechach powierzchni), zakłóceń sensora (które stanowią istotną przeszkodę gdy obiekty są widziane sensorem Kinect z odległości przekraczającej 3 m) oraz stopnia przysłonięcia. Z tego powodu korelacja histogramów nie powinna być traktowana jako pełna miara jakości potencjalnego

dopasowania. Progi korelacji mogą być dobierane lub uczone indywidualnie dla każdej części. W opisywanym systemie oprócz porównań histogramów cech, do wstępnej klasyfikacji użyto kilku dodatkowych cech podobieństwa. Sprawdzenie przybliżonej zgodności tych cech z modelem dla obserwowanych segmentów stanowi heurystykę umożliwiającą odrzucanie fałszywych klasyfikacji. Tymi cechami są pole powierzchni, maksymalna odległość punktu od geometrycznego środka ciężkości segmentu, a także wysokość nad podłogą. Dla tych cech określono dopuszczalne przedziały wartości dla każdej klasy obiektu. Niezależnie od hipotez sformułowanych opisaną metodą, do każdego segmentu jest przyporządkowywana także hipoteza zerowa (tzn. hipoteza nieznanego obiektu), co pozwala systemowi na klasyfikację segmentu jako „nie wiem" w razie braku wystarczających przesłanek do przyporządkowania go do znanej klasy lub w razie silnych przesłanek wykluczających takie przyporządkowanie.

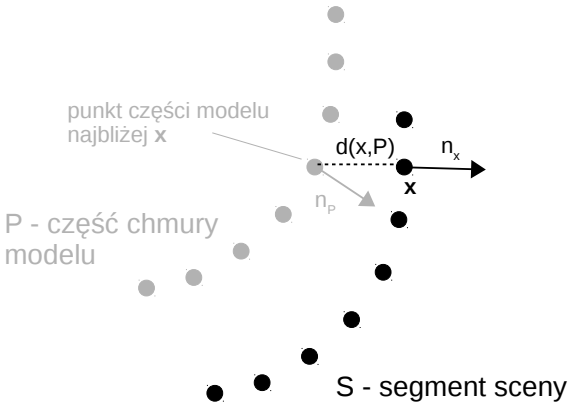
3.4 Wnioskowanie niskopoziomowe

Celem wnioskowania niskopoziomowego jest znalezienie potencjalnych złożonych, sztywnych obiektów na podstawie metrycznych relacji ich części ustalonych dla obiektów modelowych, a więc zamiana listy prostych hipotez przyporządkowanych do poszczególnych segmentów na listę „metahipotez" dotyczących całych obiektów. Odbywa się to za pomocą algorytmu typu Monte Carlo, który dąży do znalezienia transformacji maksymalizującej jakość F dopasowania, określoną równaniem:

$$F = \sum_{P \subseteq O} \sum_{S \subseteq I} \left(\frac{1}{|S|} \sum_{x \in S} \min d(x,P) \arccos(\mathbf{n_x} \mathbf{n_P}) \right)^{-1}, \tag{3.6}$$

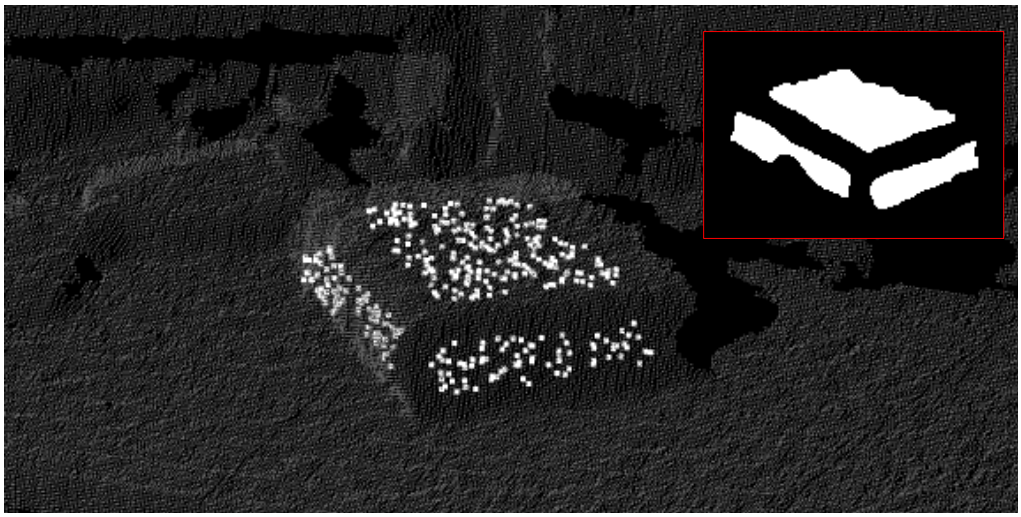
gdzie: O jest chmurą punktów modelowego obiektu, P jest chmurą punktów danej części modelu, I jest chmurą punktów sceny, S jest chmurą punktów segmentu sceny, któremu przypisano hipotezę części odpowiadającej P, d jest funkcją odległości euklidesowej. $\mathbf{n_x}$ i $\mathbf{n_P}$ są wersorami normalnymi powierzchni odpowiednio dla punktu x i odpowiadającego mu najbliższego punktu części modelu (Rys. 3.4).

Jak można zauważyć, przedstawiony algorytm dąży jednocześnie do minimalizacji odległości, jak i nierównoległości dopasowywanych powierzchni. Znając centroidę obserwowanego segmentu i zakładając, że obiekt może się obracać jedynie wokół osi pionowej (co jest sensownym założeniem dla większości obiektów wnętrza budynku), dopasowanie może zostać przeprowadzone w stosunkowo krótkim czasie. Prosty przykład dopasowywania został zilustrowany na Rys. 3.5, gdzie ukazano scenę z poszukiwanym obiektem (książką). W prawym



Rysunek 3.4: Schematyczna ilustracja wyliczania jakości dopasowania modelu (punkty szare) do segmentu sceny (punkty czarne)

górnym rogu pokazano modelowy widok obiektu, który składa się z 3 wyodrębnionych segmentów, odpowiadających widocznym gładkim powierzchniom. Takich różnych modelowych widoków może być kilka dla pojedynczego obiektu. Skutecznie dopasowaną do sceny chmurę punktów modelu (przerzedzoną w celu przyspieszenia obliczeń), pokazano w postaci białych kropek.



Rysunek 3.5: Dopasowywanie modelu książki. W tym przypadku została sformułowana metahipoteza łącząca 3 segmenty w obiekt złożony

Każdy segment sceny dla którego jakość dopasowywania jest powyżej ustalonego progu zostaje uznawany za znalezioną część i zostaje dodany do metahipotezy. Po tak opisanym procesie dopasowywania i łączenia hipotez, identyczne metahipotezy są łączone w celu uniknięcia powtórzeń.

3.5 Wnioskowanie wysokopoziomowe

Celem ostatniego kroku przetwarzania jest znalezienie najlepszej teorii sceny (zbioru hipotez dotyczących poszczególnych segmentów). Aby to osiągnąć, system przeprowadza wnioskowanie dla otrzymanych metahipotez przy użyciu bazy znanych relacji semantycznych między obiektami. Relacje użyte w opisywanym systemie to: *pod, nad* i *obok*. Używane relacje są jednokierunkowe, a więc *biurko pod książką* oznacza, że oczekujemy znalezienia biurka pod książką, ale niekoniecznie książki nad biurkiem. Do relacji przyporządkowana jest pewna waga oraz maksymalny zasięg (odległość między obiektami). Przed właściwym wnioskowaniem, możliwe relacje semantyczne między segmentami sceny zostają wykryte i zapisane w postaci grafu z połączeniami warunkowymi (zależnymi od sformułowanych hipotez poszczególnych segmentów). Gdy połączenia zostaną już określone, scena jest dzielona na niepowiązane klastry segmentów (bez żadnych połączeń między klastrami) – dla każdego z nich wnioskowanie może być przeprowadzone niezależnie dla ograniczenia złożoności obliczeniowej.

Ogólny problem wnioskowania jest opisany następująco:

Dana jest lista N węzłów (segmentów) $s_1,...,s_N$ i lista M etykiet $x_1,...,x_M$. Każdy węzeł może przyjmować dowolną etykietę od 1 do M. Etykiety są ukrytymi właściwościami węzłów, które chcemy wywnioskować na podstawie wektora obserwacji $(u_1,...,u_N)$.

Klasycznym sposobem rozwiązania problemu jest jego sformułowanie w kategoriach problemu optymalizacyjnego. W tym celu konieczne jest określenie funkcji energii F:

$$F : (\vec{x}, \vec{u}) \rightarrow R, \tag{3.7}$$

gdzie: $\vec{x} = \{x_1,...,x_N\}$, $\vec{u} = \{u_1,...,u_N\}$. Funkcja F przyporządkowuję „energię" do każdej możliwej kombinacji $(\vec{x}, \vec{u})$, która dla danego zbioru węzłów stanowi miarę niezgodności wektora etykiet z wektorem obserwacji. Celem optymalizacji jest znalezienie wektora etykiet $\vec{x}_{opt}$ który daje najniższą energię wśród wszystkich możliwych $\vec{x}$.

$$\vec{x}_{opt} = argmin_x F(\vec{x}, \vec{u}). \tag{3.8}$$

Ponieważ optymalizacja dotyczy tylko ukrytych właściwości elementów, zaś obserwacje $\vec{u}$ są dla danej sceny stałe, równanie (3.8) przyjmuje formę:

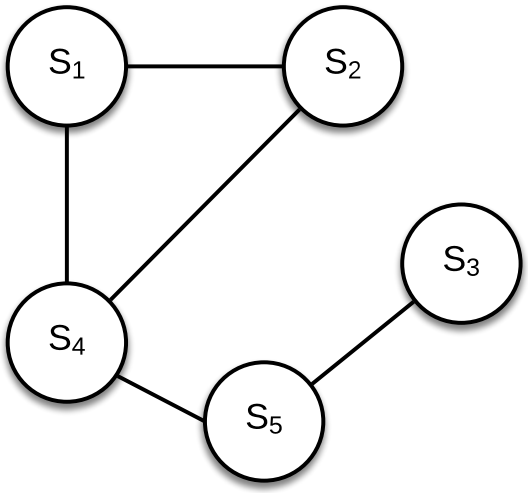
$$\vec{x}_{opt} = argmin_x F(\vec{x}). \tag{3.9}$$

Określając funkcję energii F dla rozważanego wektora etykiet $\vec{x}$, powinny zostać wzięte pod uwagę zarówno właściwości obserwowanych segmentów, jak i zależności kontekstowe wynikające z ich wzajemnego ułożenia na scenie. Bardzo dobrym sposobem modelowania takich sytuacji jest użycie dyskretnego losowego pola Markowa (ang. *Markov Random Field – MRF*) [95].

MRF jest zbiorem zmiennych losowych posiadających własność Markowa (czyli takich, których rozkład prawdopodobieństwa determinują jedynie sąsiadujące stany). Sposób powiązania między stanami opisuje graf G, który definiuje się następująco:

$$G = (V, E), \tag{3.10}$$

gdzie: $V = \{s_1, .., s_N\}$ odpowiada jego węzłom. Każdy węzeł s_i jest związany z etykietą x_j. Krawędzie E oznaczają ograniczenia kontekstowe sąsiadujących węzłów. Przykładowe losowe pole Markowa pokazano na Rys. 3.6.



Rysunek 3.6: Przykładowe losowe pole Markowa. Krawędzie grafu oznaczają ograniczenia kontekstowe: S_1 zależy od S_2 i S_4; S_5 zależy od S_4 i S_3, itd.

W losowym polu Markowa funkcja energii $F(.)$ składa się z dwóch członów:

$$F(\vec{x}) = F_{data}(\vec{x}) + F_{prior}(\vec{x}). \tag{3.11}$$

Pierwszy człon (ang. *data term*) i zdefiniowany jest następująco:

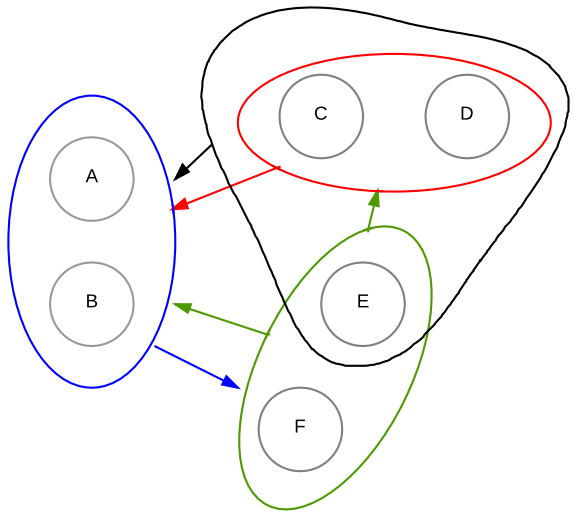
$$F_{data}(\vec{x}) = \sum_{s \in V} f_s(x_s), \tag{3.12}$$

gdzie funkcja $f_s(x_s)$ zawiera informację o stopniu, w jakim przypisanie etykiety x_s do węzła s przeczy obserwacji.

Drugi człon (ang. *prior term*) zawiera informację na temat ograniczeń kontekstowych między węzłami grafu.

$$F_{prior}(\vec{x}) = \sum_{(p,q)\in E} f_{pq}(x_p, x_q). \tag{3.13}$$

Główna różnica między użytym modelem i konwencjonalnym (opisanym powyżej) losowym polem Markowa polega na łączeniu metahipotez sceny (które mogą zawierać więcej niż jeden segment), w przeciwieństwie do łączenia węzłów prostych, jak ma to miejsce w MRF. Stanowi to istotną modyfikację, gdyż nie chcemy uzależniać wpływu metahipotezy od liczby segmentów, które zawiera obiekt „wpływający" na teorię (np. pożądany może być jednakowy wpływ krzesła i taboretu na rozpoznanie stołu, niezależnie od faktu, że krzesło posiada więcej wykrywanych części). Jednocześnie w opisywanym modelu relacje semantyczne mogą nieść informację o różnym poziomie istotności dla poszczególnych obiektów, które różnią się liczebnością segmentów. Z tych powodów połączenia grafu (w odróżnieniu od MRF) opisano jako skierowane. Ilustracja zastosowanego modelu została przedstawiona na Rys. 3.7.



Rysunek 3.7: Połączenia między metahipotezami (hipotezami pełnych obiektów) w uogólnionym losowym polu Makrowa. Segmenty (oznaczone lierami) mogą popierać różne metahipotezy jednocześnie

Opisane zmiany skutkują zmodyfikowanym członem *prior term*:

$$F_{prior}(\vec{x}) = \sum_{(A,B)\in R} f_{AB}(x_A, x_B)|A|, \tag{3.14}$$

gdzie:

- A i B są obiektami (zbiorami węzłów należących do tej samej grupy o przypisanej wspólnej etykiecie),
- R określa zbiór skierowanych ograniczeń.
- Moc zbioru $|A|$ (liczba segmentów należących do obiektu) została wprowadzona w celu uniknięcia dysproporcji energii pomiędzy metahipotezami o różnej liczbie części.

Na podstawie równań (3.12) i (3.14) możemy zdefiniować $F(\vec{x})$ następująco:

$$F(\vec{x}) = \sum_{s \in V} f_s(x_s) + \sum_{(A,B) \in R} f_{AB}(x_A, x_B)|A|, \tag{3.15}$$

z powodów praktycznych zdecydowano, że w implementacji zostanie wykorzystana maksymalizacja zamiast minimalizacji funkcji energii. Funkcja $f(x_s)$ użytą w *data term* jest zdefiniowana następująco:

$$f_s(x_s) = r(x_s)\frac{|A|}{|M|}, \tag{3.16}$$

gdzie: $r(x_s)$ jest funkcją wagi, określoną dla każdej znanej etykiety, zaś $|A|, |M|$ są odpowiednio liczbą segmentów metahipotezy dotyczącej x_s oraz liczbą części modelu obiektu. Co do funkcji użytej w *prior term*, jej wartości dla każdej relacji są dostarczane bezpośrednio do systemu. Mogą one być dobrane przez człowieka na podstawie posiadanej wiedzy o świecie oraz przykładowych scen (jak ma to miejsce w eksperymentach systemu przedstawionych w tym rozdziale). Parametry takich relacji semantycznych mogłyby też być nauczone na podstawie opisanego, dużego zbioru treningowego, czy nawet w trakcie pracy systemu robotycznego w rzeczywistym środowisku.

Poza wyborem odpowiedniej funkcji energii, ważnym elementem systemu jest algorytm optymalizacji, który będzie wykorzystany do znalezienia rozwiązania równania (3.9). Podstawowe techniki optymalizacyjne mogą zostać podzielone na dwie kategorie: lokalne i globalne. Metody lokalne są z reguły szybsze, ale mają tendencje do wpadania w minima lokalne. W przypadku dyskretnego losowego pola Markowa metody lokalne są mało skuteczne [8], a minima lokalne znacznie pogarszają jakość znalezionych rozwiązań. W opisywanym systemie dążymy do uniknięcia algorytmów lokalnych, ale także wykładniczej zależności złożoności obliczeniowej od liczby segmentów. Taka metoda byłaby niepraktyczna dla rzeczywistych scen o wielu segmentach (wykładnicza zależność ma miejsce w przypadku pełnego przeszukiwania przestrzeni rozwiązań). W tym celu przygotowano prosty ewolucyjny algorytm globalno-lokalny,

który, jak pokazują testy, okazał się wystarczający do szybkiego znalezienia najlepszego przyporządkowania etykiet na testowanych scenach.

W świetle rozważanego problemu ważną zaletą metod ewolucyjnych jest wykorzystanie czynników losowych, które przy odpowiedniej liczbie kroków optymalizacji zapobiegają wpadaniu w minima lokalne. W użytym algorytmie każdy osobnik koduje w swoich genach klasy segmentów klastra sceny (zbioru segmentów nie połączonego relacjami z innymi zbiorami). Oprócz zastosowania tradycyjnego schematu ewolucji z krzyżowaniem i mutacją, każdy osobnik dokonuje optymalizacji „lokalnej". Optymalizacja ta polega na próbie zmiany każdego ze swoich genów (testując wszystkie alternatywne jego wartości), przechodząc na nowe wartości w wypadku, gdy zostaje znalezione lepsze rozwiązanie niż dotychczasowe. Procedura optymalizacji lokalnej dla danego osobnika jest powtarzana cyklicznie aż do osiągnięcia stabilności. Wprowadzenie etapu optymalizacji lokalnej pozwala na znaczne przyspieszenie procesu znalezienia najlepszego rozwiązania.

3.6 Eksperymenty

3.6.1 Proste wnioskowanie: schody czy parapet?

Wybór właściwych relacji semantycznych oraz wag połączeń między obiektami jest ważny. W doświadczeniach przedstawionych w niniejszej pracy zależności semantyczne zostały dobrane na podstawie wiedzy autora, dodatkowo posiłkując się niewielkim zestawem scen treningowych. Jak pokazano w tym rozdziale, stosowanie wnioskowania semantycznego poprawia wyniki klasyfikacji systemu. Ustawienie zbyt wysokich dodatnich wag relacji semantycznych obniża jednak jakoś klasyfikacji – prowadzi do fałszywych dodatnich rozpoznań, nadmiernie przekładając wiedzę systemu o świecie nad dokonanymi obserwacjami.

Rozważmy sytuację, w której co najmniej dwie klasy obiektów są niemożliwe do rozróżnienia na podstawie cech powierzchni i kształtu, jak ma to miejsce dla schodka (widzianego z góry) oraz długiego parapetu. Jeżeli np. obserwujący scenę robot znajduje się na klatce schodowej, postrzegana względna wysokość obiektów nad ziemią może być różna zależnie od położenia robota, a więc także nie może być rozstrzygającą cechą klasyfikacji.

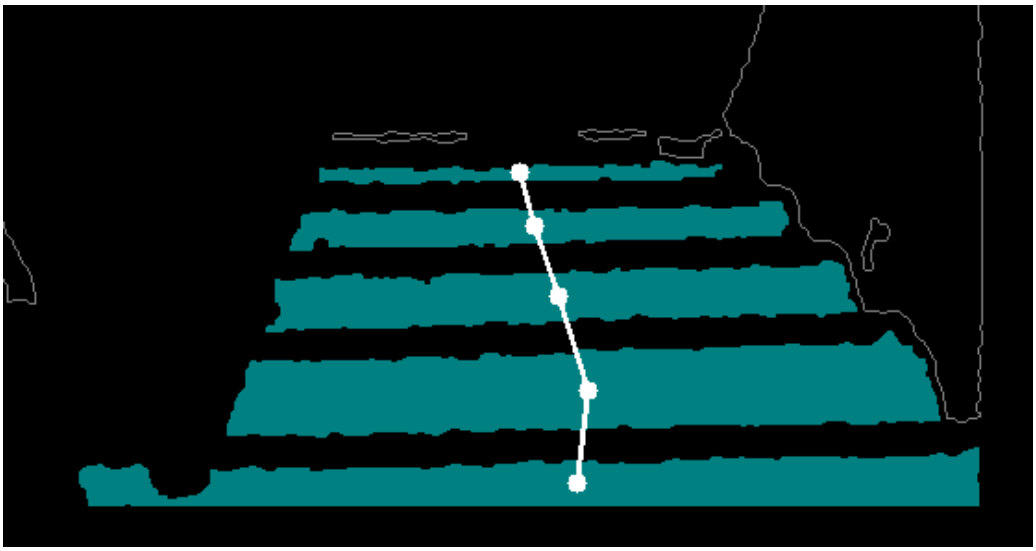
Na potrzeby testu zaaranżowany został scenariusz, w którym konieczne jest wnioskowanie na podstawie tylko dwóch relacji semantycznych: *schodek nad schodkiem* oraz *schodek pod schodkiem*, którym przypisano wagi $w = 2.0$. Waga „a priori" dla parapetu została ustalona na

poziomie 1.0, zaś 0.7 dla schodka, co odzwierciedla przekonanie, że obserwacja odizolowanego występu na ścianie świadczy bardziej o parapecie niż o schodku. Prawdopodobieństwo „a priori" dla klasy nieznanego obiektu zostało ustalone na poziomie 0.5 (tak, aby było ono niższe niż dla rozpoznań na podstawie obserwacji, lecz niezerowe – wartość ta jednak nie jest szczególnie istotna w omawianym przykładzie).

Dla prostej sceny, na której widać tylko 3 schodki, dla każdego z nich program sformułował 3 hipotezy: *schodek*, *parapet* oraz *nieznany*. Energia najlepszej teorii, w której wszystkie 3 widoczne obiekty były schodkami wyniosła 10.1, zaś energia teorii, w której wszystkie były parapetami wyniosła 3.0. Zamiana hipotezy jednego schodka na hipotezę parapetu (w teorii samych schodków) skutkowała 42% spadkiem energii (jak wyjaśniono wcześniej, wyższa energia w tym systemie oznacza lepszą teorię).

Dla sceny, gdzie widoczny był parapet przy braku innych obiektów, zgodnie z oczekiwaniami, najlepsza była teoria parapetu z energią 1.0, podczas gdy teoria schodka miała energię 0.7.

Jak widzimy, wykorzystanie opisanej metodologii pozwala w prosty sposób uwzględnić relacje semantyczne w celu dokonania lepszej klasyfikacji. Różnice w całkowitej energii teorii byłyby oczywiście jeszcze większe gdybyśmy zdefiniowali relacje o wadze ujemnej dla parapetów będących bezpośrednio nad innymi parapetami. Rys. 3.8 pokazuje prostą sytuację, podobną do opisanej (z większą liczbą schodków).



Rysunek 3.8: Wizualizacja wykrytych relacji *pod* – *nad* dla sklasyfikowanych segmentów schodów

3.6.2 Wpływ wnioskowania na klasyfikację wielu obiektów dla realistycznych scen

Aby zbadać jak zaproponowana metoda wnioskowania wpływa na klasyfikację obiektów typowej sceny wnętrza budynku, system został nauczony przez wprowadzenie 10 modeli obiektów użytku codziennego wraz z odpowiadającymi im relacjami semantycznymi. Etykiety obiektów użytych w eksperymencie były następujące *książka miska, krzesło, kubek, biurko, czajnik, blat kuchenny, lampka, piórnik* oraz *regał*.

