

Politechnika Warszawska

WYDZIAŁ ELEKTRONIKI
I TECHNIK INFORMACYJNYCH



Instytut Automatyki i Informatyki Stosowanej

Praca dyplomowa magisterska

na kierunku Automatyka i Robotyka

Stolik wyposażony w czucie taktylne w zadaniach robota asystującego

Tomasz Indeka

Numer albumu 293457

promotor
dr inż. Tomasz Winiarski

WARSZAWA 2024

Stolik wyposażony w czucie taktylne w zadaniach robota asystującego

Streszczenie.

Niniejsza praca stanowi kontynuację badań prowadzonych w trakcie mojej pracy inżynierskiej. Celem pracy magisterskiej była modyfikacja wykonanego przeze mnie rozwiązania w zakresie matrycowych czujników taktylnych, tak aby wykorzystać je na stoliku umieszczonym na robocie. Wykonany stolik ma za zadanie w jak najszerzym stopniu rozpoznawać co i kiedy zostało na nim położone.

Wykonany przeze mnie inteligentny stolik będzie umieszczony i testowany na robocie TIAGo produkowanego przez PAL Robotics. Sam robot działa na systemie ROS, dlatego projektowane przeze mnie rozwiązanie musi być z tym systemem kompatybilne.

Wykonane badania obejmowały ponowne badanie odpowiedzi pojedynczego czujnika taktylnego oraz sposób kompensacji szumów pomiarowych w ramach całego inteligentnego stolika. Opisany został też wykorzystany sprzęt, wraz z tym, co zostało zmienione w stosunku do pracy inżynierskiej. Dodatkowo, po raz kolejny przeprowadziłem przegląd rozwiązań, tylko tym razem bardziej skupiłem się on na alternatywnych rozwiązaniach dostępnych na rynku i zmianach od czasu poprzedniego przeglądu literatury. Na ostatnim etapie został przeprowadzony scenariusz testowy, wykonany na robocie w rzeczywistym środowisku.

Słowa kluczowe: TIAGo, ROS, Velostat, czujnik taktylny, inteligentny stolik, robot asystujący

Tactile sensing table in the tasks of an assistive robot

Abstract.

This thesis is a continuation of the research I have done during my bachelor's degree. The purpose of the master's thesis was to modify a matrix of tactile sensors I had made, in order to use it on a table placed on a robot. The table has been designed to recognize as much as possible about what and when is placed on it.

The Smart Table I made will be placed and tested on the TIAGo robot produced by PAL Robotics. The robot itself runs on ROS, so the solution I design has to be compatible with this system.

The performed research included a re-examination of the response of a single sensor field and how to compensate the noise within the smart table. It also describes the hardware I used, including what I changed from my bachelor's degree work. In addition, I once again did a literature review but this time it was focused more on commercially available alternatives and changes since my previous literature review. As a final step, I made a test scenario, performed on a robot in a real environment.

Keywords: TIAGo, ROS, Velostat, tactile sensor, smart table, assistive robot

Spis treści

1. Wprowadzenie	7
1.1. Cel pracy	8
1.2. Zakres pracy	9
1.3. Definicja robota asystującego	11
1.4. Definicja czujnika taktylnego	12
1.5. Struktura pracy	12
2. Przegląd istniejących rozwiązań	14
2.1. Badania na uczelniach	14
2.1.1. Trendy w ostatnich latach	14
2.1.2. Badania obejmujące pracę z Velostatem	16
2.2. Wykorzystanie sztucznej skóry na rynku	17
2.2.1. Novel	18
2.2.2. Komercyjnie dostępne czujniki sztucznej skóry	18
3. Wykorzystane narzędzia i materiały	20
3.1. Robot TIAGo	20
3.1.1. Specyfikacja techniczna robota TIAGo	20
3.1.2. Modyfikacje robota wykonane w laboratorium	21
3.2. Materiał wrażliwy na nacisk - Velostat	23
3.3. Mikrokontroler STM32	23
3.4. Python	24
3.5. ROS	24
3.6. MeROS i Visual Paradigm	25
4. Budowa stolika z wbudowanym czujnikiem taktylnym	26
4.1. Budowa mechaniczna stolika	26
4.2. Sterownik czujnika taktylnego	27
4.3. Integracja mechaniczna stolika z robotem	33
5. Przeprowadzone badania i testy Velostatu	34
5.1. Badanie zależności pomiędzy siłą nacisku a rezystancją	34
5.1.1. Przeprowadzenie badań nad Velostatem i zależnością pomiędzy siłą nacisku a jego rezystancją	34
5.1.2. Porównanie otrzymanych wyników z wynikami otrzymanymi przez innych badaczy	38
5.2. Kompensacja szumów czujnika taktylnego	41
5.2.1. Zależność od powierzchni styku z czujnikiem	46
5.2.2. Aproxymacja pomiarów do wykorzystania w pomiarach wagi przedmiotu	47
6. Scenariusze wykorzystania inteligentnego stolika	49

6.1. Scenariusz dostarczenia herbaty	50
6.2. Scenariusz odwiezienia pustego naczynia	51
6.3. Pokrycie przypadków wykorzystania inteligentnego stolika przez scenariusze	53
7. Implementacja węzłów ROS wykonujących zaplanowane scenariusze	54
7.1. Struktura projektu	54
7.2. Struktura intrasystemu inteligentnego stolika	56
7.2.1. Możliwe statusy węzła	60
7.2.2. Kalibracja czujnika taktylnego	61
7.2.3. Przetwarzanie informacji ze stolika	61
7.2.4. Wykrywanie obecności obiektu	63
7.2.5. Wykrywanie położenia obiektu	63
7.2.6. Szacowanie wagi obiektu	65
7.2.7. Rozpoznawanie położonego przedmiotu	66
7.2.8. Szacowanie wagi obiektu za pomocą sieci neuronowej	71
7.3. Struktura intrasystemu zarządzającego scenariuszami	73
7.4. Problemy integracji węzłów z systemem ROS robota	79
8. Wykonanie zaplanowanych scenariuszy	80
8.1. Środowisko wykorzystane w scenariuszach	80
8.2. Wykonanie zaplanowanych scenariuszy	81
9. Wnioski	84
9.1. Podsumowanie założonych przypadków użycia inteligentnego stolika	84
9.2. Wnioski dotyczące technologii wykonania czujnika taktylnego	85
9.3. Wnioski dotyczące wykorzystanych metod detekcji parametrów położonego obiektu	86
9.4. Wnioski dotyczące potencjału na rynku konsumenckim	87
9.5. Wnioski dotyczące możliwości wykorzystania czujników taktylnych w robotyce	87
9.6. Możliwości rozwoju projektu inteligentnego stolika	88
Bibliografia	89
Spis rysunków	97
Spis tabel	98

1. Wprowadzenie

W dzisiejszym świecie roboty stają się coraz powszechniejsze. Coraz większy staje się ich udział w naszym codziennym życiu. Część z nich ma za zadanie pomóc nam w codziennych obowiązkach (roboty sprzątające), wykonać ciężką pracę (roboty przemysłowe, rolnicze), transportować przedmioty (roboty magazynowe) czy nawet przetransportować nas z miejsca na miejsce (autonomiczne samochody). Bądź co bądź robotów w naszym życiu przybywa i to w każdym aspekcie naszego życia.

Jednym z typów robotów są roboty asystujące, których jednym z zadań jest pomoc człowiekowi w jego codziennych czynnościach. Roboty te, z uwagi na dzielenie środowiska pracy z człowiekiem, muszą posiadać wiele czujników, aby nie zaszkodzić człowiekowi, sobie ani środowisku. Są to podstawowe zasady, które każdy robot powinien przestrzegać ponad wszystkim [1].

Jednym z czujników, jakie robot może wykorzystywać do odbierania bodźców z otoczenia, są czujniki taktyльne. Zagadnienia poświęcone tym czujnikom zgłębiałem już podczas mojej pracy inżynierskiej [2]. Teraz postanowiłem kontynuować ten temat, tylko w innym zastosowaniu. Jak się w praktyce okazało, sztuczna skóra na obwodzie robota miała ograniczone zastosowania praktyczne. Finalny prototyp sztucznej skóry na obwodzie robota, który wykonałem na potrzeby pracy inżynierskiej, jest widoczny na rysunku 1.1 [2], [3].



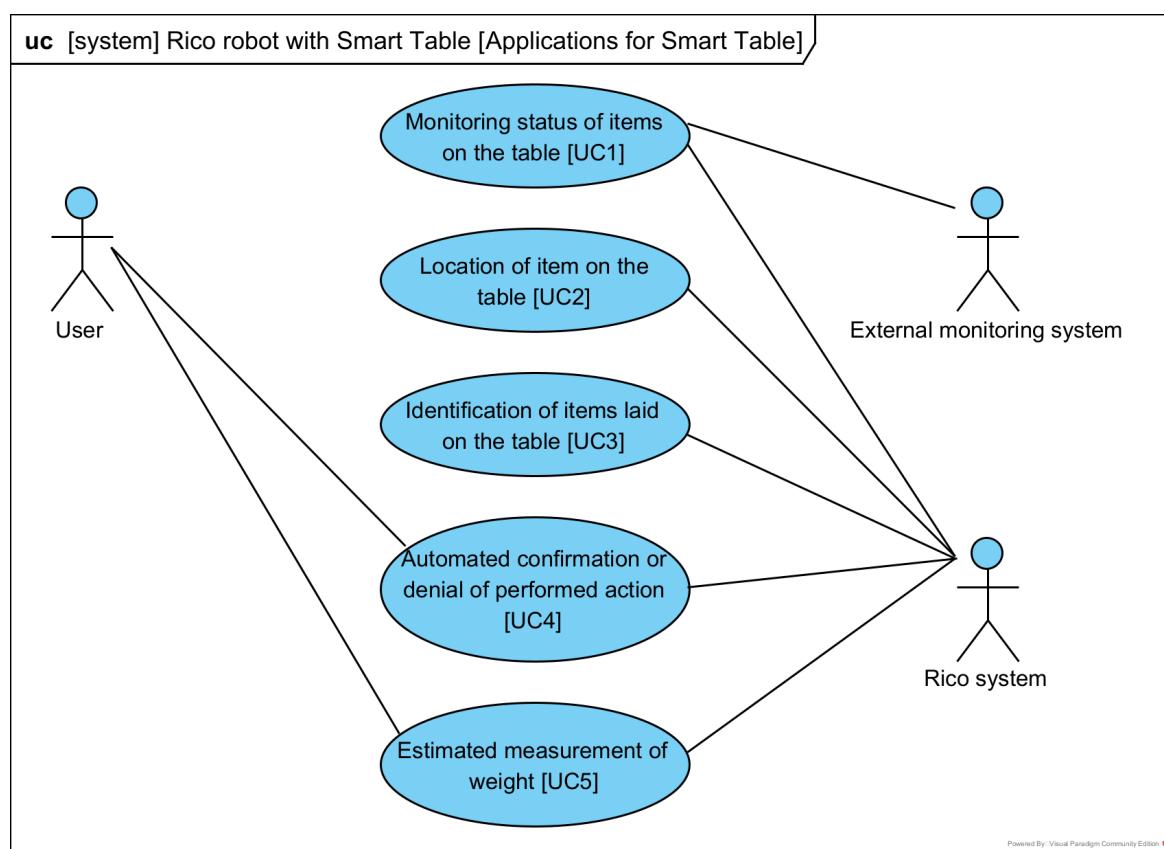
Rysunek 1.1. Prototyp sztucznej skóry wykonany na potrzeby pracy inżynierskiej [2]

Zanim jednak przystapię do dalszej części warto jasno sprecyzować cele tej pracy, jej zakres, jak również to czym są czujniki taktyльne (dotyku) oraz roboty asystujące. Ta praca opiera się na nich więc warto sprecyzować czym dokładnie są.

1.1. Cel pracy

Praca ta nie skupia się na wnikliwym pogłębianiu wiedzy w zakresie jednej dziedziny - czucia taktylnego przez roboty, ale na wykorzystaniu wiedzy z wielu różnych dziedzin i ich połączeniu, w celu stworzenia nowego rozwiązania. Doprecyzowując, celem pracy było rozszerzenie będącego już w laboratorium robota asystującego TIAGo, zwanego także Rico, o inteligentny stolik. Inteligentny stolik zastąpił zwykły stolik, rozszerzając jego możliwości o zbieranie danych o położonych na stoliku przedmiotach.

Przed projektowanym inteligentnym stolikiem zostały także postawione zadania, które powinien on spełniać. Zostały również określone szczegółowe przypadki jego użycia, które zostały zebrane na rysunku 1.2.



Rysunek 1.2. Diagram przypadków użycia inteligentnego stolika i danych przez niego zbieranych

Wyszczególnione zostało również 5 konkretnych przykładów wykorzystania inteligentnego stolika na robocie Rico:

- UC1 - monitorowanie statusu przedmiotów położonych na stoliku,
- UC2 - lokalizacja obiektów na stoliku,
- UC3 - identyfikacja przedmiotów położonych na stoliku,
- UC4 - zautomatyzowane potwierdzenie bądź zaprzeczenie wykonanej przez człowieka akcji,
- UC5 - szacunkowa waga położonego przedmiotu.

Na diagramie tym widać 3 aktorów: użytkownika, zewnętrzny system monitorujący pracę robota oraz samego robota. Użytkownikiem systemu jest człowiek, który również bezpośrednio będzie korzystał z inteligentnego stolika. Najbardziej będzie zauważał on jego działanie przy automatyzacji procesu potwierdzenia wykonanej przez człowieka akcji. Kolejnym ważnym aktorem, który zawsze musi występować razem z inteligentnym stolikiem, jest robot Rico. Jest on jednym z głównych odbiorców informacji przetwarzanych przez inteligentny stolik. Ostatnim elementem jest układ monitorowania systemu, który jest aktorem, który nie musi występować. Aktor ten umożliwia zapisywanie stanu robota i monitorowanie jego pracy zdalnie.

1.2. Zakres pracy

Zakres prowadzonych przeze mnie prac był szeroki i obejmował wiele różnych dziedzin. Zobrazowanie zróżnicowania tych prac zostało przedstawione na rysunku 1.3. Poszczególne prace obejmowały:

- ponowny przegląd literatury,
- projekt oraz budowę fizycznego prototypu stolika,
- wykonanie elektroniki sterownika czujnika taktylnego,
- implementację węzła, w systemie ROS, obsługującego stolik,
- integrację robota Rico z inteligentnym stolikiem,
- implementację przykładowego węzła wykorzystującego dane z inteligentnego stolika,
- przeprowadzenie scenariusza faktycznego użycia inteligentnego stolika na robocie.

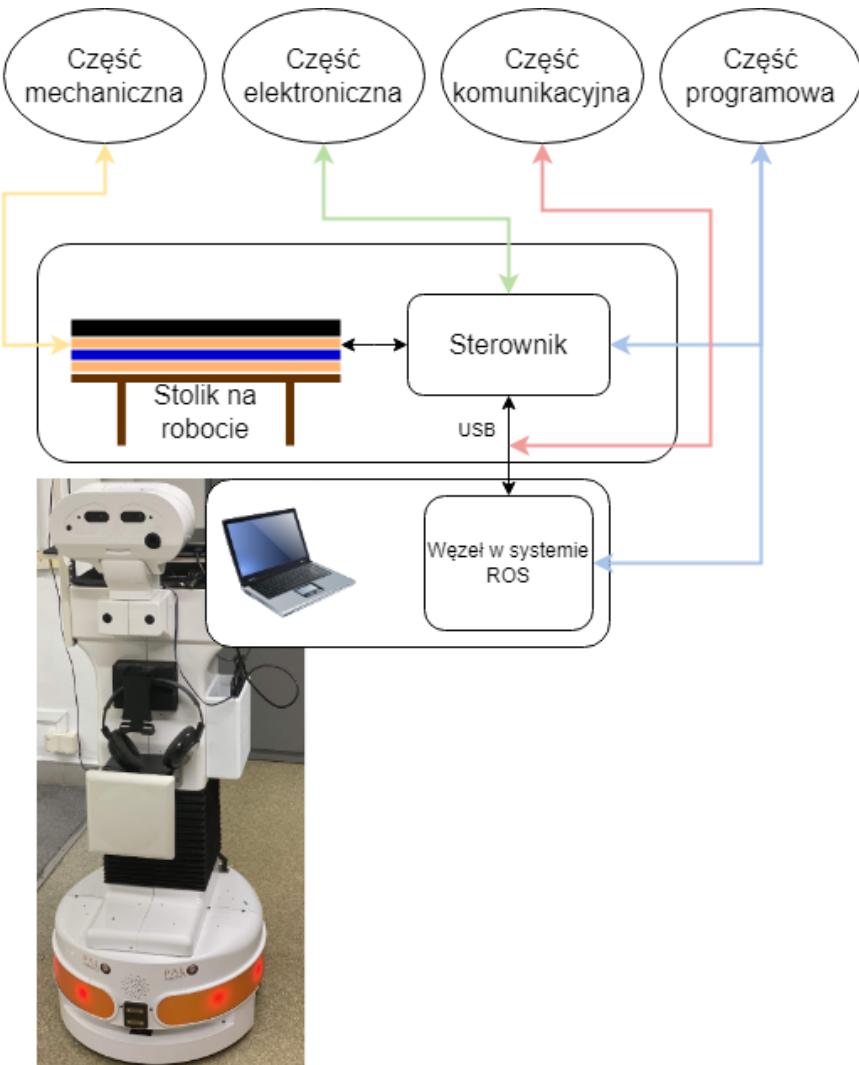
Wykonany przeze mnie ponowny przegląd literatury skupił się głównie na postępie jaki został wykonany w środowisku naukowym w ciągu ostatnich kilku lat, od czasu obrony pracy inżynierskiej. Przegląd ten dodatkowo objął również rozeznanie rynku pod kątem czujników taktylnych dostępnych komercyjnie oraz zakresem oferowanych przez producentów rozwiązań.

Budowa stolika została wykonana od podstaw, na wzór aktualnie znajdującego się na robocie stolika. Docelowo nowo zbudowany stolik miał na celu całkowicie zastąpić poprzedni.

Projekt elektroniki zakładał poprawienie układu zaprojektowanego na potrzeby pracy inżynierskiej. Wyeliminowane zostały wady, zbędne komponenty oraz całość została lepiej zorganizowana i przemyślana.

Węzeł w systemie ROS ma za zadanie odczytywać regularnie dane przesypane przez czujnik taktylny oraz przetwarzać je. Węzeł ten powinien łączyć się z robotem z wykorzystaniem odpowiednich tematów ROSowych. Tematy te pozwalają zarówno na podstawową kontrolę węzła, jak również transport przetworzonych danych z czujnika taktylnego do systemu robota. Wykonanie węzła ROSowego jest sporym kamieniem milowym tej pracy i jest jedną z kluczowych wartości tej pracy. Podczas projektowania i implementacji węzła

1. Wprowadzenie



Rysunek 1.3. Plan rozszerzenia robota Rico o inteligentny stolik

ROSowego nastąpiły różne przeciwności i obszary do badań, które zostały pogłębione w trakcie mojej pracy.

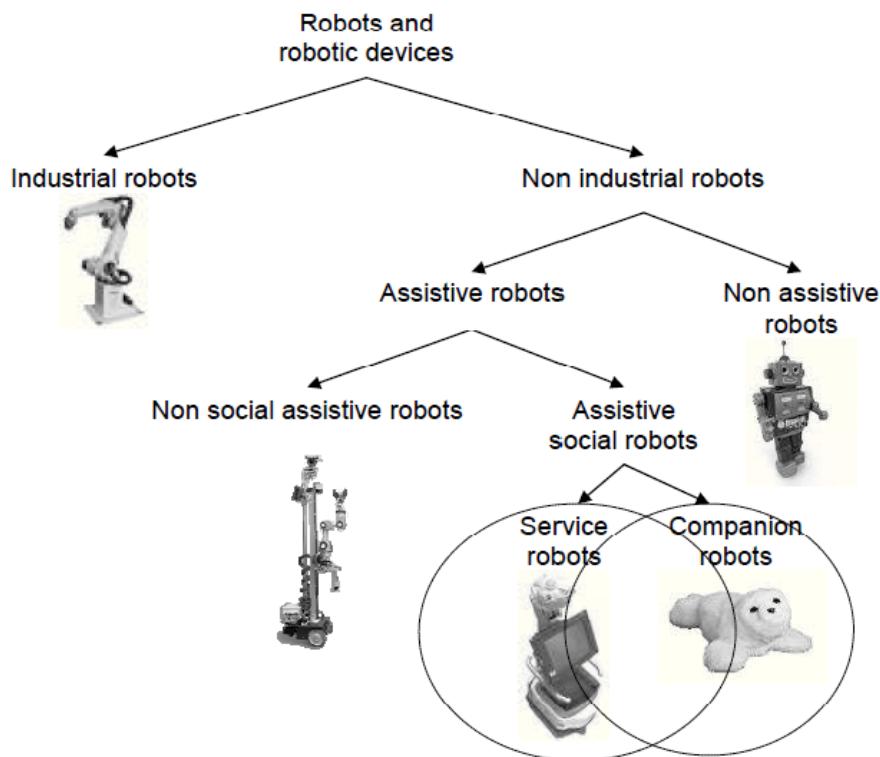
Integracja stolika pokrytego czujnikami taktylnymi również jest znacznym osiągnięciem tej pracy. Pokazuje ona, że projekt nie jest tylko teoretycznym rozważaniem, ale może również z powodzeniem zostać wykorzystany w praktyce.

Jednym ze sposobów prezentacji działania inteligentnego stolika jest napisanie i wykorzystanie przykładowego węzła, który wykorzystuje sygnały wysyłane przez inteligentny stolik do robota. Celem jest prezentacja w jaki sposób inne osoby, w przyszłości, mogą wykorzystać stolik w swoich zastosowaniach i dalej go rozwijać.

Finalnie, wisienką na torcie w przypadku zakresu prowadzonych przeze mnie prac, jest wykorzystanie inteligentnego stolika w praktycznym scenariuszu, gdzie robot oprócz zwykłego testu działania stolika będzie również współpracował z innymi podsystemami. Celem jest pokazanie w pełni zintegrowanego z robotem rozwiązania, które jest użyteczne i w pewnym stopniu automatyzuje pracę robota.

1.3. Definicja robota asystującego

Robot asystujący jest tylko jednym z wielu różnych rodzajów robotów. Jeden z głównych podziałów robotów został przedstawiony na rysunku 1.4. Warto zauważyć, że na rysunku tym zostały wyszczególnione roboty asystujące mające kontakt z człowiekiem oraz te, które tego kontaktu nie mają. W tej pracy pojęcie robota asystującego będzie się odnosić tylko do robotów przystosowanych do współistnienia z człowiekiem [4].



Rysunek 1.4. Podział robotów na podkategorie zaproponowany przez Ioana Orha [4]

Robotem asystującym możemy zatem nazwać każdego robota, który odbiera i przetwarza informację ze środowiska, i jest w stanie podejmować działania, które służą człowiekowi. Taka definicja jest bardzo ogólna, ale jej podstawową częścią jest fakt, że obejmuje roboty, które współdziałają z człowiekiem w jednym środowisku i wykonują polecenia tego człowieka. Do głównych użytkowników takich robotów można zaliczyć między innymi osoby starsze czy też osoby chore lub o ograniczonych możliwościach ruchu [5], [6].

Do głównych zadań robotów asystujących można zaliczyć:

- wykonywanie poleceń zleconych przez człowieka,
- swobodna i wygodna komunikacja z użytkownikiem,
- dotrzymywanie osobie towarzystwa,
- wyrażanie i rozpoznawanie emocji,
- swobodne i autonomiczne poruszanie się w zakresie wykonywanych obowiązków,
- posiadanie własnej osobowości i możliwość odzwierciedlania zachowań społecznych [4], [5].

1.4. Definicja czujnika taktylnego

Czujnik taktylny to czujnik, który wykrywa dotyk oraz siłę oddziałującą na niego. Jest to jeden z podstawowych czujników do wykrywania nacisku. Jest również podstawą do budowy bardziej skomplikowanych układów, jak na przykład sztuczna skóra. Fizyczna zasada działania czujnika taktylnego może być bardzo różna. Istnieją czujniki rezystancyjne, pojemnościowe, piezoelektryczne, indukcyjne, optyczne, ale dziedzina ta nie ogranicza się tylko do tych zasad działania [7], [8].

Czujniki taktyльne mogą również spełniać i wspomagać wiele różnych zadań robotycznych, takich jak:

- kontrola siły ścisłu przedmiotów przez manipulator,
- wykrywanie przesunięć i uśлизgów,
- estymacja miejsca kontaktu,
- rozpoznanie obiektów,
- rozpoznanie cech obiektu,
- wykrywanie położenia stawów robota,
- wykorzystanie jako wyłącznik krańcowy lub bezpieczeństwa,
- mierzenie nacisku w systemach rozproszonych [9].

Czujniki taktyльne rzadko występują samodzielnie. Zazwyczaj są one częścią większego systemu czy to robota, który wykrywa w ten sposób kontakt z otoczeniem, czy też większych systemów mapujących nacisk na powierzchnię. Czujniki taktyльne są w tych zastosowaniach środkiem do celu, a nie celem samym w sobie.

1.5. Struktura pracy

Niniejsza praca składa się z 9 rozdziałów. Rozdział 1 skupia się na wprowadzeniu czytelnika do tematu. Przedstawia cele i założenia niniejszej pracy. Wyjaśnia również czym są robot asystujący oraz czujnik taktylny.

Rozdział 2 skupia się na przeglądzie literatury. Opisuję tutaj to co zmieniło się w zakresie badań nad czujnikami taktylnymi od czasu, kiedy wykonywałem przegląd literatury na potrzeby pracy inżynierskiej. Wykonałem również przegląd firm, które oferują sprzedaż czujników taktylnych na rynku konsumenckim i to w jakich zastosowaniach są te czujniki używane w przemyśle.

Rozdział 3 zbiera informacje o wykorzystywanych w trakcie prac nad niniejszą pracą narzędziach. Przede wszystkim opisuje robota TIAGO, na który projektowany był inteligentny stolik i jego aktualną, laboratoryjną konfigurację.

Rozdział 4 opisuje budowę inteligentnego stolika – od koncepcji do jego finalnej wersji. Pisywana jest w tym rozdziale jego budowa mechaniczna, elektroniczna i programistyczna. Tutaj została przedstawiona również metoda integracji stolika z robotem.

Rozdział 5 przedstawia testy na materiale Velostat, które przeprowadziłem na potrzeby niniejszej pracy. Pierwsze badanie polegało na wyznaczeniu zależności siła nacisku –

rezystancja Velostatu, drugie natomiast na wybraniu najlepszej metody kompensacji szumów pomiarowych czujnika.

Rozdział 6 prezentuje pierwszy, koncepcyjny podział systemu robota na intrasystemy. W rozdziale tym przedstawione są dwa też scenariusze wykorzystania inteligenckiego stolika w praktycznym wykorzystaniu. Podsumowane jest również jakie wykorzystania stolika są testowane przez scenariusze.

Rozdział 7 przedstawia projekt oraz wykonanie intrasystemów inteligenckiego stolika oraz managera scenariuszy. Opisane są tutaj dokładnie węzły, które działają w ramach tych intrasystemów i to jak komunikują się z pozostałą częścią robota. Opisane są również wszystkie informacje otrzymywane z czujnika taktylnego oraz sposób ich uzyskania, przetwarzania i ekstrakcji danych.

Rozdział 8 opisuje fizyczne wykonanie scenariuszy w laboratorium na rzeczywistym robocie. Opisany jest szczegółowy ich przebieg, testowane systemy oraz wstępne wnioski z działania systemu.

Rozdział 9 podsumowuje pracę. Są w nim zawarte wnioski na kilka aspektów poruszanych w tej pracy. Skonfrontowane zostały przypadki wykorzystania postawione na początku pracy. Przedstawione są moje doświadczenia wynikające z pracy z zaprojektowanym czujnikiem oraz zaimplementowanym oprogramowaniem. Przedstawione są również moje wnioski na temat dalszych możliwości wykorzystania tego rozwiązania.

2. Przegląd istniejących rozwiązań

Przegląd istniejących rozwiązań znalazł się również w mojej poprzedniej pracy dyplomowej, ale z uwagi na czas jaki minął od jej zakończenia, postanowiłem przegląd ten wykonać jeszcze raz. Nie będę się tutaj jednak skupiał jeszcze raz na tych samych przykładach tylko dokonam przeglądu literatury z naciskiem na ostatnie lata, jak również przejrzę rozwiązania istniejące na rynku konsumenckim i przemysłowym. Przegląd istniejących rozwiązań nie jest jednak podstawowym celem tej pracy, dlatego też nie zostały przedstawione wszystkie istniejące na rynku i w laboratoriach rozwiązania, a tylko wybrane koncepcje.

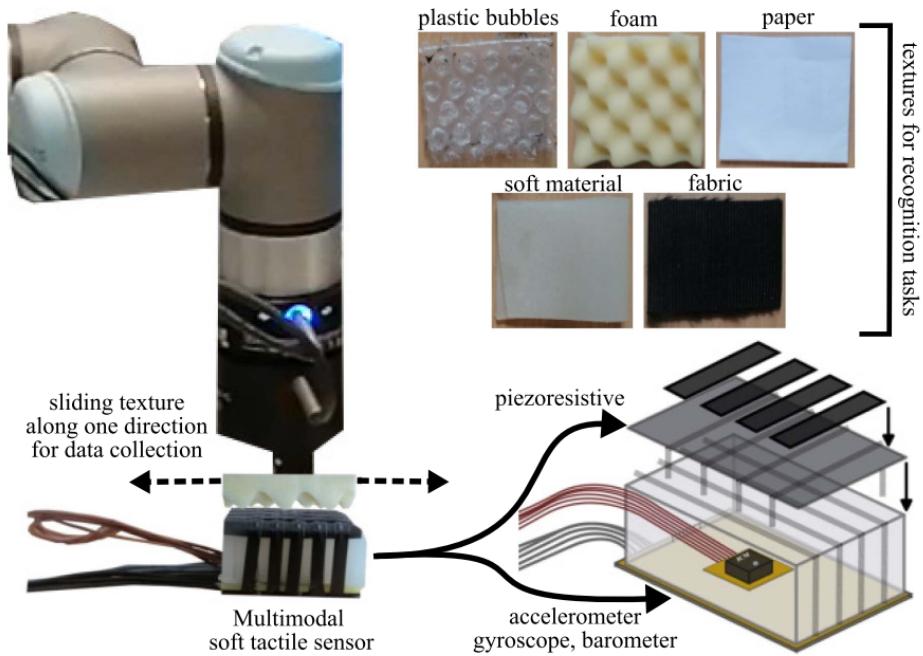
2.1. Badania na uczelniach

2.1.1. Trendy w ostatnich latach

W ostatnich latach nastąpił duży wzrost zainteresowania tematem sztucznej skóry, jak i samym materiałem Velostat. Najlepszym dowodem na potwierdzenie tej tezy jest fakt, że w serwisie IEEE Xplore, po wyszukaniu słowa „Velostat” we wszystkich źródłach, prace z lat 2021 – 2023 stanowią ~ 61% wszystkich wyświetlonych pozycji (stan na 18.10.23r). Te dane świadczą o tym, że od mojego ostatniego przeglądu literatury na potrzeby pracy inżynierskiej w badanej dziedzinie powstało sporo nowych artykułów, które warto wziąć pod uwagę podczas prowadzenia dalszych prac. Nie wszystkie z rozwiązań, które jako warstwę sensoryczną wykorzystują Velostat zostały jednak uwzględnione w tym wykazaniu, co zauważylem dopiero na późniejszym etapie przeglądu prac. Tak duży wzrost zainteresowania tym materiałem może być spowodowany przez jego stosunkowo niewielką cenę, jak również dużą dostępność [10].

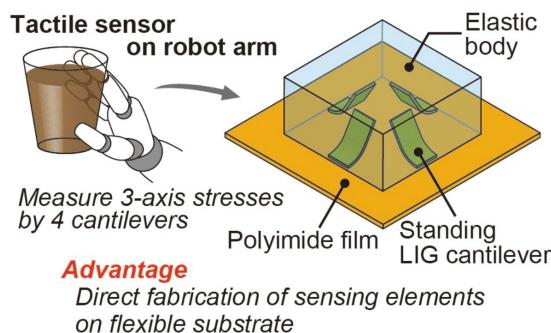
Wiele zespołów badaczy na całym świecie zajmowało się badaniami w zakresie czujników taktylnych. Widoczna jest duża różnorodność zarówno w konstrukcji samego czujnika taktylnego, w wykorzystywanych do badań robotów oraz zastosowaniach, gdzie czujniki są wykorzystywane. Coraz bardziej widoczne jest zainteresowanie tematem przeplatania się robotyki z codziennym życiem człowieka. Skutkuje to dużą liczbą prób zintegrowania czujnika z odzieżą lub innymi codziennie wykorzystywanyimi przedmiotami. Zwiększa się również liczba robotów w otoczeniu człowieka, systemów autonomicznych, systemów IoT, zdalnej kontroli i zdalnego monitorowania. W każdym z tych zastosowań można w pewnym stopniu wykorzystać czujniki taktyльne. Dlatego też wiele zespołów chce taki czujnik stworzyć i wykorzystać w produkcji komercyjnej.

Prezentowane rozwiązania są zarówno bardzo zaawansowane technologicznie, wymagające specjalistycznego laboratorium, jak i możliwe do wykonania praktycznie przez każdego, w warunkach domowych. Większość publikowanych prac opiera się na tworzeniu własnych wariacji czujników taktylnych oraz na analizie ich działania i wykorzystania. Głównymi kierunkami badań w zakresie czujników taktylnych są:



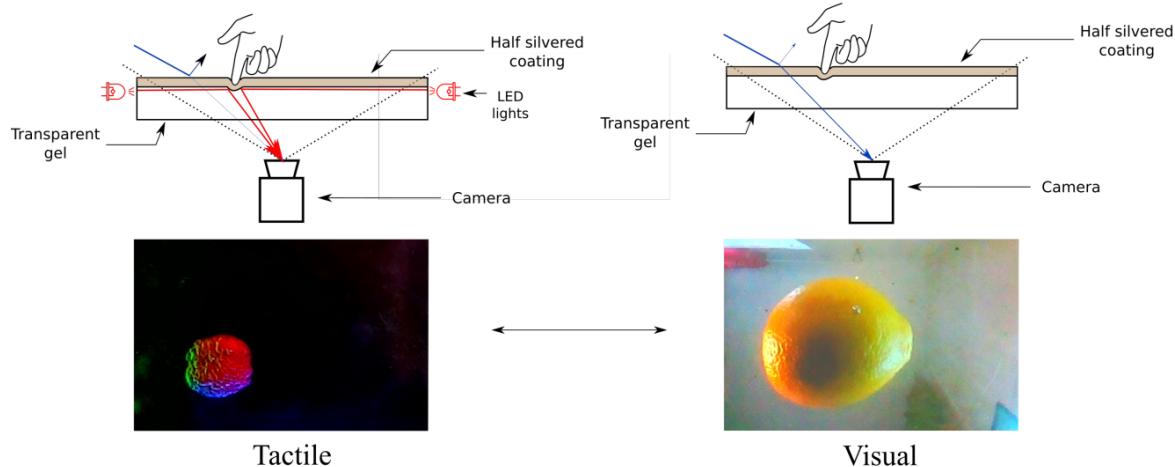
Rysunek 2.1. Przykładowe podejście do rozpoznawania faktury materiału [11]

- projekty do noszenia, umieszczane w ubraniach i gadżetach [13],
- rozwiązania giętkie, do stosowania na krzywych powierzchniach [14], [15],
- rozpoznawanie materiałów i ich tekstury (przykład na rysunku 2.1) [11], [16], [17],
- zastosowania medyczne, zwiększenie możliwości monitorowania czynności organizmu [18],
- wykrywanie i mierzenie siły ścinającej (przykład na rysunku 2.2) [12], [19],
- zwiększenie rozdzielczości i minimalizacja sensora [20],
- wykorzystanie systemów wizyjnych i opartych na kamerach (przykład na rysunku 2.3) [21]–[23],
- wykorzystanie sieci neuronowych do rozpoznawania przedmiotów i kształtów (przykład na rysunku 2.4) [24].

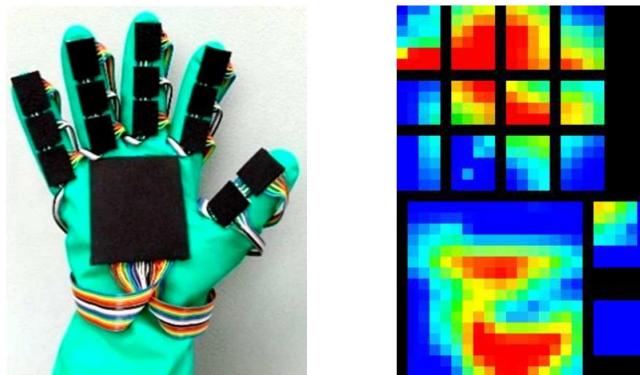


Rysunek 2.2. Przykładowe podejście do pomiaru siły tnącej [12]

2. Przegląd istniejących rozwiązań



Rysunek 2.3. Przykładowe podejście do wykrywania i rozpoznawania przedmiotów wspomaganego przez system wizyjny [23]



Rysunek 2.4. Przykładowe podejście do rozpoznawania przedmiotów chwyconych dlonią wykorzystując przygotowaną rękawicę [24]

Nie są to jednak wszystkie obszary, w których badacze rozwijają swoje rozwiązania, ponieważ jest ich zdecydowanie więcej. Wymienione przykłady są jednak najczęściej i najmocniej poruszane w tej branży.

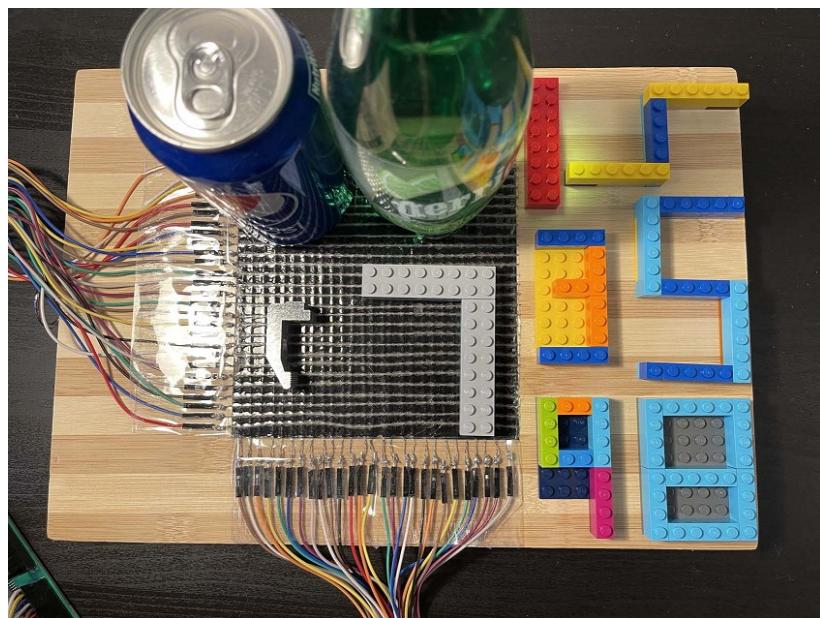
2.1.2. Badania obejmujące pracę z Velostatem

Prace nad sztuczną skórą i czujnikami taktylnymi prowadzone na uniwersytetach całego świata obejmują wiele obszarów. Te, które opierają się na właściwościach materiału Velostat skupiają się głównie wokół:

- wykrywania i rozpoznawania rozkładu nacisku stóp, umieszczanie czujników w obuwiu lub na osobnej macie [25]–[28],
- minimalizacji czujników o wysokiej rozdzielczości, które nastawione są na rozpoznawanie przedmiotów bądź ruchów [29], [30],
- czujników elastycznych i możliwości ich wykorzystania na sferycznych powierzchniach [31],

- czujników, które można nosić lub zintegrować z ubraniem [32], [33].

Jedna z przeglądanych prac zawiera częściowo elementy, nad którymi pracuję w ramach swojej pracy magisterskiej. Mam na myśli pracę Yuana Liangqi [29], który w swojej pracy wykorzystuje folię Velostat do budowy czujnika rozpoznającego przedmioty. Testowane przez niego rozwiązanie składa się z 729 pól taktylnych (27×27) rozmieszczonych równomiernie na kwadracie o boku 140mm. Pula testowanych obiektów zawierała 10 przedmiotów, pośród których znajdowały się m.in.: puszka, woda czy cyfry wykonane z LEGO. Yuan w swoich badaniach osiągnął skuteczność rozpoznawania przedmiotów na poziomie 0,9854. Wykonany czujnik wraz z testowanymi przedmiotami został przedstawiony na rysunku 2.5. Jego prace, mimo iż dużo dokładniej skupiały się na problemie rozpoznawania przedmiotów, są dla mnie ważnym odniesieniem w moich testach [29].



Rysunek 2.5. Prototyp czujnika taktylnego wykonanego przez Yuana Liangqi [29]

2.2. Wykorzystanie sztucznej skóry na rynku

Mimo iż temat sztucznej skóry jest badany od wielu lat przez wielu badaczy, na rynku konsumenckim nie znalazło się wiele firm zajmujących się tym tematem. Jednym z powodów jest fakt, iż nie ma ona aż tylu praktycznych zastosowań, a sama sztuczna skóra jest dość droga w produkcji. Dlatego też jest to wciąż zagadnienie głównie akademickie. Mimo to, są firmy które posiadają w swoim portfolio ciekawe rozwiązania opierające się na idei rozproszonych czujników taktylnych.

Sektor, w których działają firmy zajmujące się sztuczną skórą oraz czujnikami taktylnymi obejmują przede wszystkim sektor medyczny, gdzie wykorzystywane są do diagnostowania nieprawidłowości w funkcjonowaniu organizmu człowieka. Poza sektorem medycznym są one wykorzystywane również przez profesjonalnych sportowców, do obser-

2. Przegląd istniejących rozwiązań

wacji własnych ruchów i tego jak je udoskonalić. Istnieje również sektor zastosowań bardzo specjalistycznych w innych dziedzinach, jak na przykład badanie jakości spawów. Niektóre firmy oferują także całkowite przystosowanie produkowanych czujników taktylnych do różnorodnych wymagań konsumenta [34]–[37].

Problemem dla wykorzystywania czujników taktylnych dostępnych na rynku, na potrzeby mojej pracy magisterskiej, jest interfejs, którym dane są przesyłane do komputera. Większość z dostępnych rozwiązań na rynku dostarczane jest z własnym oprogramowaniem do odczytywania danych z czujnika. Nie ustaliłem jednak czy, i w których przypadkach jest możliwość wykorzystać te dane we własnym zakresie i przesłać je do dalszego przetwarzania. Te czujniki, które nie są dostarczane z dedykowanym oprogramowaniem wymagają zazwyczaj wykonania do ich obsługi własnego układu elektronicznego [38], [39].

2.2.1. Novel

Ta niemiecka firma już od wielu lat zajmuje się produkcją i sprzedażą czujników siły, nacisku i dotyku. Oferują oni zarówno czujniki zintegrowane dla specyficznych zastosowań, jak i same czujniki. Jednocześnie wydają się mieć najciekawszą i najbardziej zróżnicowaną ofertę ze wszystkich producentów, których sprawdziłem. Ich rozwiązania są wykorzystywane w wielu branżach i dziedzinach, od profesjonalnych sportów, przez samochody, do zastosowań medycznych i kosmicznych. Przykładowe zastosowania, gdzie wykorzystywane są czujniki firmy Novel obejmują:

- siodło dla konia - pomaga ustalić, czy jeździec ma prawidłową postawę [40],
- podologia - pomaga ustalić deformacje stóp oraz wady postawy [41],
- pomiar rozkładu sił dloni podczas chwytania [42],
- dostosowanie siedzenia kierowcy do panujących warunków,
- precyzyjny system pomiaru siły, który można można nosić,
- pomiar rozkładu sił na dowolnym innym przedmiocie (materac, kij golfowy, itd) [35].

Zaprojektowane przez nich przykładowe rozwiązanie do pomiaru rozkładu nacisku stopy na podłożu jest widoczne na rysunku 2.6 [35].

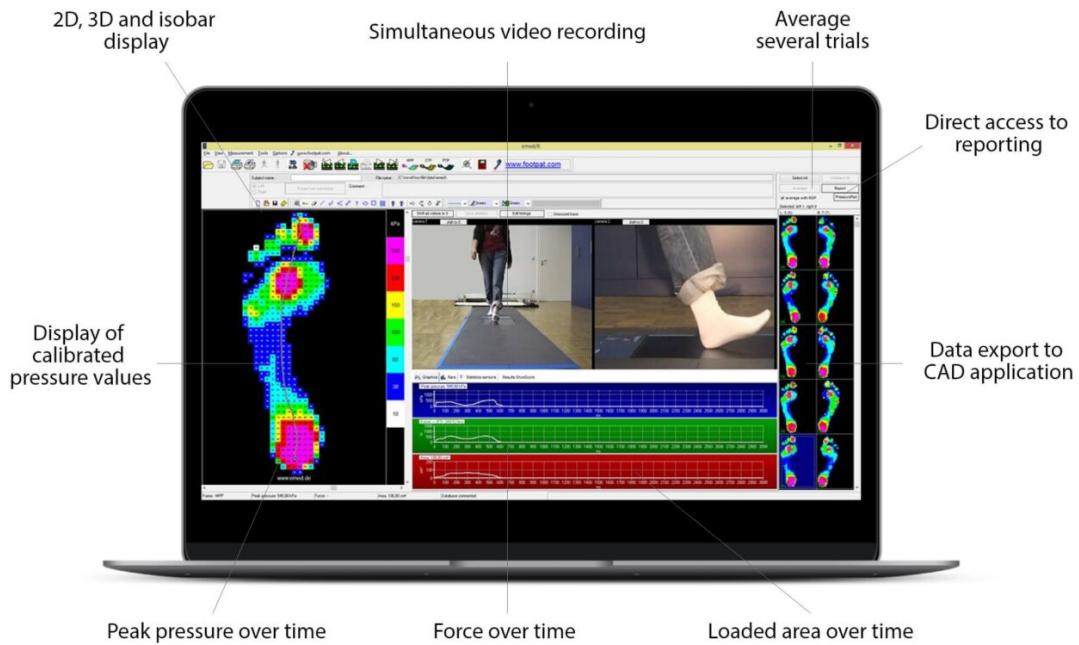
2.2.2. Komercyjnie dostępne czujniki sztucznej skóry

Rynek nie jest przesycony firmami, które zajmują się dystrybucją czujników, które mogą posłużyć jako baza dla budowy sztucznej skóry. Głównie są to firmy, które sprzedają swój produkt jako maty do pomiaru wywieranego ciśnienia. Swoją działalność skupiają wokół sprzedaży czujników w stanie surowym lub przystosowanych dla zastosowań medycznych.

Przykładowi dystrybutorzy czujników taktylnych:

- Kitronyx¹,

¹ Strona internetowa firmy Kitronyx: <https://www.kitronyx.com/>



Rysunek 2.6. Rozwiązanie firmy Novel do badania rozkładu nacisku stopy podczas chodzenia [35]

- Tekscan²,
- Loomia³,
- InnovationLab⁴,
- PPS⁵,
- Taiwan Alpha Electronic⁶,
- medilogic⁷,
- CONTEMLPLAS⁸,
- chiński sklep internetowy⁹.

W znacznej liczbie przypadków problem jest również z wyznaczeniem ceny takiego czujnika. Wielu producentów wykonuje wycenę dopiero po wysłaniu do nich bezpośredniego zapytania ofertowego.

² Strona internetowa firmy Tekscan: <https://www.tekscan.com/>

³ Strona internetowa firmy Loomia: <https://www.loomia.com/>

⁴ Strona internetowa firmy InnovationLab: <https://www.innovationlab.de/gedruckte-elektronik/>

⁵ Strona internetowa firmy PPS: <https://pressureprofile.com/>

⁶ Strona internetowa firmy Taiwan Alpha Electronic: <http://www.taiwanalpha.com/en>

⁷ Strona internetowa firmy medilogic: <https://medilogic.com/en/home/>

⁸ Strona internetowa firmy CONTEMLPLAS: <https://contemplas.com/en/motion-analysis/>

⁹ Strona internetowa chińskiego sklepu internetowego:
<https://www.aliexpress.com/item/1005003091157680.html>

3. Wykorzystane narzędzia i materiały

3.1. Robot TIAGo

3.1.1. Specyfikacja techniczna robota TIAGo

Robot, na który przeznaczony został zaprojektowany stolik to TIAGo - robot zaprojektowany i wyprodukowany przez hiszpańską firmę PAL Robotics. Robot ten (widoczny na rysunku 3.1) jest robotem asystującym, który posiada:

- platformę jezdnią (w moim przypadku robot o napędzie różnicowym klasy (2,1)),
- wbudowane czujniki odległości oraz Lidar,
- kamerę RGB-D zamocowaną na głowie robota,
- mikrofon stereo oraz głośnik,
- dodatkowe porty (USB, Ethernet, HDMI) dostępne dla użytkownika,
- opcjonalne manipulatory (w moim przypadku niedostępne),
- opcjonalny panel dotykowy (w moim przypadku niedostępny),
- komunikację po Bluetooth oraz Wi-Fi,
- ruchomy tułów oraz głowę [43].



Rysunek 3.1. Robot TIAGo [43]

Robot ten wyposażony jest także we wspierany przez producenta interfejs w systemie ROS, jak również sam działa na tym systemie. To znacznie ułatwia rozbudowywanie

robot o dodatkowe moduły i testowanie rozwiązań we własnym środowisku. Dzięki wymienionym cechom robot ten jest bardzo dobrą platformą laboratoryjną, ponieważ pozwala na bardzo dużą swobodę pracy przy nim i implementowania modyfikacji [44].

3.1.2. Modyfikacje robota wykonane w laboratorium

W laboratorium nie jestem jedyną osobą pracującą nad rozwijaniem modyfikacji dla robota TIAGo. W momencie pisania tej pracy robot został już znacząco zmodyfikowany, w porównaniu do wersji pierwotnej, i na ten moment posiada wyszczególnione modyfikacje:

- dodatkowy komputer na robocie,
- dodatkowy omnikierunkowy mikrofon [45],
- dodatkowy głośnik [45],
- połączenie z usługami rozpoznającymi mowę (Speech to Text) i generującymi głos (Text to Speech) [45], [46],
- połączenie z lokalną siecią Wi-Fi [47],
- kamerę termowizyjną [48],
- dodatkową kamerę głębi [49],
- zmodyfikowany system działania robota [50],
- dodatkowe możliwości sterowania robotem [49],
- sztuczną skórę na obwodzie robota – wykonaną w ramach mojej pracy inżynierskiej [2].

Nie wszystkie z wymienionych modyfikacji są zawsze obecne i zamocowane na robocie, dlatego jego dokładna konfiguracja ulega zmianom. Zdjęcie robota na moment spisywania pracy widoczne jest na rysunku 3.2 [2], [47].

Robot TIAGo znajdujący się w laboratorium otrzymał również swoje imię - Rico, i tak też będę się do niego czasami odnosić w tej pracy. Nadanie robotowi swojego imienia pozwala nadać mu swojej tożsamości i łatwiej się do niego odnosić w trakcie codziennej pracy. Pokazuje ono również, że ma się na myśli dokładnego robota, a nie kolejny wypuszczony przez producenta egzemplarz [51].

Jedną z najważniejszych modyfikacji jest laptop, który jest połączony z robotem Rico za pośrednictwem sieci Wi-Fi. Laptop ten pozwala na bardzo swobodne dołączanie własnych węzłów działających w sieci ROS robota i komunikujących się bezpośrednio z robotem. Jego umiejscowienie na robocie sprawia, że jest on idealną platformą do obsługi dodatkowego sprzętu, który jest do robota doczepiony (mikrofon, dodatkowe kamery, czujniki itd.). Na tym właśnie sprzęcie zostanie uruchomiony węzeł intelligentnego stolika i bezpośrednio do niego będzie podłączony czujnik taktylny. Laptop ten zaspokoi również potrzeby energetyczne wykonanego przeze mnie czujnika.

Mikrofon i dodatkowy głośnik zostały zainstalowane na robocie, pomimo iż posiada on własne. Jednak te zainstalowane przez producenta nie są najwyższej jakości i w praktycz-

3. Wykorzystane narzędzia i materiały



Rysunek 3.2. Robot Rico - z modyfikacjami wykonanymi w laboratorium

nej komunikacji głosowej z robotem nie wypadały najlepiej. Wyzszej jakości sprzęt audio został wykorzystany do komunikacji głosowej z robotem, która dodatkowo została zintegrowana z usługami, które oferują możliwość wykorzystania narzędzi typu Text-to-Speech oraz Speech-to-Text. Narzędzia te pozwalają na obustronną konwersję: tekstu na głos, jak i wypowiadanych poleceń na tekst. Te usprawnienia znacznie poprawiają komfort korzystania z robota poprzez dodanie do niego intuicyjnego interfejsu głosowego [45], [46].

Oprogramowanie Rico zostało również wzbogacone o dodatkowe węzły ROS opracowane w laboratorium, które usprawniają i rozszerzają jego działanie. Do tych węzłów należy między innymi system planowania zadań TaskER, który umożliwia odpowiednie szeregowanie oraz priorytetyzowanie zadań. System ten pozwala na dość swobodne podejście do planowania i priorytetyzacji zadań robota. Pozwala na przykład na zawieszenie wykonywania mało ważnego zadania na rzecz wykonania ważniejszego i późniejszy, automatyczny powrót do poprzednio wykonywanego zadania. Robot ten posiada również usprawnioną nawigację, która pozwala na nadpisanie komend ruchu przez wykorzystanie sterowania o wyższym priorytecie (np. sterowanie ręczne) niż dla nawigacji. Posiada on również wgrany aktualną mapę laboratorium, która umożliwia mu bezproblemowe poruszanie się po środowisku oraz planowanie ścieżki w tym środowisku [50].

W przypadku Rico, dużym ułatwieniem prac z nim są dodatkowe możliwości sterowania robotem. Posiada on kilka dodatkowych sposobów obsługi, z czego najwygodniejszymi

dla użytkownika zdecydowanie są sterowanie za pomocą pady do konsoli oraz sterowanie za pomocą telefonu. Te dwa sposoby sterowania są bardzo wygodne i intuicyjne dla użytkownika, dzięki czemu pozwalają na bezpieczne i sprawne operowanie robotem ręcznie z zadajnika [49].

3.2. Materiał wrażliwy na nacisk - Velostat

Podstawą działania czujnika taktylnego jest materiał czuły na nacisk. W moim przypadku materiałem tym jest Velostat, znany również pod nazwą Linqstat. Velostat jest cienką folią, która pod wpływem nacisku zmienia swoją rezystancję – im wyższa siła nacisku tym mniejsza rezystancja folii. Materiał ten jest tani i łatwy w użyciu przez co w sieci można znaleźć dużo różnych przykładów wykorzystania go w projektach hobbyistycznych [52].

Materiał ten nie jest jednakże wykorzystywany tylko w zastosowaniach hobbyistycznych. Na jego temat powstało również kilka badań, jak również niektórzy naukowcy wykorzystują go do budowy swoich czujników taktylnych i sztucznej skóry [28], [32], [53], [54].

3.3. Mikrokontroler STM32

Wybór układu do sterowania czujnikiem taktylnym był dość ważną decyzją. Tak jak w pracy inżynierskiej mój wybór padł na produkt od STMicroelectronics - STM32F103RB. Jest to dokładnie ten sam model co wykorzystałem w pracy inżynierskiej. Różnice w układach sterowania wykonanych na potrzeby tych dwóch prac opisałem szerzej w rozdziale 4.2 [2], [55], [56].