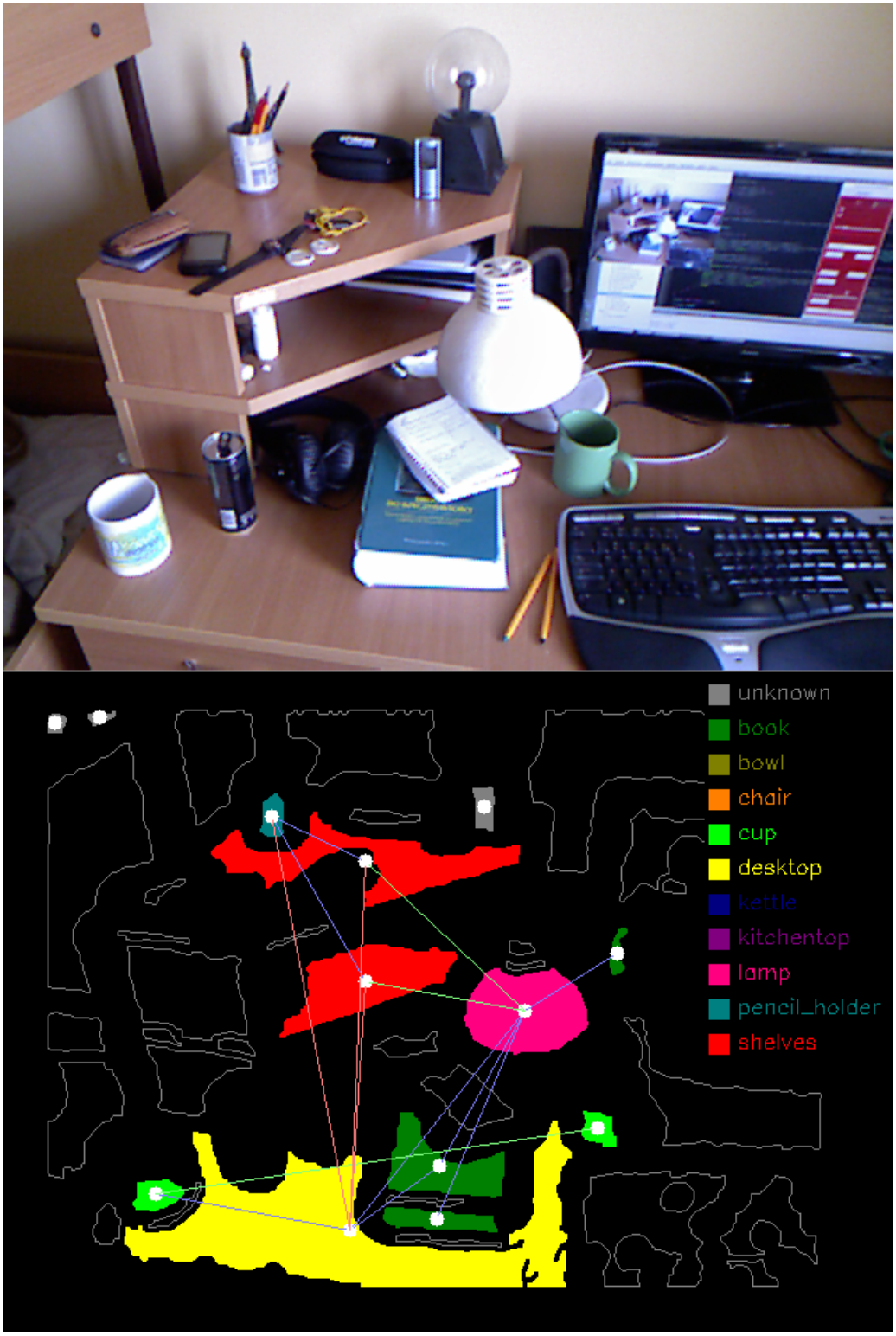
Modele obiektów składają się z widoków zarejestrowanych sensorem Kinect. Dla większości obiektów wykorzystany został tylko pojedynczy widok. Wyjątkami były krzesło i regał, dla których ze względu na większy rozmiar i złożoność zostały wprowadzone po 2 widoki. Jak widzimy, dane uczące są ograniczone do minimum.

Trzy z wymienionych obiektów zostały opisane jako złożone: *książka* (która składa się ze sztywnych, płaskich powierzchni nie połączonych w sposób gładki), *krzesło* oraz *regał*. Biurko oraz blat kuchenny funkcjonują w doświadczeniu jako obiekty proste (płaskie powierzchnie), gdyż ze względu na duży ich rozmiar wiele innych elementów biurka czy szafek kuchennych znajduje się w typowych widokach poza kadrem Kinecta.

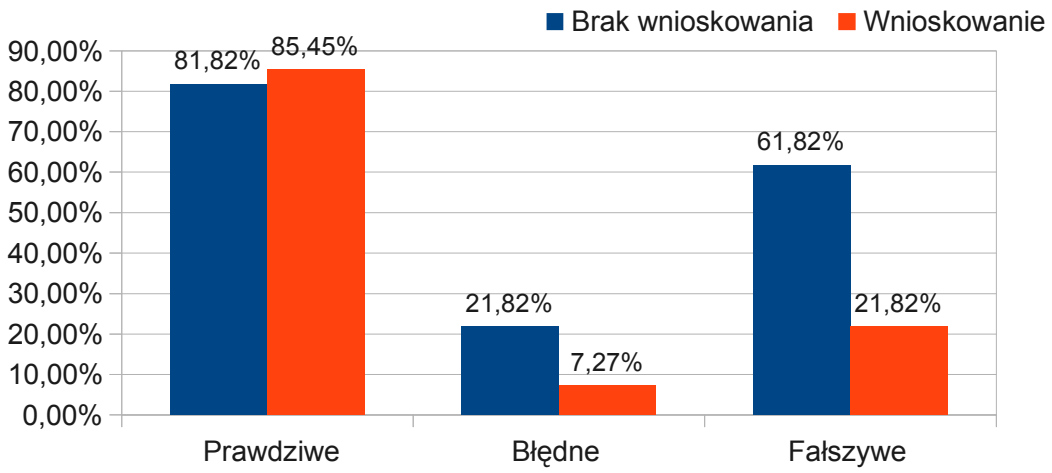
Aby przetestować system wykorzystano 10 scen wnętrza budynku (których nie rozpatrywano w procesie uczenia i określania relacji semantycznych). Sześć z tych scen przedstawia pokój z biurkiem, zaś 4 przedstawiają kuchnię. Sceny zostały zarejestrowane z różnych pozycji kamery, a obiekty były zmieniane i przestawiane w prawdopodobnych konfiguracjach. Sceny testowe zawierały 55 widoków instancji znanych obiektów oraz wielokrotnie więcej nieznanych. Przykładowa scena środowiska pokoju pracy wraz z wynikami przetwarzania została przedstawiona na Rys. 3.9.

Wskaźniki rozpoznawania (rozumianego jako prawidłowa klasyfikacja co najmniej jednej części), błędnej klasyfikacji segmentów oraz fałszywego rozpoznania zostały porównane dla systemu bez wnioskowania (klasyfikującego segmenty osobno, na podstawie ich cech) oraz systemu z wnioskowaniem. Wartości liczbowe wskaźników wyrażają częstość występowania odpowiadającego im rodzaju klasyfikacji w stosunku do liczby znanych obiektów na poszczególnych scenach (a zatem ich wartości mogą przekraczać 100%). Uczenie systemu uproszczonego (bez wnioskowania i łączenia części) przeprowadzono w taki sposób, że wszystkie obiekty

są traktowane jako proste. Poszczególne części obiektów na widokach modelowych (dla obiektów złożonych) są uwzględniane jako oddzielne widoki. Wyniki przedstawiono na Rys. 3.10.



Rysunek 3.9: Klasyfikacja chaotycznej sceny okolic biurka – 7 obiektów rozpoznanych prawidłowo i jedno fałszywe wykrycie. Relacje *pod, nad* oraz *obok* zostały zwizualizowane odpowiednio kolorem różowym, niebieskim i zielonym



Rysunek 3.10: Porównanie wyników klasyfikacji bez wnioskowania z pełnym systemem (z wnioskowaniem). Pokazano proporcjonalnie do liczby znanych obiektów na scenie „prawdziwe" rozpoznania, „błędne" (przyporządkowanie znanemu obiektowi błędnej etykiety innego znanego obiektu) oraz „fałszywe" (przyporządkowanie błędnej etykiety nieznanemu obiektowi)

Jak widzimy, przedstawiona metoda wnioskowania pozwoliła na obniżenie wskaźników błędnej klasyfikacji i fałszywych wykryć około 3-krotnie oraz lekko podniosła wskaźnik rozpoznania. Testy algorytmu zostały przeprowadzone z użyciem procesora wielordzeniowego, dla którego czasy przetwarzania dla złożonych scen sięgały około 30 sekund.

3.7 Wnioski

W tym rozdziale zaprezentowano autorską metodę trójwymiarowego rozpoznawania obiektów opartą na segmentacji, której głównym oryginalnym elementem jest wnioskowanie z wykorzystaniem kontekstu segmentów (opublikowaną w [41]). Wykorzystano zarówno relacje metryczne między potencjalnymi częściami obiektów złożonych, jak i relacje semantyczne miedzy różnymi obiektami. Wykonane doświadczenia pokazują użyteczność zaproponowanej metody wnioskowania.

Utrudnieniem stosowania tej techniki jest konieczność jakościowego oraz ilościowego „ręcznego" definiowania relacji semantycznych. W przypadku dużej liczby obiektów zainteresowania, zadanie to mogłoby zostać ułatwione wprowadzając kategorie obiektów, dla których relacje mogą być opisywane zbiorczo. Alternatywą jest uczenie na dużej, opisanej bazie treningowej scen lub (w systemie robotycznym) wprowadzenie mechanizmu pozwalającego na automatyczne uczenie się typowych relacji w trakcie działania. Ostatecznie jednak jakaś forma dostarcze-

nia lub zdobycia wiedzy o relacjach semantycznych jest nieunikniona, aby możliwe było ich wykorzystanie.

Wybrane podejście segmentacyjne jest zarówno ułatwieniem, jak i utrudnieniem. Z jednej strony rozbija trudne zadanie rozpoznawania obiektów na prostsze etapy i pozwala na wnioskowanie w ograniczonej przestrzeni dyskretnej, lecz z drugiej naraża cała metodę na błędy segmentacji. Potrzebne jest znalezienie kompromisu między ryzykiem omyłkowego połączenia części obiektów w całość oraz ryzykiem znaczącej nadsegmentacji, która zmniejszy powtarzalność części (a więc utrudni uczenie) i wydłuży procedurę optymalizacji. Dla stosunkowo prostych obiektów, które nie znajdują się zbyt daleko od kamery, jakość segmentacji jest zazwyczaj wystarczająca.

Znalezienie najlepszej teorii sceny jest wymagającym problemem optymalizacyjnym o wielowymiarowej, dyskretnej przestrzeni poszukiwań (bez ciągłej funkcji celu). Najbardziej wymagającymi obliczeniowo operacjami były etapy wnioskowania nisko- i wysokopoziomowego (całe przetwarzanie sceny trwało do 30 sekund). Przedstawiona metoda, ze względu na mnogość sprawdzanych niezależnie dopasowań metrycznych oraz teorii złożonych sceny jest możliwa do implementacji masowo-równoległej.

4. Ograniczenia percepcyjne, wiedza niepewna i niepełna

4.1 Wprowadzenie

Niewiedza i niepewność są naturalnymi, intuicyjnymi pojęciami wykorzystywanymi bezustannie w ludzkim myśleniu [4, 94]. Umysł ludzki jest świadomy faktu, że większość informacji dotyczącej otaczającego świata jest dla niego w danej chwili niedostępna. Ta „wiedza o braku wiedzy" jest używana w rozumowaniu z myślową eksploracją różnych możliwych scenariuszy i, w razie potrzeby, planów działania, które doprowadzą do zdobycia dodatkowej informacji. Podczas przetwarzania sygnałów wzrokowych ten rodzaj świadomości pozwala nam na szybkie odróżnianie przestrzeni, w których interesujący nas obiekt może się znajdować od przestrzeni, w których go z całą pewnością nie ma.

Kolejnym czynnikiem odgrywającym ważną rolę w procesie ludzkiego rozpoznawania obiektów jest niepewność [25, 46, 72]. Można to zobrazować na trywialnym przykładzie: z pewnością nie uznalibyśmy odosobnionego, dobrze widocznego kija opartego o ścianę za miotłę. Jednak gdyby ten kij wystawał zza dużego wiadra lub innego przysłaniającego obiektu, moglibyśmy go uznać za prawdopodobną miotłę. W razie potrzeby postanowilibyśmy podejść w celu dokładniejszego sprawdzenia, gdyż jesteśmy świadomi swojej niewiedzy wynikającej z przysłonięcia. Jeśli obiekt jest częściowo przysłonięty, obserwowany z oddali, jest bardzo mały lub warunki oświetleniowe są niewystarczające, jesteśmy w jakimś stopniu niepewni co do jego tożsamości. Takie mechanizmy myślowe są zupełnie oczywiste i na co dzień powszechnie przez nas używane zarówno świadomie, jak i nieświadomie. Jednakże jawne wykorzystywanie niepewności i niewiedzy jest rzadkością w widzeniu maszynowym.

System zbudowany tak, jak omawiany w rozdziale 3 (i w artykule [41]), mimo wnioskowania o relacjach metrycznych i semantycznych, ignoruje fakt występowania przysłonięć i innych rodzajów ograniczeń percepcyjnych. Ograniczenia te są kompensowane w sposób niejawny i

ograniczony, poprzez kumulatywne obliczanie i porównywanie energii poszczególnych teorii. W tym rozdziale zostanie omówione rozwinięcie systemu rozpoznawania obiektów przedstawionego w rozdziale 3. Proponowane rozwinięcie, zaprojektowane przez autora pracy i opisane w [42], jawnie przetwarza informacje o swoich ograniczeniach percepcyjnych w celu poprawienia jakości rozpoznawania obiektów.

Przy zdobywaniu danych przez inteligentny system w rzeczywistym otoczeniu, mamy do czynienia z dwoma rodzajami niepewności: niepewnością stochastyczną, będącą wynikiem losowych zakłóceń oraz epistemologiczną, która wynika z braku wiedzy. Klasycznym podejściem w robotyce przy podejmowaniu decyzji w niepewnych warunkach jest stosowanie teorii prawdopodobieństwa. Ograniczeniem prawdopodobieństwa jednak jest niezdolność uchwycenia niepewności epistemologicznej, która zdaje się być kluczową częścią realistycznego postrzegania świata, a także rozumienia złożonej sceny w robotyce usługowej.

Niektóre prace badają wpływ możliwych przysłonięć na system wizyjny 2D w konkretnych zadaniach nawigacji [33, 92, 36] i rozpoznawania ludzi [26]. Autorzy [63] korzystają z faktu występowania przysłonięć w celu odróżniania możliwych od niemożliwych ułożeń znanych obiektów na scenie. Jeden z najbardziej zaawansowanych sposobów wykorzystania przysłonięć został zaprezentowany w [105], gdzie autorzy proponują używania maski pierwszoplanowych obiektów przysłaniających. Maska ta ma wpływ na wyniki klasyfikacji w oparciu o cechy na drugim planie. Inny ciekawy sposób wykorzystania przysłonięć pokazano w [64], gdzie autorzy używają automatycznie nauczonych wzorców przysłonięć. Metoda wykluczania obszarów przysłoniętych na mapie głębi podobna do jednej z metod autorskich przedstawianych w tym rozdziale została opublikowana (praktycznie jednocześnie z publikacją metody autora tej pracy) w ramach artykułu [87].

W celu uwzględnienia informacji o niewiedzy w systemie rozpoznawania opartym o wnioskowanie, zaproponowane przez autora pracy i opisywane w tym rozdziale podejście wykorzystuje teorię Dempstera-Shafera (ang. *Dempster-Shafer theory – DST*) oraz uogólnione losowego pola Markowa opisane w rozdziale 3. Możliwość stosowania DST do prostego zadania rozpoznawania odosobnionych obiektów została poprzednio zasugerowana w [21] i [40]. Aby użyć DST w bardziej ogólnym rozpoznawaniu obiektów należy się zmierzyć z kilkoma ważnymi problemami omawianymi w tym rozdziale. Do tych problemów należą: agregacja przekonań pochodzących z segmentów mogących należeć do różnych obiektów, optymalizacja funkcji

przekonania dla całej sceny składającej się z wielu znanych i nieznanych obiektów oraz agregacja dużej liczby źródeł wiedzy (segmentów) bez doprowadzenia do saturacji przekonania.

4.2 Teoria Dempstera-Shafera

Wnioskowanie w warunkach niepewnych i korzystanie z niedokładnej wiedzy jest często konieczne w rzeczywistych problemach. [92]. Obserwacje, które są głównym źródłem informacji o środowisku robota mobilnego, są obarczone jakimś stopniem niepewności. Część danych może być zaszumiona lub niejednoznaczna. Reprezentacja wiedzy może także być nieprecyzyjna, niespójna lub częściowa. Gdy używane są reguły wnioskowania, często zdarzają się sprzeczności, redundancje lub sytuacje nie objęte tymi regułami. Formalizm odpowiedni do radzenia sobie z takimi sytuacjami ze świata rzeczywistego powinien pozwalać na ilościowe wyrażanie tych wszystkich aspektów problemu oraz na integrację danych z różnych źródeł, które są obarczone różnymi stopniami niepewności.

Jak wspomniano wcześniej, w celu radzenia sobie z sytuacjami niepewnymi, zwykle stosuje się teorię prawdopodobieństwa. Może być to prawdopodobieństwo wyprowadzone doświadczalnie (oparte na częstotliwości pewnych sytuacji), teoretyczne (gdy dysponujemy modelem w przybliżeniu opisującym rzeczywistą sytuację) lub subiektywne (oparte o zdanie ekspertów). Istotnym problemem prawdopodobieństwa jest trudność wyrażania niewiedzy. We wnioskowaniu probabilistycznym prawdopodobieństwo zajścia interesującej nas sytuacji musi być określane nawet w przypadku braku informacji. Innymi słowy, prawidłowe określenie prawdopodobieństw wymaga uwzględnienia całej przestrzeni zdarzeń, co jest niemożliwe w przypadku niepewności epistemologicznej. Ponadto, ponieważ prawdopodobieństwo danego zdarzenia określane jest jedną liczbą, niemożliwe jest (bez wprowadzania dodatkowych parametrów) odróżnienie sprzeczności przesłanek od niewiedzy. Takie odróżnienie może się okazać szczególnie przydatne w dziedzinach wizji aktywnej i robotyki mobilnej, gdyż pozwala np. na podjęcie decyzji o zgromadzeniu dodatkowych informacji w przypadku niewiedzy (eksploracji), weryfikacji dotychczas zgromadzonych w sytuacji sprzeczności lub zgłoszenie niezgodności sytuacji z wiedzą systemu.

Teoria Dempstera-Shafera [12, 16, 83] jest formalizmem wnioskowania w warunkach niepewności, który pasuje do wyżej opisanych wymagań. Teoria ta została opracowana właśnie w celu umożliwienia matematycznego opisu przekonania, niepewności i konfliktu. W porównaniu

do probabilistyki, zamiast przypisywać prawdopodobieństwa do zdarzeń (hipotez), wiedza jest kodowana przez przypisanie mas m do podzbiorów zbioru potęgowego T wszystkich możliwych zdarzeń.

$$m : 2^T \rightarrow [0,1] \tag{4.1}$$

Masy spełniają następujące założenia:

$$\sum_{A \subseteq 2^T} m(A) = 1, \tag{4.2}$$

$$m(\phi) = 0, \tag{4.3}$$

gdzie: A jest dowolnym podzbiorem zbioru potęgowego, ϕ oznacza zbiór pusty.

Masa podzbioru A ($m(A)$) wyraża więc udział przesłanek (wśród wszelkich dostępnych) wspierających bezpośrednio hipotezę A, lecz nie dotyczących konkretnych podzbiorów A (zatem masy podzbiorów zbioru A nie wchodzą w skład $m(A)$). Przykładową sytuacją, która zgodnie z tą właściwością może mieć miejsce, jest przypisanie niezerowej masy hipotezie $A \cup B$, podczas gdy $m(A) = m(B) = 0$. Możliwa jest też sytuacja odwrotna, gdy masy hipotez A i B są niezerowe, podczas gdy $m(A \cup B) = 0$.

Widzimy więc, że DST dostarcza prostego sposobu wyrażania niewiedzy poprzez masę przypisaną bezpośrednio do przestrzeni zdarzeń T, gdyż masa ta wskazuje na „jakiekolwiek" (a więc nieznane) zdarzenie.

Funkcja $bel(A) : 2^T \rightarrow [0,1]$ określa całkowitą miarę przekonania zbioru (hipotezy) A, uwzględniając wszystkie jego podzbiory B (w tym także samo A):

$$bel(A) = \sum_{B \subseteq A, B \neq \phi} m(B) \tag{4.4}$$

Funkcja $pl(A) : 2^T \rightarrow [0,1]$ określa miarę możliwości A:

$$pl(A) = \sum_{B \cap A \neq \phi} m(B), \tag{4.5}$$

gdzie A i B są dowolnymi podzbiorami zbioru potęgowego przestrzeni zdarzeń (zatem możliwością A jest suma mas wszystkich zdarzeń niesprzecznych z A).

Dopełnienia przekonania i możliwości zwane są odpowiednio *wątpliwością* (ang. *doubt*) i *zaprzeczeniem* lub *niewiarą* (ang. *disbelief*). Różnica $pl(A) - bel(A)$ określa niepewność.

Proces agregacji danych według DST polega na następujących krokach:

- Określenie poziomów przekonania poszczególnych hipotez na podstawie obserwacji (przesłanek), które stanowią źródła wiedzy – np. obserwacja klamki drzwi popiera hipotezę stanowiącą, że robot obserwuje drzwi i przeczy hipotezie obserwacji lodówki.

- Stosowanie reguły agregacji (reguły Dempstera lub innej) w celu łączenia przekonania i zaprzeczenia. Każde źródło wiedzy może popierać daną hipotezę w stopniu pomiędzy 0 a 1, a także jej zaprzeczać w stopniu z tego samego zakresu (co jest jednoznaczne z przekonaniem o nieprawdziwości hipotezy). Suma przekonania i zaprzeczenia może być mniejsza lub równa jeden (lecz nie większa).

Istnieją różne możliwe metody agregacji parametrów różnych źródeł wiedzy. Najbardziej znaną, która będzie stosowana w omawianym systemie, jest reguła agregacji Dempstera, która dla dwóch źródeł wiedzy (1 i 2) opisana jest następująco:

$$m_{1,2}(a) = \frac{\sum\limits_{b \cap c = a} m_1(b) \cdot m_2(c)}{1 - \sum\limits_{b \cap c = \emptyset} m_1(b) \cdot m_2(c)} \tag{4.6}$$

gdzie a, b, c $\subseteq 2^T$.

Rozważmy prostą sytuację, gdzie źródła wiedzy dotyczą pojedynczej hipotezy h, która jest prawdziwa lub fałszywa. Źródła te mogą dostarczać przesłanek wspierających tę hipotezę ($m(h)$) oraz przeczących tej hipotezie, a więc popierających jej negację h_n ($m(h_n)$). Aby spełnione było równanie (4.2) musimy uwzględnić masę przypisaną bezpośrednio całej przestrzeni zdarzeń (którą dla tak skonstruowanego źródła oznaczymy h_u). Masę tą ($m(h_u)$) będziemy nazywać niepewnością źródła. Na podstawie równania (4.6) dla 2 źródeł wiedzy otrzymujemy następujące praktyczne wzory:

$$\begin{aligned}
m_{1,2}(h) &= \frac{m_1(h)\cdot m_2(h) + m_1(h)\cdot m_2(h_u) + m_1(h_u)\cdot m_2(h)}{1 - m_1(h)\cdot m_2(h_n) - m_1(h_n)\cdot m_2(h)} \\
m_{1,2}(h_n) &= \frac{m_1(h_n)\cdot m_2(h_n) + m_1(h_n)\cdot m_2(h_u) + m_1(h_u)\cdot m_2(h_n)}{1 - m_1(h)\cdot m_2(h_n) - m_1(h_n)\cdot m_2(h)} \\
m_{1,2}(h_u) &= 1 - m_{1,2}(h) - m_{1,2}(h_n)
\end{aligned} \tag{4.7}$$

4.3 Wnioskowanie w warunkach niepewności

Jak wspomniano wcześniej, system przedstawiany w tym rozdziale (opisany także w artykule [42]) bazuje na schemacie przetwarzania omówionym w rozdziale 3, zilustrowanym na Rys. 3.1. Pierwsze cztery etapy przetwarzania (wydobycie cech, segmentacja, formułowanie hipotez i wnioskowanie niskopoziomowe) zostały rozszerzone o algorytmy mające na celu zdobycie informacji o niepewności i przysłonięciach. Piąty etap (wnioskowanie wysokopoziomowe) został zaprojektowany od nowa w taki sposób, by za pomocą DST uwzględniać te informacje.

Pierwszy etap przetwarzania polega na wyznaczeniu trzech dość prostych, lecz niosących ważną informację i wydajnych obliczeniowo trójwymiarowych cech lokalnych powierzchni (nachylenie, wypukłość, anizotropia wypukłości). Następnie cechy te są wprowadzane do 3-warstwowej mapy 2D, na której odbywa się wygładzanie za pomocą filtracji *mean-shift* zaimplementowanej w bibliotece OpenCV oraz segmentacja przy użyciu algorytmu *Canny'ego* i operacji morfologicznych. Segmenty. których powierzchnia nie przekracza wymaganego progu nie są analizowane. Pozostałe podlegają dalszemu przetwarzaniu.

Kolejny etap polega na sformułowaniu hipotez prostych – dotyczących tożsamości segmentów z częściami znanych obiektów. Odbywa się to przez porównanie histogramów cech segmentów i histogramów cech części obiektów wzorcowych za pomocą współczynnika korelacji Pearsona (3.5). Jeżeli współczynnik korelacji przekracza wymaganą wartość ($corr_{min}$), hipoteza stanowiąca, że dany segment jest częścią obiektu zostaje sformułowana.

Podobnie jak w rozdziale 3, przeprowadzone jest wnioskowanie niskopoziomowe, które polega na iteracyjnym dopasowaniu metodą Monte Carlo pełnego modelu obiektu do segmentów sceny dla każdej sformułowanej hipotezy. Dopasowanie to dąży do maksymalizacji jakość dopasowania (3.6). Wynikiem dopasowania jest zbiór metahipotez, czyli hipotez pełnych obiektów na scenie. Metahipotezy mogą dotyczyć zarówno jednego, jak i kilku segmentów (jeśli zostało znalezione dla nich dopasowanie do wspólnego modelu). Identyczne metahipotezy są łączone w celu uniknięcia powtórzeń. Metahipotezy te są podstawą do obliczania parametrów, które posłużą do optymalizacji we wnioskowaniu wysokopoziomowym.

4.3.1 Źródła wiedzy

Rozpatrując scenę jako pojedyncze źródło wiedzy, zgodnie z klasyczną teorią Dempstera-Shafera, musielibyśmy przypisać znormalizowane masy wszystkim możliwym podzbiorom prze-

strzeni zdarzeń. Takie podejście jest mało praktyczne dla przestrzeni o wielu zdarzeniach (każde możliwe przypisanie etykiet elementom sceny stanowi zdarzenie). Wobec tego w proponowanym systemie każda część hipotetycznego obiektu (będącego jedną ze sformułowanych metahipotez) jest traktowana jako źródło wiedzy. Dotyczy to zarówno części przyporządkowanych do konkretnego segmentu sceny, jak i nie przyporządkowanych do żadnego segmentu (czyli wynikających tylko ze sformułowania metahipotezy na podstawie innej części modelowego obiektu złożonego). Każde z tych źródeł wiedzy zawiera tylko dwa zdarzenia elementarne: obiekt występuje lub nie występuje w analizowanym miejscu sceny. Masa musi więc zostać rozdzielona tylko pomiędzy tymi zdarzeniami oraz ich sumą, oznaczającą niewiedzę (obiekt występuje lub nie występuje).

Aby przedstawić algorytm obliczenia parametrów DST dla znalezionych części (powiązanych z widocznym segmentem), dla każdej hipotezy zdefiniujemy następujące parametry pomocnicze:

$$q = 2 \cdot \frac{corr(H_S, H_M)}{(1 + corr_{min})}. \tag{4.8}$$

Parametr q (określony wzorem (4.8)) przedstawia korelację histogramu cech (H_S) segmentu z histogramem (H_M) części modelu, podzieloną przez współczynnik skalujący, uwzględniający minimalną korelację wymaganą do uznania hipotezy (tzn. progu korelacji $corr_{min}$). Jeśli wartość q jest ujemna, przypisywane jest $q = 0$.

Następnie, dla każdej hipotezy o indeksie j analizowanego segmentu definiujemy:

$$s_j = q_j \cdot \frac{\sum_{i=0}^{N} q_i - q_j}{\sum_{i=0}^{N} q_i}, \tag{4.9}$$

gdzie: $\sum_{i=0}^{N} q_i$ jest sumą wartości q wszystkich kandydujących hipotez dla danego segmentu (w tym hipotezy o indeksie j). Wartość s odpowiada za zgromadzoną możliwość alternatywnych hipotez segmentu sceny. Uwzględnienie współczynnika q_j w tym parametrze jest potrzebne do zapewnienia $q_j > s_j$, a więc (jak będzie uzasadnione w równaniach (4.12)) dodatniego wpływu hipotez części obiektów odpowiadających znalezionym segmentom na wynik agregacji (tzn. $m(h) > m(h_n)$).

Kolejny parametr, u_{area} oznacza niepewność wynikającą z różnicy pola powierzchni segmentu (A_S) i powierzchni hipotetycznej części obiektu (A_M). Tego typu różnice są dopuszczalne do pewnego stopnia ze względu na ograniczoną precyzję algorytmu segmentacji, danych z

sensora i częściowe przysłonięcia. Parametr ten definiuje wzór:

$$u_{area} = 1 - \frac{|A_M - A_S|}{A_M}. \tag{4.10}$$

Kolejnym pomocniczym parametrem jest u_{dist}:

$$u_{dist} = 1 - \frac{d}{d_{max}}, \tag{4.11}$$

gdzie: d jest średnią odległością sensora do segmentu, natomiast d_{max} jest maksymalną odległością sensora pozwalającą na stosunkowo dokładny pomiar. Wartość u_{dist} oznacza niepewność wynikającą z malejącej dokładności i rozdzielczości wraz ze wzrostem odległości sensora do obserwowanych obiektów. Należy zauważyć, że ten model liniowego spadku precyzji jest jedynie praktycznym uproszczeniem i nie odzwierciedla dokładnie optyki sensora Kinect. Wartości parametrów u_{area} i u_{dist} są progowane tak, by ich wartości pozostały w zakresie $[0,1]$.

Na podstawie tych definicji możemy wprowadzić główne parametry DST hipotezy h – hipotezy obecności danej części obiektu otrzymane na podstawie widocznego segmentu sceny:

$$\begin{aligned} m(h_u) &= 1 - u_{area} \cdot u_{dist}, \\ m(h) &= q^{\frac{1-m(h_u)}{q+s}}, \\ m(h_n) &= s^{\frac{1-m(h_u)}{q+s}} = 1 - m(h) - m(h_u). \end{aligned} \tag{4.12}$$

Jak widzimy, niepewność hipotezy składa się ze składowych wyprowadzonych na podstawie widocznego pola powierzchni i odległości. Wartości $m(h)$ i $m(h_n)$ są proporcjonalne do parametrów q i s. Przeskalowano je w taki sposób, by ich suma stanowiła dopełnienie do wartości jeden dla niepewności $m(h_u)$. Masa (poparcie) obliczana jest na podstawie dostępnych przesłanek – podobieństwie cech segmentu do modelu (parametr q). Zaprzeczenie jest obliczone na podstawie skumulowanych przesłanek popierających alternatywne hipotezy (parametr s).

Zauważmy, że tak sformułowany model nie uwzględnia bezpośrednio segmentów jako źródła wiedzy, lecz rozbija je na wiele źródeł wiedzy odpowiadających pojedynczym hipotezom sformułowanym dla tych segmentów. Pozwala to w znacznie bardziej praktyczny sposób agregować tylko interesujące nas źródła wiedzy (gdyż różne segmenty dotyczą różnych zbiorów hipotez).

Części obiektu, które nie zostały znalezione na scenie stanowią także cenne źródła informacji. Ich nieobecność może dostarczyć silnych przesłanek przeciw poszczególnym hipotezom obiektów. Stwierdzenie, że nie widać danej części niekoniecznie jednak dowodzi, że tej części nie ma. Dwoma głównymi czynnikami mogącymi ograniczyć widoczność części są przysłonięcia oraz granice kadru. Jedną z nowatorskich cech opisywanego systemu jest zdolność uwzględnienia tych czynników w sposób ilościowy jako niepewność poprzez użycie podstawowej formy „wyobraźni przestrzennej", która zostanie dalej opisana.

Aby wyznaczyć ilościowo widoczność obiektów, chmura punktów modelu obiektu (modelowego widoku) dotycząca danej hipotezy jest dopasowana do chmury punktów sceny. Odbywa się to z wykorzystaniem transformacji znalezionej podczas etapu wnioskowania niskopoziomowego. Następnie model obiektu jest rzutowany na płaszczyznę obrazową (znając wewnętrzne parametry kamery). Głębia zrzutowanego modelu jest odejmowana od faktycznego obrazu głębi, dzięki czemu możliwe jest zliczenie punktów zrzutowanego obiektu będących istotnie bliżej kamery niż obserwowane punkty sceny. Punkty te odpowiadają fragmentom obiektu, które, gdyby hipoteza byłą prawdziwa, powinny być widoczne, lecz nie są. Jeśli zrzutowane części są faktycznie przysłonięte, nie powinny wpływać negatywnie na hipotezę (system nie dysponuje wiedzą na ich temat). Przykład takiej projekcji został pokazany na Rys. 4.1.

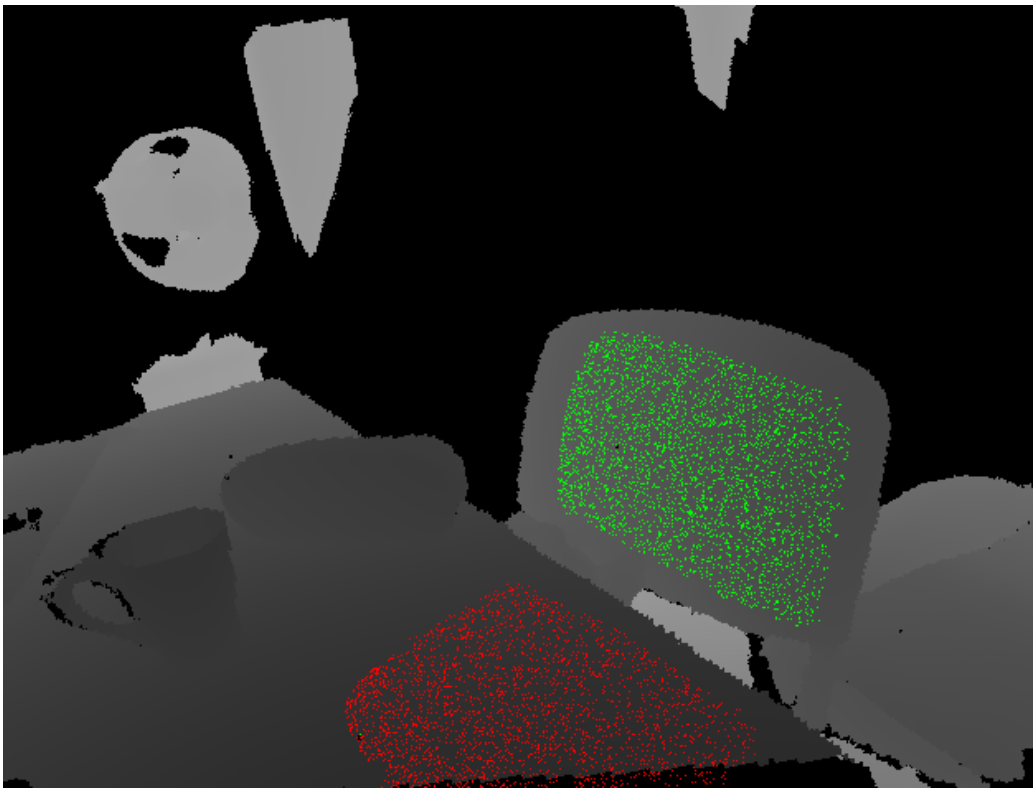
Opiszmy pikselową mapę głębi jako macierz $\mathbf{K} = [k_{ij}]$. Projekcja (rzut) przekształconej chmury punktów modelu jest zbiorem N punktów, składających się ze współrzędnych x i y oraz głębi: $P = \{(x_i, y_i, d_i)\}$. Liczba punktów modelu, które powinny być widoczne może zostać obliczona następująco:

$$N_{VMP} = \sum_{i=0}^{N} T(k_{y_i x_i} - d_i - \varepsilon). \tag{4.13}$$

gdzie: ε jest niewielkim marginesem odległości służącym skompensowaniu niedokładności dopasowania. T jest operacją progowania, która przypisuje 1 dodatnim, zaś 0 niedodatnim argumentom. Gdy niektóre punkty modelu znajdą się poza granicami obrazu, zostają naturalnie wykluczone z tej sumy. Hipotetyczna widoczność danej części, która nie została znaleziona jest obliczana jako:

$$v = \frac{N_{VMP}}{N_{TMP}}, \tag{4.14}$$

gdzie: N_{TMP} jest całkowitą liczbą punktów części modelu.



Rysunek 4.1: Obszary hipotetycznego obiektu uznane za widoczne (zielone) oraz przysłonięte (czerwone), otrzymane w opisanym procesie wnioskowania przestrzennego na mapie głębi

Proponowanym sposobem obliczenia parametrów DST dla hipotezy h – hipotezy obecności części obiektu – na podstawie *nie znalezienia* danej części są następujące wzory:

$$m(h) = 0,$$

$$m(h_u) = 2(1-v), \tag{4.15}$$

$$m(h_n) = 1 - m(h_u),$$

gdzie wartości $m(h_u)$ są progowane tak, by zawierały się w przedziale $[0,1]$. Zatem wartość $m(h_n)$ jest wyprowadzona z przesłanek wskazujących, że przestrzeń, w której dana część jest oczekiwana, jest w istocie pusta i nie przysłonięta. Potwierdzenie hipotezy jest dla takich przypadków zerowe.

Jak można zauważyć, dla części o widoczności poniżej 0.5 otrzymujemy $m(h_u) = 1$ (całkowitą niepewność, czyli brak wpływu na dalsze obliczenia). Takie zachowanie wprowadzono aby skompensować niedokładność wskazań Kinecta w pobliżu krawędzi częściowo przysłoniętych obszarów (oraz idącą za tym niedokładność segmentacji), a także kątową niedokładność

dopasowania modelu. Jeśli jednak wedle hipotezy powierzchnia danej części powinna być dobrze widoczna ($v = 1$) w danym obszarze, lecz obszar ten jest pusty, stanowi to silną przesłankę świadczącą przeciw hipotezie (skutkującą $m(h_n) = 1$), co oznacza całkowite zaprzeczenie (skutkujące odrzuceniem hipotezy).

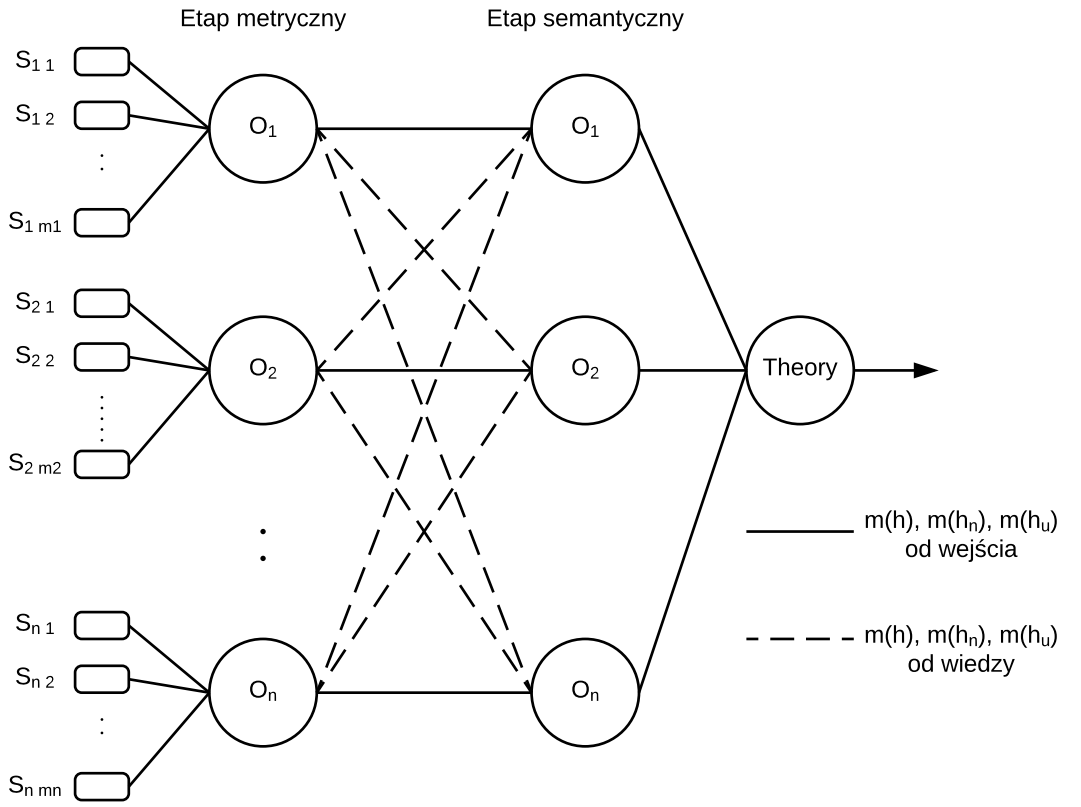
4.3.2 Agregacja

Obserwacje poszczególnych części obiektów nie muszą być jednakowo istotne, gdyż ich cechy charakteryzują się różnym stopniem unikalności i wykrywalności przez sensor Kinect. Dla przykładu, płaskie powierzchnie są mało unikalne i nie pozwalają, bez dodatkowych przesłanek, na rozstrzygnięcie klasy obiektu. Elipsoida natomiast jest powierzchnią znacznie bardziej charakterystyczną, mogącą dostarczać silnych przesłanek na korzyść konkretnego obiektu, takiego jak lampka. Aby uchwycić te zależności w wiedzy systemu wnioskującego, wprowadzono „ważoną" modyfikację zwykłej reguły agregacji DST. Modyfikacja ta polega na stosowaniu wag w w parametrach DST, które stopniują wpływ wykrycia lub stwierdzenia braku segmentu pasującego do danej części modelu na przekonanie o występowaniu obiektu. Wagi wpływają na masy w sposób opisany równaniami (4.16). Wagi mogą być uczone automatycznie, optymalizując rozpoznania na dużym zbiorze treningowym lub, jak w przypadku eksperymentalnego systemu tu opisanego, dobrane na podstawie wiedzy ludzkiej i testów na niewielkim zbiorze treningowym. Poniżej przedstawiono wzory ważonej modyfikacji parametrów DST:

$$
\begin{aligned}
m_w(h) &= \frac{w \cdot m(h)}{w \cdot m(h) + w \cdot m(h_n) + m(h_u)}, \\
m_w(h_n) &= \frac{w \cdot m(h_n)}{w \cdot m(h) + w \cdot m(h_n) + m(h_u)}, \\
m_w(h_u) &= 1 - m_w(h) - m_w(h_n).
\end{aligned}
\tag{4.16}
$$

Wprowadzone wagi mogą się różnić dla tych samych części w przypadku ich wykrycia (wagi „pozytywne") i w przypadku nie wykrycia (wagi „negatywne"). Rozróżnienie to odzwierciedla różne znaczenie obecności i nieobecności w świecie rzeczywistym, co można dobrze wyjaśnić na przykładzie drzwi: wykrycie pionowego panelu jest słabszym dowodem na obecność drzwi niż wykrycie klamki, gdyż taki panel może być np. częścią szafy, regału, itp. Nie wykrycie panelu w sytuacji, gdy powinien być widoczny jest silniejszym dowodem na nieobecność drzwi niż stwierdzenie braku klamki, gdyż mogą istnieć drzwi bez klamki, ale nie bez panelu.

Wnioskowanie wysokopoziomowe zaprezentowane w tym rozdziale korzysta z 3-stopniowego algorytmu agregacji Depmstara-Shafera, który zilustrowano na Rys. 4.2. Prostokąty przedstawiają parametry DST obliczone dla części poszczególnych obiektów teorii sceny zarówno dla obecnych, jak i nieobecnych części.



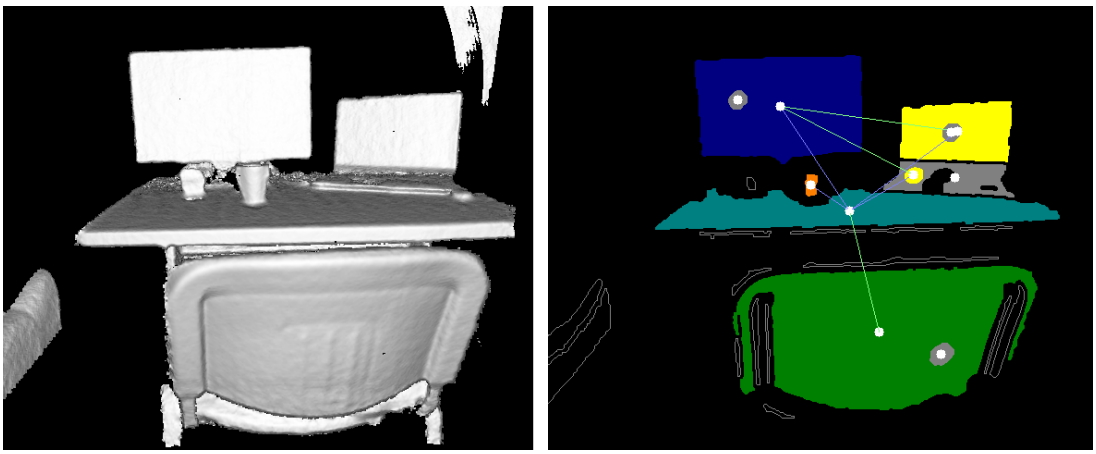
Rysunek 4.2: Schemat obliczenia jakości teorii sceny stosując DST. Dane przepływają od lewej do prawej. S_{nm} oznaczono części (zarówno wykryte, jak i nie wykryte) hipotetycznych obiektów teorii; O_n oznaczono pełne obiekty uwzględnione w teorii; jako *Theory* oznaczono całą teorię sceny

Parametry te są agregowane w pierwszym etapie metody agregacji (metrycznym), dla każdego obiektu stosując opisaną (ważoną) modyfikację agregacji Dempsterowskiej. Wynikowe parametry$(m(h), m(h_n), m(h_u))$ dla każdej hipotezy obiektu są przekazywane do następnego etapu (semantycznego) i agregowane z parametrami a-priori wynikającymi z wykrytych relacji semantycznych z innymi obiektami w rozpatrywanej teorii. Parametry te (podobnie jak człon *prior* opisywany w rozdziale 3) są stałe dla każdej znanej relacji, stanowią część wiedzy systemu i mogą określać pozytywne $(m(h) > 0, m(h_n) = 0)$ lub negatywne $(m(h) = 0, m(h_n) > 0)$ relacje semantyczne. W ten sposób możliwe jest zagwarantowanie całkowitego zaprzeczenia sytuacji np. „krzesło nad monitorem" jeśli wiedza systemu dla takiej relacji będzie stanowić $m(h_n) = 1$. Przypisanie takiej masy zaprzeczenia hipotezy oznacza pełne przekonanie o tym,

że sytuacja nie występuje (bez pozostawienia miejsca na niepewność), co będzie po agregacji skutkowało odrzuceniem teorii sceny.

Parametry wyjściowe obliczone dla poszczególnych obiektów także podlegają agregacji, dając pojedynczy zestaw parametrów DST dla całej teorii sceny. Przy agregacji licznych źródeł, uwidacznia się niekorzystna tendencja reguły Dempstera do zbiegania masy do wartości jeden. Aby temu zapobiec, wprowadzono modyfikację tej reguły agregacji, która polega na stosowaniu współczynnika skalującego dla $m(h)$ oraz $m(h_n)$. Wartość tego współczynnika została ustalona doświadczalnie na poziomie 0.1. Współczynnik ten jest stosowany tylko jeśli wartości $m(h_n)$ agregowanych hipotez wynoszą mniej iż 1, co pozwala zachować możliwość pewnego wykluczania sytuacji całkowicie sprzecznych z wiedzą systemu.

Znalezienie najlepszej teorii sceny stanowi nietrywialny problem optymalizacyjny, który, analogicznie jak w rozdziale 3, rozwiązano za pomocą globalno-lokalnego algorytmu ewolucyjnego. Algorytm ten dąży do ustalenia teorii o najwyższym potwierdzeniu $m(h)$, obliczanym metodą przedstawioną w tym rozdziale (potwierdzenie jest traktowane jako funkcja celu). Algorytm inicjalizuje populację teorii klasyfikujących wszystkie segmenty sceny (lub grupę segmentów bez powiązań zewnętrznych), która podlega ewolucji. Przykładową teorię sceny będącą wynikiem optymalizacji przedstawiono na Rys. 4.3.



Rysunek 4.3: Wyniki rozpoznawania obiektów dla prostej sceny (kolory: *monitor* – niebieski, *laptop* – żółty, *kubek* – pomarańczowy, *stół* – turkusowy, *krzesło* – zielony, *nieznany* – szary). Wykryte relacje semantyczne zilustrowano kolorowymi liniami (kolory: *obok* – zielony, *nad* – niebieski)

4.3.3 Złożoność obliczeniowa

Aby oszacować złożoność obliczeniową najmniej korzystnego scenariusza dla przedstawionych algorytmów, oznaczmy liczbę znanych części obiektów jako N. Przyjmijmy, że liczba ta jest w przybliżeniu proporcjonalną do liczby znanych obiektów.