Układy oferowane przez STMicroelectronics są szeroko używane w różnych zastosowaniach zarówno przemysłowych, konsumenckich jak i hobbyistycznych. Występują w wielu wariantach i modelach, które różnią się od siebie mocą obliczeniową, poborem mocy czy liczbą dostępnych peryferiów. Niektóre z nich oferują nawet wbudowane układy do komunikacji bezprzewodowej. Liczba możliwości w doborze mikrokontrolera sprawia, że można znaleźć odpowiedni dla praktycznie każdego zastosowania [55].

Układ do sterowania czujnikiem taktylnym, aby móc go wykorzystać do celów pracy magisterskiej, musiał posiadać kilka zasadniczych cech:

- możliwość łatwego i wygodnego programowania,
- obsługa 16 kanałów analogowych ADC,
- możliwość łatwej komunikacji z komputerem,
- względnie niski pobór mocy,
- niewielki rozmiar docelowego układu.

Dość łatwe programowanie jest zapewnione przez dostarczaną przez producenta bibliotekę HAL, narzędzie konfigurowania portów STMCubeMX, możliwość programowania wspieraną przez wiele IDE oraz moje wcześniejsze spore doświadczenia z tymi układami.

3. Wykorzystane narzędzia i materiały

Dodatkowo kod napisany na jeden układ można również w łatwy sposób przenieść na inny układ z rodziny STM32. Ułatwia to wstępna pracę na płytach rozwojowych, a w rozwiązaniu docelowym migrację na inny układ [55].

3.4. Python

Python jest powszechnie stosowanym skryptowym językiem programowania. Jest on wspierany przez szeroką społeczność, dzięki czemu jest bardzo mocno rozwinięty w wielu różnych dziedzinach naukowych. Dzięki temu wiele bibliotek wspiera ten język lub istnieje tylko w tym języku [57]. Było to dla mnie ważne ponieważ wiele z tych bibliotek było przeze wykorzystywanych, w tym:

- *PyQT5* - stworzenie prostego interfejsu graficznego do podglądu danych przychodzących z czujnika,
- *TensorFlow* - stworzenie i pracowanie nad sieciami neuronowymi [58], [59],
- *PIL* i *OpenCV* - przetwarzanie i obrabianie danych z czujnika jako obraz,
- *serial* - obsługa komunikacji ze sterownikiem przez port USB,
- *ros_numpy*¹⁰ - konwersja obrazu na formę możliwą do przesyłania tematami ROSowymi,
- *rospy* - obsługa funkcji systemu ROS i komunikacji z innymi węzłami.

3.5. ROS

ROS (Robot Operating System) jest systemem zaprojektowanym specjalnie do wygodnego sterowania, obsługi i programowania robotów. Zawiera wiele przyjaznych dla użytkownika narzędzi i rozwiązań, które ułatwiają wspomniane rzeczy. ROS jest również systemem open-source, co oznacza, że nie ma jednej firmy, która nad pracuje, każdy może wziąć udział w rozwijaniu go i jest darmowy. Są to dodatkowe cechy, które czynią ten system atrakcyjny [60].

System ten opiera się na węzłach, które reprezentują poszczególne części składowe robota. Pojedynczy węzeł zazwyczaj odpowiada za jedną z funkcji robota, jak na przykład poruszanie się czy też planowanie zadań. Dzięki temu łatwo jest dodać do robota dodatkową funkcję – poprzez napisanie kolejnego węzła współpracującego z tymi już istniejącymi. Same węzły komunikują się ze sobą za pomocą tematów oraz serwisów, na których wysypane są odpowiednie informacje czy też polecenia [60]

Jednymi z podstawowych narzędzi udostępnianych przez ROS są: narzędzie do symulacji – Gazebo oraz narzędzie do podglądu stanu robota – RViz. Gazebo jest wygodnym symulatorem, który pozwala testować robota oraz algorytmy w bezpiecznym środowisku, bez ryzyka uszkodzenia człowieka, robota czy też środowiska. RViz jest natomiast narzędziem do wygodnego obserwowania stanu robota. Odczytuje ono informacje przepływające po wybranych tematach i wizualizuje w formie wygodnej dla człowieka. RViz

¹⁰ Strona projektu ros_numpy: https://github.com/eric-wieser/ros_numpy

jest w stanie wizualizować zarówno stan robota w symulacji jak i rzeczywistym środowisku [60]–[62].

3.6. MeROS i Visual Paradigm

Do usystematyzowanego tworzenia dokumentacji wybrałem MeROS (w najnowszej dostępnej wersji 1.2.3¹¹) - język modelowania systemów oparty na języku SysML oraz UML. MeROS został przystosowany specjalnie do opisywania systemów opartych na ROS i zawiera elementy przystosowane do tego środowiska. MeROS jest również językiem rozwijanym na Politechnice Warszawskiej i używanie go rozwija prestiż tej uczelni [63]–[65].

MeROS jest metamodelem, który definiuje i systematyzuje relacje pomiędzy poszczególnymi elementami istniejącymi w tym środowisku, takimi jak system, obszary robocze, repozytoria, pakiety, węzły czy kanały komunikacji. Dopuszcza również możliwość umieszczania na schematach elementów, które nie istnieją w środowisku ROS [65].

Do tworzenia diagramów wykorzystywany był program Visual Paradigm. Program ten wspiera wiele wielu języków modelowania i nie tylko. Wspiera przede wszystkim języki, na których opiera się metamodel MeROS, czyli SysML i UML. Dzięki temu możliwe jest w tym programie wygodne tworzenie wszystkich potrzebnych diagramów. Diagramami, które zostały wykorzystane w niniejszej pracy są:

- *Block Definition Diagram (bdd)* - podstawowy diagram definiujący strukturę systemu i identyfikujący poszczególne bloki systemu,
- *Internal Block Diagram (ibd)* - diagram definiujący wewnętrzną strukturę wewnątrz bloku lub pewnej struktury,
- *Activity Diagram (act)* - diagram aktywności pokazujący przebieg pewnego procesu od początku do końca,
- *Sequence Diagram (sq)* - diagram sekwencji pokazuje interakcję pomiędzy poszczególnymi elementami systemu i jej przebieg w czasie,
- *State Machine Diagram (stm)* - diagram maszyny stanów prezentujący stany w jakich może znaleźć się układ i czynniki jakie wpływają na zmianę aktualnego stanu,
- *Use Case Diagram (uc)* - diagram przedstawiający przypadki użycia systemu [65], [66].

¹¹ Strona GitHub projektu MeROS: <https://github.com/twiniars/MeROS>

4. Budowa stolika z wbudowanym czujnikiem taktylnym

Jako że jedną z funkcji robota asystującego jest przewożenie przedmiotów, to zaprojektowany stolik został zamocowany bezpośrednio nad miejscem przeznaczonym do przewożenia przedmiotów. Starsza konfiguracja robota posiadała już dodatkowy stolik, który pokryty był gumą, aby zwiększyć jego przyczepność, jak również chronić robota przed potencjalnymi uszkodzeniami. Stolik ten generował również swojego rodzaju półkę, w którą można włożyć laptopa nie zabierając w ten sposób robotowi możliwości przewożenia przedmiotów. Nowa iteracja, którą w ramach tej pracy wykonywałem, miała na celu bezpośrednie zastąpienie wcześniej używanej wersji, która jest widoczna na rysunku 4.1.

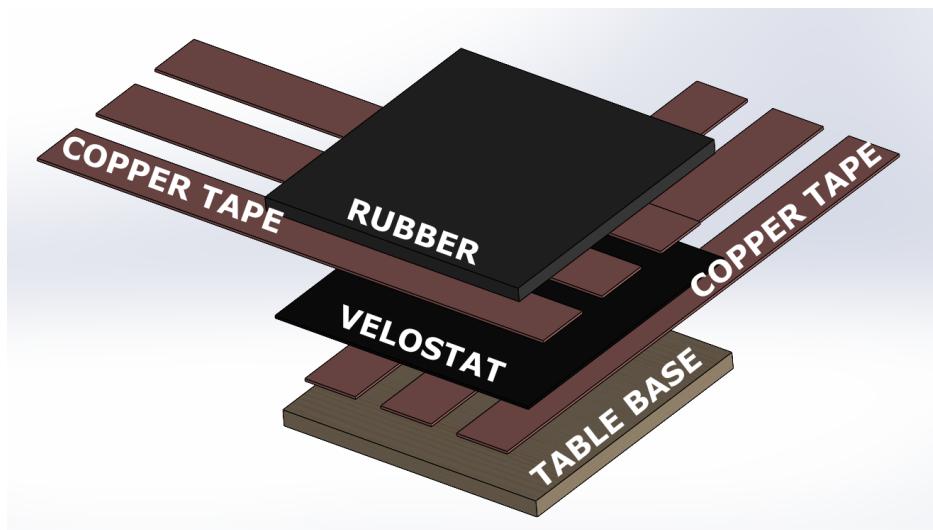


Rysunek 4.1. Zastąpiona wersja stolika na robocie

4.1. Budowa mechaniczna stolika

Z uwagi na konieczność zastąpienia istniejącego już stolika, nowy stolik miał za zadanie być jak najbardziej podobny do tego już istniejącego. W tym celu zmierzyłem stary stolik i na jego podstawie zaprojektowałem nowy, którego wymiary widoczne są na rysunku 4.3a. Na tej postawie byłem w stanie dokładnie określić jakich materiałów użyć i jak je rozmieścić, aby otrzymać jak największą liczbę pól taktylnych, przy jednoczesnym jak największym pokryciu powierzchni [67].

Sam koncept na wykonanie sztucznej skóry nie zmienił się od początku, czyli od momentu kiedy włączyłem się w prace prowadzone wcześniej w zespole. Ogólny schemat budowy sztucznej skóry widoczny jest na rysunku 4.2. Jedyna zmiana, którą wykonałem względem oryginalnego podejścia, to zamiana jednej z warstw tłumiących na twardą



Rysunek 4.2. Zasadnicza struktura sztucznej skóry

warstwę konstrukcyjną. Warstwa ta jednocześnie pełni funkcję twardej i stabilnej podstawy stolika [2], [68].

Jako warstwy przewodzące zostały wykorzystane taśmy miedziane, które zapewniają niewielką grubość, dobre przewodzenie prądu oraz są one stosunkowo łatwe w montażu. Warstwy przewodzące są ułożone względem siebie prostopadle aby tworzyć możliwie jak najwięcej pól taktylnych.

Warstwa Velostatu bezpośrednio odpowiada za zdolność wykrywania nacisku. Materiał ten ma tą właściwość, że pod wpływem nacisku zmienia on swoją rezystancję, która jest później mierzona przez sterownik [52].

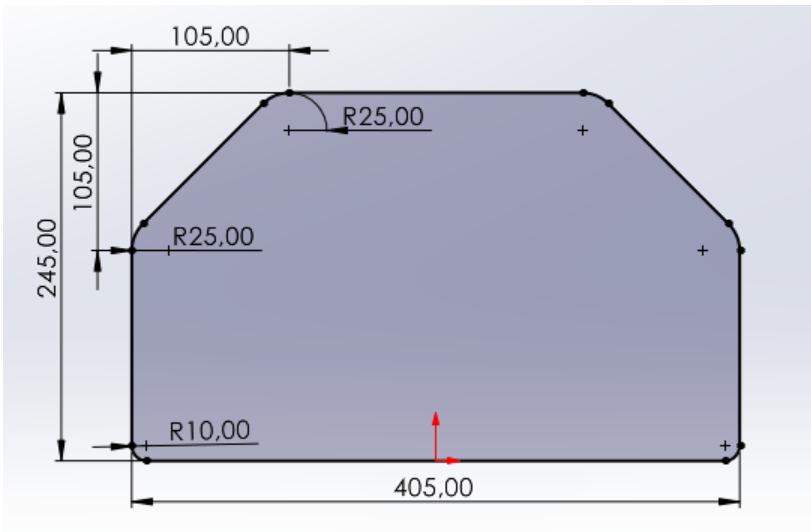
Finalnie wykonany został stolik składający się z 224 pól taktylnych rozmieszczonej równomiernie na całej powierzchni stolika. Każde pole ma wymiary $1x2\text{cm}$. Do stworzenia całego stolika została wykorzystana taśma miedziana w konfiguracji $16x16$ prostopadle ułożonych pasków taśmy. Daje to potencjalny wymiar 256 pól, ale część z nich nie istnieje ze względu na kształt samego stolika. Do każdej z taśm miedzianych został doprowadzony przewód, który później jest poprowadzony bezpośrednio do sterownika.

Góra pokryta została gumą mikroporową o grubości 5mm wybraną jeszcze na etapie badań nad pracą inżynierską. Głównym zadaniem tej warstwy jest ochrona wnętrza czujnika przed uszkodzeniami. Całość stolika została dodatkowo zabezpieczona taśmą po bokach oraz miejscami została wykorzystany klej, usztywniający całą konstrukcję. Wykonany stolik został przedstawiony w całej okazałości na rysunku 4.3 [2].

4.2. Sterownik czujnika taktylnego

Sterownik czujnika taktylnego jest tym elementem, który łączy sprzętową część stolika z jego programową częścią. Dlatego też stoją przed nim odpowiednio z tym związane zadania:

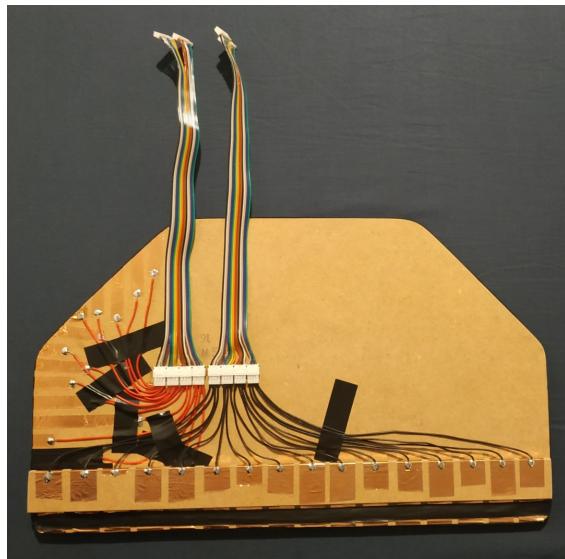
4. Budowa stolika z wbudowanym czujnikiem taktylnym



(a) Projekt stolika wraz z jego zewnętrznymi wymiarami



(b) Widok stolika z przodu



(c) Widok stolika z tyłu

Rysunek 4.3. Zaprojektowany i wykonany stolik pokryty sztuczną skórą

- zasilenie czujnika taktylnego i odczytywanie nacisku na każdym z jego pól,
- wstępne przetworzenie danych oraz przygotowanie ich do wysłania do komputera,
- automatyczny restart w przypadku zawieszenia systemu lub krytycznych błędów,
- możliwość zdalnej konfiguracji, bez konieczności budowania kodu mikrokontrolera, przy zmianie parametrów pracy skóry.

Do kontroli opisanego czujnika taktylnego został zaprojektowany sterownik, oparty na mikrokontrolerze STM32. Zaprojektowany układ założeniowo opierał się na tym skonstruowanym na potrzeby mojej pracy inżynierskiej, ale wprowadziłem do niego dodatkowe ulepszenia i usprawnienia:

- zmiana układu kluczującego z ULN2803 na dyskretne tranzystory MOS,

- dobór rezystancji dzielnika napięciowego na podstawie spodziewanego obciążenia na sztucznej skórze,
- wyeliminowanie z układu elementów, które nie są wykorzystywane przez inteligentny stolik.

Kluczową zmianą, która została wykonana, była zamiana układu kluczującego ULN2803 na dyskretne tranzystory MOSFET. Układ ULN2803 miał tą własność, że podczas kluczowania sygnału spadek napięcia na układzie wynosił około $0,7 - 0,8V$. W przeciwnieństwie do tego układu cyfrowego, wykorzystane tranzystory MOS charakteryzują się niską rezystancją kanału przewodzącego ($< 100m\Omega$). W efekcie daje to pomijalnie niskie straty napięcia na tranzystorze kluczującym. W przypadku wykorzystywanego przeze mnie zasilania $3,3V$ zmiana ta jest bardzo korzystna, ponieważ poprzedni układ wprowadzał straty około 23% na maksymalnym zakresie pomiarów, a aktualne rozwiązanie nie wprowadza zauważalnych strat [69], [70].

Kolejna zmiana miała na celu poprawienie jakości otrzymywanych pomiarów. Wartość rezystancji została dobrana na podstawie wcześniejszych moich prac ze sztuczną skórą tak, aby zwiększyć czułość pomiarów w okolicy docelowego punktu pracy czujnika taktylnego. Wartość ta została tak dobrana, aby skóra była bardziej czuła na niewielki nacisk. Wartość dzielnika rezystancji ostatecznie została zmieniona z 330Ω na 470Ω [2].

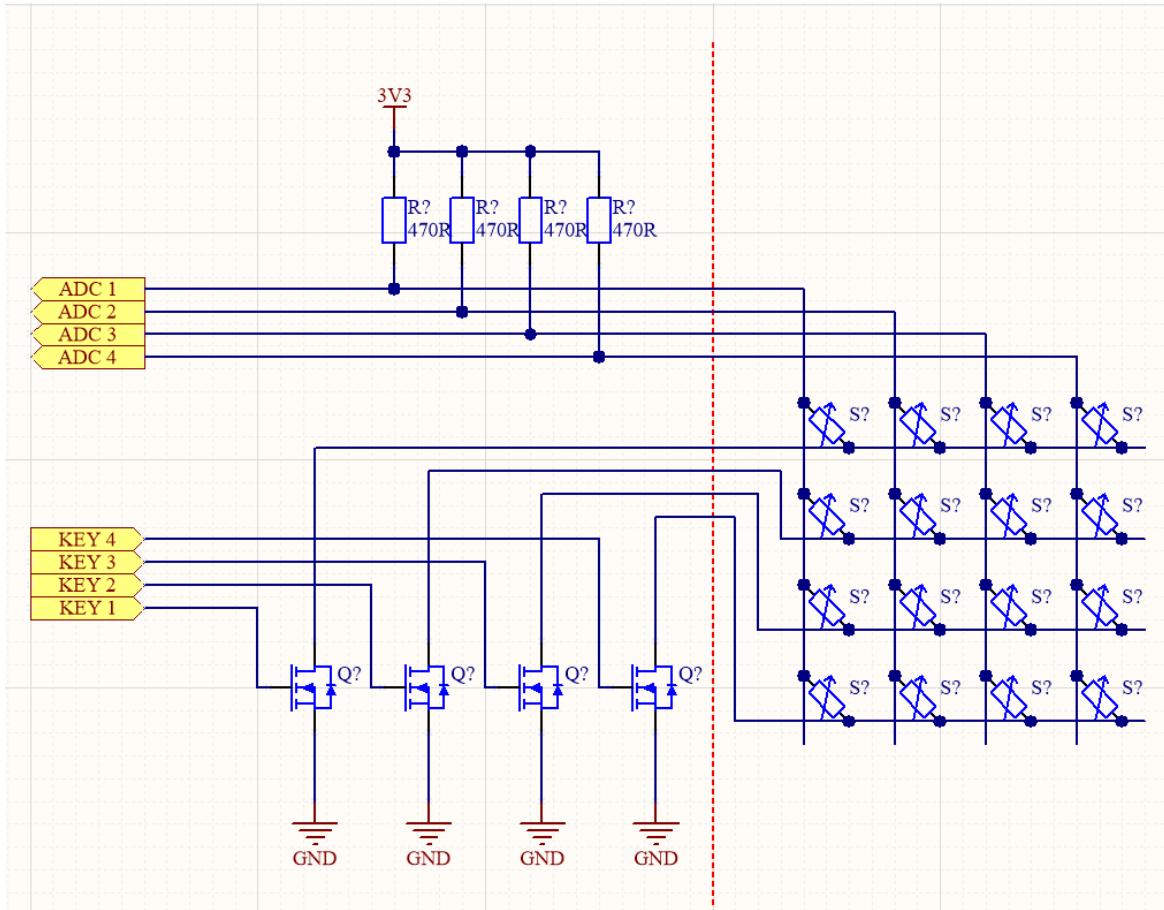
Z układu zostały usunięte nieużywane elementy: wyprowadzenie na moduł Bluetooth i dodatkowe złącze zewnętrznego zasilania. Pierwotnie były one przewidziane do ewentualnego użycia w przyszłości. Okazały się one jednak niepotrzebne, a zajmowały one cenne miejsce na płytce [2].

Sama zasada działania układu i sposób zbierania danych z poszczególnych pól czujnika pozostała niezmienny. Ideowy schemat działania układu zbierającego dane został przedstawiony na rysunku 4.4. Prawa strona schematu ilustruje czujnik taktylnego, gdzie każde pole taktyльne jest zobrazowane przez osobny rezystor ze zmienną rezystancją. Lewa część ilustruje część pomiarową zrealizowaną na płytce PCB. Zostały przedstawione na niej sygnały kluczujące, wysyłane przez mikrokontroler, jak również odczytywane przez ten sam mikrokontroler napięcie na poszczególnych polach. Należy wziąć pod uwagę, że w danym momencie włączony jest tylko jeden klucz, a nad poprawnością tego czuwa mikrokontroler.

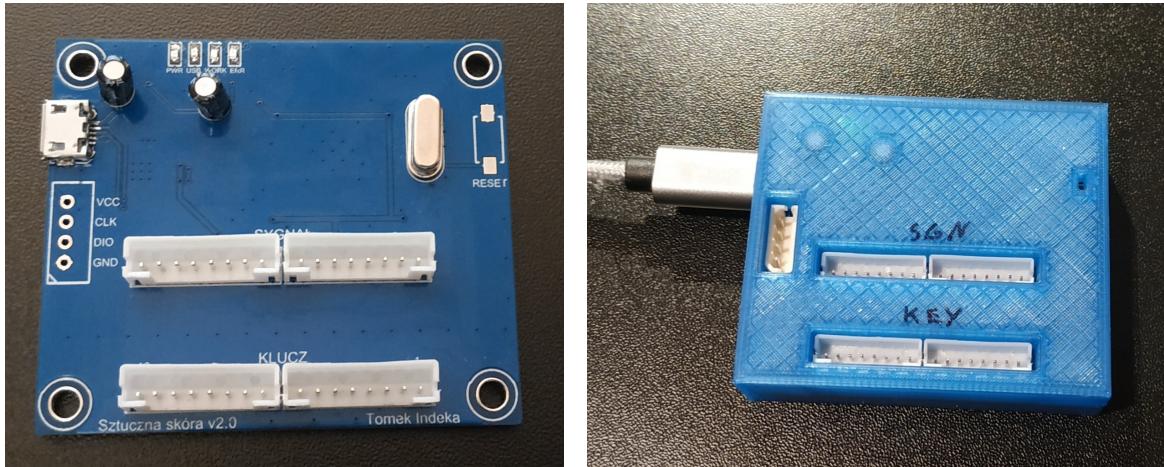
Uwzględniając opisane modyfikacje powstał nowozaprojektowany układ elektroniczny przedstawiony na rysunku 4.5. Ponadto została do niego zaprojektowana i wydrukowana obudowa, osłaniająca cały układ.

Komunikacja sterownika z komputerem odbywa się przez wykorzystanie szeregowego portu USB. Za pomocą tego portu sterownik jest również zasilany. Pozwala to na minimalizację liczby przewodów, które przebiegają pomiędzy komputerem a czujnikiem taktylnym. Po stronie sterownika komunikacja ta jest obsługiwana przez dodatkowy moduł CP2102, który stanowi swojego rodzaju bufor pomiędzy mikrokontrolerem a komputerem.

4. Budowa stolika z wbudowanym czujnikiem taktylnym



Rysunek 4.4. Schemat elektroniczny przedstawiający zasadę działania i odczytywanie wartości rezystancji z pól czujnika taktylnego

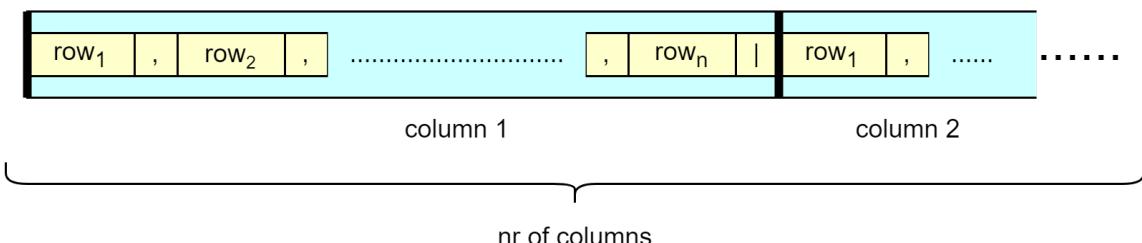


Rysunek 4.5. Zaprojektowana i wykonana elektronika do kontroli stolika

Wysyłanie odbywa się regularnie z częstotliwością 10Hz i w tym czasie wysyłane są do komputera dane odczytane ze wszystkich pól stolika. Ramka wysyłana do komputera jest bardzo prosta i nie zawiera żadnych zabezpieczeń poprawności przesłania danych. Jest to

zrealizowane w taki sposób, aby możliwy był podgląd danych w oknie zwykłego terminala systemowego. Przesyłana ramka widoczna jest na rysunku 4.6. Składa się ona z pomiarów z poszczególnych rzędów oddzielonych przecinkami. Kolejne kolumny danych oddzielone są od siebie liniami pionowymi.

n - nr of rows



Rysunek 4.6. Diagram ramki wysyłanej do komputera

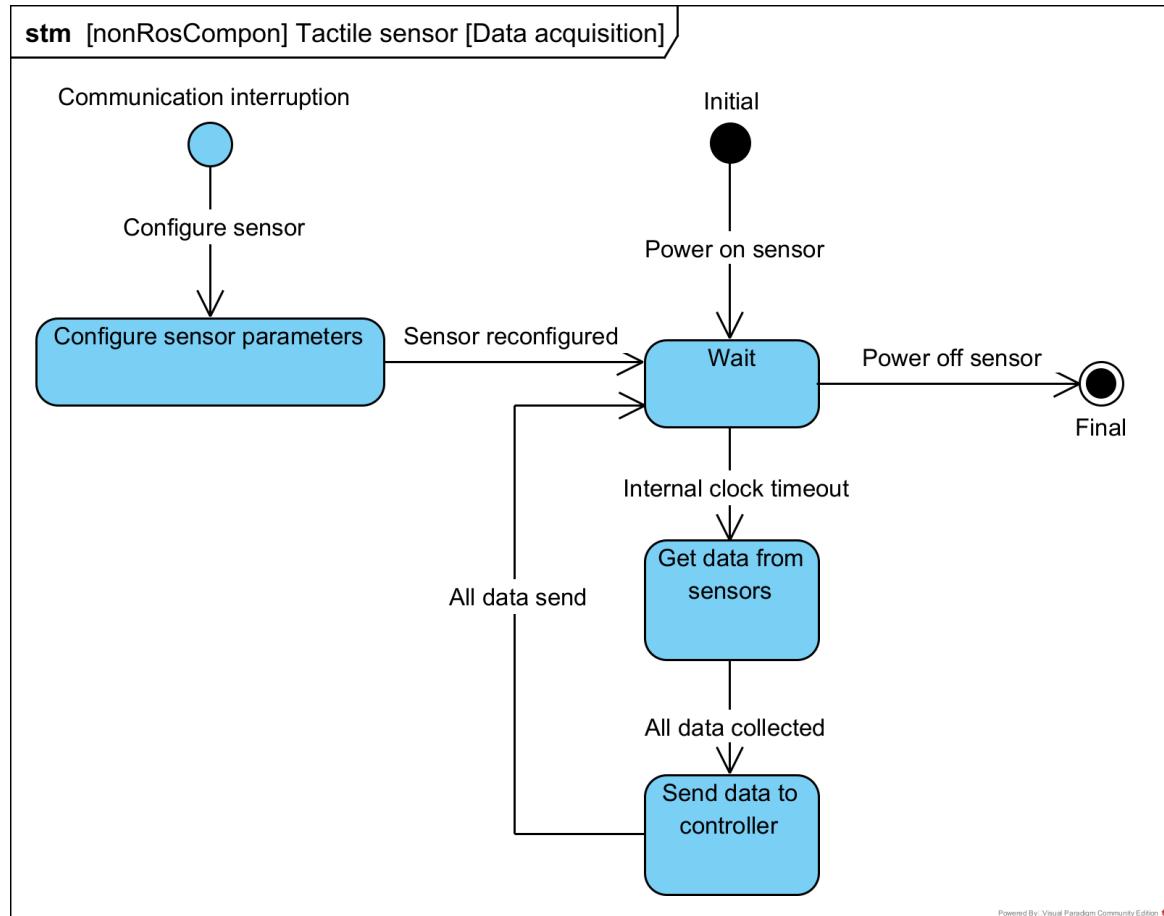
Programowanie mikrokontrolera bywa kłopotliwe. Potrzeba do tego specjalnego programatora, programu do wgrywania plików binarnych, kodu źródłowego programu, specjalnego kompilatora i bibliotek danego mikrokontrolera. Konfigurowanie takiego środowiska jest czasochłonne i wymaga sporej determinacji, co może być kłopotliwe jeśli potrzeba dokonać niewielkich modyfikacji. Dlatego postanowiłem, że sterownik powinien w jakimś stopniu wspierać możliwość zdalnej jego konfiguracji przez USB. Zaimplementowana przeze mnie opcja pozwala na skonfigurowanie liczby rzędów i kolumn sztucznej skóry, ich typu i tego jakie wcześniej zdefiniowane obliczenia mają być wykonane przez mikrokontroler. W praktyce pozwala to na używanie tego samego oprogramowania w różnych zastosowaniach, w różnych konfiguracjach i przy różnych rozmiarach sztucznej skóry, w szczególności sztucznej skóry w wersji zaprojektowanej w trakcie pracy inżynierskiej.