Koszt obliczenia cech geometrycznych i przeprowadzenia segmentacji jest proporcjonalny do liczby punktów (P) obecnych na scenie. Zakładając stałą rozdzielczość i rozmiar kadru mamy rząd złożoności $O(P) = O(1)$.

Formułowanie hipotez i wnioskowanie niskopoziomowe porównują i dopasowują (dla złożonych obiektów) każdy segment sceny (zawierający maksymalnie M punktów) do każdej z N modelowych części. Te dwa etapy mają wspólny rząd złożoności obliczeniowej najgorszego scenariusza, wynoszący $O(MN)$, gdyż maksymalna liczba iteracji użytego algorytmu Monte Carlo jest stała.

Ostatni etap przetwarzania w przedstawionej implementacji ma najwyższą złożoność obliczeniową najmniej korzystnego przypadku. Pomiary faktycznego przebiegu algorytmu pokazują jednak, że zajmuje on średnio około 1/3 czasu etapu wnioskowania niskopoziomowego. Część globalna algorytmu genetycznego ma złożoność $O(M)$, wynikającą tylko z kosztu obliczenia funkcji przystosowania (zakładając stałą liczbę osobników i dopuszczalnych iteracji). Wyższą złożonością charakteryzuje się część lokalna optymalizacji, dla której w najgorszym wypadku wszystkie segmenty sceny pasowały (pod względem histogramów cech) do wszystkich części znanych obiektów i na dodatek optymalizacja musiałaby przejść przez wszystkie możliwe teorie sceny. Dla takiego scenariusza rząd złożoności byłby taki, jak dla pełnego przeszukiwania przestrzeni rozwiązań, czyli $O(M \cdot M^N) = O(M^{N+1})$. Mimo braku absolutnej gwarancji niższej złożoności, w praktyce algorytm genetyczny jest o wiele rzędów wielkości szybszy niż pełne przeszukiwanie (dokładny czas obliczeń zależy od sceny). Optymalizacja lokalna w testach praktycznych prowadzi do szybszego znalezienia rozwiązania (co było motywacją do jej wprowadzenia). Możliwe by było również odgórne ograniczenie liczby kroków optymalizacji lokalnej osobników. Stosując takie rozwiązanie, maksymalna złożoność wynosiłaby $O(M^2N)$ (choć niekoniecznie zmniejszyłoby to średnią ilość faktycznie wykonywanych obliczeń).

4.4 Eksperymenty

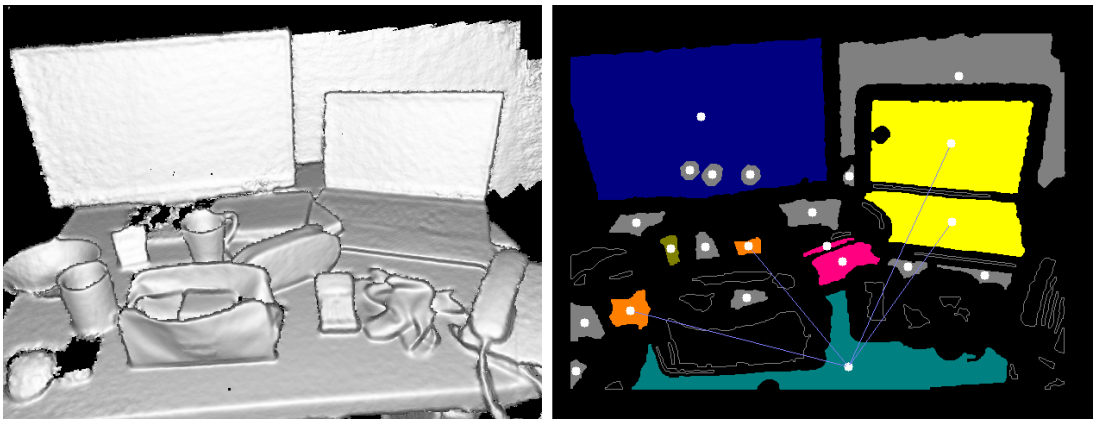
Przedstawiona metoda rozpoznawania obiektów została zaimplementowana i przetestowana na chmurach punktów otrzymanych z sekwencji sensora Kinect, połączonych i przefiltrowanych algorytmem Kinect Fusion [47]. Algorytm ten jest metodą SLAM (jednoczesnej lokalizacji i tworzenia mapy – ang. *simultaneous localization and mapping*) przeznaczoną na procesory graficzne (GPGPU), która znalazła zastosowanie w opisywanym systemie. Dzięki tej technice możliwe jest otrzymywanie chmur punktów o wyższej jakości (wyższym poziomie szczegółowości, mniejszej ilości braków i zakłóceń) w porównaniu do tych pochodzących bezpośrednio z Kinecta. Po kilku-sekundowej obserwacji możliwe jest zarejestrowanie bardziej różnorodnych powierzchni, w tym częściowo odbijających światło (jak monitory) oraz powierzchni czarnych, z którymi Kinect ma duże trudności. Kinect Fusion umożliwia pobieranie i przetwarzanie scen w czasie rzeczywistym (z pewną bezwładnością, która dla rozpoznawania obiektów statycznych nie jest problemem). Wymaga to jednak użycia układu GPGPU, co w dzisiejszych czasach jest możliwe do realizacji na pokładzie niewielkiego robota mobilnego, np. za pomocą układów Nvidia Tegra K1, X1, X2. Ze względu na otwartość kodu i wieloplatformowość, wykorzystano wersję Kinect Fusion zaimplementowaną przez Anatolia Bakszejewa w bibliotece PCL.

Do celów eksperymentu zarejestrowano 3 zestawy scen (głównie w środowisku biurowym):

1. Zbiór treningowy, składający się z 18 scen, który zawiera w pełni widoczne obiekty obserwowane z niewielkiej odległości, a także ich zestawienia podobne do warunków docelowego działania systemu. Widoki te zostały użyte do przygotowania modeli obiektów i weryfikacji parametrów stanowiących wiedzę systemu.

2. Zbiór testowy 14 realistycznych scen środowiska biurowego, zawierających znane i nieznane obiekty w różnych ułożeniach i widziane z różnych odległości. W zbiorze występują przysłonięcia, ale otoczenie jest stosunkowo uporządkowane, z racji czego ten zbiór będziemy nazywać zbiorem „łatwych” scen.

3. Zbiór testowy 12 scen z licznymi, przeważającymi obiektami nieznanymi i wieloma przysłonięciami. Część scen i obiektów jest niezwiązana ze środowiskiem biurowym, w którym osadzone są sceny ze zbioru treningowego. Zbiór ten będziemy nazywać zbiorem „trudnych” scen (przykład takiej sceny jest pokazany na 4.4).



Rysunek 4.4: Przykładowa chmura punktów (przefiltrowana algorytmem Kinect Fusion) ze zbioru scen trudnych wraz z wynikami rozpoznawania (kolory: monitor – niebieski, laptop – żółty, kubek – pomarańczowy, stół – turkusowy, budzik – oliwkowy, piórnik – różowy, nieznany – szary)

Systemowi dostarczono 20 modelowych widoków obiektów i 21 relacji semantycznych, a także pozytywnych i negatywnych wag części dla obiektów złożonych. Parametry te, jak wspomniano, zostały dobrane na podstawie zbioru treningowego. W eksperymencie użyto 9 semantycznych klas obiektów: *krzesło, budzik, lampka, laptop, monitor, myszka, piórnik* oraz *stół*, z których 5 jest traktowanych jako obiekty złożone (składające się z więcej niż 1 części). Wskaźniki skuteczności rozpoznawania obiektów zostały porównane używając 26 scen testowych dla następujących wersji systemu:

1. System bez żadnej wiedzy o swoich ograniczeniach percepcyjnych. W tym systemie poparcie dla poszczególnych hipotez opiera się tylko na korelacjach histogramów cech z modelami części różnych klas obiektów (z użyciem wag). Ponadto, tak jak w systemie opisanym w rozdziale 3, hipotezy nie zostają sformułowane dla segmentów znacznie przekraczających rozmiar danej części modelowego obiektu.

2. System uwzględniający przysłonięcia i ograniczone pole wiedzenia (tzn. korzystający z parametrów DST dla części nieobecnych w sposób opisany w poprzednim podrozdziale). System ten nie uwzględnia niewiedzy dla części znalezionych (a więc niewiedzy wynikającej z ograniczonej dokładności sensora i algorytmów).

3. System uwzględniający niewiedzę z części znalezionych (ograniczonej dokładności), lecz nie uwzględniający przysłonięć i ograniczonego pola widzenia.

4. System pełny, uwzględniający oba rodzaje niepewności (korzystający z parametrów DST zarówno dla części znalezionych, jak i nie znalezionych).

Wszystkie te wersje korzystają z relacji semantycznych na etapie wnioskowania wysokopoziomowego. Jakość rozpoznawania została zmierzona używając 4 wskaźników:

1. Liczba prawidłowych rozpoznań w stosunku do liczby znanych obiektów na scenie.

2. Liczba błędnych rozpoznań w stosunku do liczby znanych obiektów na scenie (zarówno znanych z przypisaną błędną klasą, jak i nieznanych, błędnie oznaczonych jako znane).

3. Całkowity wskaźnik błędów $E1$ zdefiniowany jako:

$$E1 = \frac{FN + FP}{NO}, \tag{4.17}$$

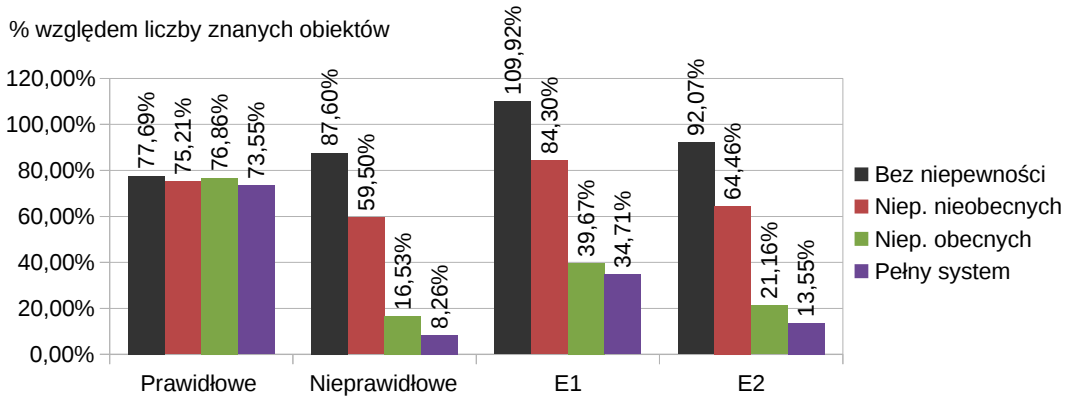
gdzie: FN stanowi liczbę znanych, nie rozpoznanych obiektów widocznych na scenie (fałszywie ujemne przypadki), FP jest liczbą błędnych rozpoznań (fałszywie dodatnie), zaś NO jest liczbą widocznych znanych obiektów.

4. Wskaźnik ważony $E2$, obliczany podobnie jak $E1$, lecz zmodyfikowany przez wprowadzenie wagi dla nie rozpoznanych obiektów:

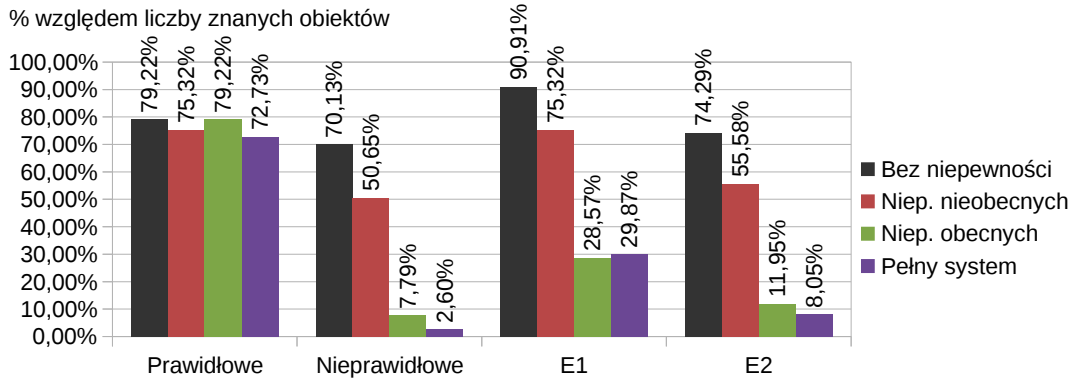
$$E2 = \frac{0.2 \cdot FN + FP}{NO}. \tag{4.18}$$

Ten drugi wskaźnik błędu jest bardziej odpowiedni dla sytuacji, w której koszt fałszywych rozpoznań jest wyższy niż koszt pominięcia znanego obiektu. Ma to miejsce np. w typowych dla robotyki zadaniach lokalizacji wspomaganej semantycznie: jeśli dany znany obiekt zostanie pominięty, robot może nadal lokalizować się prawidłowo, tracąc tylko niektóre przesłanki. W przypadku błędnych rozpoznań robot może łatwo stracić orientację.

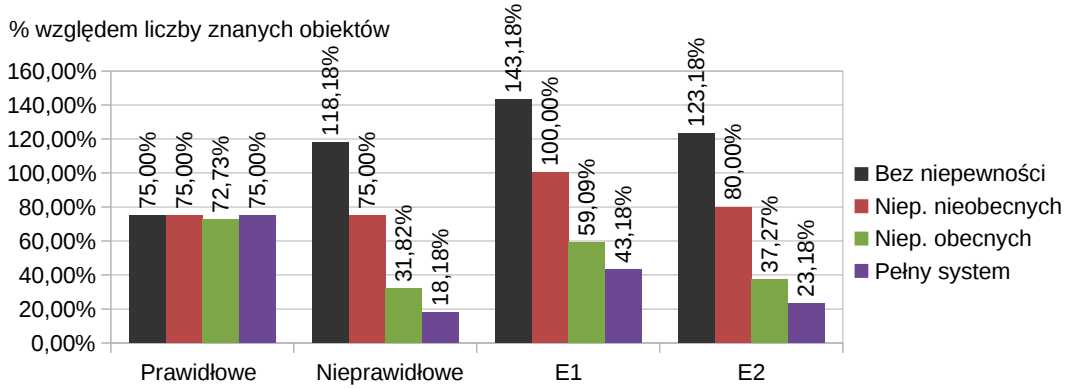
Rys. 4.5 podsumowuje wyniki eksperymentu. Jak widać na wykresach, wykorzystanie informacji o niepewności tylko nieznacznie obniżyło liczbę prawidłowych rozpoznań, lecz znacząco obniżyło liczbę rozpoznań nieprawidłowych. Jest to oczekiwany rezultat, gdyż zwiększanie niepewności w parametrach DST działa na rzecz hipotez zerowych poszczególnych segmentów, co skutkuje większą skłonnością do przydzielenia im klasy „nieznanej”. Wyniki sugerują, że niepewność dla części znalezionych w większym stopniu wpływa na system niż niepewność dla części nie znalezionych (dla łatwego zbioru testowego nawet użycie tego drugiego rodzaju niepewności jest w niewielkim stopniu niekorzystne pod względem wskaźnika $E1$). Jednak połączenie obu rodzajów niepewności daje średnio najlepsze wyniki. Jak można było się spodziewać, wnioskowanie z uwzględnieniem przysłonięć w większym stopniu poprawia wyniki



(a)



(b)



(c)

Rysunek 4.5: Wyniki eksperymentu dla pełnego (a), „łatwego” (b) oraz „trudnego” (c) zbioru testowego. Wykresy porównują wskaźniki prawidłowych, nieprawidłowych rozpoznań (które są względne do liczby znanych obiektów, więc mogą przekraczać 100%) oraz błędów E1 i E2 dla różnych sposobów wykorzystania niepewności

dla scen trudnych niż dla łatwych, gdyż na scenach trudnych przysłonięcia są znacznie częstszym problemem.

Całkowite obniżenie wskaźników błędów $E1$ i $E2$ pełnego systemu w porównaniu z wersją bez uwzględnienia niepewności części nieobecnych wynosiło odpowiednio 12.5% i 35.9%; w porównaniu z wersją bez niepewności części obecnych 58.8% i 79.0%; w porównaniu do systemu nie używającego niepewności 68.4% i 85.3%.

Omówione algorytmy zaimplementowano w języku C++ w wersji szeregowej (CPU), zrównoleglając na kilka rdzeni tylko etap wnioskowania niskopoziomowego. Nie dotyczy to algorytmu Kinect Fusion, który jest oddzielną częścią systemu działającą na karcie graficznej, nie zaimplementowaną przez autora tej pracy. Czas przetwarzania pojedynczej sceny na średniej klasy procesorze 4-rdzeniowym Intel Core i7 zależał od złożoności sceny i wynosił od pół minuty do minuty.

4.5 Wnioski

Uwzględnienie informacji o niepewności i niewiedzy, zgodnie z oczekiwaniami, okazało się mieć korzystny wpływ na skuteczność rozpoznawania obiektów (szczególnie dla scen trudnych). Choć użyty model ograniczeń percepcyjnych systemu jest bardzo prosty, rachunek prawdopodobieństwa nie dostarcza naturalnego sposobu jego uwzględnienia we wnioskowaniu. Pomocna okazuje się teoria Dempstera-Shafera, zawierająca odpowiednie narzędzia matematyczne do przetwarzania wiedzy niepewnej. Jednak niektóre efekty będące implikacją użycia DST były negatywne (np. zbieganie potwierdzenia teorii do 1 przy wielu obiektach na scenie) i wymagały wprowadzenia współczynników kompensujących.

Nasuwającym się rozwinięciem opisanego systemu byłoby dokładniejsze i pełniejsze zamodelowanie źródeł niepewności, np. uwzględniając właściwości fizyczne sensora, a także badając wpływ innych znanych czynników na dokładność algorytmów segmentacji i wyznaczania cech. Innym elementem mogącym podlegać modelowaniu niepewności jest algorytm agregacji danych wejściowych z kamery (w omawianym systemie Kinect Fusion).

Z punktu widzenia wydajności obliczeniowej, ta metoda ma podobne wady i zalety do przedstawionej w rozdziale 3, gdyż ogólna koncepcja przestrzeni poszukiwań i optymalizacji w obu systemach jest taka sama.

5. Szybkie rozpoznawanie obiektów

5.1 Wprowadzenie

Szybkość poruszania się, lokalizacji, planowania i przyswajania informacji o środowisku jest kluczowym aspektem w mobilnej robotyce usługowej. O ile robot autonomiczny wykonujący np. pomiary terenu może przemieszczać się i przetwarzać informacje bardzo wolno pozostając użytecznym, robot usługowy powinien ruszać się i reagować w akceptowalnych dla człowieka przedziałach czasowych. Przyczyną tego są względy praktyczne, takie jak nie blokowanie korytarzy, nie stwarzanie zagrożenia, szybkie odpowiadanie na polecenia człowieka, wydajne wykonywanie zadań, ale także względy psychologiczne. Zbyt wolne reakcje lub ruchy w stosunku do ludzkich (a także zbyt gwałtowne ruchy) mogą wywoływać zniecierpliwienie, stres i utrudniać użytkowanie robota. Z tych powodów, jak zauważono we wstępie, szybkość działania algorytmów jest ważnym aspektem widzenia maszynowego w robotyce.

Nie jest jednak całkiem jasne jak szybkie muszą być roboty i ich algorytmy planowania lub rozumienia sceny, by człowiek uznał je za akceptowalne. Jednym z czynników wartych uwzględnienia jest czas przetwarzania informacji przez samego człowieka, do którego to czasu jesteśmy najbardziej przyzwyczajeni. Na podstawie badań neurobiologicznych i neuropsychologicznych [65, 17, 23] zauważamy, że wzrokowe rozumienie sceny przez człowieka jest stopniowe. Choć jest to uzależnione od rodzaju rozpoznawanego obrazu, czas potrzebny do uchwycenia ogólnego semantycznego sensu sceny na zdjęciu wynosi ok. 100-200 ms. Pełniejsza analiza sceny może jednak trwać znacznie dłużej. Człowiek analizując obraz skupia wzrok w nowych miejscach 3-4 razy na sekundę, wykonując ruchy sakadowe. Jeśli więc zamiarem obserwatora jest dokładne przyjrzenie się wielu obiektom na złożonej scenie, może mu to zająć wiele sekund. W życiu codziennym człowiek nie ma jednak konieczności częstego zatrzymywania się celem pełnej analizy sceny, gdyż w mózgu jest budowana reprezentacja otoczenia w sposób inkrementalny. Gdy już znamy środowisko, nie musimy obserwować wciąż tych samych miejsc, lecz zwracamy szczególną uwagę na ruch (a więc zmiany w środowisku), który jest

bardzo szybko wykrywany. Zawdzięczamy to wyspecjalizowanemu, szybkiemu przetwarzaniu obrazu drogą grzbietową kory wzrokowej mózgu oraz szybkiej aktywności neuronów siatkówki odpowiadających za widzenie peryferyjne. Ponadto, człowiek podczas przyswajania informacji wzrokowej wykorzystuje szereg mechanizmów heurystycznych. Będąc w stanie bardzo szybko uchwycić ogólny sens i zgrubny zarys sceny, wiemy gdzie się spodziewać interesującego nas obiektu. Zaawansowane mechanizmy percepcyjne człowieka skutkują bardzo szybką orientacją przestrzenną, kategoryzacją i przyswajaniem informacji wzrokowej.

Z praktycznego punktu widzenia, nasze wymagania czasowe względem robota mobilnego zdaje się najlepiej ujmować stosowane w środowisku technicznym i przemysłowym pojęcie przetwarzania „w czasie rzeczywistym" [114]. Choć pojęcie to nie jest zgodne z informatycznym znaczeniem systemów czasu rzeczywistego [89] (czyli takich, które gwarantują uzyskanie odpowiedzi w stałych, określonych z góry ramach czasowych), używa się go do opisu systemu dającego odpowiedź w czasie ledwo zauważalnym dla człowieka (a więc w „ułamku sekundy").

Przetwarzanie obrazu 3D jest z natury wymagającym obliczeniowo zadaniem. Jak zauważono we wstępie pracy, szybkość działania najbardziej dokładnych technik wykrywania klas semantycznych obiektów lub określania ich pozycji w 3D nie jest wystarczająca dla potrzeb robotyki usługowej. Można wyróżnić kilka różnych sposobów dążenia do szybszego rozwiązywania zadania, jakim jest rozpoznawanie obiektów w warunkach rzeczywistych:

- przerzedzając chmurę punktów (zmniejszając rozdzielczość i liczbę punktów),
- stosując szybsze algorytmy, w swojej istocie cechujące się niższa złożonością obliczeniową,
- optymalizując algorytmy na poziomie kodu,
- stosując obliczenia masowo-równoległe,
- stosując techniki heurystyczne,
- agregując informację, inkrementalnie tworząc model otoczenia.

Zależnie od skali interesujących nas obiektów sceny i początkowej rozdzielczości, często można do pewnego stopnia zmniejszyć liczbę punktów, np. poprzez zastosowanie filtru wokselowego (jest to przybliżenie niewielkich, sześciennych obszarów sceny pojedynczymi punktami umiejscowionym w centroidach poszczególnych sześcianów). Zbyt duże uproszczenie sceny w sposób oczywisty powoduje jednak utratę niezbędnej informacji.

Stosowanie algorytmów o niższej złożoności obliczeniowej to kolejny dość oczywisty sposób przyspieszenia przetwarzania. Przede wszystkim korzystnie jest unikać metod wnioskowania o złożoności eksponencjalnej względem liczby obiektów na scenie, metod wymagających pełnego przeszukania przestrzeni wielowymiarowych lub wykonania zbyt skomplikowanych obliczeń dla każdego punktu chmury 3D. Jak zostanie jednak uzasadnione, czasochłonność algorytmów czasami jest możliwa do złagodzenia dzięki zastosowaniu dodatkowych metod heurystycznych.

Optymalizacja algorytmów na poziomie kodu polega na stosowaniu bardziej wydajnych implementacji algorytmów składowych, szybszych bibliotek, języków programowania, metod zarządzania pamięcią, obliczeń przybliżonych, itp. Optymalizacje tego typu są opłacalne do pewnego stopnia: np. przepisanie na język C++ jeśli program był początkowo napisany w Matlabie, używanie zoptymalizowanych bibliotek zamiast własnych, prostych implementacji algorytmów. Jednak po rozwiązaniu głównych problemów wydajności, korzyści z dalszej optymalizacji stają się niewielkie i program zaczyna się zbliżać do swojej nieprzekraczalnej granicy wydajności.

Możliwą drogą do zwiększenia wydajności programu jest stosowanie obliczeń masowo-równoległych. Ze względu na ograniczenia fizyczne, nie można liczyć na znaczący dalszy wzrost szybkości procesorów CPU opartych na tranzystorach krzemowych. Jednak dzięki rozpowszechnieniu się procesorów graficznych ogólnego przeznaczenia (GPGPU) możliwa jest stosunkowo łatwa implementacja algorytmów masowo-równoległych, które zależnie od używanych metod i sprzętu, mogą być od kilku do kilkuset razy szybsze od swoich odpowiedników szeregowych. Problemy wizyjne są zwykle podatne na zrównoleglenie, gdyż często czasochłonne operacje polegają na wykonaniu podobnych obliczeń dla wielu pikseli, punktów chmury, klastrów lub hipotez. Niekiedy jednak problem zrównoleglania nie jest trywialny i wymaga dużo pracy badawczo-testowej. Dodatkowym zagadnieniem przy technikach masowo-równoległych jest osiąganie możliwie jak największej wydajności na konkretnym sprzęcie: ze względu na różnice w liczbie jednostek obliczeniowych, ilości pamięci RAM karty graficznej, a także czasu dostępu do tej pamięci, różne wersje algorytmu mogą się okazać bardziej lub mniej wydajne na różnym sprzęcie. Pisanie kodu masowo-równoległego, w szczególności uniwersalnego i łatwo konfigurowalnego wymaga zaawansowanych umiejętności specjalistycznych. W tym rozdziale zostaną przedstawione niektóre prace autora dotyczące metod masowo-równoległych.

W problemach wizyjnych często możliwe jest podejście heurystyczne. Spotykane są metody odrzucania niskim kosztem znaczącej części hipotez rozwiązania, a także metody zawężania uwagi do wybranych fragmentów sceny bez utraty dokładności wyników. Projektowanie systemów heurystycznych wymaga dodatkowych prac badawczych, ale często może skutkować radykalnym zmniejszeniem czasu potrzebnego do osiągnięcia odpowiedzi systemu. Niniejszy rozdział zawiera omówienie prac autora w tym zakresie.

Dzięki inkrementalnemu agregowaniu informacji, możliwe jest tworzenie reprezentacji środowiska w pamięci komputera. Jest to działanie podobne do zachodzącego w umyśle ludzkim. Założenie, ze środowisko nie ulega częstym, całościowym zmianom, pozwala skupiać się na jego nowych (lub nie zauważonych wcześniej) elementach. W robotyce realizuje się to przez budowanie map semantycznych i metryczno-semantycznych [53].

5.2 Masowo-równoległa segmentacja obszarów gładkich

Jak zauważono we wstępie, techniki oparte na segmentacji stanowią jedno z podstawowych podejść do problemu rozpoznawania obiektów. Jednym z najczęściej wykorzystywanych rodzajów segmentacji sceny 3D jest segmentacja obszarów płaskich lub gładkich. Obszary tego typu mają szczególne znaczenie we wnętrzu budynku, a w związku z tym w robotyce usługowej. W środowisku wnętrza budynku występują płaskie podłogi, ściany, sufity oraz duże płaskie powierzchnie mebli. Tego typu segmentacja może być użyta bezpośrednio do znalezienia oddzielonych przestrzennie klastrów punktów, które na prostych scenach stanowią rozpoznawane obiekty. Segmentacja obszarów gładkich może posłużyć również do heurystycznego ograniczania obszaru poszukiwań innych algorytmów rozpoznawania, zmniejszając ich koszt obliczeniowy [88, 37].

Najczęściej spotykane podejście do przeprowadzenia segmentacji 3D obszarów płaskich opiera się na wykorzystaniu algorytmu RANSAC (ang. *random sample consensus*), za pomocą którego dopasowuje się model płaszczyzny do chmury punktów sceny [76, 19, 44, 60]. Po skutecznym dopasowaniu procedura jest powtarzana iteracyjnie z wykluczeniem punktów już uznanych za część płaszczyzn sceny. Algorytm kończy działanie, gdy zostaną odsegmentowane wszystkie znaczące obszary płaskie (o pewnej minimalnej powierzchni lub liczbie punktów). Ta procedura posiada istotne ograniczenia:

- Mimo, że dopasowywanie metodą RANSAC płaszczyzny do chmury odbywa się szybciej niż wzajemne dopasowanie dwóch chmur punktów, nadal jest to metoda czasochłonna, gdyż wymaga sprawdzania licznych hipotez płaszczyzny, za każdym razem obliczając odległość do każdego z punktów sceny.
- Znaleziona płaszczyzna może być mało precyzyjnie dopasowana do sceny, co wymaga użycia dodatkowych metod zwiększenia dokładności segmentacji (takich jak ICP – *ang. iterative closest point*).
- Procedura wymaga tym więcej powtórzeń, im więcej jest dużych płaskich obszarów na scenie, co może skutkować znaczącą zmiennością czasową lub pomijaniem części obszarów.
- Czas dopasowania pojedynczej płaszczyzny jest losowy (co wynika z losowej natury algorytmu).
- Metoda nie będzie działać dla lekko zaokrąglonych lub inaczej zakrzywionych ścian.

W efekcie czasy dopasowania wielu płaszczyzn do pojedynczej sceny tą metodą wynoszą typowo kilka sekund nawet przy wykorzystaniu masowo-równoległej implementacji metody RANSAC (jest to silnie zależne od sceny). Zaproponowane zostało też wykorzystane dekompozycji głównych składowych [101] oraz klasycznej metody rozrostu ziarna [27] w zastosowaniu do scen 2,5D (pojedynczej klatki sensora głębi), segmentując w oparciu o wektory normalne do powierzchni. Autorzy ostatniej zacytowanej publikacji wspominają o możliwości użycia metody rozrostu ziarna dla niezorganizowanej chmury punktów (bez uporządkowania na mapie 2D), choć nie omawiają implikacji czasowych takiego podejścia. Implikacje te byłyby znaczące, gdyż dla niezorganizowanej chmury punktów nie jest możliwe używanie optymalizacji korzystających z trywialnych relacji przestrzennych pikseli na obrazie 2D i w pamięci komputera. W dodatku, występuje konieczność nietrywialnego wyznaczenia sąsiedztwa dla każdego punktu. Możliwość sprawnego działania takiego algorytmu segmentacji na pełnej chmurze 3D (danej w postaci zbioru punktów 3D) jest wysoce pożądana. Taka wydajna metoda segmentacji umożliwia przetwarzanie *on-line* chmur o dokładności zwiększonej z użyciem metod SLAM (takich jak Kinect Fusion czy Kintinuous [97]), a także chmur uzyskanych agregacją danych z kilku sensorów.

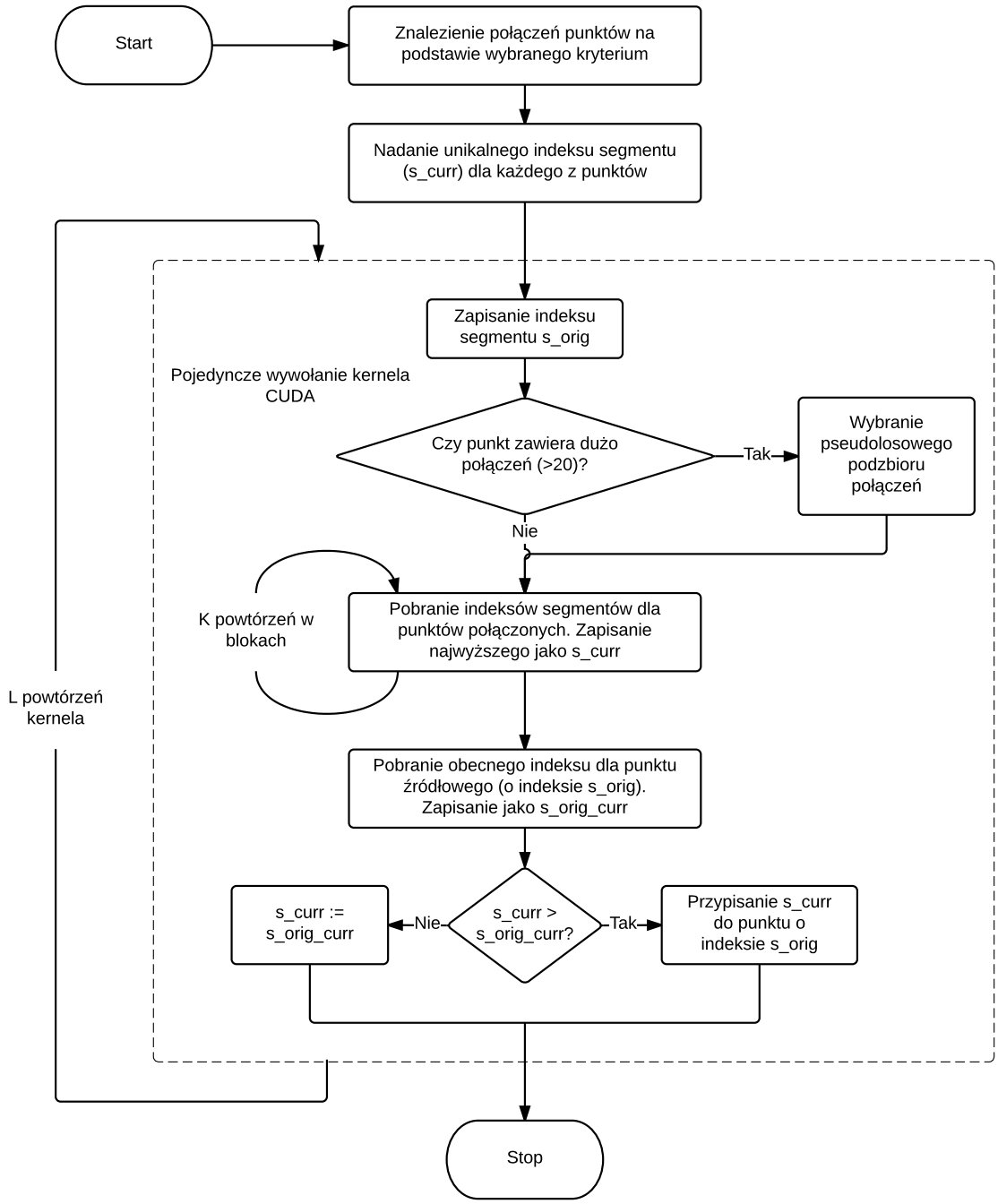
Nasuwającym się rozwiązaniem jest wykorzystanie obliczeń masowo-równoległych do przeprowadzenia segmentacji metodą rozrostu ziarna. Technika ta nie jest obarczona silną losowo-

ścią i zależnością od sceny jak metoda RANSAC. Algorytm rozrostu ziarna jest jednak z natury szeregowy: w klasycznej wersji polega on na rozroście tylko jednego segmentu na raz, zaś próba przyłączenia każdego kolejnego punktu do segmentu wiąże się z koniecznością sprawdzenia, czy nie został już wcześniej dodany do któregoś segmentu. Autor niniejszej rozprawy zaproponował masowo-równoległą, iteracyjną metodę rozwiązania tych trudności [38], w której poszczególne obszary konkurencyjnie przyłączają nie przydzielone punkty i inne obszary na podstawie funkcji ciągłości i wartości swoich indeksów.

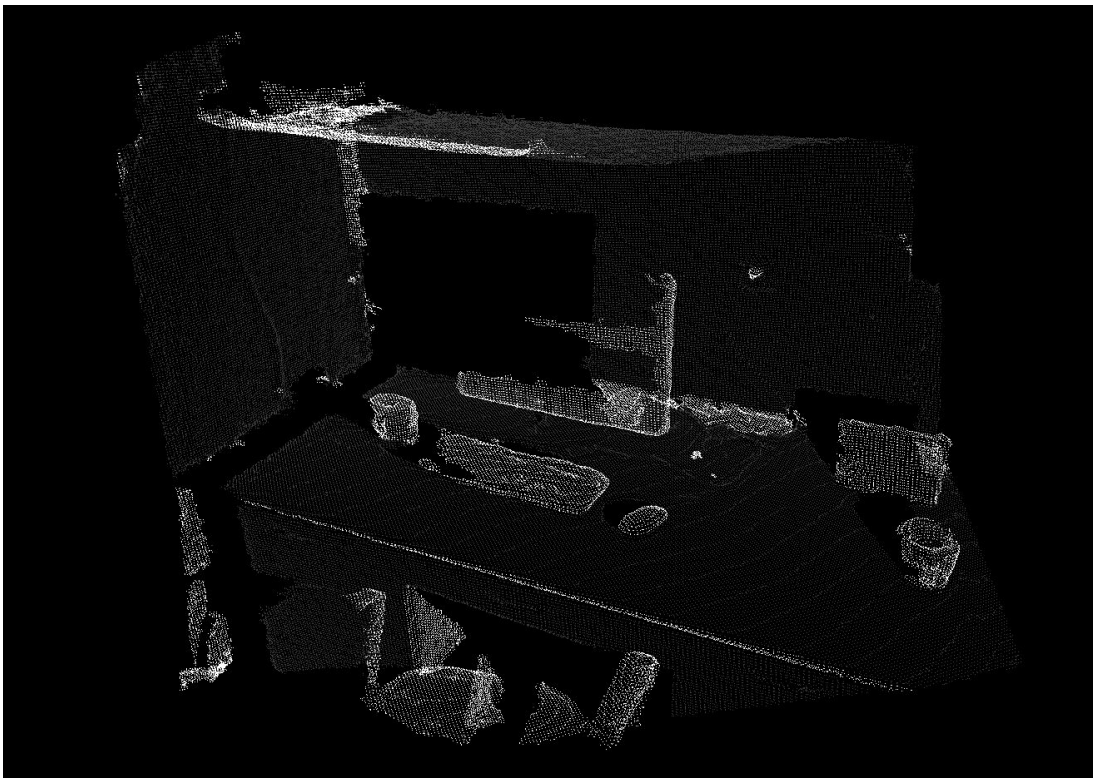
Przed właściwym przeprowadzeniem rozrostu ziarna odbywa się wyznaczenie (wybranego doświadczalnie) 20-milimetrowego sąsiedztwa każdego z punktów za pomocą drzewa ósemkowego (także w implementacji masowo-równoległej z biblioteki PCL). Ten konieczny rozmiar sąsiedztwa jest niemały pod względem zawartej w nim liczby punktów. Dla chmury już przetworzonej filtrem wokselowym o długości boku woksela 5 mm liczba punktów sąsiedztwa wynosi ok. 50 punktów dla powierzchni płaskiej, zaś dla prostych narożników może przekraczać 100 punktów. Algorytm rozrostu ziarna wyjaśnia Rys. 5.1.

Zaproponowana metoda nadaje się do masowo-równoległego przeprowadzenia rozrostu ziarna jednocześnie dla dowolnej liczby ziaren (w zaprezentowanej wersji każdy punkt jest początkowo ziarnem). Możliwe jest także stosowanie niemal dowolnego kryterium ciągłości. W przedstawionej wersji jako kryterium ciągłości między sąsiednimi punktami przyjęto spełnienie przybliżonej równoległości ich wektorów normalnych do powierzchni (w pewnym otoczeniu tych punktów). Do innych zastosowań równie dobrze można by było przyjąć kryterium podobieństwa koloru, wypukłości lub innej cechy.

Wynik segmentacji (a następnie odrzucenia) obszarów gładkich dla testowej sceny przedstawiono na Rys. 5.2. Na testach przeprowadzonych z użyciem karty graficznej GTX Titan Black czas przetwarzania scen składających się z ponad 100000 punktów wynosił zwykle poniżej 50 ms. Wliczono w to czas obliczenia wektorów normalnych i znalezienia 2-centymetrowego sąsiedztwa każdego z punktów za pomocą drzewa ósemkowego (także w implementacji masowo-równoległej z biblioteki PCL). Największy wpływ na czas przetwarzania miała (w sposób oczywisty) liczba punktów oraz ich zagęszczenie, które determinowało średni rozmiar sąsiedztwa, a więc liczbę sprawdzanych połączeń. Czynniki te dają się jednak łatwo ograniczyć odgórnie. Liczba płaskich powierzchni na scenie miała bardzo mały wpływ na czasy przetwarzania.



Rysunek 5.1: Diagram zaproponowanego algorytmu masowo-równoległego rozrostu ziarna. Bloki decyzyjne (warunki) stanowią heurystykę: pierwszy (górny) warunek zmniejsza liczbę redundantnych sprawdzeń dla punktów silnie połączonych z sąsiedztwem. Drugi warunek przyspiesza propagację najwyższych indeksów dzięki dwukierunkowej wymianie informacji z zapamiętanym „ziarnem”



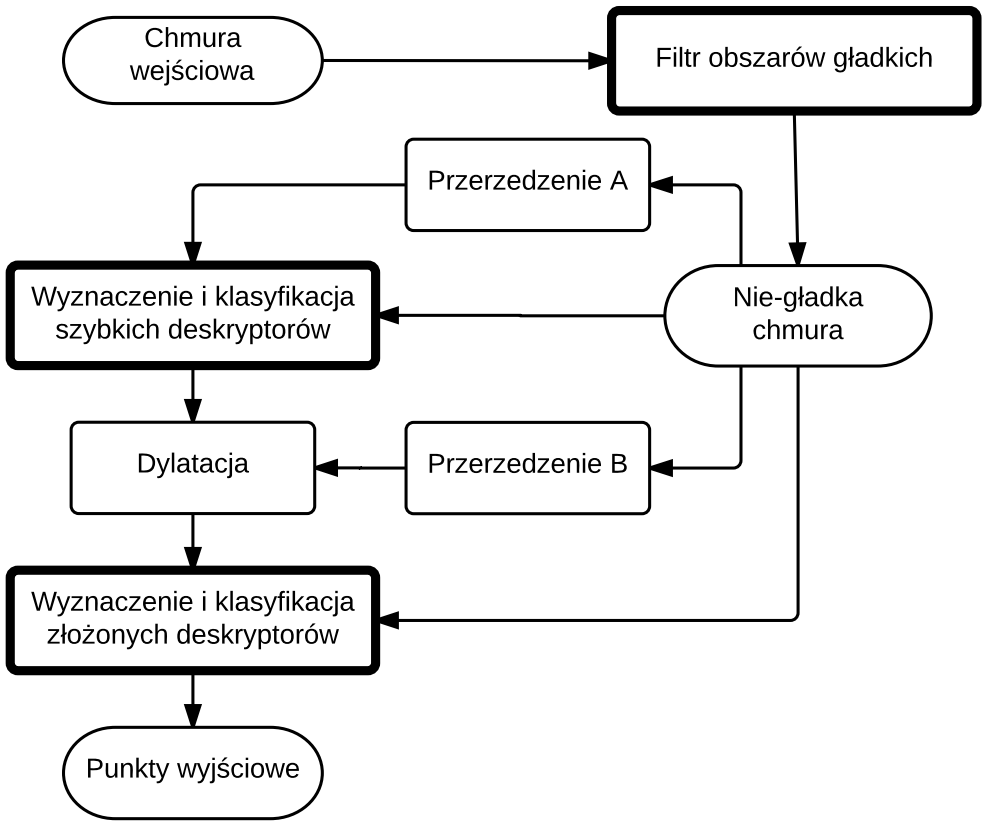
Rysunek 5.2: Wynik masowo-równoległej segmentacji i odrzucenia obszarów gładkich, przeprowadzonej w $51.17ms$ **dla** 136765 **punktów. Odsegmentowane niepłaskie klastry zostały wyróżnione białymi punktami**

5.3 Heurystyczne użycie deskryptorów

Jak opisano w rozdziale 2, ważną grupą metod rozpoznawania obiektów 3D są techniki oparte na deskryptorach lokalnych powierzchni. Metody te umożliwiają detekcję obiektów bez wstępnej segmentacji, a także same w sobie mogą być podstawą do przeprowadzenia segmentacji korzystającej z informacji wysokopoziomowych. Aby rozpoznawać obiekty za pomocą dekskryptorów, niezbędne jest dokonanie ich klasyfikacji. Polega to na zastosowaniu wybranego algorytmu określania, czy należą one do rozważanego obiektu (lub jednego z kilku rozważanych obiektów). Klasyfikacja deskryptorów może też funkcjonować jako heurystyka dla innych, wymagających obliczeniowo metod rozpoznawania. Utrudnieniem, które wiąże się z używaniem deskryptorów lokalnych jest wysoki koszt obliczeniowy niektórych złożonych deskryptorów w sytuacji, gdy wyznaczane są dla wielu obszarów. Skrajnym, lecz często spotykanym zadaniem jest wyznaczanie deskryptorów dla obszarów gęsto pokrywających całą scenę. Oczywistym, regularnie stosowanym uproszczeniem jest ograniczenie obliczania deskryptorów do występujących powierzchni, z pominięciem obszarów pustych. W wielu wypadkach nie jest

to jednak wystarczające, by uzyskać satysfakcjonujące czasy przetwarzania. Ugruntowaną metodą dalszego upraszczania sceny jest wyznaczenie punktów charakterystycznych [30, 34]. Jak wynika z zacytowanych publikacji, wybór detektora punktów charakterystycznych ma wpływ na jakość późniejszej klasyfikacji za pomocą deskryptorów lokalnych. Detektory, poza różną wrażliwością na określone kształty, charakteryzują się różną powtarzalnością wskazywanych punktów przy zmieniających się widokach [34]. Jak sugerują autorzy [2], gęste rozmieszczenie obszarów przetwarzanych przez deskryptory lokalne może być skuteczniejsze niż przetwarzanie wyłącznie punktów charakterystycznych, choć, oczywiście, znacznie bardziej czasochłonne.

W związku z tymi wadami używania detektorów punktów charakterystycznych, autor tej rozprawy zbadał alternatywne podejście heurystyczne w zadaniu detekcji obiektów na pełnych scenach wnętrza budynku [37]. Zaproponowana metoda wybierania obszarów do obliczenia deskryptorów została zilustrowana na Rys. 5.3.



Rysunek 5.3: Schemat zaproponowanej metody hierarchicznej redukcji sceny i obliczania deskryptorów. Pogrubioną ramką wyróżniono trzy główne etapy algorytmu

Algorytm operuje na pełnych chmurach punktów sceny, które ulegają redukcji w każdym kroku procedury. Pierwszy krok polega na usunięciu dużych, gładkich (najczęściej płaskich)

obszarów, w których nie spodziewamy się znalezienia istotnych elementów. Do przeprowadzenia tej operacji wykorzystano metodę opisaną w poprzednim podrozdziale.

Po usunięciu tych obszarów, kolejny krok polega na uproszczeniu chmury z wykorzystaniem filtru wokselowego w celu otrzymania zbioru punktów o stosunkowo dużej częstotliwości, zwanych dalej *punktami kotwiczącymi*. Różnica między punktami kotwiczącymi a puntami charakterystycznymi (używanymi w klasycznej analizie deskryptorów) polega na tym, że punkty charakterystyczne zostają wyznaczone przez algorytm ogólny przed jakimkolwiek użyciem informacji na temat obiektu zainteresowania. Punkty kotwiczące natomiast są początkowo równomiernie rozmieszczone blisko siebie na całej scenie (poza obszarami pustymi i dużymi, gładkimi fragmentami), więc jest ich znacznie więcej. Liczba punktów kotwiczących jest następnie ograniczana z użyciem informacji o konkretnym, klasyfikowanym obiekcie, jak zostanie dalej opisane.

Kuliste otoczenia początkowych punktów kotwiczących zostają znalezione na pełnej chmurze w sposób masowo-równoległy za pomocą zbudowanego i wykorzystanego wcześniej drzewa ósemkowego (wykorzystanego do segmentacji obszarów gładkich). Otoczenia kuliste zostają dalej wykorzystane do obliczania deskryptorów lokalnych. W pierwszej kolejności obliczane są deskryptory, które na schemacie oznaczono jako „szybkie". Na tym etapie klasyfikacji pożądane jest użycie deskryptorów o niskim koszcie obliczeniowym. Deskryptory te cechują się niskim udziałem klasyfikacji fałszywie ujemnych, lecz dopuszczalny jest dla nich dość wysoki wskaźnik klasyfikacji fałszywie dodatnich.

Wynikiem tych dwóch etapów heurystycznej redukcji jest obszar zainteresowania, w którym znajduje się przeważająca część punktów kotwiczących należących do poszukiwanych obiektów. Dla tych punktów będą dalej obliczane i klasyfikowane deskryptory złożone, bardziej dokładne i czasochłonne. Podobnie jak przy użyciu punktów charakterystycznych, przetwarzanie tych deskryptorów wymaga jedynie ułamka mocy obliczeniowej potrzebnej do ich przetworzenia dla całej sceny.

Po filtracji punktów kotwiczących za pomocą szybkich deskryptorów, zostaje przeprowadzony dodatkowy krok dylatacji. Polega to na propagacji pozytywnych klasyfikacji na sąsiednie punkty kotwiczące. Operacja ta opiera się na założeniu przestrzennej ciągłości obiektów i ma na celu zapobieganie pominięciom istotnych punktów kotwiczących na ostatnim etapie detekcji.

Na rysunku 5.3 operacje upraszczania chmury za pomocą filtrów wokselowych oznaczono jako „przerzedzenie" A i B. Te dwie operacje zostały rozróżnione, gdyż możliwe jest zastosowanie w nich wokseli o różnych rozmiarach (choć w eksperymentach użyto takich samych).