Program napisany w celu kontrolowania sztucznej skóry jest stosunkowo prosty. Jest on cały czas w gotowości i oczekuje na sygnał przerwania z wewnętrznego zegara, po którym dokonuje on odczytu wszystkich pól sztucznej skóry, przygotowuje dane do wysłania i umieszcza je w buforze nadawczym. Pętla działania sterownika została zaprezentowana na schemacie maszyny stanów na rysunku 4.7.

Wewnętrzny zegar generuje sygnał przerwania dokładnie co $100ms$. Ten czas jest wystarczający, aby sterownik mógł zebrać dane ze wszystkich pól czujnika i je przetworzyć. Z obliczeń wynika, że czas ten może być jeszcze bardziej skrócony, ale nie zostało to wykonane z uwagi na obsługujący go węzeł, który również musi przetworzyć otrzymywane dane. Dane te przesyłane są później również do całego systemu ROS, gdzie częstotliwość $10Hz$ jest standardową częstotliwością pracy wielu modułów. Z tego powodu zdecydowałem, że w tym przypadku dopasuję częstotliwość pracy mikrokontrolera do częstotliwości wykorzystywanej w systemie ROS.

Pola czujnika taktylnego dostarczają sygnał napięciowy, który jest odczytywany przez wbudowane w mikrokontroler przetworniki ADC. Przetworniki te mają rozdzielcość

4. Budowa stolika z wbudowanym czujnikiem taktylnym



Rysunek 4.7. Diagram automatu skończonego sterownika sztucznej skóry

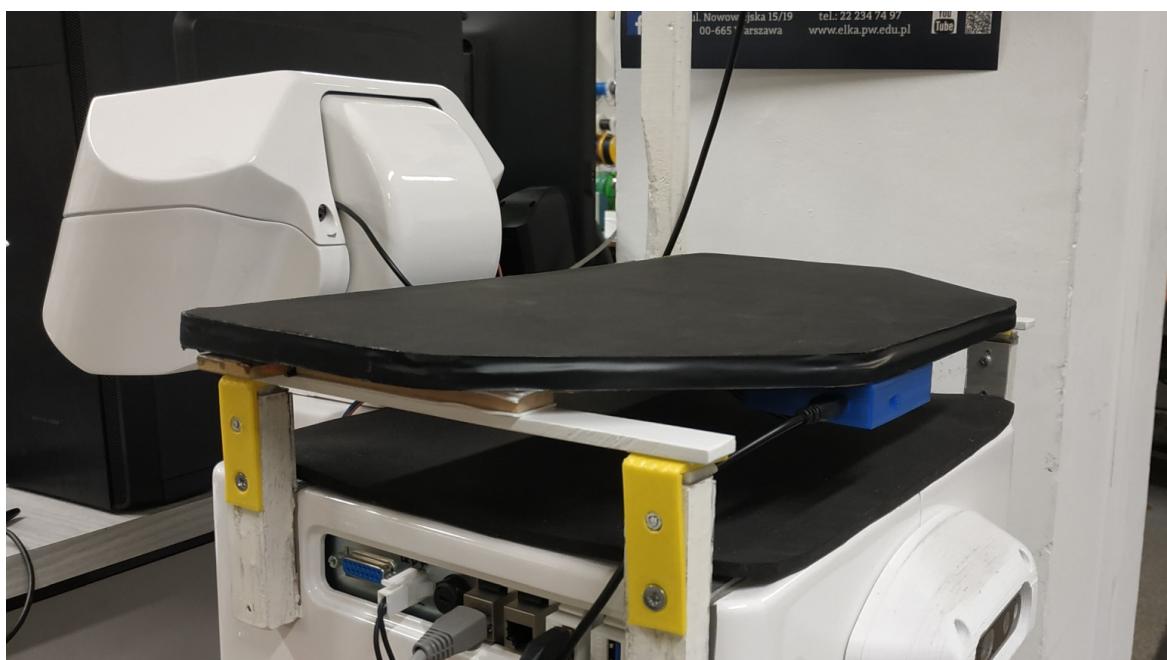
12 bitów, co w praktyce daje wartości w zakresie 0 – 4095. W przypadku sztucznej skóry zastosowanej na stoliku otrzymywane dane nie są przetwarzane przez sterownik, tylko wysyłane w surowej postaci do węzła ROSowego, który sam zajmuje się ich interpretacją. Dodatkowo, wraz z każdym wysłaniem danych zmieniany jest stan zapalenia diody oznaczającej pracę układu. Jest to informacja dla operatora o tym, że sterownik czujnika taktylnego działa poprawnie.

Zaimplementowany został także watchdog (nie został on wyszczególniony na diagramie 4.7), którego zadaniem jest wykrywanie zawieszenia się systemu. Jeśli system się zawiesi i nie zresetuje licznika watchdoga, wtedy sterownik zostanie zrestartowany, a program uruchomi się na nowo. Dodatkowo, w celu informacji dla użytkownika, po restartie systemu przez watchdoga zapalana jest na PCB czerwona dioda sygnalizująca error. Palenie się tej diody nie ma żadnego wpływu na działanie programu, jest tylko informacją dla operatora, że w trakcie pracy wystąpił błąd i system się restartował. Licznik watchdoga resetowany jest za każdym razem po wysłaniu wszystkich danych z czujników stolika do komputera. Czas restartu watchdoga został ustawiony na 1 s.

4.3. Integracja mechaniczna stolika z robotem

Stolik, który był pierwotnie na robocie, był przymocowany na stałe za pomocą taśm montażowych. Aby zmienić stolik na nowy musiały one zostać zerwane. Narodziła się w tamtym momencie koncepcja, aby wykonać mocowanie pozwalające na demontaż stolika bez konieczności zrywania taśm.

W tym celu zostało zaprojektowane i wydrukowane specjalne mocowanie, które pozwala w prosty sposób, za pomocą śrub, zmienić wybrany stolik na inny. Mocowanie zostało przyklejone na stałe do obu stolików: starego i nowego. Jednocześnie do systemu mocowania na robocie zostały wprowadzone mufy, pozwalające na swobodne przykręcenie stolika do robota. Nowa wersja stolika umieszczona i przykręcana na górze robota jest widoczna na rysunku 4.8.



Rysunek 4.8. Wykonana przeze mnie wersja stolika na robocie

Porównując starszy stolik (rys. 4.1) z nowym (rys. 4.8) można zauważyć, że ten, nowy stolik, jest wyraźnie mniejszy. Można to zaobserwować zwracając uwagę na pionowe punkty mocowania, które w przypadku nowego stolika znajdują się poza powierzchnią samego stolika. Wprowadził to konieczność montażu nowej wersji na odpowiednich przedłużeniach. Wykonane one zostały z plexy, która wprowadza dodatkową, niepożądaną, elastyczność i sprężystość jeśli chodzi o sam montaż stolika. Jest to uchybienie, które nie powinno mieć wpływu na wyniki dalszych badań, jednakże w przyszłości w miarę możliwości powinno zostać ono poprawione i usztywnione.

5. Przeprowadzone badania i testy Velostatu

5.1. Badanie zależności pomiędzy siłą nacisku a rezystancją

Podobne badanie przeprowadzałem również na etapie prac nad moją pracą inżynierską. Pojawiła się jednak potrzeba wykonania tych badań ponownie ze względu na trudność użycia poprzednich badań w praktyczny sposób.

Główną różnicą utrudniającą wykorzystanie pomiarów z mojej pracy inżynierskiej jest fakt, że wykonana została wtedy charakterystyka wartości odczytywanej przez sterownik w zależności od przyłożonego nacisku. Wykonywana teraz charakterystyka jest zależnością rezystancji od przyłożonego nacisku. W przypadku odczytywania wartości zmierzonej przez sterownik wartość ta zależy od użytego przetwornika ADC, rezystancji dzielnika napięcia, użytego układu kluczującego i potencjalnych innych szumów w układzie. Zebrała w ten sposób charakterystyka może być rzetelnie użyta tylko na bardzo podobnym układzie jak ten, na którym została wykonana. Charakterystyka bazująca na rezystancji pola taktylnego jest dużo bardziej uniwersalna i może być z dużą łatwością przeniesiona na inny sterownik, nawet taki, o zupełnie innej budowie.

Wspomniany został też inny układ kluczujący (opisany szczegółowo w rozdziale 4.2), który znacznie zmniejszał zakres pomiarów przetworników ADC. Użycie starej charakterystyki opierałoby się na pomiarach, które muszą zostać w praktyce przeskalowane i rozciągnięte na pełny zakres przetworników. Zmieniona została również wartość rezystora w dzielniku napięciowym, więc przekształcenie starej charakterystyki na nową wprowadza konieczność wykonania wielu dodatkowych obliczeń.

Finalnie, inne jest również pole powierzchni pojedynczego pola czujnika. W trakcie prac nad pracą inżynierską badałem pola o wymiarach $5x5\text{cm}$, czyli 12,5 raza większe niż wykorzystuję obecnie na stoliku. Taka zmiana pola powierzchni również może zakłócić otrzymywane pomiary. Wszystkie wspomniane różnice składałyby się na kolejne błędy obliczeniowe, które po skumulowaniu mogą wprowadzać duże przekłamania w stosunku do stanu faktycznego [2].

5.1.1. Przeprowadzenie badań nad Velostatem i zależnością pomiędzy siłą nacisku a jego rezystancją

Celem wykonanych badań było jak najlepsze wyznaczenie charakterystyki rezystancji pojedynczego pola czujnika taktylnego w zależności od przyłożonego nacisku. Folia Velostat, która jest podstawą budowy czujnika, posiada tą własność, że wraz ze zwiększeniem nacisku na nią zmniejsza się jej rezystancja. Dobrze wykonana charakterystyka zachowania się czujnika jest kluczowa przy późniejszych próbach szacowania wag przedmiotów położonych na stoliku [52].

Stanowisko do badania rezystancji czujnika taktylnego było dość proste, składało się ono z:

- pojedynczego pola czujnika,
- multimetru z możliwością mierzenia rezystancji,
- różnych wcześniej zważonych obciążen.

Pojedyncze pole czujnika zostało wykonane w 3 wymiarach: $1x1\text{cm}$, $1x2\text{cm}$ oraz $2x2\text{cm}$. Każde z tych pól zostało wykonane na twardym materiale. Rozważany był również miękki materiał, taki jakiego został użyty w budowie stolika, ale ze względu na rozkładanie masy po bokach czujnika oraz niestabilność odważników przy dużych obciążeniach nie została wykonana charakterystyka na nim. Testowe podejścia nie wykazały jednak większych różnic w danych pomiędzy twardym, a miękkim materiałem. Stanowisko do badań zostało przedstawione na rysunku 5.1.



Rysunek 5.1. Stanowisko badawcze do pomiarów zależności pomiędzy naciskiem a rezystancją

Pomiary zostały zebrane dla pól o wielkości $1x1\text{cm}$, $1x2\text{cm}$ oraz $2x2\text{cm}$. Dla każdego z tych pól zostało zebranych po 6 kolejnych odczytów rezystancji dla każdego z wykorzystanych obciążen. Wykorzystane obciążenia były takie same dla każdego testowanego pola.

W praktyce podczas wykonywania badań pojawiły się spodziewane, na podstawie moich wcześniejszych doświadczeń, problemy. Odczytywane wartości rezystancji pola czujnika zmieniały się w zależności od tego jak długo obciążenie pozostawało na czujniku. Wraz z biegiem czasu odczytywana rezystancja spadała. Aby zbierane dane mogły być do siebie bezpośrednio odnoszone wszystkie zebrane próbki danych zostały zapisane w niewielkim odstępie czasu (kilka sekund) po położeniu obciążenia na czujniku [2].

Podobne zjawisko zaobserwowali również inni badacze na całym świecie. Niektórzy podeszli do tematu bardzo rzetelnie i sprawdzili jak dokładnie folia zmienia swoją rezy-

5. Przeprowadzone badania i testy Velostatu

stancję w czasie. Z wykonanych przez nich badań wynika, że przez pierwsze $\sim 10\text{s}$ wartość rezystancji się zauważalnie zmienia. Po tym czasie staje się ona bardziej stabilna, ale wciąż systematycznie i konsekwentnie spada. Po dłuższym czasie, $\sim 10\text{min}$, wartość rezystancji jest już praktycznie ustalona. Sama wartość rezystancji może osiągać wtedy wartość nawet 37,5% niższą niż wynosiła ona w momencie przyłożenia obciążenia do czujnika, co jest bardzo dużą zmianą. Jest to informacja bardzo ważna, którą warto uwzględnić w programie, a przynajmniej trzeba być świadomym tej wady materiału podczas korzystania z niego [29], [53], [71].

Dodatkowo zauważałem, że pojedyncze pomiary czasami potrafią znacznie odbiegać od innych pomiarów, wykonanych w tych samych warunkach. Do tej pory nie jestem w stanie jednoznacznie określić, co było przyczyną takich rozbieżności.

Pomiary wykazywały również tendencję do trochę innych wartości rezystancji, kiedy obciążenie było zmniejszane, a nie zwiększane, co świadczy o histerezie materiału. Jej charakterystyka nie została jednak przeze mnie dokładnie zbadana i opisana. Problem ten był natomiast bardziej szczegółowo analizowany przez innych badaczy, którzy wykazali, że Velostat faktycznie cechuje się histerezą i swojego rodzaju pamięcią materiału. Wykazali oni, że podczas zmniejszania obciążenia przykładanego do czujnika wykazuje on mniejszą rezystancję niż podczas zwiększania obciążenia, dla tej samej wartości obciążenia [71].

W celu uśrednienia zebranych przeze mnie danych, biorąc pod uwagę problemy podczas samego badania, zrezygnowałem ze standardowej średniej arytmetycznej na rzecz średniej obciążonej. Jest to średnia arytmetyczna z pominięciem najwyższych i najniższych wartości w zbiorze, w moim przypadku odrzuciłem jedną największą wartość oraz jedną najmniejszą wartość w zbiorze. Zdecydowałem się skorzystać z niej, ponieważ zauważałem w pobranych danych pojedyncze wartości znacznie odstające od pozostałych przy danym obciążeniu. Nie chciałem, aby wartości te znieksztalciały otrzymane wyniki. Uśrednione wyniki pomiarów dla testowanych pól znajdują się w tabeli 5.1. Dane zawarte w tabeli zostały również przedstawione na wykresie widocznym na rysunku 5.2, gdzie dodatkowo widoczna jest próba aproksymacji zebranych danych.

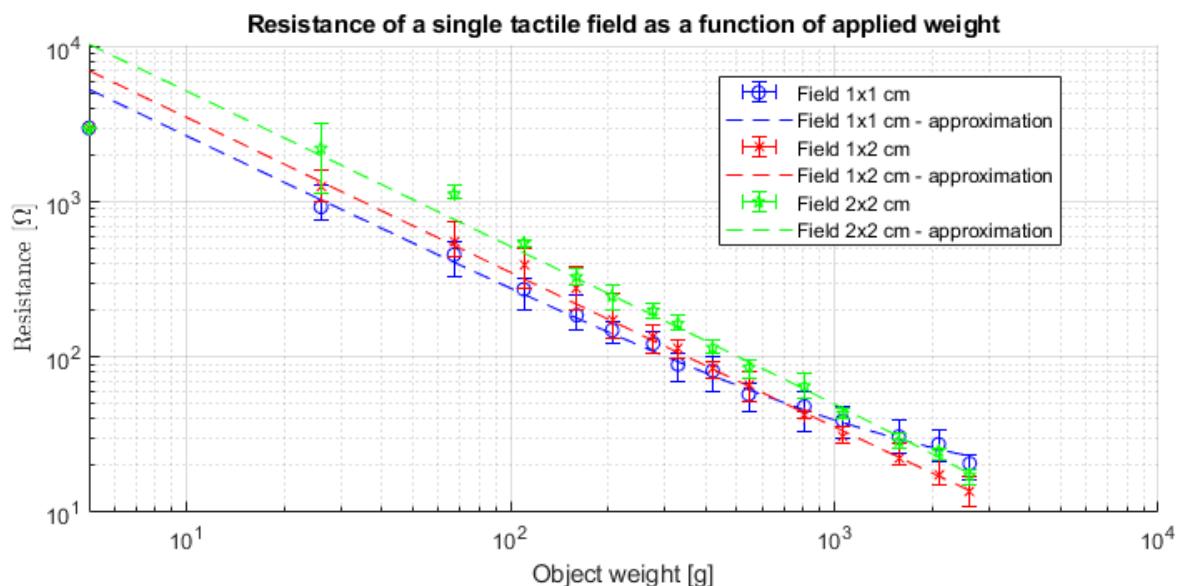
Na podstawie średnich arytmetycznych zebranych w tabeli 5.1 można od razu zauważyć, że rezystancja odczytywana przy polach o większej powierzchni jest większa. Prawdopodobnie wynika to z rozłożenia nacisku na większą powierzchnię, przez co folia Velostat jest mniej naciskana na swojej przestrzeni. Różnica ta zaciera się wraz ze zwiększaniem przykładanego obciążenia.

Widoczne jest również pewne odchylenie w danych dotyczących pola $1 \times 1\text{cm}$, które przestaje prezentować najniższe pomiary dla obciążień powyżej $\sim 500\text{g}$. Pochodzenie tych odchyleń wymaga jednak dalszych badań. Analizując wykres 5.2 opisane odchylenie można zauważać jeszcze dokładniej. Widać również, jak równolegle do siebie układają się zmierzone wartości rezystancji, szczególnie przy niższym nacisku.

Warto zwrócić uwagę na logarytmiczną skalę na obu osiach wykresu, która pozwala

Tabela 5.1. Rezystancja pola czujnika dla różnych pól powierzchni badanego czujnika (średnia 4 z 6)

Obciążenie [g]	Pole 1x1cm	Pole 1x2cm	Pole 2x2cm
26	930,00	1 250,00	2 212,50
67	457,50	558,50	1 125,00
110	273,75	392,50	542,50
160	186,25	277,50	330,00
206	149,25	175,50	247,50
276	122,75	135,75	200,00
329	89,75	114,75	165,00
421	81,75	85,75	115,00
544	57,63	66,00	85,88
808	48,10	42,75	65,25
1062	39,00	31,00	43,25
1580	30,75	22,50	28,75
2098	27,50	17,50	24,75
2616	20,68	13,50	17,25

**Rysunek 5.2.** Rezystancja pola czujnika dla różnych pól powierzchni badanego czujnika (średnia 4 z 6)

w klarowny sposób przedstawić zależności pomiędzy dużymi zmianami w pomiarach. Oprócz samych pomiarów przedstawione zostały również słupki błędów dla każdego z badanych punktów. Dla zerowej siły nacisku nie zostały wykonane żadne pomiary, ponieważ sama górną część badanego czujnika posiadała już zauważalną masę. Część wykresu w bliskim otoczeniu 0 jest przedłużeniem istniejącego wykresu.

Aproxymacja widoczna na rysunku 5.2 została wykonana hiperbolą. Wzór ogólny hiperboli został przedstawiony na równaniu (1), gdzie y jest przesunięciem hiperboli w osi pionowej, x_0 jest przesunięciem hiperboli w osi poziomej, a parametr a jest parametrem

5. Przeprowadzone badania i testy Velostatu

hiperboli i określa jej zakrzywienie. $f(x)$ jest wartością rezystancji, która obliczana jest na podstawie siły nacisku x .

$$f(x) = y + \frac{a}{(x - x_0)} \quad (1)$$

Optymalne wartości a , y , oraz x_0 obliczane są w Matlabie na podstawie zebranych wcześniej danych. Do optymalizacji wykorzystywana jest wbudowana funkcja *fmincon*, która pozwala na wprowadzenie ograniczeń podczas wykonywania obliczeń. Ograniczeniem, które wprowadziłem do algorytmu było $x_0 = 0$, ponieważ na późniejszych etapach prac okazało się, że kiedy ta wartość odbiegała od zera to wykorzystanie modelu wprowadzało problemy przy szacowaniu wagi dla niewielkich obiektów. Same obliczenia, w celu zminimalizowania błędów losowości i niedeterminiczności funkcji *fmincon*, zostały wykonywane po 100 razy. Jako wynik zostały wybrane parametry, dla których funkcja aproksymująca zwracała najmniejszy błąd [72].

Funkcja, która była minimalizowana w trakcie obliczeń, nie była standardową normą Euklidesową, lecz musiała zostać odpowiednio zmodyfikowana. Z uwagi na bardzo duże rozbieżności pomiędzy wartościami odczytów i koniecznością wykonania charakterystyki na całej dziedzinie musiałem wprowadzić czynnik, który nie pozwoli niskim wartościom nacisku (dużym pomiarom rezystancji) na zdominowanie charakterystyki. W tym celu wykorzystałem funkcję błędu (2), gdzie wektor różnic wartości faktycznej y i oczekiwanej $f(y', a', x'_0, x)$ jest dodatkowo mnożony przez wektor sił nacisku x . Wartości y' , a' oraz x'_0 są chwilowymi wartościami funkcji, które w trakcie obliczeń są optymalizowane.

$$e = \|(y - f(y', a', x'_0, x)) * x\| \quad (2)$$

Obliczone w Matlabie aproksymacje dobrze oddają charakterystykę przedstawianą przez pomiary, mimo iż nie zawsze przechodzi ona idealnie przez zakres odczytanych wartości. Obliczone parametry dla każdej aproksymaty zostały zestawione ze sobą w tabeli 5.2. Do dalszego użytku zostały wybrane parametry dla pola $1x2cm$, ponieważ są one najmniej zakłócone, jak również pola na stoliku mają dokładnie takie wymiary.

Tabela 5.2. Wyniki aproksymacji pomiarów przy założeniu, że $x_0 = 0$

Wymiary testowanego pola	y	a	x_0
Pole $1x1cm$	12,8559	26519,0	0
Pole $1x2cm$	0,4172	35108,0	0
Pole $2x2cm$	-2,0821	51905,0	0

5.1.2. Porównanie otrzymanych wyników z wynikami otrzymanymi przez innych badaczy

Otrzymane wyniki postanowiłem porównać i odnieść do rezultatów badań naukowców z całego świata. W tym celu wybrałem kilka prac, które opisują i badają właściwości Velostatu.

statu. Charakterystyką, którą wybrałem jest zależność pomiędzy pomiarami rezystancji pola taktylnego a naciskiem na to pole. To kryterium odrzuciło wiele prac, ponieważ wielu badaczy prezentowało zależność odczytywanego napięcia w zależności od nacisku. Ta charakterystyka bez znajomości szczegółów fizycznej realizacji czujnika była nieodwracalna i nie nadawała się do wykonywanych przeze mnie porównań.

Ostatecznie, aby móc dokonać porównania pomiędzy różnymi pracami, wybrałem kilka wartości nacisku, dla których odczytywałem z wykresu rezystancję czujnika, jeśli było to możliwe. Dodatkowo dla każdego z wybranych przypadków zapisałem w tabeli powierzchnię czujnika, który został wykorzystany w pomiarach.

Wybrane punkty pomiarowe zostały zebrane w tabeli 5.3. Nie dla każdego z analizowanych źródeł dostępne były wszystkie wartości pomiarów, dlatego też odczytałem tak wiele, jak to tylko było możliwe. Podczas przeglądania literatury można też było odnieść wrażenie, że niektórzy badacze nie wiedzą czym jest ciśnienie i jak jest ono obliczane na podstawie nacisku na badany obiekt. Z tego powodu musiałem odrzucić również kilka artykułów.

Tabela 5.3. Wybrane punty obciążenia Velostatu do porównania moich wyników z wynikami innych badaczy

Masa obciążnika [g]	Szacowany ciężar [N]
25	0,25
50	0,50
75	0,75
100	1,00
250	2,50
500	5,00
750	7,50
1000	10,00
1500	15,00
2000	20,00
2500	25,00

Wyniki porównania przeze mnie prac kilku badaczy można znaleźć w tabeli 5.4. Bardzo ważną uwagą dotyczącą tych badań jest fakt, że odczytane wartości są przybliżone, ponieważ w większości odczytywane są one z wykresów, a nie są one obliczane ze wzorów. Wykresy prezentowane przez różnych badaczy posiadają różne skale, dokładności oraz fizyczne rozmiary w publikacji, gdzie każda z wymienionych przyczyn wprowadza swoje niepewności odczytu wartości. Jednak pomimo tych niedokładności można wyraźnie zauważać pewne tendencje w przestudiowanej literaturze.

Na pierwszy rzut oka otrzymane porównanie jest bardzo chaotyczne i niejednoznaczne. W praktyce, faktycznie pozostawia wiele pola do interpretacji, ale jednocześnie pokazuje też wyraźne trendy.

5. Przeprowadzone badania i testy Velostatu

Tabela 5.4. Porównanie odpowiedzi rezystancyjnej materiału Velostat z wynikami innych badaczy

Ciążar [N]	Niniejsza praca (200mm ²) [Ω]	A. -R. A. Laaraibi (100mm ²) [Ω] [73]	A. Dzedzickis, nowy materiał (25mm ²) [Ω] [53]	A. Dzedzickis, stary materiał (25mm ²) [Ω] [53]
0,25	1405		6300	3500
0,50	703		4000	2700
0,75	469		2900	2000
1,00	351	200	2000	1500
2,50	141	80	800	500
5,00	71	50		
7,50	47	40		
10,00	36	30		
15,00	24	25		
20,00	18	20		
25,00	14	20		

Tabela 5.4. (c.d.) Porównanie odpowiedzi rezystancyjnej materiału Velostat z wynikami innych badaczy

Ciążar [N]	Marco Reps (40mm ²) [Ω] [54]	Anis Fatema (1600mm ²) [Ω] [74]	S. S. Suprapto (49mm ²) [Ω] [25]	Zaccaria Del Prete (24mm ²) [Ω] [75]	Xiaoyou Lin (100mm ²) [Ω] [76]
0,25				3000	
0,50				1250	
0,75				700	
1,00	600		5900	500	2500
2,50	370		3100	200	1900
5,00	170	2500	1800	150	1400
7,50	120		1200	100	1100
10,00	80	1200	800	75	900
15,00	70	800	600		600
20,00		620			550
25,00		500			500

Pierwszą wspólną i wyraźną cechą, dla wszystkich zestawionych wyników, jest charakterystyka pomiarów dla kolejnych wartości nacisku. Rezystancja ma w tym przypadku charakterystykę hiperbowiczną. Badacze jednak częściej przeliczali pomiary rezystancji na konduktancję, która przyjmowała wtedy charakterystykę liniową. Na potwierdzenie, że te charakterystyki są sobie równoważne, na równaniu (3) zapisałem zależność pomiędzy rezystancją R a konduktancją G .

$$G = \frac{1}{R} \quad (3)$$

Patrząc na same wartości rezystancji, jakie są odczytywane po przyłożeniu nacisku, można zauważyc, że występują bardzo duże rozbieżności pomiędzy poszczególnymi badaniami. Widać również, że powierzchnia pola czujnika nie była w tym przypadku kluczowym czynnikiem – nie ma wyraźnie zarysowanej zależności pomiędzy rezystancją a polem powierzchni czujnika.