Jako deskryptory „szybkie" wykorzystano histogramy jednowymiarowe o 20 przedziałach oraz histogramy 2D o 20x20 przedziałach, utworzone z prostych cech lokalnych chmury punktów. Liczba przedziałów została dobrana eksperymentalnie tak, by histogramy różnych widoków obiektów zawierały wysoki poziom szczegółowości i jednocześnie były powtarzalne.

Użytymi cechami są: *Inklinacja (I)* (nachylenie) powierzchni, *wypukłość (C) i anizotropia wypukłości (A)*, które opisano wcześniej, a także *odcień (H) i saturacja (S)* z przestrzeni barw HSV. Wykorzystano też cechy geometryczne *D2, D3, D4*, będące znormalizowanymi histogramami odpowiednio: odległości między dowolnymi 2 punktami klastra, powierzchni rozpiętych na dowolnych 3 punktach klastra oraz objętości rozpiętych na dowolnych 4 punktach [61]. Histogramy te są estymowane przy użyciu losowych krotek punktów. Deskryptory „złożone" stanowią cechy histogramowe uchwytujące pełniejsze relacje między punktami i ich wektorami normalnymi: *PFH* [79], *FPFH* [75] oraz *PFHRGB* [3]. Do badań deskryptorów PFH, FPFH i PFHRGB posłużono się masowo-równoległymi implementacjami modułu GPU z bibliotek PCL, natomiast pozostałe cechy zostały zaimplementowane przez autora rozprawy oraz Łukasza Chechlińskiego (współautora cytowanej pracy [37]). Dla deskryptorów prostych, oprócz ich wykorzystania w testach w sposób bezpośredni, uwzględniono ich wersje łączone, polegające na prostym uśrednianiu podobieństwa do modelu powierzchni lokalnych obliczonego dla kilku różnych deskryptorów. Jak zostanie pokazane przy omówieniu eksperymentów, to połączenie przynosi dobre wyniki klasyfikacji.

Klasyfikacja deskryptorów odbywa się przez obliczanie współczynnika korelacji Pearsona między rozpatrywanymi deskryptorami sceny, a deskryptorami obliczonymi dla widoków modelowych. Dla widoków modelowych deskryptory są obliczane z gęstszym rozmieszczeniem punktów kotwiczących niż dla sceny. Takie działanie kompensuje negatywne efekty kwantyzacji i pozwala na nieco rzadsze rozmieszczenie punktów kotwiczących sceny (w celu przyspieszenia obliczeń). Baza deskryptorów modelowych dodatkowo była redukowana w taki sposób, że deskryptory w wysokim stopniu podobne do już istniejących w bazie nie były do niej dodawane (przy korelacji większej niż 0,99 z którymś z deskryptorów bazy). Aby zapobiec błędom spowodowanym przez uwzględnianie powszechnych, płaskich powierzchni, w procedurze automatycznego przygotowania bazy („uczenia" algorytmu) zostały wykorzystane „modele ne-

gatywne" powierzchni płaskiej, których funkcja polega na zapobieganiu dodawania do bazy podobnych do nich deskryptorów.

Eksperymenty

System był badany w środowisku realistycznym, z wykorzystaniem scen przechwyconych sensorem Kinect i przetworzonych w czasie akwizycji algorytmem Kinect Fusion [47]. Wybrano przygotowanie własnych zbiorów scen zamiast użycia gotowych. Jak wspomniano we wstępie pracy, dostępne w Internecie bazy składają się z surowych obrazów Kinecta, a te charakteryzują się niską dokładnością kształtów oraz licznymi brakami w chmurach punktów. Użycie Kinect Fusion i wiążący się z nim znaczny wzrost dokładności i kompletności obserwowanych powierzchni umożliwia rozpoznawanie przedmiotów niewielkich, mogących stanowić obiekt manipulacji robota usługowego.

Sceny były skanowane obserwując obiekty tylko z jednej strony przez około 3 sekundy, podczas których kamera wykonywała niewielki ruch (tę samą metodę wykorzystano do skanowania widoków modeli). Sceny były zróżnicowane pod względem odległości kamery od obiektów oraz pod względem kierunku obserwacji. Komputer przetwarzający był wyposażony w układ GPGPU (ang. *general-purpose graphics processing unit*) Nvidia GTX Titan Black (obliczenia deskryptorów przeprowadzano masowo-równolegle – dlatego najbardziej istotna jest karta graficzna). Zbiór testowy scen został dobrany tak, by odzwierciedlał trudne warunki nieuporządkowanego środowiska domowego. W takim środowisku występują liczne obiekty, wzajemne przysłonięcia, przedmioty przewrócone, stykające się i ułożone jedne na drugich, a także zróżnicowana widoczność i odległość obiektów. Zbiór treningowy zawiera 6 przedmiotów w różnych ułożeniach i 6 widoków każdego z nich: *książka, kubek, żelazko, butelka keczupu, płyn hamulcowy, myszka komputerowa*. Zbiór testowy zawiera 21 scen z obiektami zawartymi w zbiorze treningowym oraz wieloma innymi, nie znanymi dla systemu. Dane eksperymentalne są ograniczone, lecz stwarzają wymagające warunki. W celu oszacowania dokładności algorytmu, znane obiekty zostały dokładnie oznaczone ręcznie.

W proponowanym systemie klasyfikacji deskryptor zostaje uznany za należący do obiektu jeżeli odległość od najbliższego deskryptora w bazie modeli nie przekracza pewnego ustalonego progu. To podejście jest bardzo korzystne przy niewielkiej liczbie przykładów uczących (np. kilku widokach pokazanych robotowi). Przeszukiwanie bazy modeli zostało w pełni zrównoleglone z wykorzystniem technologii CUDA. Aby porównać skuteczność różnych deskryptorów,

wykreślono wykresy precyzji i kompletności (ang. *recall*) oraz miary skuteczności $F1$. Miary te są powszechnie wykorzystywane w ocenie klasyfikatorów binarnych:

$$precyzja = \frac{TP}{TP+FP}, \tag{5.1}$$

$$kompletnosc = \frac{TP}{TP+FN}, \tag{5.2}$$

$$F1 = 2 \cdot \frac{precyzja \cdot kompletnosc}{precyzja + kompletnosc}, \tag{5.3}$$

gdzie:

- TP (ang. *true positive*) to klasyfikacje prawdziwie pozytywne,
- FP (ang. *false positive*) to klasyfikacje fałszywie pozytywne,
- TN (ang. *true negative*) to klasyfikacje prawdziwie negatywne,
- FN (ang. *false negative*) to klasyfikacje fałszywie negatywne.

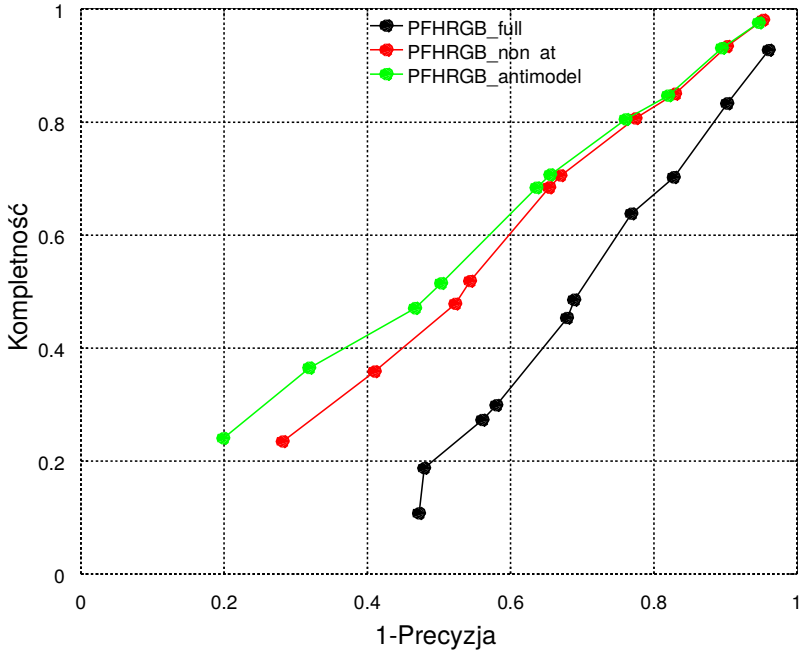
W ramach eksperymentów zbadano różne wartości promienia sąsiedztw dla deskryptorów. Najlepsze rezultaty uzyskał promień zależny od rozmiaru poszukiwanego obiektu, zdefiniowany następująco:

$$R = \sqrt{N/k}, \tag{5.4}$$

gdzie: k to średnia liczba sąsiadów w promieniu $1cm$ (uśredniona dla całej sceny), zaś N jest średnią liczbą punktów obiektu w widoku modelowym.

Stałe promienie o wartości $6-7cm$ sprawdziły się tylko nieznacznie gorzej. Stały promień byłby preferowany ze względu na wydajność obliczeniową, gdyby poszukiwano kilku różnych obiektów jednocześnie. Gęstość filtru wokselowego została dobrana w taki sposób, by na płaskiej powierzchni otoczenie danego punktu kotwiczącego zawierało średnio 7 punktów kotwiczących. Wartość ta została dobrana eksperymentalnie na etapie wstępnych badań jako wystarczająca do uniknięcia dużych błędów kwantyzacji.

W ramach walidacji pierwszego mechanizmu heurystycznego, porównano jakość klasyfikacji deskryptorów dla pełnej sceny, dla sceny bez obszarów gładkich oraz dla sceny z dodatkowym wykorzystaniem modeli negatywnych. Obydwa wymienione mechanizmy (usuwanie obszarów gładkich i modele negatywne) wpłynęły pozytywnie na jakość klasyfikacji (szczególnie usunięcie obszarów gładkich). Pokazano to na Rys. 5.4 na przykładzie deskryptora PFHRGB



Rysunek 5.4: Porównanie klasyfikacji deksyptorem PFHRGB dla scen pełnych, z usuwaniem obszarów gładkich (*nonflat*) oraz dodatkowo uczonych modelami negatywnymi lub ujemnymi (ang. *antimodel*)

Tabela 5.1: Wyniki filtracji „niegładkiej”

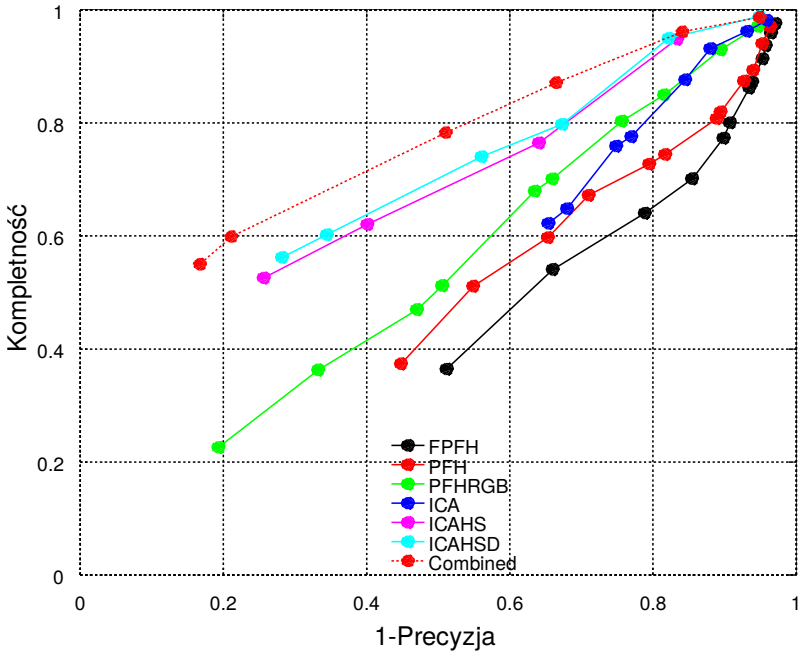
Średnia liczba punktów kotwiczących	1975
Średnia liczba punktów po filtracji niegładkiej	1010
Średni czas wykrywania obszarów gładkich	38.5 ms

(ocenionego jako najdokładniejszy w [29, 2]). Wyniki ilościowe filtracji „niegładkiej” uzyskane z eksperymentów przedstawiono w Tab. 5.1.

Po uwzględnieniu filtracji obszarów gładkich i modeli ujemnych, porównano deskryptory FPFH, PFH, PFHRGB oraz wiele kombinacji wymienionych wcześniej histogramów cech prostych – I, C, A, H, S, D2, D3, D4 w wersji 1D i 2D. Pominięto pełne przedstawienie wyników szczegółowych, gdyż nie wnoszą wiele dodatkowych informacji. Testy wykazały, że histogramy 2D nie mają w rozważanym zastosowaniu zauważalnej przewagi nad ich jednowymiarową wersją.

Na Rys. 5.5 przedstawiono wykresy (1-precyzja) x kompletność dla wybranych deskryptorów (dających najlepsze wyniki). Jak widzimy, deskryptor łączący wszystkie histogramy jed-

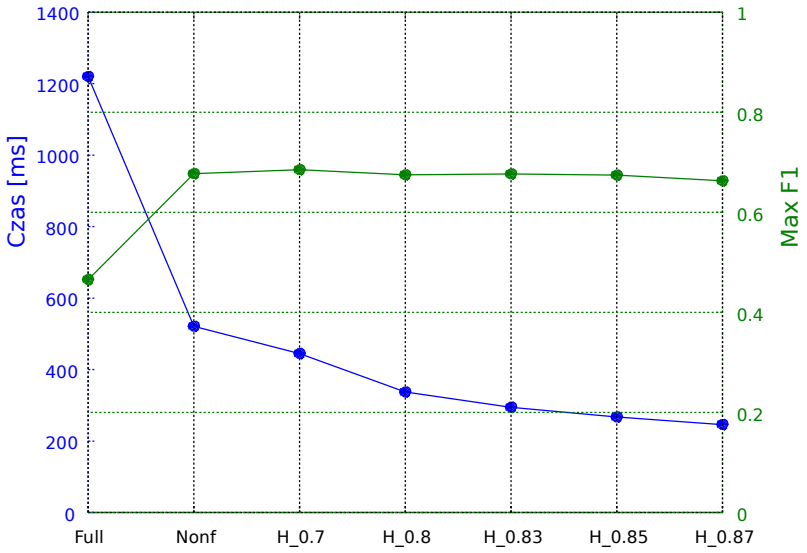
nowymiarowe cech prostych (nazwany ICAHSD) osiągnął najlepsze wyniki (przewyższające PFHRGB – najlepszy deskryptor złożony). Jednakże w połączeniu z PFHRGB (w sposób naiwny, przez uśrednianie korelacji dla obu deskryptorów), deskryptor osiągnął jeszcze lepsze wyniki (*Combined*) niż każdy z nich osobno. Warto zauważyć, że krzywe nie zawsze sięgają lewej krawędzi wykresu (odpowiadającej wysokiej precyzji). Oznacza to, że po przekroczeniu pewnego progu wymaganej korelacji nie występowały już żadne pozytywne klasyfikacje w bazie testowej.



Rysunek 5.5: Porównanie wybranych deskryptorów (wykorzystano niegładką heurystykę i modele ujemne dla wszystkich). *Combined* oznacza ICAHSD + PFHRGB

W oparciu o te wyniki, przeprowadzono testy pełnego systemu – z filtracją obszarów gładkich, modelami negatywnymi oraz deskryptorami ICAHSD jako heurystyką. Warto zaznaczyć, że obliczenie deskryptorów ICAHSD na niegładkich obszarach trwa średnio ok. 100*ms*. Do klasyfikacji końcowej (jako deskryptora złożonego) wykorzystano deskryptor łączony, który okazał się najdokładniejszy (ICAHSD + PFHRGB). Rys. 5.6 przedstawia zmierzony wpływ proponowanego mechanizmu heurystycznego na czas przetwarzania sceny przy różnych progach decyzyjnych korelacji dla niższego etapu klasyfikacji heurystycznej. Tabela 5.2 podsumowuje przeprowadzone badania, uwzględniając stopień heurystyczny: 0 – pełna scena, 1 – heurystyka

obszarów niegładkich, 2 – heurystyka obszarów niegładkich i deskryptorów pierwszego poziomu.



Rysunek 5.6: Czasy przetwarzania i maksymalne (w kontekście różnych progów) wskaźniki F1 dla deskryptora *Combined* bez heurystyki, czyli dla pełnej sceny (*Full*), z niegładką heurystyką (*Nonf*) i całościową heurystyką (H) dla różnych progów klasyfikacji ICAHSD

Jak wynika z wykresu 5.6 i tabeli 5.2, początkowy czas przetwarzania sceny został zmniejszony ponad dwukrotnie w wyniku samego odrzucenia obszarów gładkich (zwiększając też wskaźnik jakości F1). Dalsza dwukrotna redukcja czasu została osiągnięta dzięki klasyfikacji pierwszego poziomu (ICAHSD). Wynik F1 prawie się nie zmieniał, aż do osiągnięcia progu 0.87. Można stąd wnioskować, że wartość progu nieco poniżej tego poziomu, np. 0.85 dla wymaganej korelacji deskryptorów „szybkich" wydaje się być optymalna. Z tabeli 5.2 wynika, że czas przetwarzania deskryptorów prostych nie zmniejsza się w wyniku wykorzystania heurystyki obszarów niegładkich, co jest ciekawym, nieintuicyjnym zjawiskiem. Wynika to z niskiego kosztu obliczeniowego deskryptorów „szybkich" użytych w tych testach. Badania pokazały jednak, że heurystyka obszarów niegładkich zwiększa jakość klasyfikacji, zmniejszając fałszywie pozytywne klasyfikacje, więc ostatecznie pozostaje korzystną częścią algorytmu.

Przykładową scenę ze zbioru testowego, przetworzoną pełnym zaproponowanym algorytmem, przedstawiono na Rys. 5.7.

Tabela 5.2: Porównanie poziomów heurystycznych, maksymalnych (w kontekście różnych progów) wskaźników F1 i średnich czasów przetwarzania dla wybranych deskryptorów w konfiguracjach: bez heurystyki (full), niegładka filtracja (N), modele ujemne (A), wszystkie etapy heurystyczne (H)

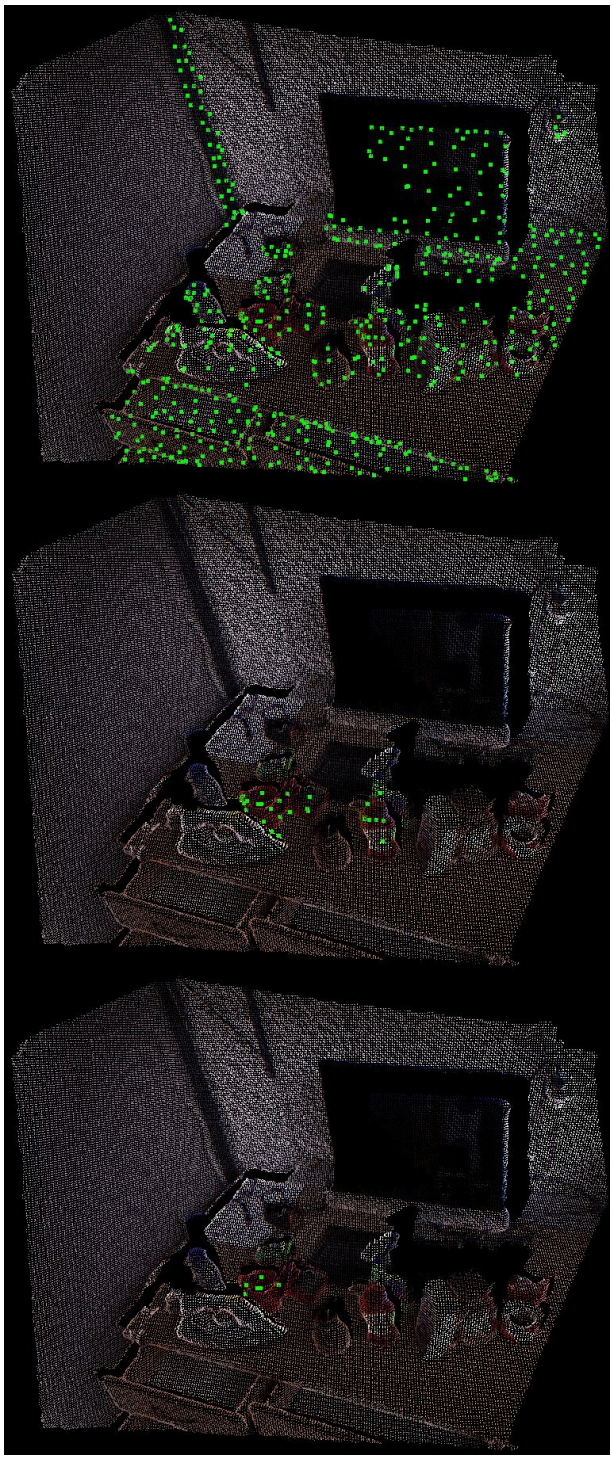
Deskryptor	Poz H	Maks F1	Śr T [ms]
FPFH full	0	0,229	**254,3**
FPFH N + A	1	0,367	**166,2**
PFH full	0	0,261	671,3
PFH N + A	1	0,369	**290,8**
PFHRGB full	0	0,283	1053,1
PFHRGB N	1	0,403	439,1
PFHRGB N + A	1	0,460	440,2
ICA full	0	0,279	**94,5**
ICA N + A	1	0,327	**103,5**
ICAHS full	0	0,422	**105,3**
ICAHS N + A	1	0,542	**109,1**
ICAHSD full	0	0,382	**192,6**
ICAHSD N + A	1	0,567	**147,2**
Combined full	0	0,466	1220,3
Combined N + A	1	**0,677**	521,5
Combined H 0.7	2	**0,685**	445,2
Combined H 0.8	2	**0,675**	337,7
Combined H 0.83	2	**0,677**	**294,9**
Combined H 0.85	2	**0,674**	**267,6**
Combined H 0.87	2	**0,663**	**246,6**

5.4 Wnioski

W tym rozdziale zaprezentowano kilka metod służących do szybkiego rozpoznawania obiektów w chmurach punktów o wysokiej dokładności. Poza możliwościami przyspieszania znanych technik, doświadczenia pokazują wysoką użyteczność łączenia różnych prostych deskryptorów lokalnych w polepszaniu dokładności rozpoznawania. Aby dokonać dalszej optymalizacji czasowej i jakościowej, należałoby przetestować większą liczbę deskryptorów lokalnych i użyć dużej bazy testowych scen realistycznych. Jednak, jak wspomniano we wstępie, mało jest ogólnodostępnych materiałów pozwalających na uzyskanie chmur wysokiej dokładności. Takie

chmury są niezbędne do rozpoznawania niewielkich, istotnych w robotyce usługowej obiektów. Wartościowe byłoby także zbadanie możliwości połączenia przedstawionych metod z podejściem wykorzystującym detektory punktów charakterystycznych.

Naturalnym zastosowaniem omówionych algorytmów byłoby ich połączenie z dokładną techniką weryfikacji hipotez obiektów, wykorzystującą dopasowanie metryczne (np. algorytm typu RANSAC). Wymagający, lecz dokładny algorytm dopasowania mógłby używać wyników wypracowanych przez metodę deskryptorów jako heurystyki.

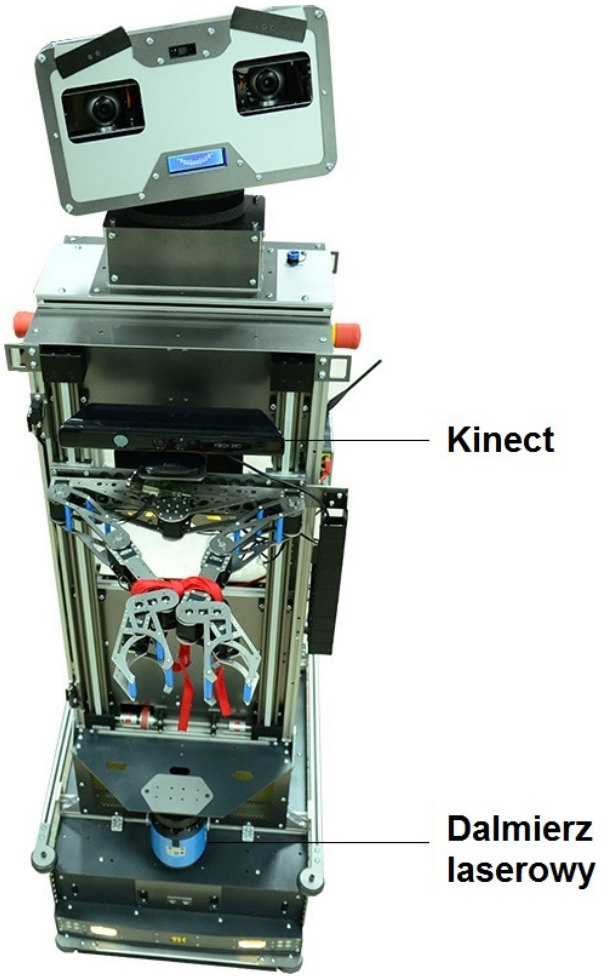


Rysunek 5.7: Przykładowa scena testowa z przygotowanego zbioru Kinect Fusion przetworzona w proponowany, hierarchiczny sposób dla detekcji nauczonego kubka. Zielone kropki przedstawiają punkty kotwiczące dla sceny: (od góry do dołu) po usunięciu obszarów gładkich, po klasyfikacji deskryptorem ICAHSD oraz po klasyfikacji deskryptorem łączonym (ICAHSD+PFHRGB)

6. Rozpoznawanie obiektów 3D w systemie robotycznym

Aby lepiej zapoznać się z rzeczywistą problematyką rozpoznawania obiektów w robotyce usługowej, zaimplementowano prosty system robotyczny integrujący nawigację metryczną, tworzenie mapy semantyczno-metrycznej oraz rozpoznawanie obiektów 3D. Do przeprowadzenia eksperymentów użyto badawczej platformy mobilnej, jaką jest Robot Kurier [84], zaprojektowany i zbudowany na wydziale Mechatroniki Politechniki Warszawskiej (Rys. 6.1). Istotnymi z punktu widzenia opisywanego systemu sensorami robota są: skaner laserowy (wykorzystywany do nawigacji metrycznej) oraz kamera głębi Kinect, za pomocą której możliwe jest trójwymiarowe rozpoznawanie obiektów.

Omawiany system został opisany w [66]. Moduły nawigacji i obsługi map semantycznych zostały przygotowane głównie przez Macieja Przybylskiego. Problemy tworzenia map oraz wizualizacji robota i obiektów na mapie budynku zostały rozwiązane adaptując ogólnodostępne komponenty systemu ROS (ang. *Robot Operating System*). W opracowaniu systemu brali udział także Daniel Koguciuk, Barbara Siemiątkowska i Łukasz Chechliński. System wizyjny, przygotowany głównie przez autora tej rozprawy, ograniczono do podstawowych, szybkich metod, by umożliwić płynne działanie on-line, w warunkach stosunkowo uporządkowanego środowiska laboratoryjnego. Algorytm rozpoznawania opiera się na opisanej poprzednio segmentacji obszarów gładkich, po której uzyskane ciągłe przestrzennie klastry są klasyfikowane za pomocą prostych deskryptorów (opisanych w rozdziale 5) metodą najbliższych sąsiadów. Obliczanie odległości do sąsiadów odbywa się w metryce korelacyjnej, gdzie odległość jest rozumiana jako odwrotność korelacji Pearsona. Procedura rozpoznawania została zaimplementowana w autorskiej platformie badawczej *Heuros*, osadzonej z kolei w ROS. Platforma ta zawiera wiele pomocnych narzędzi wizualizacyjnych, obliczeniowych i komunikacyjnych. Oprogramowanie *Heuros* zostało udostępnione do użytku społeczności naukowej jako „otwartoźródłowe" (ang. *open source*) w wersji używanej do części eksperymentów omawianych w tej rozprawie [108].

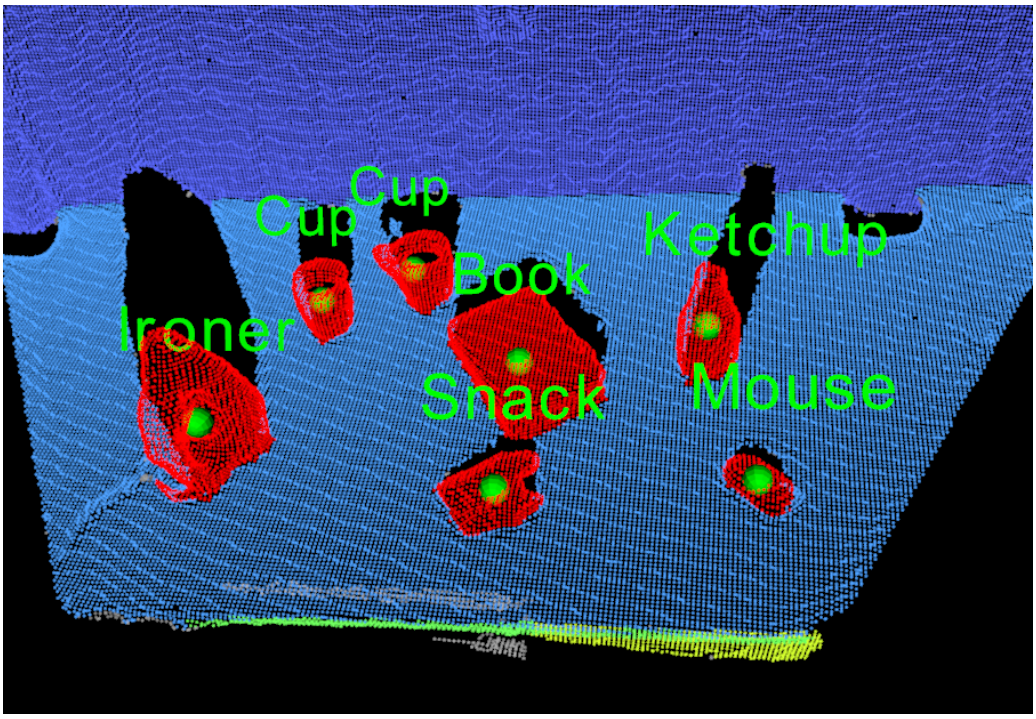


Rysunek 6.1: Robot Kurier – platforma mobilna wykorzystana w implementacji systemu robotycznego

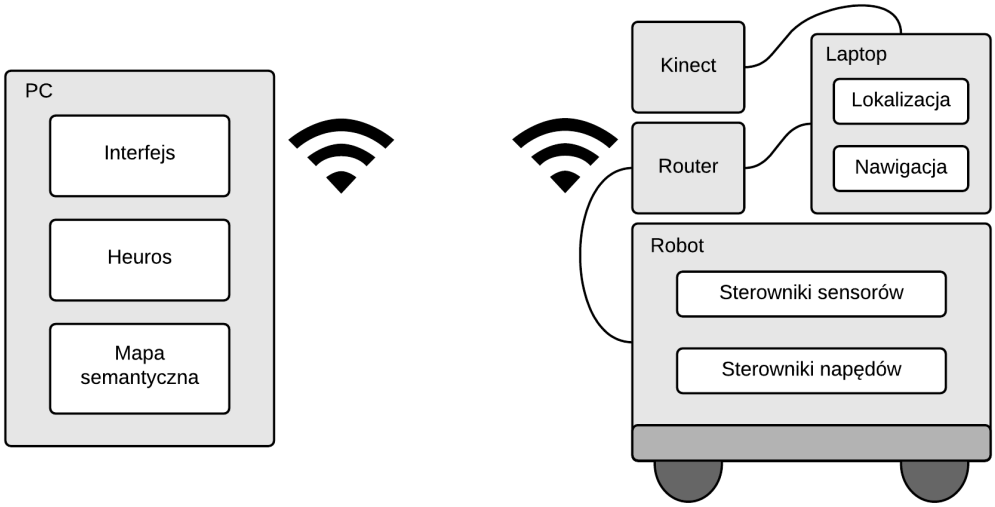
Platformę udostępniono też w wersji bardziej wydajnej, lecz z bardziej ograniczoną funkcjonalnością [107]. Na Rys. 6.2 pokazano rozpoznania użytego systemu wizyjnego w warunkach dobrej widoczności i nie stykających się obiektów. Takie warunki ustanowiono na potrzeby implementacji dość prostego systemu interakcji z mapą semantyczno-metryczną, omawianego w tym rozdziale. Schemat pełnego systemu robotycznego przedstawiono na Rys. 6.3.

Jak pokazano na schemacie, przetwarzanie obrazu 3D i aktualizacja mapy semantycznej (nanoszenie rozpoznanych obiektów na mapę) odbywa się poza pokładem robota, na komputerze skomunikowanym z robotem przez sieć bezprzewodową o wysokiej przepustowości. Ograniczenie to wynikło z braku karty graficznej odpowiedniej do wymagających obliczeń masowo-równoległych na pokładzie robota.

Istotnym elementem przygotowanego systemu jest zastosowanie algorytmu Kinect Fusion [47], który pozwala na znaczące podniesienie dokładności i uzupełnienie braków chmury punk-



Rysunek 6.2: Przykładowe obiekty rozpoznawane przez system wizyjny (nazwy angielskie są wyświetlane w interfejsie systemu nad geometrycznymi środkami odsegmentowanych klastrów)

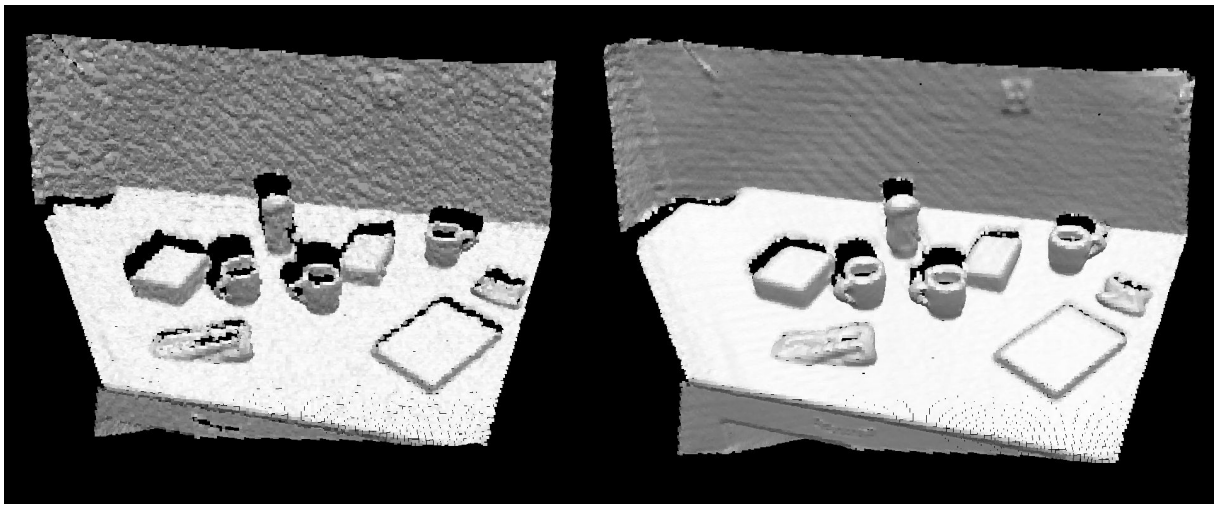


Rysunek 6.3: Schemat systemu robotycznego. Blok „Heuros" stanowi trójwymiarowy system wizyjny – ze względu na ograniczenia komputera pokładowego, przetwarzanie obrazu odbywa się zdalnie

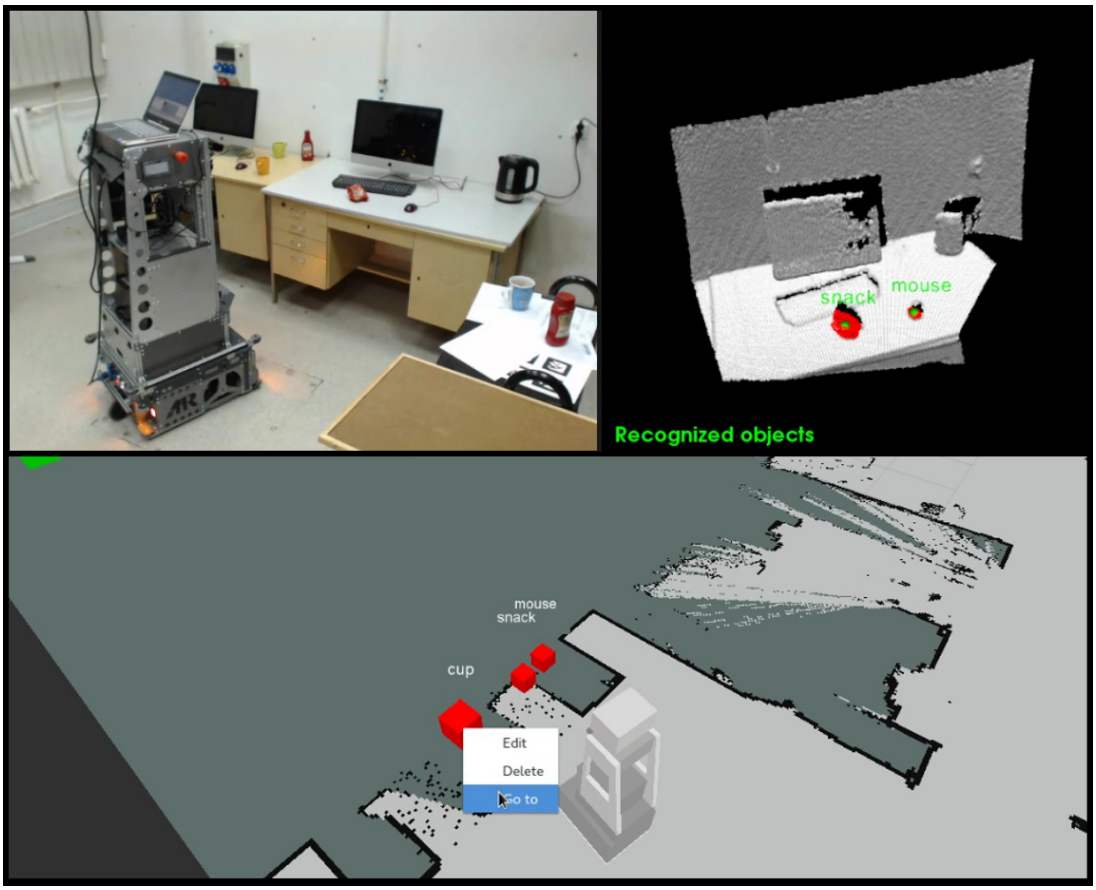
tów (Rys. 6.4). Algorytm ten, jak wspomniano wcześniej, łączy informację pochodzącą z wielu klatek kamery głębi w sposób bardzo wydajny, korzystając z obliczeń masowo-równoległych. Przeprowadzone testy pokazały, że nawet jednosekundowa obserwacja i scalanie klatek sceny wprowadza wyraźny wzrost dokładności względem chmury punktów z jednej klatki. Ze względu na ograniczoną prędkość robota i chęć uzyskania jak najwyższej dokładności sceny, postanowiono przeprowadzać analizę sceny w cyklu 3-sekundowym. W jedynym cyklu robot (ruchomy lub nieruchomy) przetwarza klatki za pomocą Kinect Fusion, po czym następuje rozpoznawanie obiektów i aktualizacja mapy semantycznej. Zaletą metody Kinect Fusion jest to, że kamera może być w ruchu: jest to nawet bardzo wskazane, gdyż pozwala na kompensację jej błędu systematycznego (dla nieruchomej kamery poprawa jakości jest mniejsza niż dla ruchomej). W związku z tym, aby uczynić maksymalny użytek z dostępnego sprzętu, użyto wbudowanego silnika kamery Kinect, który zapewnia jej bezustanny, powolny ruch góra-dół. Pozwala to uzyskać wspomniany efekt kompensacji błędów systematycznych również w sytuacji, gdy robot się nie przemieszcza, zwiększając jednocześnie efektywne pole widzenia.

Scalanie klatek za pomocą Kinect Fusion nie jest jednak pozbawione wad. Jedną z nich jest przydatność takiego podejścia tylko do scen statycznych. Podczas obserwacji sceny z obiektami ruchomymi, musiałyby one być wykrywane i usuwane niezależnym algorytmem (co nie zostało zaimplementowane w opisywanym systemie). Innym problemem jest ograniczona objętość obserwowanej przestrzeni, po wyjściu z której musi nastąpić restart algorytmu. Nie jest to problemem przy niewielkich prędkościach obrotowych robota, lecz w bardziej dynamicznym systemie należałoby skorzystać z uogólnienia użytej metody agregacji, takiego jak w pracy [97]. Ostatecznie jednak przeprowadzone testy pokazały, że używanie Kinect Fusion jest wręcz niezbędne do uzyskania satysfakcjonującej jakości rozpoznań, zwłaszcza niewielkich lub czarnych obiektów. Czarne powierzchnie są słabo rejestrowane przez sensor Kinect i na pojedynczych klatkach powodują duże braki w chmurze punktów.

Używana mapa semantyczno-metryczna stanowi bazę do przechowywania aktualnych informacji o środowisku robota, w tym najnowsze rozpoznania obiektów (w momencie rozpoznania obiekty są automatycznie nanoszone na mapę). Obiekty poza polem widzenia robota pozostają na mapie aż do czasu automatycznego stwierdzenia ich braku lub zmiany pozycji, czy też usunięcia ich z mapy przez użytkownika. Mapa ta umożliwia robotowi nawigację, ale również umożliwia człowiekowi wgląd w stan wiedzy robota i wydawanie poleceń. Umożliwia to interfejs markerów przygotowany w wizualizatorze systemu ROS – Rviz. Używając mapy



Rysunek 6.4: Porównanie sceny (bez koloru) z pojedynczej klatki Kinecta (po lewej) ze sceną uzyskaną po 3-sekundowym działaniu Kinect Fusion (z wielu klatek)



Rysunek 6.5: Zestawienie widoków w eksperymencie systemu robotycznego. Lewy górny obraz pokazuje poglądowy widok robota i środowiska z kamery zewnętrznej; prawy górny obraz pokazuje ostatnią przetworzoną scenę i rozpoznania systemu Heuros; dolny obraz przedstawia widok mapy semantyczno-metrycznej, która stanowi zarówno podstawę do nawigacji, jak i do komunikacji z użytkownikiem

możliwe jest dodawanie, usuwanie lub zmiana położenia obiektów, a także zlecenie robotowi, by podjechał autonomicznie do wybranego obiektu lub węzła na mapie semantycznej. Po zadaniu celu robot dojeżdża i zatrzyma się w pewnym przedziale odległości od wskazanego miejsca. Za pomocą tego interfejsu możliwe jest też polecenie robotowi, by przyjął wskazaną pozycję i orientację na mapie metrycznej.

W przeprowadzonym scenariuszu testowym robot podjeżdża do mebli, na których znajdują się różne znane systemowi obiekty spożywcze i biurowe: kubki, mysz komputerowa, butelka i opakowane przekąski. Użytkownik wydaje polecenie podjechania do danego obiektu, zaś podczas wykonywania polecenia przez robota, wprowadzane są fizycznie zmiany w niewidocznej dla niego części środowiska. Niektóre obiekty są dodawane, a niektóre zabierane. Następnie robot otrzymuje polecenie podjechania do obiektów, w których zaszły zmiany, dzięki czemu możliwa jest ocena skuteczności automatycznej aktualizacji mapy semantycznej i ogólnej pracy robota w takim systemie.

Eksperymentalny system działał zgodnie z oczekiwaniami – mapa semantyczna była w większości wypadków prawidłowo aktualizowana z wykorzystaniem rozpoznań obiektów. Zdarzały się przypadki pominięcia obiektu, który zostawał rozpoznany w kolejnym cyklu obserwacji lub dopiero gdy robot podjeżdżał bliżej. Niedoskonałością okazał się mocno ograniczony zasięg Kinecta, a dokładniej zakres odległości, dla którego przechwycona chmura punktów jest wystarczająco dokładna do rozpoznawania obiektów. Maksymalna odległość robota od obserwowanego obiektu, która umożliwiała stosunkowo pewne rozpoznawaniem wynosiła niewiele ponad 3 m mimo zastosowania Kinect Fusion. Odległość ta była trochę większa dla dużych lub dobrze widocznych obiektów. Rozpoznania najbardziej oddalonych od kamery przedmiotów były niestabilne (obiekty nie były rozpoznawane w każdym cyklu). Aby zwiększyć wiarygodność systemu wprowadzono prostą filtrację dolnoprzepustową przy usuwaniu obiektów z mapy (obiekty usuwane są dopiero po stwierdzeniu ich braku w kilku kolejnych cyklach).

Konkluzja eksperymentu była taka, że zaproponowany system jest realizowalny w praktyce i może być użyteczny w robotyce usługowej, lecz wskazane byłoby zastosowanie kamery głębi o większym zasięgu (np. Kinect 2), a także ciągłej techniki filtracji chmury punktów (np. Kintinuous) zamiast podstawowej metody Kinect Fusion. Konieczność „ostrego” przechodzenia z jednego obserwowanego obszaru na inny stwarza pewne utrudnienie, którego można by było uniknąć.

7. Podsumowanie i wnioski

Rozprawa została poświęcona jednemu z najbardziej kluczowych czynników do osiągnięcia wysokiej autonomii robotów mobilnych: percepcji. W szczególności praca eksploruje techniki wykorzystania współczesnych sensorów trójwymiarowych – kamer głębi, w semantycznym etykietowaniu elementów otoczenia robota usługowego. Dla usługowego robota mobilnego, semantyczne etykietowanie stanowi podstawę do użytecznej interakcji z otoczeniem.

Autor zaproponował szereg założeń dotyczących systemu rozpoznawania obiektów, który będzie użyteczny w mobilnej robotyce usługowej. Rozróżniono zadanie rozpoznawania obiektów w robotyce mobilnej od rozpoznawania obiektów dla innych zastosowań, np. automatycznej interpretacji zdjęć w Internecie. Wśród sformułowanych założeń najważniejsza jest szybkość przetwarzania danych oraz szybkość uczenia się nowych obiektów (zdolność uczenia się na podstawie niewielu widoków). Ważnym założeniem z punktu widzenia wyboru algorytmów jest też niewielki (w większości sytuacji) zbiór obiektów jednoczesnego zainteresowania robota. Założenia te wynikają z domniemanych, a także spotykanych dzisiaj warunków pracy robotów mobilnych. Te spostrzeżenia stanowiły swojego rodzaju „drogowskaz” w badaniach autora nad różnymi konkretnymi algorytmami.

Praca zawiera klasyfikację technik rozpoznawania obiektów (2D i 3D) opartą o szeroki przegląd literatury naukowej – zarówno nowej, jak i „zakorzenionej” w dziedzinie rozpoznawania obiektów. Podczas omawiania metod trójwymiarowych, zostały porównane wyróżnione przez autora grupy technik najbardziej użytecznych w rozpatrywanych warunkach: oparte o segmentację, wykorzystujące deskryptory lokalne oraz metod z dopasowaniem modelu. Dla każdej z nich krótko omówiono przykłady konkretnych algorytmów.

Duża część pracy opisuje badania prowadzone przez autora rozprawy, które dotyczą rozpoznawania obiektów 3D na potrzeby robotyki usługowej. Większość tych badań została (przynajmniej częściowo) opublikowana w postaci artykułów naukowych. Opis poszczególnych prac został uzupełniony i zaktualizowany w kontekście najnowszych osiągnięć w dziedzinie. Oryginalny wkład autora dotyczy następujących zagadnień:

- Rozpoznawania obiektów z wykorzystaniem kontekstu: główny wkład autora polega na opracowaniu metody uwzględniania kontekstu semantycznego (hipotez innych pobliskich obiektów) podczas klasyfikacji i kojarzenia poszczególnych fragmentów sceny. Dzięki wykorzystaniu uogólnienia losowych pól Markowa i algorytmu ewolucyjnego, zaproponowana metoda umożliwia znalezienie „teorii" najlepiej wyjaśniającej scenę. Teorie sceny są oceniane zarówno pod względem zarejestrowanych cech indywidualnych obiektów, jak i ich konfiguracji przestrzennej, opisanej w sposób semantyczny.

- Jawnego wykorzystania informacji o ograniczeniach percepcyjnych systemu w rozpoznawaniu obiektów: autor proponuje metodykę wnioskowania opartą na teorii Dempstera-Shafera, która pozwala na modelowanie niepewności i niewiedzy systemu. Uwzględnione rodzaje niepewności wynikają z takich utrudnień percepcyjnych, jak odległość od obiektów oraz występujące na scenie przysłonięcia.

- Autorskiego, masowo-równoległego algorytmu rozrostu ziarna, który w pokazanym zastosowaniu umożliwia szybkie znalezienie dużych obszarów gładkich w chmurze punktów. Algorytm ten stanowi użyteczną innowację w wydajnym przetwarzaniu chmur punktów i umożliwia m.in. praktyczne czasowo rozpoznawanie obiektów w chmurach niezorganizowanych. Chmury te, w przeciwieństwie do obrazów 2,5 D stanowią w pełni trójwymiarowy zapis sceny i cechuje je znacznie wyższa elastyczność metod przechwytywania sceny i fuzji danych.

- Metod heurystycznego ograniczania uwagi (przestrzeni poszukiwań) dla technik rozpoznawania obiektów z użyciem deskryptorów lokalnych. Techniki te, choć kosztowne obliczeniowo, posiadają bardzo użyteczne właściwości dla rozpoznawania obiektów w środowisku nieuporządkowanym. Ta zaleta wynika z faktu, że zasada działania deskryptorów lokalnych nie jest oparta na segmentacji, więc sprawdzają się one tam, gdzie segmentacja jest trudna. Przedstawione badania związane z tym tematem skupiają się na autorskich metodach przyspieszających analizę sceny poprzez kilkustopniową redukcję. Metody te mogą zamieniać lub uzupełniać powszechnie stosowanie podejście ogólnych (niezależnych od obiektów zainteresowania) detektorów punktów charakterystycznych. Użyteczność zaproponowanych technik została potwierdzona doświadczalnie.