Pewne światło na ten problem rzucają badania Andriusa Dzedzickisa [53], który wykonał pomiary zarówno nieużywanego, jak i wielokrotnie używanego materiału. W swojej pracy posiłkował się również zdjęciami AFM tych próbek materiału. Wyraźnie widać, że dłużej użytkowany materiał ma inną charakterystykę niż ten, który dopiero co został otwarty. Pomimo, że są to zmiany bardzo wyraźne (szczególnie w przypadku niewielkiego nacisku) to nie uzasadniają one aż tak dużych rozbieżności pomiędzy wynikami otrzymanymi przez poszczególnych badaczy [53].

Pomimo prób znalezienia racjonalnego wyjaśnienia dla tak dużych rozbieżności pomiędzy pomiarami nie udało mi się znaleźć jednoznacznej przyczyny. Na pewno na pomiary wpływała w pewnym stopniu metodologia samych pomiarów – nie wszystkie były wykonywane w odpowiednio wyposażonych warsztatach. Dodatkowo, dużą rolę mogła też odgrywać sama konstrukcja badanego czujnika. Ostatnim możliwym czynnikiem jest możliwość, że firma 3M produkuje folię Velostat o zróżnicowanych charakterystykach w poszczególnych partiach produktu.

5.2. Kompensacja szumów czujnika taktylnego

Pomiary odczytywane z czujnika taktylnego nie są pozbawione wad. Są one mocno zaszumione, szczególnie w momencie, kiedy nacisk na pole jest niewielki. Nawet przy braku jakiegokolwiek przedmiotu na stoliku pomiary nacisku nie są zerowe (wartość 4095) tylko wykazują niewielki nacisk, który się delikatnie waha. Sytuacja ta widoczna jest na rysunku 5.3, gdzie pomimo braku żadnego nacisku na stolik widoczne są pomiary wynoszące nawet ~ 3900 . Wartość ta oznacza teoretyczny nacisk o wartości $\sim 4\text{g}$, tylko na to pojedyncze pole czujnika. Wartość ta wydaje się niewielka, ale warto zauważyc, że każde z kilkuset pól czujnika wprowadza niewielki nacisk, który po zsumowaniu daje znaczące wartości.

Widoczne na pomiarach szумy, występujące mimo prób mechanicznego ich zniwelowania (przez modyfikację i poprawę samego stolika), mogą mieć kilka powodów. Najważniejszym powodem jest guma, która jest położona na górze stolika. Guma ta sama w sobie posiada masę i jest ona widoczna na odczytach sensora. Kolejnym powodem są niedokładności w budowie mechanicznej czujnika. Dodatkowo, czujnik został wykonany tylko w jednym egzemplarzu, który na dodatek był ręcznie składany. Niedoskonałości na czujniku mogą być spowodowane przez ewentualne fałdy na nieprecyzyjnie przyklejonych paskach taśmy miedzianej, naprężeniach gumy ochraniającej czujnik, czy też sztywnych (klejonych) ściankach bocznych czujnika. Dodatkowo, jak w każdym elektronicznym

5. Przeprowadzone badania i testy Velostatu

```
▶ 00 = {list} <class 'list'>: [4020.0, 4039.0, 4041.0, 4023.0, 4023.0, 4017.0, 3999.0, 4032.0, 4012.0, 4033.0, 4029.0, 4035.0, 4047.0, 4051.0, 4049.0, 4049.0]
▶ 01 = {list} <class 'list'>: [4025.0, 4049.0, 4047.0, 4027.0, 4030.0, 4041.0, 4013.0, 4003.0, 4013.0, 4007.0, 4006.0, 3994.0, 4037.0, 4043.0, 4037.0, 4050.0]
▶ 02 = {list} <class 'list'>: [4012.0, 4030.0, 4032.0, 4013.0, 4026.0, 4033.0, 4028.0, 4004.0, 4014.0, 4020.0, 3999.0, 3985.0, 4020.0, 4019.0, 4040.0, 4034.0]
▶ 03 = {list} <class 'list'>: [4035.0, 4040.0, 4023.0, 4020.0, 4021.0, 4045.0, 4028.0, 3999.0, 4006.0, 4008.0, 4001.0, 4008.0, 4038.0, 4036.0, 4014.0, 4021.0]
▶ 04 = {list} <class 'list'>: [4049.0, 4039.0, 4039.0, 4028.0, 4027.0, 4037.0, 4037.0, 4009.0, 4013.0, 4029.0, 4005.0, 4043.0, 4051.0, 4041.0, 4051.0, 4046.0]
▶ 05 = {list} <class 'list'>: [4036.0, 4026.0, 3999.0, 3970.0, 3987.0, 4035.0, 4015.0, 3995.0, 3952.0, 3982.0, 3974.0, 4002.0, 4030.0, 4036.0, 4034.0, 4032.0]
▶ 06 = {list} <class 'list'>: [4036.0, 4038.0, 4030.0, 4025.0, 4042.0, 4046.0, 4023.0, 4007.0, 4010.0, 4020.0, 4008.0, 4026.0, 4034.0, 4032.0, 4039.0, 4019.0]
▶ 07 = {list} <class 'list'>: [4040.0, 4040.0, 4017.0, 4025.0, 4027.0, 4045.0, 4039.0, 4021.0, 4015.0, 4024.0, 4007.0, 4021.0, 4049.0, 4037.0, 4035.0, 4033.0]
▶ 08 = {list} <class 'list'>: [4043.0, 4039.0, 4023.0, 4035.0, 4035.0, 4038.0, 4033.0, 4037.0, 4035.0, 4031.0, 4021.0, 4021.0, 4034.0, 4031.0, 4006.0, 4029.0]
▶ 09 = {list} <class 'list'>: [4023.0, 4029.0, 4017.0, 4018.0, 4019.0, 4016.0, 4025.0, 4005.0, 3977.0, 4000.0, 3974.0, 3997.0, 4022.0, 4038.0, 4031.0, 4036.0]
▶ 10 = {list} <class 'list'>: [4029.0, 4041.0, 4019.0, 3989.0, 4015.0, 4023.0, 4005.0, 3971.0, 3973.0, 3987.0, 3978.0, 4009.0, 4025.0, 4037.0, 4021.0, 4034.0]
▶ 11 = {list} <class 'list'>: [4018.0, 4031.0, 4002.0, 3950.0, 3994.0, 4012.0, 4012.0, 3960.0, 3950.0, 3950.0, 3970.0, 4020.0, 4036.0, 4032.0, 4030.0, 4014.0]
▶ 12 = {list} <class 'list'>: [4019.0, 4021.0, 3971.0, 3905.0, 3967.0, 4003.0, 3964.0, 3964.0, 3979.0, 3983.0, 3993.0, 4025.0, 4039.0, 4029.0, 4005.0, 4017.0]
▶ 13 = {list} <class 'list'>: [4022.0, 4019.0, 3995.0, 3966.0, 3990.0, 4003.0, 3978.0, 3938.0, 3965.0, 3976.0, 3970.0, 4024.0, 4031.0, 3978.0, 4030.0, 4026.0]
▶ 14 = {list} <class 'list'>: [4016.0, 4026.0, 3991.0, 3999.0, 4020.0, 4037.0, 4006.0, 4004.0, 4010.0, 3991.0, 3997.0, 4038.0, 4041.0, 4046.0, 4039.0, 4047.0]
▶ 15 = {list} <class 'list'>: [3874.0, 3920.0, 3950.0, 3970.0, 3995.0, 4010.0, 3984.0, 3993.0, 4018.0, 4024.0, 4025.0, 4049.0, 4052.0, 4042.0, 4052.0, 4056.0]
```

Rysunek 5.3. Prezentacja problemu kompensacji - faktyczne odczyty czujnika taktylnego przy braku nacisku

układzie, istnieją również zakłócenia na sygnale analogowym, takie jak przesłuchy, czy różnego rodzaju upływności w układzie. Zjawisko to, na samym Velostacie, zostało opisane w pracach Lakshmanana, ale uznałem, że w moim przypadku zakłócenia te są na tyle nieznaczne, że nie potrzebują specjalnego wyodrębniania [77].

Biorąc pod uwagę opisane problemy i ich charakterystykę podjąłem próbę zniwelowania ich wpływu. Wytypowałem kilka sposobów kompensacji tych odczytów, tak aby na dalszym etapie zerowy nacisk na poszczególne pola czujnika był odzwierciedlany przez wartość 4095 (maksymalna wartość pomiaru 12-bitowego czujnika). Testowane przeze mnie metody kompensacji były następujące:

- brak kompensacji (pomiar referencyjny),
- kompensacja na podstawie najgorszego odczytu z całego stolika,
- przesunięcie odczytów o wartość kalibracji każdego pola taktylnego,
- skalowanie odczytów do wartości kalibracji każdego pola taktylnego,
- filtr odcinający szумy pomiarowe na stałej wartości 4000,
- filtr odcinający szumy pomiarowe na stałej wartości 3950,
- filtr odcinający szumy pomiarowe na stałej wartości 3900,
- filtr odcinający szumy pomiarowe na stałej wartości - 0,99 wartości kalibracji każdego pola taktylnego,
- filtr odcinający szumy pomiarowe na stałej wartości - 0,98 wartości kalibracji każdego pola taktylnego.

Dalsza część metodologii polega na zsumowaniu nacisków na każde pole czujnika taktylnego (wykorzystując zależności wykazane w rozdziale 5.1.1). To podejście umożliwiło zestawienie w jednej liczbie, pomiarów z wielu pól czujnika dla każdego pomiaru. Dodatkową zaletą jest fakt, że te same pomiary mogą zostać później wykorzystane do ustalenia zależności pomiędzy wagą obliczoną przez czujnik, a faktyczną wagę przedmiotu (opisane w rozdziale 5.2.2).

Pierwszy jest pomiar bez wykorzystania żadnej metody kompensacji. Jest to pomiar typowo referencyjny, aby zobaczyć jak względem niego zachowują się pozostałe metody.

Drugi pomiar wykorzystuje znajomość nacisków na całym stoliku (również na nieistniejących polach, które są poza stolikiem i nigdy nie zostanie na nich wywarty nacisk). Odczytywana jest najwyższa uzyskana na całym stoliku wartość i traktowana jest ona jako wartość, dla której na pole taktylne wywierany jest zerowy nacisk. Pomiary ze wszystkich pól są na tej podstawie przesuwane o tę samą wartość tak, aby dla wybranego zerowego nacisku wartość pola wynosiła maksymalne 4095. Wartość ta jest inna w każdej badanej serii danych z czujnika i opiera się tylko na aktualnych danych.

Trzeci i czwarty pomiar bazują na wartości kalibracji, która jest wykonywana automatycznie przy uruchomieniu stolika lub ręcznie przez użytkownika. Pierwsza z metod wykorzystuje przesunięcie odczytanej wartości o wartość odczytaną przy kalibracji tak, że wartość odczytana podczas kalibracji jest traktowana jako referencyjne 4095. Druga metoda na wykorzystanie wartości odczytanej podczas kalibracji zakłada przeskalowanie odczytanej wartości w taki sposób, aby proporcjonalnie zwiększyć jej wartość w takim samym stopniu, w jakim wartość z kalibracji odbiega od idealnej.

Kolejne 3 metody kompensacji polegają na brutalnym ograniczeniu czułości stolika w najbardziej czułym zakresie. Wszystkie wartości wyższe niż założona stała są sprowadzane do wartości tej stałej. Kolejnym etapem jest przeskalowanie otrzymanych pomiarów, aby odpowiadały one pełnemu zakresowi możliwym do odczytania pomiarów.

Ostatnie 2 metody kompensacji również bazują na wykonanej uprzednio kalibracji czujnika. Wykorzystują one również ucinanie czułości pól taktylnych. Ograniczanie czułości nie jest wykonywane jednak na sztywno określonych wartościach, tylko jest bezpośrednio uwarunkowane wartością odczytaną podczas kalibracji każdego pola. Każda odczytana wartość pola wyższa niż ta zdefiniowana przez ustalony procent wartości kalibracji jest zmniejszana do tego ustalonego procentu. Na kolejnym etapie każde z pól czujnika jest przemnażane przez własną wartość maksymalną, tak aby znormalizować każde pole do maksymalnej wartości odczytu czujnika (4095).

Każda z kolejnych metod kompensacji w pewien sposób ucina część informacji ze stolika, w ten sposób zmniejszając jego czułość i upośledzając go w pewnym stopniu. Dlatego też chciałem sprawdzić jaki poziom upośledzenia czujnika będzie optymalny dla tego, aby nie stracić zanadto dokładności pomiarów.

Do testów kompensacji wykonałem pomiary dla 26 przedmiotów po 20 powtórzeń (plus pomiar kalibracyjny na pustym stoliku). Powtórzenia były wykonywane w różnych miejscach stolika, z wyłączeniem samej krawędzi, gdzie pomiary mogą być przekłamane z uwagi sztywną krawędź boczną stolika. Podczas pomiarów, oprócz samej wagi położnego przedmiotu, zapisywałem również średnicę jego podstawy styku (większość testowanych przedmiotów miała okrągłą podstawę i posiadała zauważalny rant na zewnętrznej części podstawy), aby móc również sprawdzić, jak dużą rolę w dokładności pomiarów

5. Przeprowadzone badania i testy Velostatu

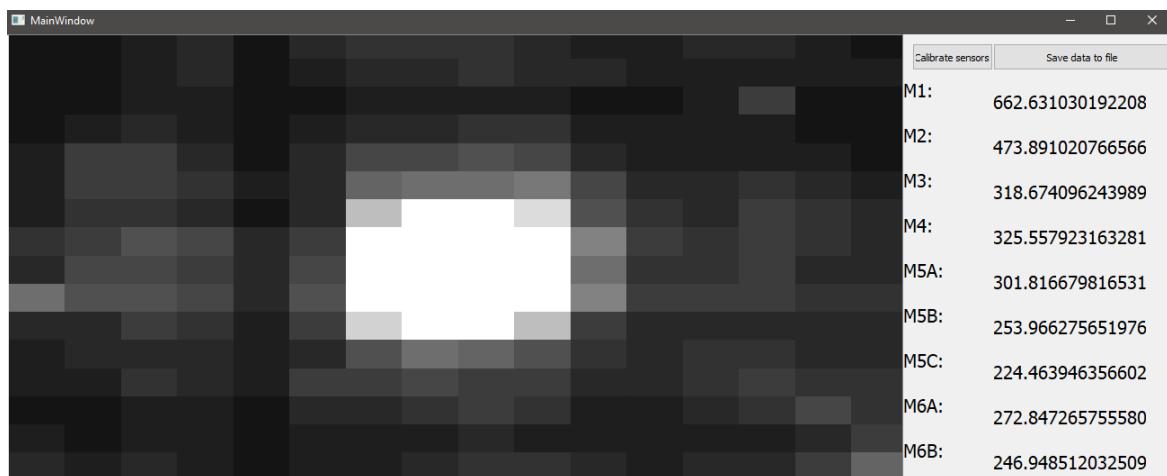


Rysunek 5.4. Stanowisko badawcze do badania faktycznej wagi przedmiotów na stoliku

gra rozłożenie nacisku na większej liczbie pól taktylnych. Stanowisko pomiarowe, wraz z położonym na stoliku przykładowym przedmiotem, widoczne jest na rysunku 5.4.

Zostało również wykonane okno podglądu tego, jakie wartości przedstawiają aktualnie poszczególne metody kompensacji, które jest widoczne na rysunku 5.5. Widoczne wyniki były podstawą do dalszych obliczeń. Do zapisu całej serii danych służy widoczny u góry przycisk. Jest tam również dostępny przycisk umożliwiający kalibrację stolika w dowolnym momencie jego korzystania. Widoczny interfejs użytkownika nie jest dopracowany i przyjazny dla użytkownika, ale był on dla mnie tylko narzędziem deweloperskim, a nie docelowym efektem pracy.

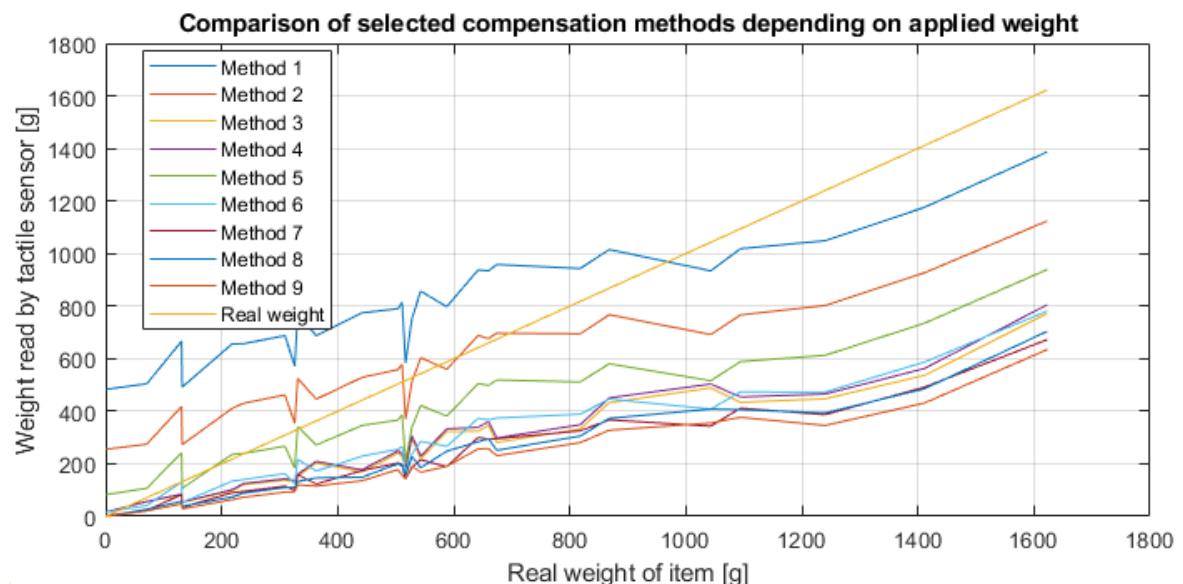
Zebrane dane zostały uśrednione dla każdego przedmiotu oraz metody kompensacji. Analizując dane nie zauważylem występowania znacznych pojedynczych wartości odsta-



Rysunek 5.5. Podgląd w czasie rzeczywistym na wartości wagi obliczane z wykorzystaniem kolejnych metod kompensacji szumów

jących, dlatego do średniej wliczają się wszystkie pobrane próbki. Wyniki i zestawienie wykonanych pomiarów widoczne jest na rysunku 5.6. Wykres przedstawia średnią wartość pomiarów wagi zmierzonych przez stolik w zależności od rzeczywistej wagi obiektu.

Na wykresie, oprócz wszystkich metod kompensacji, widoczna jest również prosta odzwierciedlająca idealny pomiar wagi, który powinien być otrzymany w wyniku obliczeń. Jak można zauważyć linia ta znacznie odbiega od tych otrzymanych podczas samej kompensacji. Tą częścią badań zająłem się jednak później, a uzyskane wyniki znajdują się w rozdziale 5.2.2.



Rysunek 5.6. Porównanie wszystkich pomiarów uzyskanych dla wszystkich metod kompensacji

Na wykresie przedstawionym, na rysunku 5.6 można też zauważać kilka wyraźnych wahnięć, gdzie przy tej samej rzeczywistej masie przedmiotu wartość odczytywana jest bardzo różna. Wspomniane rozbieżności wynikają z testowanych przedmiotów i tego, że kilka posiadało trochę inną charakterystykę spodu – nie miały wyraźnego rantu na obwodzie podstawy.

Po wykonaniu wykresów charakterystyki wyznaczyłem najważniejsze parametry, którymi kierowałem się podczas wyboru najlepszej metody kompensacji szumów:

- zerowa odczytana waga przy zerowym faktycznym nacisku na stolik,
- metody oparte na z góry określonej wartości liczbowej mają mniejszą użyteczność,
- możliwie jak najmniejsze zmniejszenie czułości stolika.

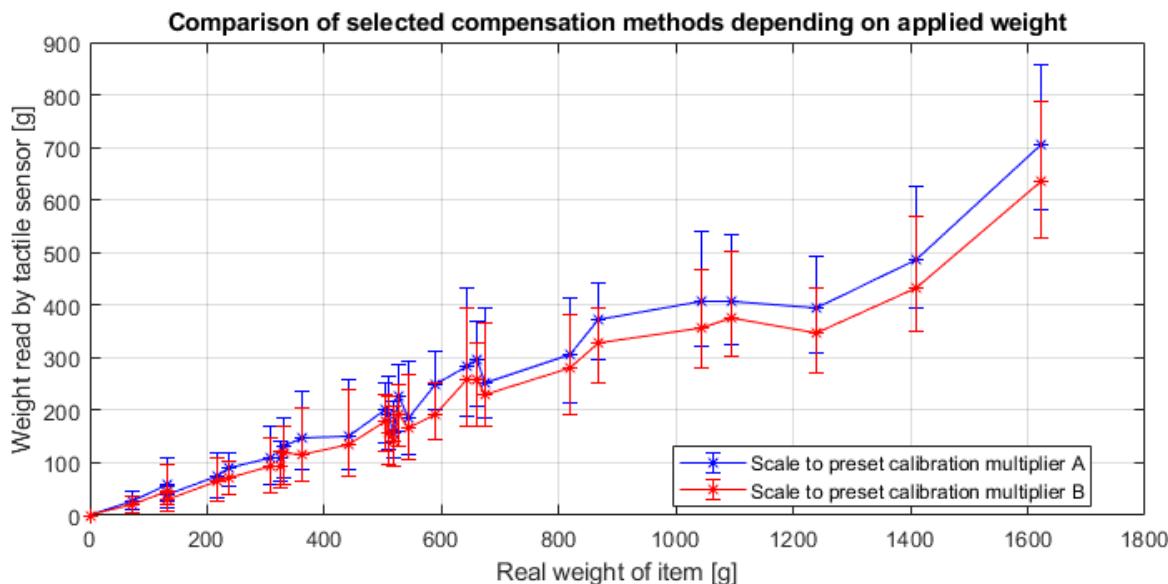
Biorąc pod uwagę pierwszy z wymienionych warunków, do wyboru zostały mi tylko ostatnie 3 metody kompensacji (na podstawie rysunku 5.6). Wszystkie pozostałe metody kompensacji, dla zerowego nacisku na stolik, nie wykazywały zerowego odczytu, a ten warunek był dla mnie kluczowy przy wyborze metody kompensacji.

Metoda siódma również posiada wadę, ponieważ oparta jest ona na z góry ustalonej wartości filtrowania, co może powodować, że w szczególnych przypadkach będzie

5. Przeprowadzone badania i testy Velostatu

to wartość niewystarczająca. Dodatkowo, stała wartość jest w tym przypadku ustalona bardzo nisko, co powoduje bardzo duże ograniczenie czułości stolika w przypadku lekkich przedmiotów. Z tych powodów metoda ta również została przeze mnie odrzucona.

Pomiędzy pozostałymi dwoma metodami wykonałem dodatkowe, szczegółowe porównanie widoczne na rysunku 5.7. Dodatkowo zaznaczone zostały na wykresie słupki błędów dla poszczególnych pomiarów.



Rysunek 5.7. Porównanie pomiędzy ostatnimi dwoma wybranymi metodami kompensacji

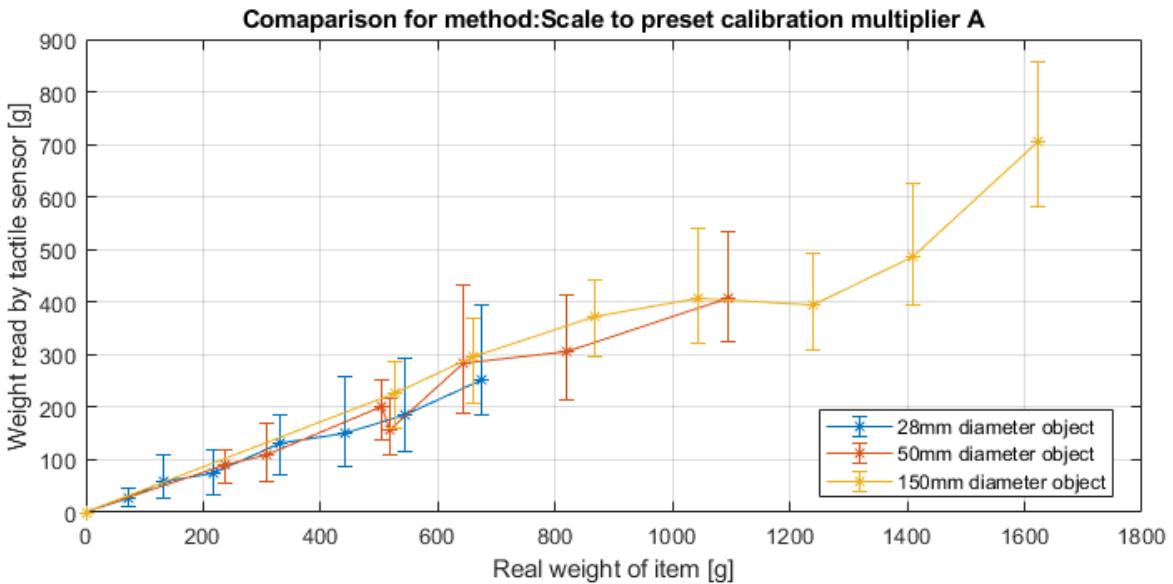
Ostatecznie, według postawionych przeze mnie kryteriów, najlepiej wypada metoda 8. Jest to metoda, która idealnie trafia w postawione przeze mnie kryteria, dodatkowo nie zmniejsza ona zbytnio czułości samego czujnika. Dlatego też ta metoda - **filtrowany szumy pomiarowe na stałej wartości - 0,99 wartości kalibracji każdego pola taktylnego**, została wybrana do dalszych badań. Miałem obawy, że czujnik po długim użytkowaniu bez kalibracji w międzyczasie będzie tracił swoje dopasowanie i samoistnie generował szумy, w tym przypadku lepiej sprawdziłaby się metoda 9, ale w toku moich badań nie zaobserwowałem takiego zachowania czujnika.

5.2.1. Zależność od powierzchni styku z czujnikiem

Przy okazji sprawdziłem, czy czujnik wykazuje widoczną zależność pomiędzy przedmiotami o różnych średnicach podstawy, ale tej samej masie. W tym celu stworzyłem wykres przedstawiający pomiary dla 3 testowanych średnic obiektu:

- 28mm,
- 50mm,
- 150mm.

Sam wykres widoczny jest na rysunku 5.8. Porównanie zostało wykonane dla wybranego na poprzednich etapach sposobu kompensacji.



Rysunek 5.8. Zależność pomiędzy przedmiotami o różnych średnicach podstawy, przy wykorzystaniu wybranej metody kompensacji

Na wspomnianym wykresie można zaobserwować, że istnieje pewna zależność pomiędzy średnicą podstawy a nachyleniem wykresu. Przy mniejszej średnicy podstawy wykres jest bardziej poziomy, co oznacza, że przy stolik jest wtedy mniej czuły na przykładany nacisk. Ciężko było mi przetestować duże obciążenia stolika dla niewielkich średnic obiektu, ponieważ obiekty te były wtedy dość niestabilne.