- Eksperymentalnego systemu robotycznego z rozpoznawaniem obiektów 3D w chmurach niezorganizowanych, przy użyciu agregacji danych z wielu klatek. System ten został sku-

tecznie zrealizowany na sprzęcie fizycznym, co pozwoliło wyciągnąć cenne wnioski. W pracy opisano różne wyzwania związane z praktyczną realizacją takiego systemu, a także rolę i użyteczność rozpoznawania obiektów 3D w połączeniu z mapami semantyczno-metrycznymi.

Trudnością w prowadzeniu badań okazał się brak ogólnodostępnych baz testowych zawierających pożądane informacje w kontekście rozważanego obszaru robotyki mobilnej. Podczas gdy w ostatnich latach zostały wprowadzone pierwsze duże zbiory scen naturalnych RGB-D, nie zawierają one etykiet pozwalających na rozróżnianie instancji obiektów (konkretnych jednostek) na przestrzeni wielu scen, a jedynie ich kategorii semantycznej lub na odróżnianie instancji na pojedynczej scenie. Po zapoznaniu się z dostępnymi bazami (NYU Dataset, Berkeley 3D Object Dataset, SUN RGB-D), przeglądzie scen i oznaczeń autor zauważył, że oznaczenia są często bardzo wyrywkowe, niekompletne, a ich jakość jest niska dla wielu scen. Ponadto, jak uzasadniono w pracy, dokładność informacji metrycznej dostarczanej przez sensor Kinect (szczególnie Kinect V1) jest bardzo ograniczona i utrudnia rozpoznawanie niewielkich obiektów, które mogłyby stanowić przedmiot manipulacji robota mobilnego. Algorytmy agregacji informacji pochodzącej z wielu klatek, takie jak Kinect Fusion są w stanie znacząco podnieść jakość analizowanych scen, co zostało wykorzystane dla części przedstawionych badań. Dane wejściowe pozwalające na testy z użyciem tego typu rozwiązań są w domenie publicznej trudno dostępne. Z tych powodów konieczne było opracowywanie własnych zbiorów scen do różnych eksperymentów, co skutkowało ich ograniczonym rozmiarem.

Zaprezentowane w tej rozprawie prace koncentrują się na różnych, komplementarnych aspektach rozpoznawania obiektów 3D na potrzeby robotyki mobilnej i były prowadzone na przestrzeni kilku ostatnich lat przez autora. Interesująca byłaby implementacja holistycznego, „najlepszego” sytemu integrującego w ujednolicony sposób wszystkie rozpatrywane zagadnienia. Stanowiłoby to jednak duże przedsięwzięcie naukowo-inżynierskie, które musiałoby zostać zrealizowane bardzo szybko, a następnie być sprawnie aktualizowane, by pozostać użytecznym. Zdaniem autora prace wybiórcze nad rozpoznawaniem obiektów w chmurach punktów, z jakimi mamy dzisiaj do czynienia w literaturze naukowej, będą w najbliższych latach wciąż konieczne, zanim omawiana dziedzina osiągnie poziom pozwalający na wdrażanie rozbudowanych systemów praktycznych.

Bibliografia

[1] A. Aldoma et al. CAD-model recognition and 6DOF pose estimation using 3D cues. W: IEEE International Conference on Computer Vision Workshops, ICCV 2011 Workshops, Barcelona, Spain, November 6-13, 2011. *Proceedings*, 2011, s. 585–592.

[2] L. A. Alexandre. 3D descriptors for object and category recognition: a comparative evaluation. W: Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RSJ International Conference on Intelligent Robots and Systems. *Proceedings*, Vilamoura, Portugal, 2012.

[3] L. A. Alexandre. Set distance functions for 3D object recognition. W: 18th Iberoamerican Congress on Pattern Recognition. *Proceedings*, Havana, Cuba, Springer, 2013. wolumen LNCS 8258 serii *Lecture Notes in Computer Science*, s. 57–64.

[4] P. G. Armour. The five orders of ignorance. Communications of the ACM, 2000, wolumen 43, numer 10, s. 17–20.

[5] H. H. Baker, T. O. Binford. Depth from edge and intensity based stereo. W: Proceedings of the 7th International Joint Conference on Artificial Intelligence – Volume 2. *Proceedings*, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc., 1981, IJCAI'81, s. 631–636.

[6] D. H. Ballard. Readings in computer vision: Issues, problems, principles, and paradigms. rozdział Generalizing the Hough Transform to Detect Arbitrary Shapes, s. 714–725. San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. 1987.

[7] H. Bay et al. Speeded-up robust features (surf). Computer Vision and Image Understanding, 2008, wolumen 110, numer 3, s. 346–359.

[8] J. Besag. On the statistical analysis of dirty pictures. Journal of the Royal Statistical Society, 1986, wolumen 48, s. 259–302.

[9] I. Biederman. Review of visual cognition. American Journal of Psychology, 1988, wolumen 101, s. 146–148.

[10] I. Biederman, R. J. Mezzanotte, J. C. Rabinowitz. Scene perception: detecting and judging objects undergoing relational violations. Cognitive Psychology, 1982, wolumen 14, numer 2, s. 143–177.

[11] L. Bo, X. Ren, D. Fox. Kernel descriptors for visual recognition. W: *Advances in Neural Information Processing Systems 23* Red. J. D. Lafferty et al, s. 244–252. Curran Associates, Inc. 2010.

[12] P. L. Bogler. Shafer-Dempster reasoning with applications to multisensor target identification systems. IEEE Transactions on Systems, Man, and Cybernetics, 1987, wolumen 17, s. 968–977.

[13] G. Bradski. The OpenCV library. Dr. Dobb's Journal of Software Tools, 2000.

[14] J. Canny. A computational approach to edge detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1986, wolumen PAMI-8, numer 6, s. 679–698.

[15] X. Chen et al. 3D object proposals for accurate object class detection. W: Advances in Neural Information Processing Systems. *Proceedings*, 2015.

[16] B. Chokr, V. Kreinovich. How far are we from complete knowledge, complexity of knowledge acquisition in the Dempster-Shafer approach. Advances in the Dempster-Shafer Theory of Evidence, 1994, s. 555–576.

[17] R. M. Cichy, D. Pantazis, A. Oliva. Resolving human object recognition in space and time. Nature Neuroscience, 2014, wolumen 17, numer 3, s. 455–462.

[18] D. Comaniciu, P. Meer, S. Member. Mean shift: A robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002, wolumen 24, s. 603–619.

[19] R. Cupec et al. Detection of planar surfaces based on RANSAC and LAD plane fitting.. W: European Conference on Mobile Robots. *Proceedings* Red. I. Petrovic, A. J. Lilienthal. KoREMA, 2009, s. 37–42.

[20] P. De Graef, D. Christiaens, G. d'Ydewalle. Perceptual effects of scene context on object identification. Psychological Researcj, 1990, wolumen 52, numer 4, s. 317–329.

[21] A. P. Dempster, W. F. Chiu. Dempster-Shafer models for object recognition and classification. International Journal of Intelligent Systems, 2006, wolumen 21, numer 3, s. 283–297.

[22] C. Desai, D. Ramanan, C. Fowlkes. Discriminative models for multi-class object layout. International Journal of Computer Vision, 2011.

[23] J. J. DiCarlo, D. Zoccolan, N. C. Rust. How does the brain solve visual object recognition? Neuron, 2012, wolumen 73, numer 3, s. 415–434.

[24] F. Endres et al. 3-D mapping with an RGB-D camera. IEEE Transactions on Robotics, 2014, wolumen 30, numer 1, s. 177–187.

[25] J. T. Enns. *The Thinking Eye, The Seeing Brain: Explorations in Visual Cognition*. W. W. Norton & Company 2004.

[26] A. Ess et al. Improved multi-person tracking with active occlusion handling. W: ICRA Workshop on People Detection and Tracking. *Proceedings*, 2009. wolumen 2.

[27] R. Farid. Region-growing planar segmentation for robot action planning. W: AI 2015: Advances in Artificial Intelligence: 28th Australasian Joint Conference, Canberra, ACT, Australia, November 30 – December 4, 2015, Proceedings. *Proceedings* Red. B. Pfahringer, J. Renz. Springer International Publishing, 2015, s. 179–191.

[28] D. J. Felleman, D. C. Van Essen. Distributed hierarchical processing in the primate cerebral cortex. Cerebral Cortex, 1991, wolumen 1, numer 1, s. 1–47.

[29] S. Filipe, L. Itti, L. A. Alexandre. BIK-BUS: Biologically motivated 3D keypoint based on bottom-up saliency. IEEE Transactions on Image Processing, 2015, wolumen 24, numer 1, s. 163–175.

[30] S. Filipe, L. A. Alexandre. A comparative evaluation of 3D keypoint detectors in a RGB-D object dataset. W: 2014 International Conference on Computer Vision Theory and Applications (VISAPP). *Proceedings*, 2014.

[31] M. A. Fischler, R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM, 1981, wolumen 24, numer 6, s. 381–395.

[32] C. Galleguillos, S. J. Belongie. Context based object categorization: A critical survey. Computer Vision and Image Understanding, 2010, wolumen 114, numer 6, s. 712–722.

[33] A. Geiger, P. Lenz, R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. W: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). *Proceedings*, Providence, USA, 2012, s. 3354–3361.

[34] V. K. Ghorpade et al. Performance evaluation of 3D keypoint detectors for time-of-flight depth data. W: 2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV). *Proceedings*, 2016, s. 1–6.

[35] R. B. Girshick et al. Rich feature hierarchies for accurate object detection and semantic segmentation. Computing Research Repository, 2013, wolumen abs/1311.2524.

[36] T. Goedemé, T. Tuytelaars, L. V. Gool. Omnidirectional sparse visual path following with occlusion-robust feature tracking. W: in: 6th Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras, OMNIVIS05, in Conjunction with ICCV 2005. *Proceedings*, 2005, s. 1806–1811.

[37] B. Harasymowicz-Boggio, L. Chechliński. Keypoint-less, heuristic application of local 3D descriptors. Foundations of Computing and Decision Sciences, 2017, wolumen 42, s. 239–155.

[38] B. Harasymowicz-Boggio, L. Chechliński, B. Siemiątkowska. Nature-inspired, parallel object recognition. W: *Progress in Automation, Robotics and Measuring Techniques. vol. 1 Control and Automation* Red. R. Szewczyk, C. Zieliński, M. Kaliczyńska, wolumen 350 serii *Advances in Intelligent Systems and Computing*, s. 53–62. Springer 2015.

[39] B. Harasymowicz-Boggio, L. Chechliński, B. Siemiątkowska. Significance of features in object recognition using depth sensors. Optica Applicata, 2015, wolumen 45, numer 4, s. 559–571.

[40] B. Harasymowicz-Boggio, B. Siemiątkowska. Object classification using Dempster-Shafer theory. W: Mechatronics 2013: Recent Technological and Scientific Advances. *Proceedings*. Springer, 2013, s. 559–565.

[41] B. Harasymowicz-Boggio, B. Siemiątkowska. Object classification with metric and semantic inference. W: 2013 European Conference on Mobile Robots. *Proceedings*. Institute of Electrical and Electronics Engineers (IEEE), 2013, s. 186–191.

[42] B. Harasymowicz-Boggio, B. Siemiątkowska. Using ignorance in 3D scene understanding. Mathematical Problems in Engineering, 2014, s. 1–11.

[43] V. Hegde, R. Zadeh. Fusionnet: 3D object classification using multiple data representations. Computing Research Repository, 2016, wolumen abs/1607.05695.

[44] D. Holz et al. Robot soccer world cup xv. rozdział Real-time plane segmentation using RGB-D cameras, s. 306–317. Berlin, Heidelberg, Springer-Verlag 2012.

[45] J. Huang, S. You. Detecting objects in scene point cloud: A combinational approach. W: 2013 International Conference on 3D Vision – 3DV 2013. *Proceedings*, 2013, s. 175–182.

[46] G. Humphreys, C. Price, J. Riddoch. From objects to names: A cognitive neuroscience approach. Psychological Research, 1999, wolumen 62, s. 118–130.

[47] S. Izadi et al. Kinectfusion: real-time 3D reconstruction and interaction using a moving depth camera. W: Proceedings of the 24th annual ACM symposium on User interface software and technology. *Proceedings*, New York, NY, USA, ACM, 2011, UIST '11, s. 559–568.

[48] A. Janoch et al. A category-level 3-D object dataset: Putting the kinect to work. W: 1st Workshop on Consumer Depth Cameras for Computer Vision (ICCV workshop). *Proceedings*, 2011.

[49] A. E. Johnson. Spin-images: A representation for 3-D surface matching. Raport instytutowy, 1997.

[50] E. Kalogerakis, A. Hertzmann, K. Singh. Learning 3D mesh segmentation and labeling. ACM Transactions on Graphics, wolumen 29, numer 4.

[51] T. Kamioka et al. Dynamic gait transition between bipedal and quadrupedal locomotion. W: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems. *Proceedings*, 2015, s. 2195–2201.

[52] S. Kole et al. SURF and RANSAC: A conglomerative approach to object recognition. International Journal of Computer Applications, 2015, wolumen 109, numer 4, s. 7–9.

[53] I. Kostavelis, A. Gasteratos. Semantic mapping for mobile robotics tasks: A survey. Robotics and Autonomous Systems, 2015, wolumen 66, s. 86 – 103.

[54] M. Kraft et al. Efficient RGB-D data processing for feature-based self-localization of mobile robots. Applied Mathematics and Computer Science, 2016, wolumen 26, numer 1, s. 63–79.

[55] A. Krizhevsky, I. Sutskever, G. E. Hinton. Imagenet classification with deep convolutional neural networks. W: *Advances in Neural Information Processing Systems 25* Red. F. Pereira et al, s. 1097–1105. Curran Associates, Inc. 2012.

[56] K. Lai, L. Bo, D. Fox. Unsupervised feature learning for 3D scene labeling. W: IEEE International Conference on on Robotics and Automation. *Proceedings*, 2014.

[57] R. Lienhart, A. Kuranov, V. Pisarevsky. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. W: Pattern Recognition: 25th DAGM Symposium, Magdeburg, Germany, September 10-12, 2003. Proceedings. *Proceedings* Red. B. Michaelis, G. Krell. Springer Berlin Heidelberg, 2003, s. 297–304.

[58] D. G. Lowe. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 2004, wolumen 60, numer 2, s. 91–110.

[59] M. Maimaitimin, K. Watanabe, S. Maeyama. Surface-common-feature descriptor of point cloud data for deep learning. W: 2016 IEEE International Conference on Mechatronics and Automation. *Proceedings*, 2016, s. 525–529.

[60] A. J. R. Neves et al. Object detection based on plane segmentation and features matching for a service robot. International Journal of Computer, Electrical, Automation, Control and Information Engineering, 2016, wolumen 10, numer 4, s. 745 – 752.

[61] R. Osada et al. Shape distributions. ACM Transactions on Graphics, 2002, wolumen 21, numer 4, s. 807–832.

[62] S. E. Palmer. The effects of contextual scenes on the identification of objects. Memory & Cognition, 1975, wolumen 3, numer 5, s. 519–526.

[63] C. Papazov, D. Burschka. An efficient RANSAC for 3D object recognition in noisy and occluded scenes. W: Proceedings of the 10th Asian Conference on Computer Vision. *Proceedings*. Springer-Verlag, 2010, s. 135–148.

[64] B. Pepik et al. Occlusion patterns for object class detection. W: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). *Proceedings*, 2013.

[65] M. C. Potter. Meaning in visual search. Science, 1975, wolumen 187, numer 4180, s. 965–966.

[66] M. Przybylski et al. Integration of qualitative and quantitative spatial data within a semantic map for service robots. W: *Progress in Automation, Robotics and Measuring Techniques. vol. 2 Robotics* Red. R. Szewczyk, C. Zieliński, M. Kaliczyńska, wolumen 351 serii *Advances in Intelligent Systems and Computing*, s. 223–232. Springer 2015.

[67] A. Rabinovich, A. Vedaldi, S. Belongie. Does image segmentation improve object categorization. Raport instytutowy, 2007.

[68] M. Raibert et al. Bigdog, the rough-terrain quadruped robot. IFAC Proceedings Volumes, 2008, wolumen 41, numer 2, s. 10822 – 10825. 17th IFAC World Congress.

[69] A. Ramezani, S.-J. Chung, S. Hutchinson. A biomimetic robotic platform to study flight specializations of bats. Science Robotics, 2017, wolumen 2, numer 3.

[70] X. Ren, L. Bo, D. Fox. RGB-D scene labeling: Features and algorithms. W: IEEE International Conference on Computer Vision and Pattern Recognition. *Proceedings*, 2012, s. 2759–2766.

[71] Z. Ren, E. B. Sudderth. Three-dimensional object detection and layout prediction using clouds of oriented gradients. W: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). *Proceedings*, 2016.

[72] M. Riddoch, G. Humphreys. *Object Recognition, Handbook of Cognitive Neuropsychology*. Hove: Psychology Press 2001.

[73] C. Romero-González et al. 3D spatial pyramid: descriptors generation from point clouds for indoor scene classification. Machine Vision and Applications, 2016, wolumen 27, numer 2, s. 263–273.

[74] E. Rublee et al. Orb: An efficient alternative to sift or surf. W: Proceedings of the 2011 International Conference on Computer Vision. *Proceedings*, Washington, DC, USA, IEEE Computer Society, 2011, ICCV '11, s. 2564–2571.

[75] R. B. Rusu, N. Blodow, M. Beetz. Fast point feature histograms (FPFH) for 3D registration. W: in In Proceedings of the International Conference on Robotics and Automation (ICRA. *Proceedings*, 2009.

[76] R. B. Rusu et al. Close-range scene segmentation and reconstruction of 3D point cloud maps for mobile manipulation in domestic environments. W: International Conference on Intelligent Robots and Systems. *Proceedings*, 2009, s. 1–6.

[77] R. B. Rusu et al. Fast 3D recognition and pose using the viewpoint feature histogram. W: Proceedings of the 23rd IEEE/RSJ International Conference on Intelligent Robots and Systems. *Proceedings*, Taipei, Taiwan, 2010.

[78] R. B. Rusu et al. Detecting and segmenting objects for mobile manipulation. W: Proceedings of IEEE Workshop on Search in 3D and Video (S3DV), held in conjunction with the 12th IEEE International Conference on Computer Vision (ICCV). *Proceedings*, Kyoto, Japan, 2009.

[79] R. B. Rusu et al. Learning informative point classes for the acquisition of object model maps. W: Proceedings of the 10th International Conference on Control, Automation, Robotics and Vision (ICARCV), Hanoi, Vietnam, December 17-20. *Proceedings*, 2008.

[80] R. B. Rusu, S. Cousins. 3D is here: Point Cloud Library (PCL). W: IEEE International Conference on Robotics and Automation (ICRA). *Proceedings*, Shanghai, China, 2011.

[81] R. Schnabel, R. Wahl, R. Klein. Efficient ransac for point-cloud shape detection. Computer Graphics Forum, 2007, wolumen 26, numer 2, s. 214–226.

[82] S. Seok et al. Design principles for energy-efficient legged locomotion and implementation on the mit cheetah robot. IEEE/ASME Transactions on Mechatronics, 2015, wolumen 20, numer 3, s. 1117–1129.

[83] G. Shafer. *The Mathematical Theory of Evidence*. Princeton University Press 1976.

[84] B. Siemiątkowska, B. Harasymowicz-Boggio. Robot Kurier – prototyp inteligentnego wózka transportowego. Logistyka, 2014, numer 4, s. 4084–4089.

[85] N. Silberman et al. Indoor segmentation and support inference from rgbd images. W: European Conference on Computer Vision. *Proceedings*, 2012.

[86] S. Song, S. P. Lichtenberg, J. Xiao. Sun RGB-D: A RGB-D scene understanding benchmark suite. W: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). *Proceedings*, 2015.

[87] S. Song, J. Xiao. Sliding shapes for 3D object detection in depth images. W: European Conference on Computer Vision. *Proceedings* Red. D. Fleet et al. Springer International Publishing, 2014, s. 634–651.

[88] S. Song, J. Xiao. Deep sliding shapes for amodal 3D object detection in RGB-D images. Computing research repository, 2015, wolumen abs/1511.02300.

[89] J. A. Stankovic. Misconceptions about real-time computing: A serious problem for next-generation systems. Computer, 1988, wolumen 21, numer 10, s. 10–19.

[90] M. J. Swain, D. H. Ballard. Color indexing. International Journal of Computer Vision, 1991, wolumen 7, numer 1, s. 11–32.

[91] C. Szegedy et al. Going deeper with convolutions. W: Computer Vision and Pattern Recognition (CVPR). *Proceedings*, 2015.

[92] A. Teichman, S. Thrun. Practical object recognition in autonomous driving and beyond. W: IEEE Workshop on Advanced Robotics and its Social Impacts. *Proceedings*, 2011, s. 35–38.

[93] S. Thorpe, D. Fize, C. Marlot. Speed of processing in the human visual system. Nature, 1996, wolumen 381, s. 520.

[94] A. Varouxaki et al. Inference neglect and ignorance denial. British Journal of Developmental Psychology, 1999, wolumen 17, numer 4, s. 483–499.

[95] A. Vedaldi, A. Zisserman. Sparse kernel approximations for efficient classification and detection. W: Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). *Proceedings*, 2012.

[96] P. Viola, M. Jones. Rapid object detection using a boosted cascade of simple features. W: Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001. *Proceedings*, 2001. wolumen 1, s. I–511–I–518 vol.1.

[97] T. Whelan et al. Kintinuous: Spatially extended kinectfusion. W: RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras. *Proceedings*, Sydney, Australia, 2012.

[98] D. Węgrzyn, L. A. Alexandre. A genetic algorithm-evolved 3D point cloud descriptor. W: Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications. *Proceedings*, New York, NY, USA, Springer-Verlag New York, Inc., 2013, CIARP 2013, s. 92–99.

[99] Z. Wu et al. 3d shapenets: A deep representation for volumetric shapes.. W: CVPR. *Proceedings*. IEEE Computer Society, 2015, s. 1912–1920.

[100] J. Yan et al. Object detection by labeling superpixels. W: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). *Proceedings*, 2015.

[101] H. W. Yoo et al. Real-time plane detection based on depth map from kinect. W: Proceedings of the 44th Internationel Symposium on Robotics, IEEE ISR 2013, Seoul, Korea (South), October 24-26, 2013. *Proceedings*, 2013, s. 1–4.

[102] P. Zanuttigh et al. *Time-of-Flight and Structured Light Depth Cameras – Technology and Applications*. Springer 2016.

[103] Z. Zhang. Microsoft kinect sensor and its effect. IEEE MultiMedia, 2012, wolumen 19, numer 2, s. 4–10.

[104] L. Zhaoping. From the optic tectum to the primary visual cortex: migration through evolution of the saliency map for exogenous attentional guidance. Current Opinion in Neurobiology, 2016, wolumen 40, s. 94–102.

[105] M. Z. Zia, M. Stark, K. Schindler. Explicit occlusion modeling for 3D object class representations. W: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). *Proceedings*, 2013, s. 3326–3333.

[106] Aethon. Tug robots in healthcare. http://www.aethon.com/tug/tughealthcare. (Dostęp 03/14/2017).

[107] Bogdan Harasymowicz-Boggio and Łukasz Chechliński and Daniel Koguciuk. Heuros 2. https://bitbucket.org/rrgwut/heuros2. (Dostęp 04/25/2017).

[108] Bogdan Harasymowicz-Boggio and Łukasz Chechliński and Daniel Koguciuk. Heuros. https://bitbucket.org/rrgwut/heuros. (Dostęp 04/25/2017).

[109] Kent State University. Pearson correlation. http://libguides.library.kent.edu/SPSS/PearsonCorr. (Dostęp 05/29/2017).

[110] Kinect with corrected depth projection. https://vvvv.org/blog/kinect-with-corrected-depth-projection. (Dostęp 08/30/2017).

[111] Pobonline. Automation in point cloud processing: The bar moves up. www.pobonline.com/blogs/23-geodatapoint-blog/post/100664-automation-in-point-cloud-processing-the-bar-moves-up. (Dostęp 08/30/2017).

[112] Rethink Robotics. Baxter collaborative robots for industrial automation. http://www.rethinkrobotics.com/baxter. (Dostęp 03/14/2017).

[113] Shuran Song. Sliding shapes for 3D object detection in depth images. http://videolectures.net/eccv2014_song_depth_images/. (Dostęp 03/21/2017).

[114] Techopedia. Real-time data processing. https://www.techopedia.com/definition/31742/real-time-data-processing. (Dostęp 03/26/2017).

[115] Tesla. Autopilot. https://www.tesla.com/autopilot. (Dostęp 03/14/2017).