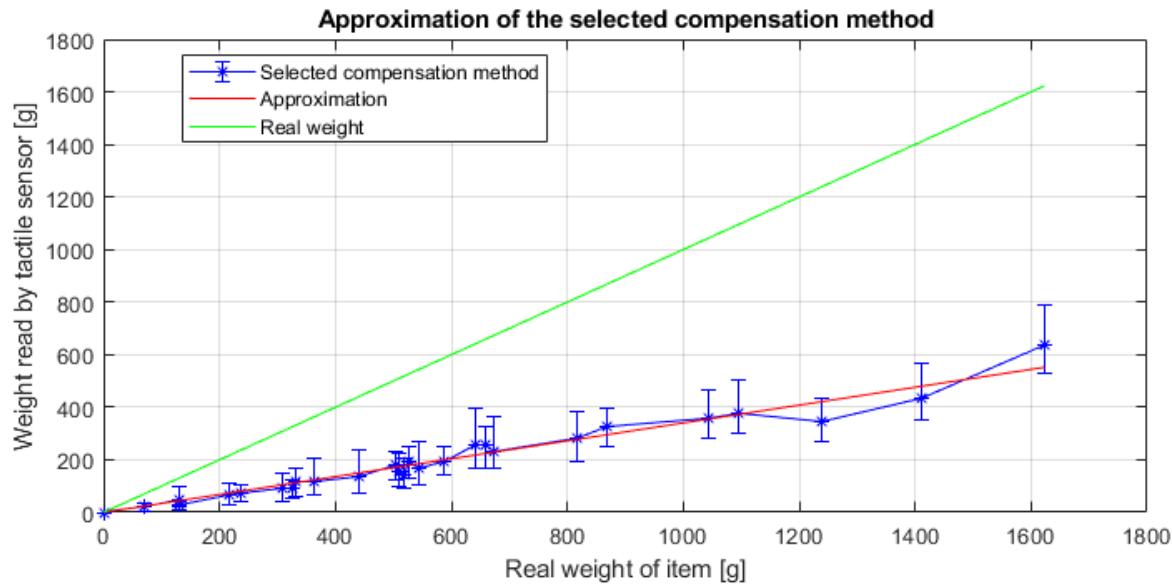
Ostatecznie postanowiłem, że zignorowanie tej zależności przy szacowaniu wagi obiektu nie wniesie dużych zakłóceń, dlatego zawiesiłem dalsze badania w tym zakresie. Kluczowe jest tutaj też zauważenie, że rozbieżności pomiarami dla każdego z obiektów wnoszą dużo większe odchylenia, niż różnice w powierzchni styku przedmiotów.

5.2.2. Aproksymacja pomiarów do wykorzystania w pomiarach wagi przedmiotu

Przeprowadzone badania dają dobrą podstawę do szacowania wagi położonego na stoliku przedmiotu. Badania te wiążą rzeczywistą wagę z odczytami czujnika taktylnego, więc nic nie stoi na przeszkodzie aby zależność tą odwrócić.

Aproksymację wykonałem linią prostą, ponieważ dobrze oddaje ona charakterystykę czujnika. Dodatkowo, pozwala to w bardzo prosty sposób powiązać rzeczywistą wagę z pomiarami – przez zwykłe przemnożenie pomiarów. Do aproksymacji wykorzystałem wszystkie dostępne pomiary potraktowane wybraną metodą kompensacji. Funkcją, którą minimalizowałem jest zwykła norma Euklidesowa (4), gdzie e jest obliczonym błędem aproksymacji, współczynnik a jest nachyleniem prostej, wektor y jest wartościami zmierzonymi przez czujnik taktylny, wektor x jest rzeczywistymi wartościami przedmiotów. Wyniki wykonanej przeze mnie aproksymacji są widoczne na wykresie przedstawionym na rysunku 5.9.

5. Przeprowadzone badania i testy Velostatu



Rysunek 5.9. Aproksymacja linią prostą wykonana na wybranej metodzie kompensacji w porównaniu do realnej wagi przedmiotu

$$e = \|(y - a * x)\| \quad (4)$$

Jak można zauważyć aproksymacja przebiegła całkiem pomyślnie. W jej wyniku otrzymałem prostą, która dość dobrze oddaje charakterystykę pomiarów. Przechodzi ona również przez wszystkie zakresy błędów, co dodaje jej wiarygodności. Sama prosta ma nachylenie **0,3401**, natomiast prosta rzeczywistej wagi ma współczynnik nachylenia wynoszący 1,0. W praktyce oznacza to, że można otrzymać szacowaną wagę przedmiotu (5), gdzie y jest finalnie szacowaną wagą przedmiotu, a x jest wstępnie szacowaną wagą przedmiotu.

$$y = \frac{x}{0,3401} \quad (5)$$

6. Scenariusze wykorzystania inteligentnego stolika

Mając platformę sprzętową, która jest stabilna, a jej działanie zostało dobrze opisane, czas wrócić do celów opisanych w rozdziale 1.1. Zaplanowane zostały scenariusze, które pozwolą sprawdzić przedstawione w rozdziale tym, na rysunku 1.2, przykłady wykorzystania inteligentnego stolika. Same scenariusze, poza samym dowodem poprawnego działania systemu, mają na celu sprawdzić również przypadki graniczne i negatywne – kiedy robot musi odmówić wykonania zadania, z powodu wysokiej możliwości niepowodzenia. Przykładowym powodem odmówienia zadania jest brak wykrytego przedmiotu lub jego położenie na krawędzi stolika.

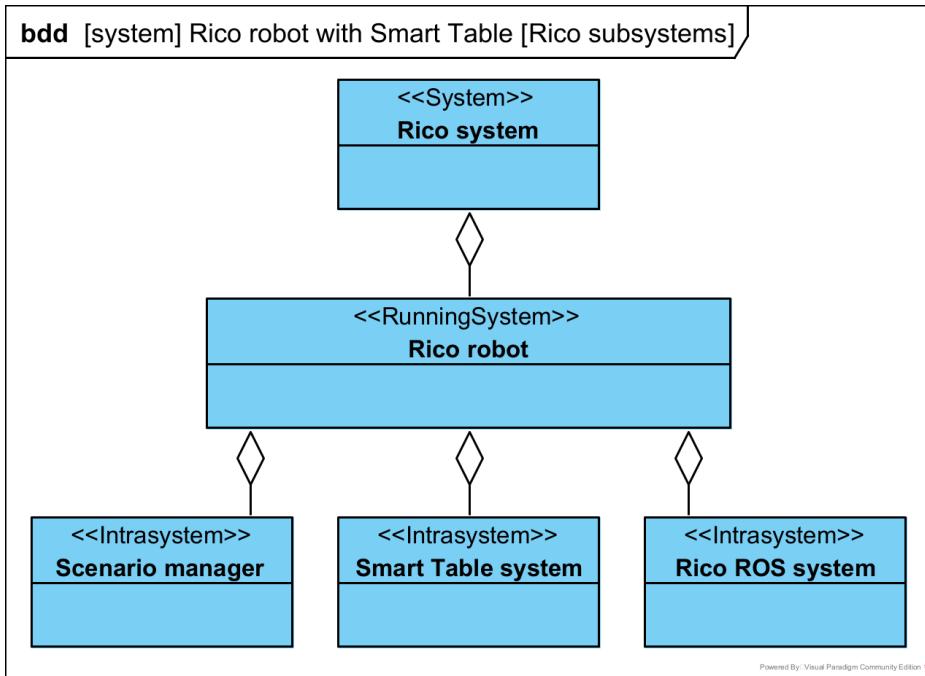
W tym celu zostały zaplanowane scenariusze użycia stolika w interakcji ze środowiskiem. Scenariusze te testują jednocześnie wiele różnych funkcji stolika. Korzystają one również z pracy innych osób z zespołu laboratoryjnego. Scenariusze, które zostały wybrane do wykonania, oparte są na scenariuszach wykonywanych wcześniej na tym robocie przez innych dyplomantów. W szczególności jeden ze scenariuszy był wzorowany na tym, który w swojej pracy inżynierskiej zaprezentował Stanislau Stankevich. Polegał on na prośbie skierowanej do robota o przywiezienie herbaty. Po modyfikacji scenariusza do moich celów przyjąłem go jako punkt kluczowy moich badań. Ulepszeniem działania robota w stosunku do scenariusza Stanislawa jest brak konieczności potwierdzenia robotowi wykonanych czynności [46].

Ostatecznie do wykonania zostały wybrane dwa scenariusze na robocie:

- dostarczenie osobie starszej herbaty (6.1),
- odwiezienie pustego naczynia do kuchni (6.2).

Przed samym należy jednak koncepcyjnie podzielić system robota na intrasystemy wykonujące poszczególne zadania i komunikujące się ze sobą. W tym celu z działającego systemu robota zostały wyodrębnione trzy mniejsze intrasystemy, z czego dwa z nich są bezpośrednio związane z inteligentnym stolikiem. Wykonany podział widoczny jest na rysunku 6.1. Poszczególne bloki widoczne na tym diagramie symbolizują:

- *Rico system* – system skupiający wszystkie elementy związane z Rico,
- *Rico robot* – cały system działającego robota Rico,
- *Scenario manager* – koncepcyjny intrasystem zawierający elementy odpowiedzialne za zarządzanie wykonaniem scenariusza,
- *Smart Table system* – koncepcyjny intrasystem zawierający elementy odpowiedzialne za prawidłowe działanie inteligentnego stolika,
- *Rico ROS system* – koncepcyjny intrasystem zawierający elementy odpowiedzialne za działanie istniejącej platformy robota Rico.



Rysunek 6.1. Podział działającego systemu robota Rico na koncepcyjne intrasystemy

6.1. Scenariusz dostarczenia herbaty

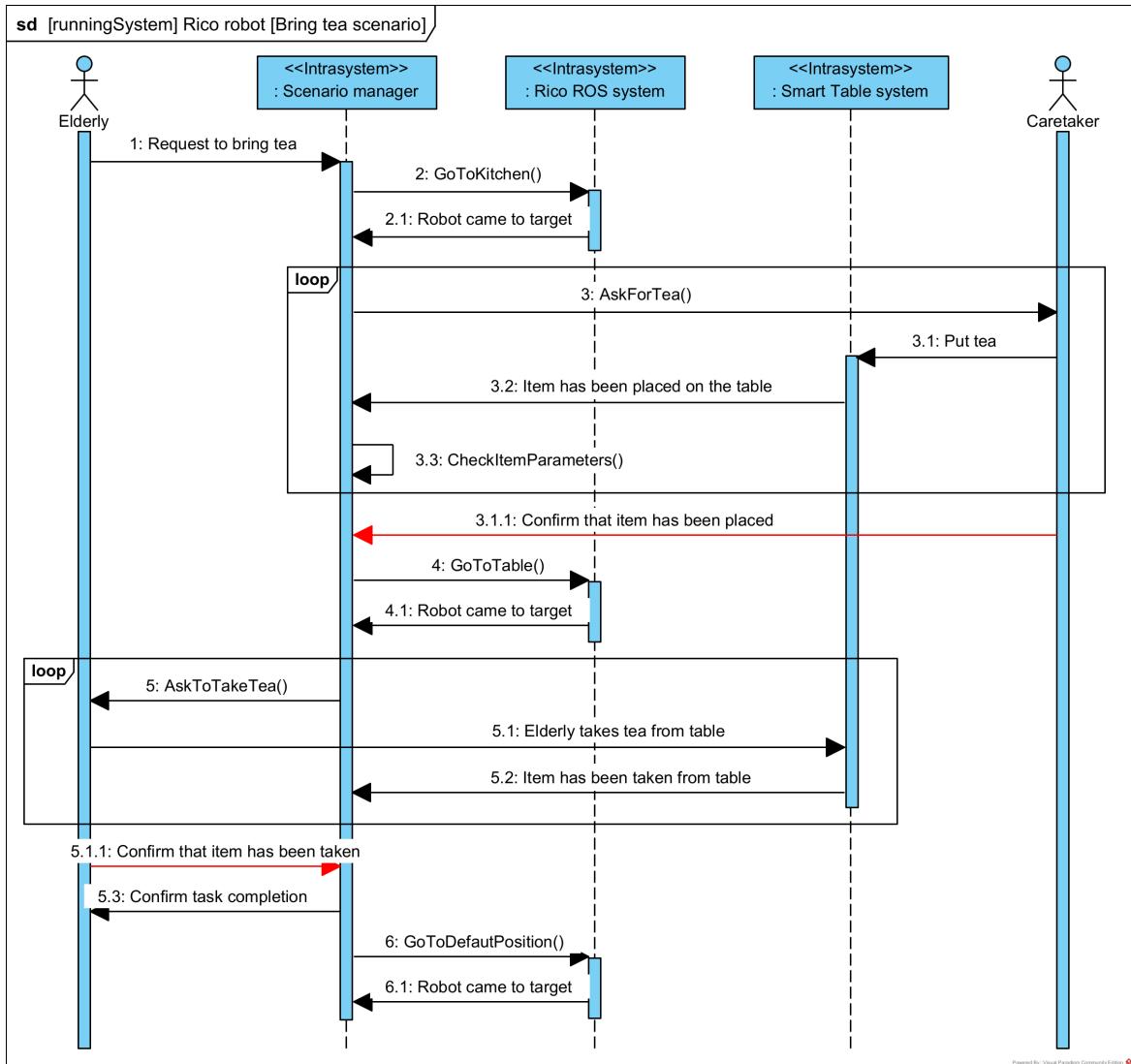
Jako najczęściej wykonywany w laboratorium scenariusz pokazowy, jest on już bardzo dobrze znany w zespole laboratoryjnym. Zakładany przebieg tego scenariusza uwzględniającego wprowadzone przez mnie modyfikacje został przedstawiony na rysunku 6.2.

Na diagramie kolorem czerwonym zostały oznaczone czynności, które musiały zostać podjęte przez człowieka w poprzednich odsłonach tego scenariusza, ale wykorzystanie inteligentnego stolika wyeliminowało tę konieczność. Z pozoru nie wygląda to jak bardzo duże ułatwienie, ale w praktyce przyspiesza to pracę z robotem i szybkość jego działania. Dodatkowo potwierdzanie każdej, najmniejszej operacji jest dla użytkownika dość uciążliwe, poprzez konieczność wypowiedzenia za każdym razem słowa klucz aktywującego rozpoznawanie mowy przez robota. Wykorzystanie stolika znacząco upraszcza tą kwestię. To zadanie automatyzacji potwierdzania położenia i zdjęcia przedmiotu ze stolika zostało postawione już na początku pracy w przypadku użycia *UC4*.

Inteligentny stolik pełni w tym zadaniu funkcję potwierdzenia poprawności położenia przedmiotu – kubka z herbatą. Najpierw stolik ocenia położenie kubka na nim. Jest ono kluczowe, ponieważ przewożony jest gorący płyn, który przy rozaniu może uszkodzić samego robota bądź też osoby w jego otoczeniu (wykorzystywany przypadek użycia *UC2*).

Kolejnym sprawdzanym parametrem jest waga położonego przedmiotu. Należy wziąć pod uwagę, że odczytana waga może być obarczona sporym błędem, więc jest głównie wskaźnikiem pomocniczym. Biorąc pod uwagę ten czynnik sprawdzane jest, czy waga przedmiotu mieści się w zakładanym zakresie (wykorzystywany przypadek użycia *UC5*).

Jeśli inteligentny stolik potwierdzi, że przedmiot jest prawidłowy to wysyła sygnał,



Rysunek 6.2. Diagram sekwencji dla przykładowego scenariusza przywożenia herbaty dla osoby starszej

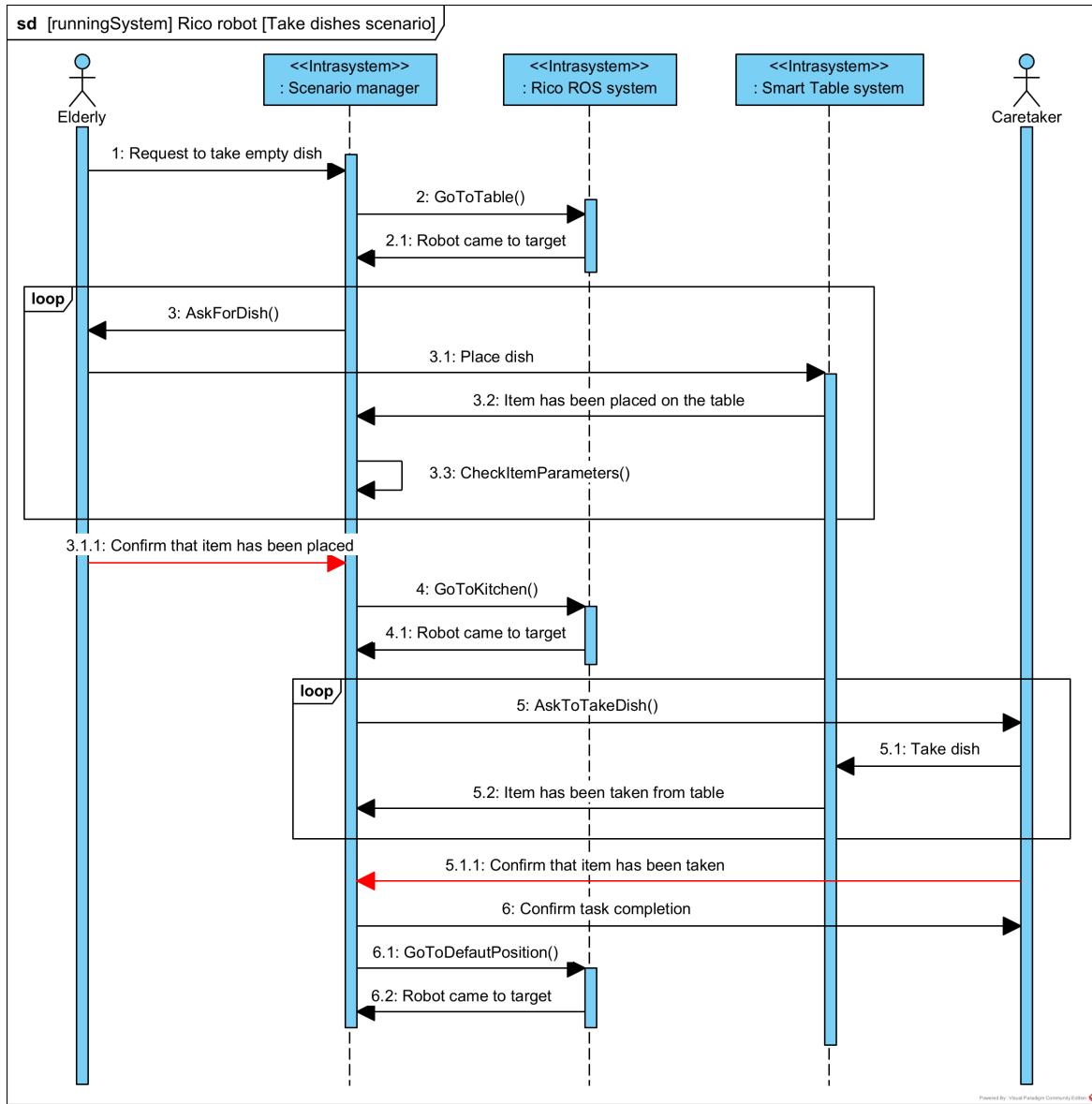
że robot może jechać do spragnionego użytkownika. W przeciwnym wypadku robot prosi opiekuna o poprawne położenie kubka z herbatą lub położenie odpowiednio zapełnionego naczynia.

Po otrzymaniu prawidłowego przedmiotu robot jedzie do osoby starszej, która odbiera kubek. Fakt odebrania jest monitorowany przez robota za pomocą intelligentnego stolika (wykorzystywany przypadek użycia *UC4*). Po wykryciu, że kubek został odebrany robot jedzie na miejsce, gdzie czeka na kolejne zadania.

6.2. Scenariusz odwiezienia pustego naczynia

Kolejny wykonywany przez robota scenariusz jest po części kontynuacją i rozszerzeniem scenariusza z przywiezieniem herbaty. Jednak zadania stawiane przed stolikiem

6. Scenariusze wykorzystania intelligentnego stolika



Rysunek 6.3. Diagram sekwencji dla przykładowego scenariusza odwożenia pustego naczynia

w tym scenariuszu są nieco inne. Pełne wykonanie tego scenariusza zostało przedstawione na rysunku 6.3.

W przypadku tego scenariusza, tak jak w przypadku 6.1, na diagramie 6.3 na czerwono zaznaczone są operacje, które zostały zautomatyzowane dzięki wykorzystaniu intelligentnego stolika i są obecnie zbędne.

Zadanie rozpoczyna się poprzez prośbę osoby starszej o odwiezienie pustego kubka do kuchni. Rico jedzie wtedy do osoby starszej, aby odebrać od niej naczynie. Podczas odbierania naczynia sprawdzane jest poprawne położenie kubka oraz jego waga. Po potwierdzeniu przez robota, że jest to poprawny przedmiot robot jedzie do kuchni, gdzie opiekun odbiera kubek. Robot po wykryciu tego faktu jedzie z powrotem do pozycji domyślnej.

Same zadania stolika są bardzo podobne, jak te opisane w rozdziale 6.1, dlatego przy opisie zadania nie wchodziłem w szczegóły jakie zadania pełni stolik. W tym scenariuszu same potwierdzenia poprawności położenia przedmiotu nie są tak krytyczne (przewożony kubek jest pusty), jeśli chodzi o bezpieczeństwo robota i otoczenia. Nie zmienia to jednak podejścia inteligentnego stolika do wykonywanych przez niego pomiarów.

6.3. Pokrycie przypadków wykorzystania inteligentnego stolika przez scenariusze

Scenariusze przedstawione w tym rozdziale zostały zaprojektowane w taki sposób, że sprawdzają 3 z 5 zdefiniowanych przypadków użycia. Są to:

- *UC2* – Lokalizacja obiektów na stoliku.
- *UC4* – Zautomatyzowane potwierdzenie bądź zaprzeczenie wykonanej przez człowieka akcji. Możliwe do sprawdzenia tylko w dynamicznych testach.
- *UC5* – Szacunkowa waga położonego przedmiotu.

W tym przypadku sprawdzone nie zostały:

- *UC1* – Monitorowanie statusu przedmiotów położonych na stoliku.
- *UC3* – Identyfikacja przedmiotów położonych na stoliku.

Monitorowanie przedmiotów nie zostało sprawdzone w scenariuszach, ponieważ wymagałoby ono dodania do scenariusza dodatkowego zewnętrznego obserwatora, którego jedynym zadaniem byłaby obserwacja wszystkich danych. Postanowiłem, że na tym etapie dodatkowy system monitorowania nie jest wymagany.

Identyfikacja położonych na inteligentnym stoliku przedmiotów również nie została wykorzystana w praktycznych scenariuszach. Dane zwarcane przez system identyfikujący miały zbyt niską dokładność, aby opierać na nich decyzje robota.

7. Implementacja węzłów ROS wykonujących zaplanowane scenariusze

Na podstawie zaplanowanych scenariuszy wykonana została implementacja poszczególnych intrasystemów w systemie działającego Rico. Implementacja ta miała na celu obsłużyć zarówno czujnik taktylny, jak i wykonać scenariusze opisane w rozdziale 6. W tym rozdziale zostało opisane działanie dwóch intrasystemów:

- *Smart Table system* – intrasystem odpowiedzialny za działanie inteligentnego stolika (7.2),
- *Scenario manager* – intrasystem odpowiedzialny za wykonanie i nadzorowanie scenariuszy (7.3).

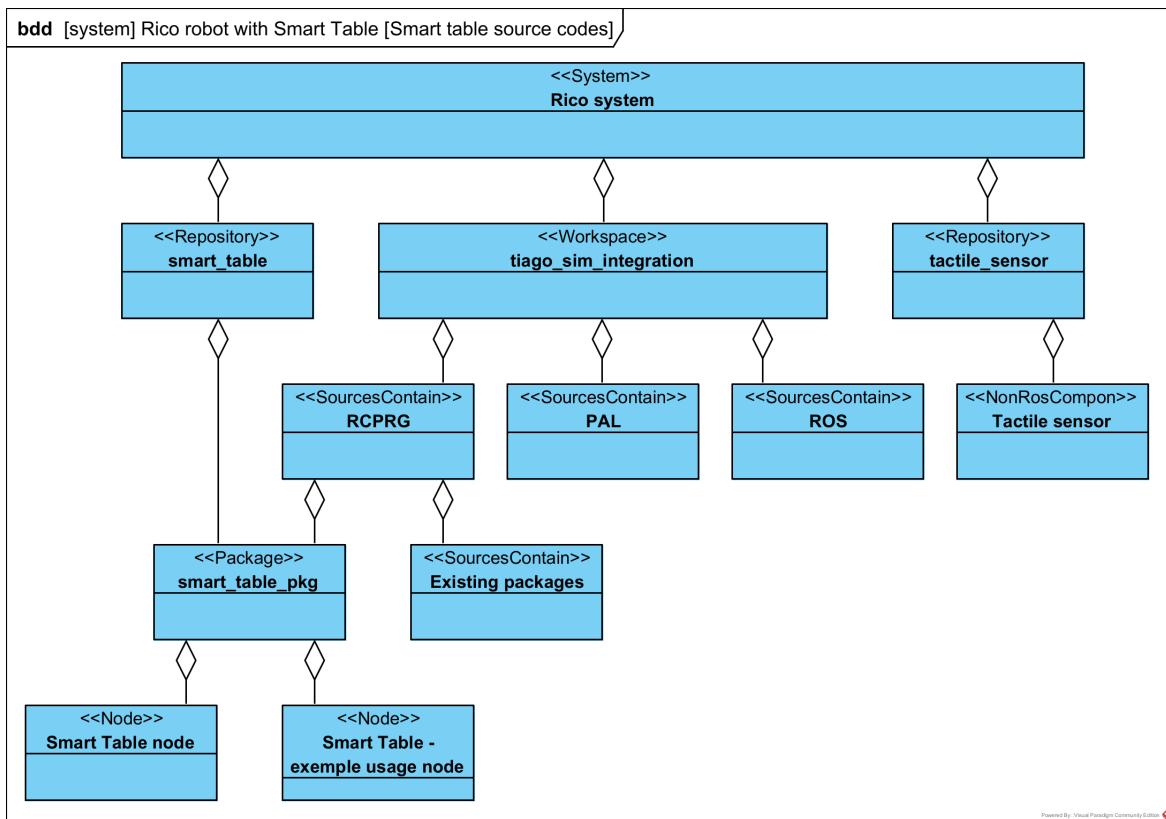
7.1. Struktura projektu

Przed podejściem do samego projektowania węzłów przyjrzałem się temu, jak zdefiniowana jest struktura projektu robota Rico. Z uwagi na to, że robot działa w systemie ROS oraz korzysta z wielu różnych bibliotek było to konieczne, aby móc prawidłowo dopisać planowane funkcje. Jest to tym ważniejsze, że w przypadku przekazywania projektu przyszłemu pokoleniu robotyków kolejnym osobom łatwiej będzie pracować z moim rozwiązaniem.

Zgodnie z założeniami ROS całość projektu podzielona jest na system, kontenery kodu (przestrzeń robocza, repozytorium), pakiety oraz węzły, które to są najniżej w wymienionej hierarchii. W moim przypadku istnieje również jeden kod źródłowy, który nie jest ujęty w strukturach systemu ROS a musiał zostać ujęty na diagramie. W przypadku projektowanego przeze mnie inteligentnego stolika końcowa, po dodaniu wszystkich wykorzystywanych elementów, struktura widoczna jest na rysunku 7.1 i składa się z następujących elementów:

- *Rico system* – system wszystkich elementów systemu ROS i nie tylko, które są potrzebne do działania inteligentnego stolika na robocie Rico,
- *tiago_sim_integration* – przestrzeń robocza wszystkich elementów systemu ROS potrzebnych do działania inteligentnego stolika,
- *ROS* – ogólnodostępne pakiety systemu ROS,
- *RCPRG* – pakiety stworzone przez zespół robotyczny z Politechniki Warszawskiej (Robot Programming and Pattern Recognition Group),
- *PAL* – pakiety stworzone przez twórców robota - PAL Robotics,
- *smart_table* – repozytorium zawierające kody źródłowe pakietów ROSowych, które zostały napisane w ramach tej pracy,
- *smart_table_pkg* – pakiet zawierająca kody źródłowe elementów ROS związanych z inteligentnym stolikiem, która została napisana w ramach niniejszej pracy,
- *Existing packages* – pozostałe pakiety stworzone w ramach grupy RCPRG,

- *Smart Table node* – węzeł ROS inteligentnego stolika,
- *Smart Table – example usage node* - węzeł obsługujący scenariusze wykonywane przez robota,
- *tactile_sensor* – repozytorium zawierające kody źródłowe elementów nieROSowych, które zostały napisane w ramach tej pracy,
- *Tactile sensor* – kod źródłowy sterownika czujnika taktylnego.



Rysunek 7.1. Struktura hierarchiczna kodów źródłowych inteligentnego stolika

Podstawą całego systemu obsługującego inteligentny stolik jest system robota Rico, który składa się z dwóch głównych kontenerów kodu. Pierwszy jest przestrzenią ROSową, która mieści pod sobą trzy główne grupy pakietów podzielone ze względu na ich twórców. Pakiety z grupy *PAL* są pakietami, które dostarczają i utrzymują twórcy robota TIAGO. Pakiety z grupy *ROS* są ogólnodostępnymi pakietami systemu ROS i są utrzymywane przez społeczność oraz organizację odpowiedzialną za ten system. Pakiety z grupy *RCPRG* zostały napisane przez laborantów z Politechniki Warszawskiej, do których ja również się w tym momencie zaliczam.

Najważniejsza, z punktu widzenia tej pracy, jest część, która została przeze mnie dopisana. Mieści się ona w pakiecie *smart_table* i zawiera kod źródłowy węzłów ROS, które obsługują inteligentny stolik. Pakiet ta zawiera w sobie dwa węzły: *Smart Table node* oraz *Smart Table - example of usage*.

7. Implementacja węzłów ROS wykonujących zaplanowane scenariusze

Drugim kontenerem, który przechowuje kod źródłowy, jest repozytorium *tactile_sensor*. Nie mogło ono zostać włączone w struktury przestrzeni roboczej, ponieważ nie zawiera się ono w przestrzeni systemu ROS. W tym repozytorium znajduje się kod źródłowy sterownika czujnika taktylnego *Tactile sensor*. Opis implementacji i działania tej części oprogramowania została szczegółowo opisana w rozdziale 4.2.

7.2. Struktura intrasystemu intelligentnego stolika

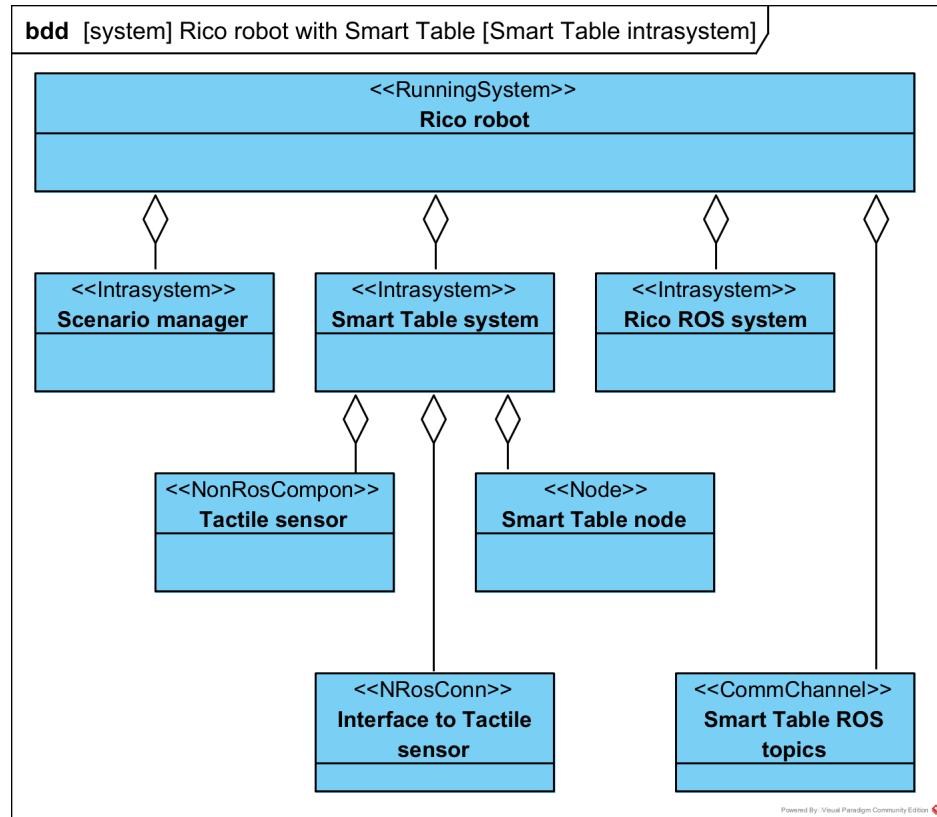
Projektowany intrasystem ma za zadanie współpracować z już istniejącym systemem robota. Dlatego ważne jest, aby na początku projektowania dobrze określić, w jaki sposób zawarty w nim węzeł będzie komunikował się z pozostałą częścią systemu. Aby uszczególnić te relacje i zobrazować je, został wykonany diagram widoczny na rysunku 7.2. Na tym diagramie widać elementy niezbędne dla prawidłowego działania intrasystemu intelligentnego stolika:

- *Rico robot* – cały system działającego robota Rico,
- *Scenario manager* – intrasystem zawierający elementy odpowiedzialne za zarządzanie wykonaniem scenariusza,
- *Smart Table system* – intrasystem zawierający elementy odpowiedzialne za prawidłowe działanie intelligentnego stolika, opisywany w tym rozdziale,
- *Rico ROS system* – intrasystem zawierający pozostałe elementy ROSowe robota Rico,
- *Tactile sensor* – mechaniczna budowa stolika z czujnikiem taktylnym wraz z jego sterownikiem,
- *Smart Table node* – węzeł ROS intelligentnego stolika,
- *Interface to Tactile sensor* – interfejs komunikacyjny pomiędzy intelligentnym stolikiem a czujnikiem taktylnym,
- *Smart Table ROS topics* – interfejs komunikacyjny pomiędzy intelligentnym stolikiem a pozostałą częścią systemu robota.

Jak widać podstawowym elementem systemu jest robot Rico, który, w rozpatrywanej przeze mnie implementacji, oprócz modułów działających na nim normalnie oznaczonych przez intrasystem *Rico ROS system*, posiada dodatkowo dołączone intrasystemy intelligentnego stolika oraz managera scenariuszy. Intrasystem intelligentnego stolika składa się z mechanicznej części projektu, czyli czujnika taktylnego na stoliku robota *Tactile sensor* oraz samego węzła intelligentnego stolika w systemie ROS *Smart Table node*. Dodatkowo w tym intrasystemie został wyodrębniony kanał komunikacyjny *Interface to Tactile sensor*. Kanał *Smart Table ROS topics* jest wyodrębniony z głównego systemu, ponieważ jest interfejsem pomiędzy poszczególnymi intrasystemami.

Przechodząc do dalszej specyfikacji jak powinno wyglądać działanie węzła ustalilem, jak powinien on komunikować się z otoczeniem. Ta część zmieniała się mocno wraz z rozwojem i klarowaniem projektu, ale ostatecznie wykłarała się wersja widoczna

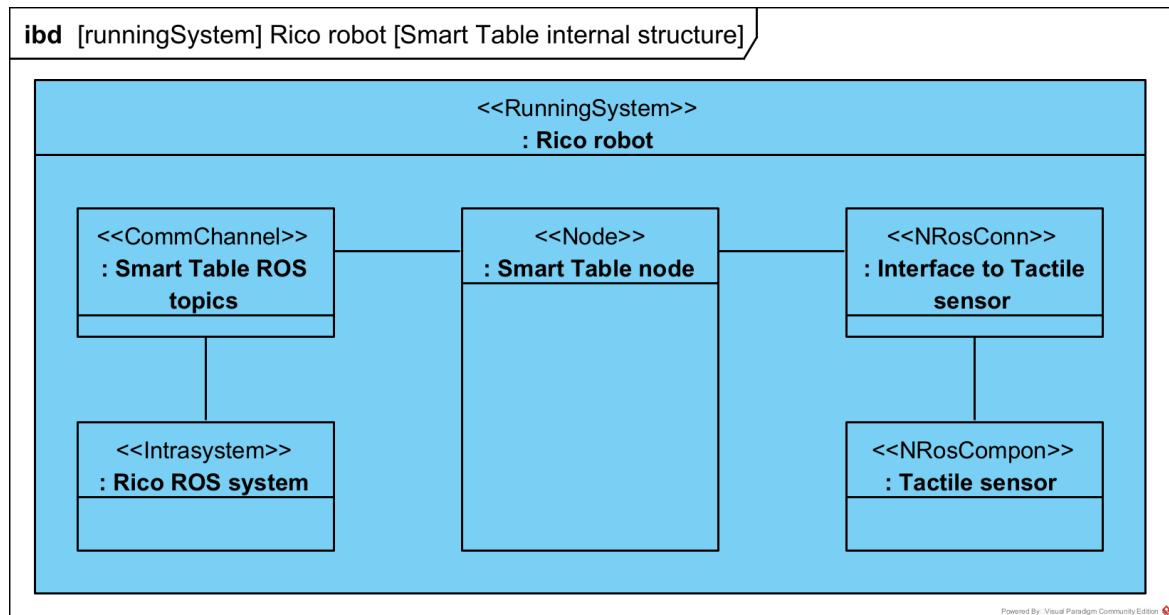
7. Implementacja węzłów ROS wykonujących zaplanowane scenariusze



Rysunek 7.2. Diagram definicji bloków dla intrasystemu inteligentnego stolika

na rysunku 7.3. Diagram ten przedstawia jak będzie przebiegała komunikacja pomiędzy węzłem inteligentnego stolika a pozostałymi elementami systemu.

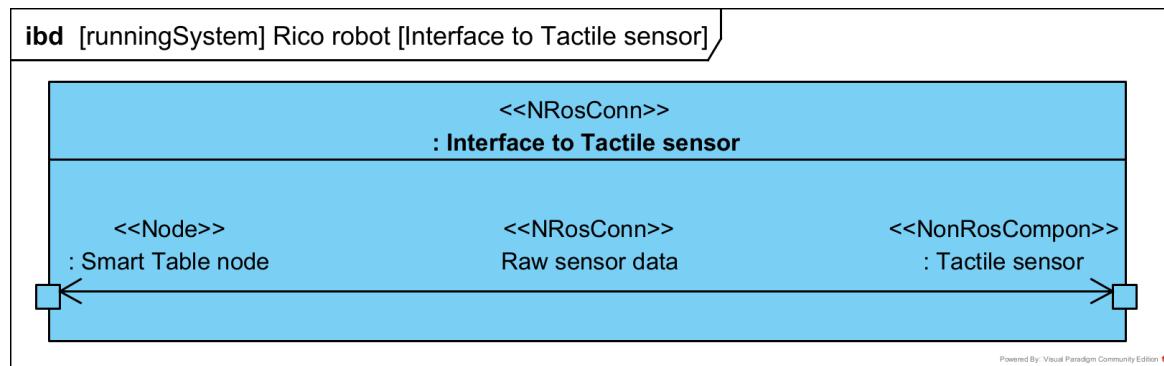
Zadaniem interfejsu do czujnika taktylnego jest przesyłanie danych z czujnika do



Rysunek 7.3. Diagram bloków wewnętrznych realizacji węzła inteligentnego stolika wewnętrz systemu Rico

7. Implementacja węzłów ROS wykonujących zaplanowane scenariusze

węzła inteligentnego stolika. Zadanie to zostało zaprezentowane na rysunku 7.4 przedstawiającym diagram wewnętrzny tego kanału komunikacji. Fizyczne właściwości tego kanału zostały sprecyzowane i opisane w rozdziale 4.2. Jak można zauważyć komunikacja została oznaczona jako dwustronna, ale w przeważającej większości czasu wykorzystywany będzie tylko przesył danych z czujnika do węzła.



Rysunek 7.4. Diagram bloków wewnętrznych obrazujący komunikację pomiędzy węzłem inteligentnego stolika a czujnikiem taktylnym

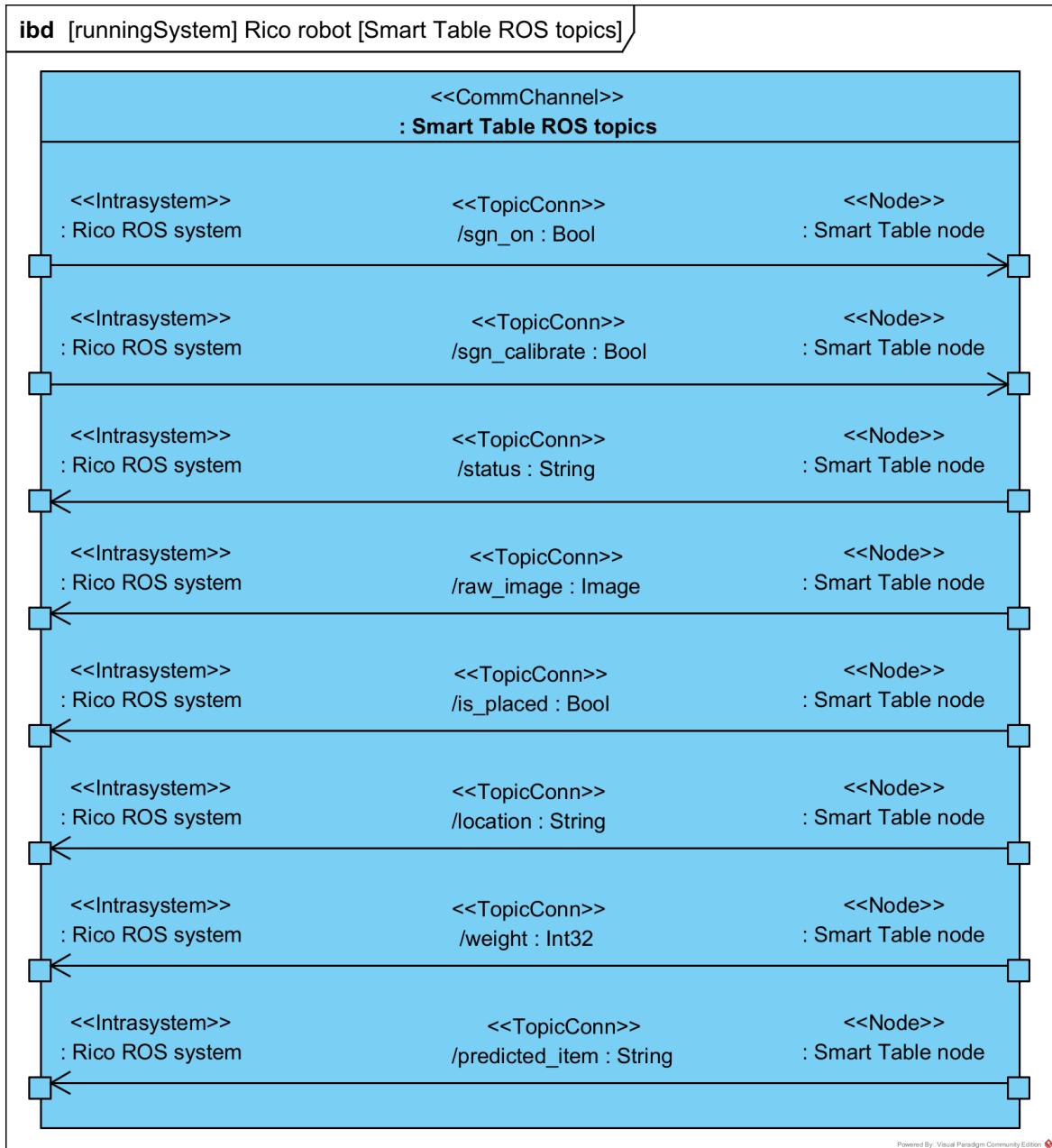
Zadaniem drugiego z wymienionych interfejsów jest przesyłanie danych z węzła do systemu robota jak również umożliwienie robotowi kontroli nad działaniem węzła inteligentnego stolika. Komunikacja odbywa się za pomocą dostępnych w systemie ROS tematów, które są odpowiednie do tego zadania. Sam kanał komunikacji składa się z ośmiu tematów:

- */sgn_on* - sygnał, który włącza lub wyłącza działanie węzła inteligentnego stolika,
- */sgn_calibrate* - sygnał, który wywołuje kalibrację węzła inteligentnego stolika,
- */status* - status węzła inteligentnego stolika,
- */raw_image* - surowe dane odczytywane z czujnika taktylnego, przedstawione w formie monochromatycznego obrazu,
- */is_placed* - sygnał informujący o tym, czy na stoliku jest aktualnie położony jakiś przedmiot,
- */location* - lokalizacja położonego na stoliku przedmiotu,
- */weight* - szacunkowa waga położonego na stoliku obiektu,
- */predicted_item* - informacja o tym, jaki przedmiot znajduje się na stoliku.

Wymienione tematy zostały również przedstawione na diagramie wewnętrznym tego kanału komunikacji, który widoczny jest na rysunku 7.5.

W trakcie ustalania szczegółów pracy węzła inteligentnego stolika wykłarowało się również jego oczekiwane działanie, które jest widoczne na rysunku 7.6. Ogólne działanie węzła nie jest skomplikowane. Jego praca zaczyna się od inicjalizacji wewnętrznej struktury, ustanowieniu połączenia z sterownikiem czujnika taktylnego oraz załadowaniu potrzebnych, wcześniej wytrenowanych, modeli. Kolejna część pracy odbywa się już w pętli głównej programu. Aby mogła ona zajść, wymagany jest sygnał */sgn_on* z wartością

7. Implementacja węzłów ROS wykonujących zaplanowane scenariusze



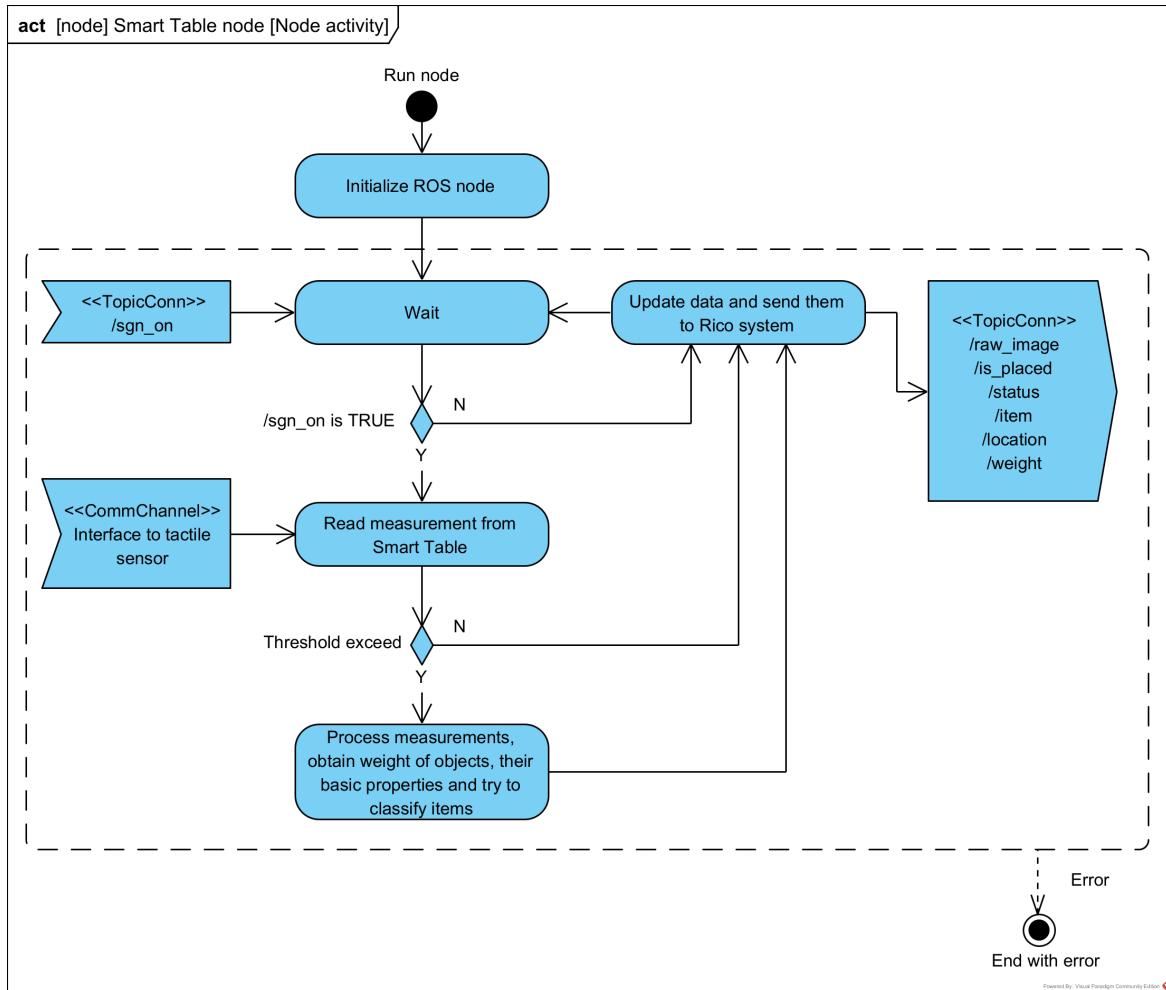
Rysunek 7.5. Diagram bloków wewnętrznych obrazujący komunikację, w postaci tematów ROS, pomiędzy węzłem inteligentnego stolika a pozostałą częścią systemu

true. Po każdym wykonaniu pętli odpowiednio aktualizowane są wszystkie publikowane tematy, w systemie ROS.

Cała praca węzła jest rozpoczynana po otrzymaniu nowych danych z portu komunikacyjnego, ze sterownika czujnika taktylnego. Jeśli zauważone zostaną błędy lub braki w danych to cała seria otrzymanych danych jest ignorowana i ustawiana jest odpowiednia flaga na temacie /status. Dane po otrzymaniu są od razu konwertowane na obraz monochromatyczny. Na tym etapie uwzględniana jest również wykonana wcześniej kalibracja (7.2.2) i kompensacja szumów stolika (5.2).

Jeśli węzeł wykryje, że na stoliku znajduje się przedmiot (algorytm opisany w rozdziale

7. Implementacja węzłów ROS wykonujących zaplanowane scenariusze



Rysunek 7.6. Diagram aktywności obrazujący sposób działania węzła inteligentnego stolika

7.2.4) dane są dalej przetwarzane. Dalsze przetworzenie danych polega kolejno na ustaleniu położenia przedmiotu na stoliku (7.2.5), oszacowaniu jego wagi (7.2.6 i 7.2.8) oraz próbie ustalenia tego, jaki przedmiot został położony na stoliku (7.2.7).

Program został w niektórych miejscach zabezpieczony przed wewnętrznymi błędami. Nie przeprowadzone zostały jednak pełne testy odporności na błędy, więc w przypadku wystąpienia takiego błędu, program zostanie zakończony.

Osobną kwestią, niezawartą na diagramie 7.6, jest kalibracja czujnika. Odbywa się ona zawsze, kiedy sygnał */sgn_calibrate* ma wartość *true*. Kalibracja została jednak bardziej szczegółowo opisana w rozdziale 7.2.2.

7.2.1. Możliwe statusy węzła

Węzeł inteligentnego stolika może przyjmować kilka predefiniowanych statusów, które mogą służyć do sprawdzenia czy działa on poprawnie. W logice węzła zostały zaimplementowane następujące statusy:

- *Unknown* – nieznany status węzła,
- *Initializing* – węzeł w trakcie inicjalizacji,

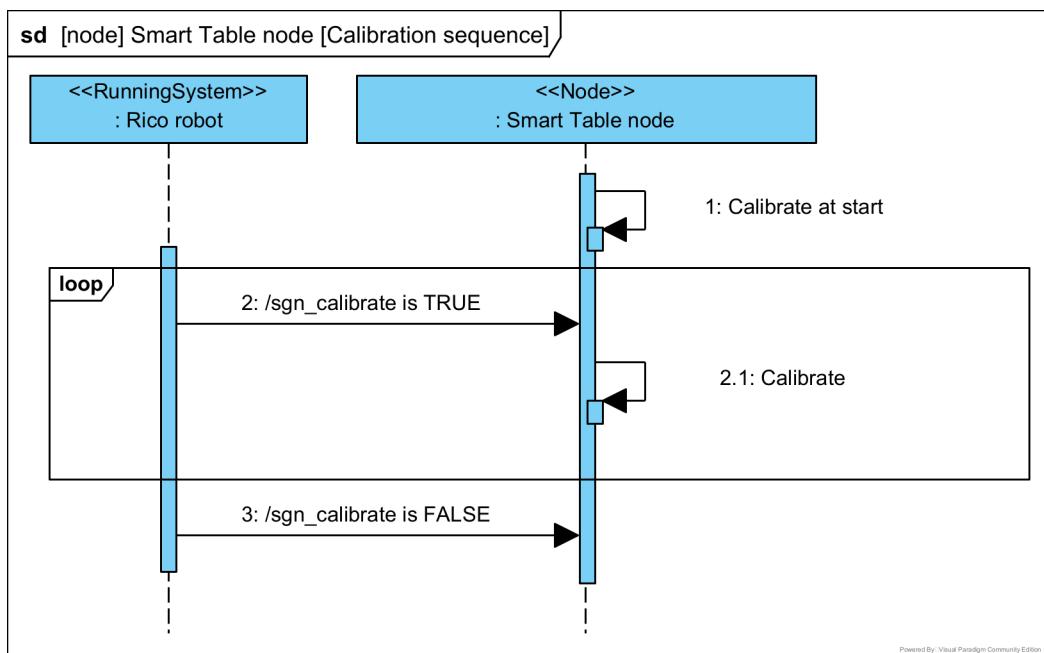
- *Working off* – węzeł prawidłowo zainicjalizowany, a sygnał */sgn_on* ma wartość *false*,
- *Working on* – węzeł prawidłowo działa,
- *Working calibrating* – węzeł w trakcie kalibracji,
- *Crashed communication* – wystąpił błąd związany z komunikacją z czujnikiem.

Wymienione powyżej statusy są publikowane regularnie na temacie *status*.

7.2.2. Kalibracja czujnika taktylnego

Kalibracja czujnika jest potrzebna, ponieważ stolik w momencie uruchomienia wykrywa pewien nacisk. Nacisk ten wynika z faktu, że czujnik pokryty jest gumą, a sama budowa czujnika jest niedoskonała. Szerzej ten temat został opisany w rozdziale 5.2, gdzie poruszona jest również kwestia kompensacji i filtrowania szumów.

Kalibracja jest wykonywana za każdym razem przy uruchomieniu węzła inteligentnego stolika oraz za każdym razem, gdy użytkownik tego zażąda - poprzez wystawienie odpowiedniego stanu na temacie */sgn_calibrate*. Ten prosty schemat działania został przedstawiony na rysunku 7.7.



Rysunek 7.7. Diagram sekwencji kalibracji pomiarów z czujnika taktylnego

Sama kalibracja polega na zapisaniu do pamięci węzła aktualnego obrazu poisanego z czujnika taktylnego. Obraz ten jest później wykorzystywany do kompensacji błędów i eliminacji szumów.

7.2.3. Przetwarzanie informacji ze stolika

Obraz, który wysyłany jest przez inteligentny stolik w surowej formie jest trudny do obrabiania cyfrowego. Jest to macierz dwuwymiarowa liczb w przedziale 0 – 4095. Dlatego

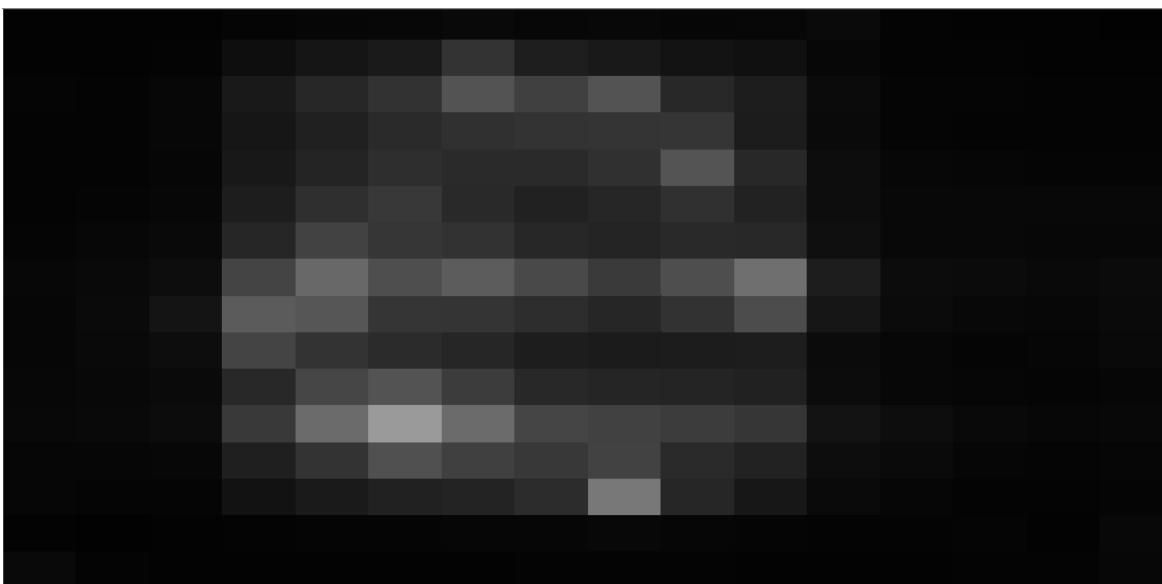
7. Implementacja węzłów ROS wykonujących zaplanowane scenariusze

też postanowiłem surowe odczyty przekształcić na 8-bitowy obraz monochromatyczny o rozdzielczości $16 \times 16\text{px}$.

Przekształcenie danych na postać obrazu pozwala od ręki użyć wielu bibliotek i metod dedykowanych dla obrazów, ich rozpoznawania, interpretacji i przetwarzania. Ten dział programowania jest dość dobrze rozwinięty i udokumentowany, co pozwala na stosunkowo łatwe użycie dostępnych narzędzi.

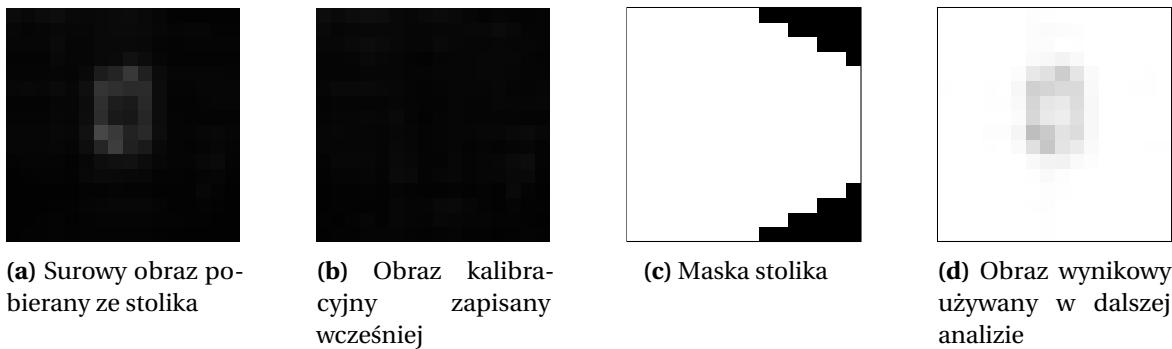
Większość wykonywanych operacji jest wykonywana na podstawie obrazu, ale nie wszystkie, ponieważ obraz zawiera mniej szczegółów, w porównaniu do surowych danych, a to może zmniejszać jakość wykonywanych obliczeń. Surowe dane są między innymi wykorzystywane przy szacowaniu wagi obiektu, opisany w rozdziale 7.2.6.

Obraz jest także łatwiejszy do interpretacji dla zwykłego użytkownika niż ciąg liczb. Z tego powodu powstał podgląd dostępny dla użytkownika. Zrzut obrazu z programu prezentującego ten podgląd jest widoczny na rysunku 7.8. Obraz ten jest prezentowany użytkownikowi w formacie 2 : 1, co odpowiada faktycznej wielkości pól na stoliku.



Rysunek 7.8. Program do podglądu nacisku wywieranego przez obiekt na stolik

Obraz, zanim będzie gotowy do wykorzystania przez program, poddawany jest najpierw kilku prostym zabiegom i przekształceniom. Surowy pobrany obraz jest odejmowany od obrazu otrzymanego podczas wykonanej wcześniej kalibracji. Do tego jest też dokładana kompensacja szumów opisana w rozdziale 5.2, tak aby obraz nabrął więcej głębi. Kolejno na otrzymany obraz nakładana jest maska pól taktylnych, które są fizycznie realizowane przez stolik – wszystkie pola, które nie posiadają fizycznej realizacji są zerowane i nieuwzględniane w dalszych przekształceniach. Tak przygotowany obraz obiektu jest gotowy do dalszej analizy przez stosowne narzędzia i funkcje. Opisane w tym akapicie poszczególne przekształcenia obrazu są widoczne na rysunku 7.9.



Rysunek 7.9. Interpretacja oraz wstępna obróbka obrazu (kubka) przez węzeł inteligentnego stolika

7.2.4. Wykrywanie obecności obiektu

Wykrycie obecności obiektu na stoliku jest kluczowe, aby móc przejść do dalszego przetwarzania obrazu i ustalania dalszych parametrów obiektu. Bez wykrycia obiektu na stoliku nie są uruchamiane i rozważane kolejne etapy przetwarzania obrazu. Wykrywanie obecności obiektu jest również podstawą do wypełnienia przypadków użycia *UC1* oraz *UC4*.

W logice węzła zawarłem jeden warunek, który pozwala określić, czy na stoliku znajduje się obiekt: dowolny piksel obrazu ma wartość większą od 10 (określona na obrazie wynikowym, maksymalną wartością jest 255). Wartość ta odpowiada naciskowi $\sim 10g$ na pojedyncze pole czujnika (biorąc pod uwagę badania wykonane w rozdziale 5.2.2). W praktyce wartość ta wynosi więcej, przez to, że nacisk nie jest w całości przykładany do pojedynczego pola taktylnego.

7.2.5. Wykrywanie położenia obiektu

Aspekt ten jest ważny, aby umieszczony na stoliku przedmiot nie spadł podczas transportu. Biorąc pod uwagę, że na stoliku mogą być przewożone gorące przedmioty, a sam robot może pracować w środowisku z osobami starszymi lub ograniczeniami ruchowymi, obiekt pod żadnym pozorem nie może spaść z robota. Dlatego też ważne jest, aby wykryć, czy przedmiot znajduje się na krawędzi stolika. Wykrywanie położenia przedmiotu na stoliku jest też bezpośrednim wykonaniem przypadku użycia *UC2*.

Zaprojektowany przeze mnie węzeł jest w stanie rozpoznać położenie obiektu spośród wymienionych:

- *Nothing* - brak obiektu na stoliku,
- *Center of table* - obiekt znajduje się w centrum stolika,
- *Side of table* - obiekt znajduje się po bokach stolika,
- *Edge of table* - obiekt znajduje się na krawędzi stolika.

Mimo prób, praktycznie niemożliwa okazała się klasyfikacja obiektu znajdującego się częściowo poza stolikiem. Obraz ten generuje odczyty bardzo podobne do tych, które daje

7. Implementacja węzłów ROS wykonujących zaplanowane scenariusze

obiekt znajdujący się na krawędzi. Dlatego też, w toku badań zrezygnowałem z rozpoznawania tego położenia i uznałem je za równoważne z obiektem na krawędzi stolika.

W pierwszym kroku wykonałem sprawdzenie, czy obiekt znajduje się na brzegu stolika. Wyznaczyłem w tym celu szereg warunków, które weryfikują każdy otrzymany obraz obiektu:

- największy generowany nacisk jest na skrajnie brzegowych polach taktylnych,
- środek ciężkości obiektu znajduje się mniej niż 5cm od krawędzi stolika,
- peak histogramu rozkładu nacisków obiektu od środka ciężkości obiektu jest w odległości większej niż odległość do krawędzi minus 2cm ,
- waga obiektu rozkłada się w większej części na zewnętrznej części stolika (2 skrajne pola taktyльne) niż w pozostałe części stolika.

Jeśli któryś z powyższych warunków zostanie spełniony to węzeł klasyfikuje przedmiot jako położony na krawędzi.

Rozwijając temat wspomnianego histogramu, to jest on dość ważny w przypadku, kiedy obiekt rozkłada się na większej liczbie pól czujnika. W takim przypadku należy sprawdzić, czy znacząca część obiektu nie znajduje się na brzegach stolika. Histogram ten pozwala oszacować średnicę położonego obiektu poprzez zmierzenie w jakiej odległości od środka ciężkości rozmieszczony jest największy nacisk. Znając szacunkową średnicę obiektu możemy założyć dodatkowy margines błędu i użyć tych danych do określenia lokalizacji. Niestety, metoda ta najlepiej sprawdza się w przypadku okrągłych przedmiotów, z rantem na zewnątrz. W przypadku przedmiotów o innych kształtach, metoda ta nie spisuje się już tak dobrze, ponieważ nie jest w stanie prawidłowo wyznaczyć wielkości obiektu.

Pozostałe obrazy są klasyfikowane, na te znajdujące się w centrum oraz po bokach stolika, na podstawie fizycznych właściwości stolika. Granicą, na której rozgraniczam centrum z bokami jest linia poprowadzona pionowo, od miejsca, gdzie rozpoczyna się ucięcie rogów stolika.

Poza organoleptycznym i jednostkowym sprawdzeniem poprawności działania algorytmu określającego lokalizację obiektu wykonałem również badanie na zbiorze danych zebranym na potrzeby trenowania sieci neuronowych. Podczas zbierania tychże obrazów zapisywałem również, z jakiej strefy zostały one zebrane. W ten sposób sprawdziłem jakość wykrywania krawędzi na podstawie dużego zbioru danych.

Obrazy zebrane w momencie kiedy obiekt był na krawędzi stolika lub częściowo poza nim oznaczyłem na potrzeby tego testu jako obiekty na krawędzi. Obrazy zebrane gdy obiekt znajdował się w centrum lub na którejś stronie stolika oznaczyłem jako wewnętrz. Badanie przeprowadzałem odpowiadając na pytanie *Czy obiekt znajduje się wewnątrz stolika?*. Na tej podstawie stworzyłem macierz błędów, która widoczna jest w tabeli 7.1. Na podstawie tej tabeli wyznaczyłem podstawowe parametry wykorzystanej metody wykrywania położenia obiektu:

- czułość: 0,85,
- swoistość: 0,64,
- precyzja: 0,67,
- dokładność: 0,73.

Tabela 7.1. Macierz błędów dla wykrywania czy obiekt znajduje się wewnątrz stolika

		Klasa rzeczywista	
		pozytywna	negatywna
Klasa predykowana	pozytywna	888 (85%)	445 (36%)
	negatywna	162 (15%)	784 (64%)

Analizując te wartości można zauważyć, że wybrana metoda detekcji położenia obiektu na stoliku nie jest najlepsza. Świadczą o tym wysoki odsetek fałszywie pozytywnych próbek (36%) oraz stosunkowo niska precyzja klasyfikacji (0,67). Wybrałem te dwie wartości jako najważniejsze w tym zadaniu, ponieważ najgorszym przypadkiem dla zadania przenożenia przedmiotów jest stwierdzenie, że obiekt znajduje się wewnątrz stolika, kiedy w rzeczywistości jest on ja jego krawędzi.

Mimo tak niesatisfakcyjnych wyników na dużej skali postanowiłem pozostać przy wybranej metodzie. Jednym z powodów był fakt, iż inne testowane metody nie dawały lepszych wyników, a drugim fakt, że testy tej metody na fizycznym stoliku przebiegły obiecująco.

7.2.6. Szacowanie wagi obiektu

Szacowanie wagi obiektu jest ważnym aspektem pracy, ponieważ jest dodatkowym czynnikiem, który pomaga ocenić czy znajdujący się na stoliku przedmiot jest prawidłowy. Szacowanie wagi jest również bezpośrednim wykonaniem przypadku wykorzystania stoliku, opisanym przez UC5.

Szacowanie wagi obiektu zostało oparte w całości na badaniach opisanych w rozdziale 5.2.2. Wyliczony tam współczynnik został bezpośrednio wykorzystany w tym węźle i jest głównym elementem obliczania szacowanej wagi obiektu. W tym przypadku, jeśli tylko jest to możliwe, do szacowania wagi obiektu wykorzystywane są surowe dane otrzymywane ze stolika, a nie ich zrzut do obrazu.

Mimo wykorzystania wykonanych przeze mnie badań, wskazania wagi mierzonej przez stolik wyraźnie odbiegały od stanu faktycznego. Różnicę pomiędzy odczytami można było wyraźnie zaobserwować również pomiędzy prawą i lewą stroną czujnika, które wykazywały różną czułość. W praktyce rozbieżności w pomiarach wynosiły czasami nawet do $-50\% / +100\%$, co wynosi bardzo dużo, ale przynajmniej daje pewne pojęcie, co do możliwego zakresu faktycznej wagi obiektu. Z uwagi, iż ta metoda obliczania wagi bazuje na surowych danych odbieranych z czujników taktylnych, nie byłem w stanie przetestować jej na dużym zbiorze danych, który został zapisany w postaci przetworzonych obrazów.

W praktyce szacowanie wagi praktycznie zawsze zaniża pomiar wagi względem faktycznej wagi. Z uwagi na dynamiczną charakterystykę Velostatu pomiar wagi regularnie się zwiększa wraz z czasem. Po ustabilizowaniu się, pomiary często są zbliżone do faktycznej wagi, ale sam proces stabilizacji może zajść nawet do kilku minut, co stanowi spory problem przy wykorzystaniu ich w szybkim szacowaniu wagi. Widoczna jest również wyraźna różnica pomiędzy pomiarami z prawej i lewej strony stolika. Ogólnie do tych pomiarów należy podchodzić z dużym zapasem zaufania.

7.2.7. Rozpoznawanie położonego przedmiotu

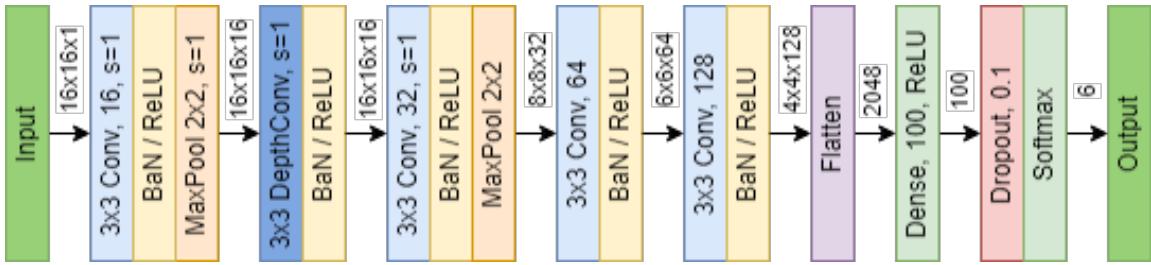
Jak już opisywałem w rozdziale 2.1.2, podobne prace są wykonywane na całym świecie. Jako bardzo dobry przykład wybrałem pracę Yuana Liangqi [29], który, podobnie jak ja, zajmuje się zagadnieniem rozpoznawania przedmiotów położonych na czujniku bazującym na folii Velostat. Testował on czujnik rozpoznający 10 przedmiotów i wykorzystując konwolucyjną sieć ResNet-PI (zmodyfikowana sieć ResNet-18) osiągnął zdolność ich rozpoznawania na poziomie 0,9854, co było dobrym odniesieniem dla moich prac. Samo rozpoznawanie położonych obiektów zostało przeze mnie zdefiniowane już w przypadku użycia UC3 i ten przypadek wykonuje [29].

Yuan w swojej pracy wykorzystał matrycę 27×27 połączeń rozmieszczonych co 5mm na obszarze o wymiarach $140 \times 140\text{mm}$. Wykonując podstawowe obliczenia wyznaczono gestość rozmieszczenia pól czujnika, a zarazem rozdzielcość czujnika, która wynosi $3,72\text{pol}/\text{cm}^2$. Dla porównania, moje rozwiązanie jest matrycją o wymiarach 16×16 , umieszczoną na płaszczyźnie o wymiarach $240 \times 400\text{mm}$. Wykonując analogiczne pomiary można zauważać, że moje rozwiązanie ma rozdzielcość tylko $0,27\text{pol}/\text{cm}^2$. Jest to około 13,78 razy mniejsza rozdzielcość niż w przypadku Yuana. Taka niska rozdzielcość przekłada się bezpośrednio na jakość i skuteczność używanej sieci [29].

Do klasyfikacji pierwotnie planowałem wykorzystać jeden z gotowych modeli dostępnych w bibliotece Keras (importowanej wraz z TensorFlow). Niestety okazało się, że z uwagi na bardzo niewielkie wymiary moich obrazów nie mogłem bezpośrednio wykorzystać żadnej z gotowych sieci. Gotowe modele sieci przyjmowały zazwyczaj dane o minimalnej wielkości $32 \times 32\text{px}$ i o 3 kanałach kolorów, a moje obrazy mają wielkość tylko $16 \times 16\text{px}$ i operują tylko na skali szarości [59]. Dlatego też, wzorując się na dostępnych w tej bibliotece oraz wykorzystywanych przez innych badaczy modelach, postanowiłem zaprojektować sieć CNN od podstaw [29], [59], [78].

Podczas projektowania wziąłem pod uwagę niewielki rozmiar obrazów oraz nieduże skomplikowanie problemu. Moje poszukiwania były usiane wieloma testami, a finalna sieć którą przyjąłem ma architekturę widoczną na rysunku 7.10.

Zaprojektowana sieć składa się z 7 głównych warstw neuronowych: 4 warstw konwolucyjnych, 1 warstwy głębokiej konwolucyjnej i 2 warstw gęstych, z czego jedna z nich bezpośrednio odpowiada za klasyfikację obiektu. Poza tymi warstwami widoczne na diagramie są też warstwy *BaN* oraz *ReLU*, które są dwoma osobnymi warstwami, ale dla



Rysunek 7.10. Architektura zaprojektowanej sieci CNN do klasyfikacji położonych obiektów

czytelności i zmniejszenia rozmiaru rysunku zostały połączone. Pomiędzy poszczególnymi warstwami, na strzałkach przepływu, zostały oznaczone rozmiary przepływających danych. Wyraźnie zostało tam zaznaczone, że na wejście wchodzi obraz monochromatyczny o wielkości $16 \times 16 \text{ px}$, a na wyjściu otrzymywana jest informacja, który przedmiot został położony na stoliku.

Podstawowym zadaniem warstw konwolucyjnych jest ekstrakcja poszczególnych cech z obrazu. Widoczny w niektórych warstwach dopisek $s = 1$ oznacza dodanie po bokach obrazu pasków, które zapobiegają przedwczesnemu zmniejszeniu się rozdzielczości badanego obrazu. Wraz ze wzrastającą głębokością sieci zmniejsza się wielkość przestrzeni danych, a zwiększa się liczba wykorzystywanych filtrów. Służy to do ekstrakcji większej ilości coraz bardziej złożonych cech obiektu, przy jednoczesnym zachowaniu rozsądnego czasu potrzebnego na obliczenia [59].

Warstwa głębszej konwolucji ma podobne zadanie co zwykła warstwa konwolucyjna, jednak wykorzystuje ona do tego inną metodę. Przeprowadza ona konwolucję niezależnie dla każdego kanału, co pozwala na szybsze i mniej obciążające uczenie sieci. Wykorzystanie przez mnie tej warstwy zostało zainspirowane jej szerokim wykorzystywaniem przez model *MobileNet* [59].

Warstwy *BaN* i *ReLU* są odpowiedzialne za uczenie i dobieranie współczynników wewnętrz sieci. Warstwa *ReLU* jest warstwą z funkcją aktywacji, która zwraca wartość współczynnika mnożenia współczynników sieci. Warstwa *BaN* odpowiada za normalizację danych tak, aby ich średnia wynosiła 0, a odchylenie standardowe 1 [59].

MaxPooling ma na celu redukcję wymiaru przestrzeni danych przetwarzanych przez sieć neuronową. Jednocześnie zachowane są przez sieć najistotniejsze dane, poprzez wybór tylko najbardziej widocznych cech, które są przekazywane dalej [59].

Na końcu modelu dodane zostały warstwy typu *Dense*, które łączą ze sobą neurony w sposób każdy z każdym. Warstwy te są wykorzystane, aby móc wyciągnąć wnioski na podstawie wszystkich dostępnych danych. Warstwy te są poprzedzone spłaszczeniem danych, realizowanym przez blok *Flatten*, który z danych wielowymiarowych tworzy pojedynczy wektor neuronów [59].

Na szczególną uwagę zasługuje również wykorzystana przeze mnie warstwa *Dropout*.

Dokonuje ona losowego wyłączania części neuronów w trakcie procesu uczenia. Zapobiega to przeuczeniu się sieci oraz zwiększa odporność na potencjalne zakłócenia [59].

Na ostatnim kroku jest warstwa typu *Dense* z tym, że jest ona specjalnie przygotowana do klasyfikacji obiektów. Posiada ona tylko 6 neuronów, co odpowiada liczbie rozpoznawanych przedmiotów. Warstwa ta posiada również specyficzną funkcję aktywacji, którą jest *Softmax* przystosowany specjalnie do klasyfikacji obrazów. Wyjściem sieci neuronowej jest prawdopodobieństwo dla każdej klasy, że obraz wejściowy do niej należy [59], [79].

Jak wspomniałem, sieć wykrywa tylko 6 różnych przedmiotów, mimo iż pierwotnie miała mieć szerszy zakres rozpoznawanych obiektów. Niektóre przedmioty zostały wyeliminowane ze względu na niewielkie różnice pomiędzy poszczególnymi klasami, które ciężko było sieci od siebie odróżnić. Ostatecznie do uczenia klasyfikatora zostało wybrane 6 przedmiotów:

- *Book* – książka,
- *Mug* – kubek pełny oraz pusty,
- *Plate* – talerz pełny oraz pusty,
- *Phone* – telefon,
- *Hand* – ręka z różnym stopniem nacisku,
- *Drugs* – pojemnik na leki (kieliszek).

Każdy z wybranych przedmiotów był umieszczany w różnych miejscach stolika, po czym zapisywany do pliku był surowy obraz, taki sam jaki jest widoczny na podglądzie opisywanym w rozdziale 7.2.3 i jest widoczny na rysunku 7.9. Dodatkowo, zapisywany był obraz wykonany podczas kalibracji, tak aby móc na późniejszym etapie odfiltrować zakłócenia stałe stolika. Podczas zapisywania obrazu, oprócz informacji o tym jakiego przedmiotu dotyczy zdjęcie, zapisywana była także informacja o jego wadze, położeniu na stoliku, powierzchni i charakterystycze styku ze stolikiem. Te dodatkowe dane umożliwiły wykorzystanie zebranego zbioru danych do innych celów, niż sama klasyfikacja obiektu.

Do dalszej analizy pod kątem klasyfikacji obiektu brane były pod uwagę tylko próbki obiektów, które całkowicie mieściły się na stoliku (według kryteriów opisanych w rozdziale 7.2.5). Warunek ten jest ważny, aby sieć błędnie nie uczyła się przedmiotów, które i tak zostaną sklasyfikowane jako niewłaściwie umiejscowione.

Aby zwiększyć liczbę dostępnych próbek, jak i możliwe kombinacje położenia obiektu próbki zostały dodatkowo poddane mnożeniu poprzez rotację oraz odbicie obrazu. W ten sposób liczba próbek została zwiększona ośmiokrotnie. Takie podejście jest często stosowane przez różnych badaczy i pozwala zwiększyć pokrycie badanego problemu. Zapobiega ono również przeuczeniu się modelu na podstawie zbyt małej próbki danych [80].

Do treningu została wzięta taka sama liczba próbek z każdej klasy przedmiotów. Kiedy sieć brała do uczenia się wszystkie dostępne próbki, niezależnie od ich liczby, to wtedy wyraźnie zauważalne było faworyzowanie przy klasyfikacji tych przedmiotów, które posiadały większą liczbę próbek w trakcie uczenia. Z tego powodu liczba próbek z każdej klasy została

zmniejszona do liczby próbek z najmniej licznej klasy (selekcja wewnętrz poszczególnych klas odbywała się losowo). Przedmiotem ograniczającym liczność próbek okazał się być pojemnik na leki, który posiadał tylko 400 reprezentacyjnych obrazów. Dlatego do uczenia sieci zostało wykorzystanych łącznie 2400 obrazów z 6 klas przedmiotów.

Zgodnie ze sztuką, cała pula dostępnych danych została podzielona na kilka mniejszych zbiorów:

- zbiór uczący (treningowy),
- zbiór walidacyjny,
- zbiór testowy.

W literaturze bardzo często całkowicie pomija się ostatni zbiór testowy i jako wynik jakości uczenia sieci bierze się wynik uzyskany na zbiorze walidacyjnym [29], [81], [82].

Literatura również nie określa jednoznacznie, jakie powinny być stosunki pomiędzy poszczególnymi zbiorami. Określa się tylko, że zbiór treningowy powinien być największy. Biorąc pod uwagę dostępne materiały oraz własną intuicję, postanowiłem podzielić posiadane dane w stosunku 7 : 2 : 1 (zbiór uczący : walidacyjny : testowy) [83], [84].

Zbiór uczący, jak sama nazwa wskazuje, jest wykorzystywany w procesie uczenia sieci. Na podstawie tego zbioru wyliczane są poszczególne wagi połączeń w jej budowie. Zbiór walidacyjny również jest wykorzystywany w procesie uczenia. Służy on do obserwacji i kontrolowania, aby sieć się nie przeuczyła, i nie dopasowała za bardzo do danych treningowych. Natomiast zbiór testowy jest wykorzystywany na samym końcu, do ponownej walidacji wytrenowanej sieci na innym zbiorze danych, niż te dostępne sieci w procesie uczenia. Wyniki uczenia się sieci są prezentowane właśnie na podstawie danych testowych. Przy podziale na te 3 zbiory należy pamiętać, że każdy z nich powinien zawierać reprezentatywną próbę danych, których sieć powinna się nauczyć.

Stosując się do opisanej metodologii wykonania badania, nauczyłem sieć na zebranych obrazach. Wyniki uczenia są widoczne w tabeli 7.2. Ogólna skuteczność sieci na zbiorze testowym wyniosła w tym przypadku ~ 0,75.

Tabela 7.2. Tabela wyników klasyfikacji dla wybranego modelu sieci

	Phone	Book	Plate	Mug	Hand	Drugs
Phone	0,391996	0,011368	0,107422	0,065497	0,004382	0,419335
Book	0,068280	0,894872	0,028242	0,006057	0,001872	0,000677
Plate	0,027491	0,000596	0,679883	0,182186	0,031878	0,077966
Mug	0,000009	0,000001	0,000072	0,922031	0,000001	0,077888
Hand	0,075417	0,064943	0,169018	0,026235	0,664094	0,000291
Drugs	0,000033	0,000000	0,000093	0,082082	0,000001	0,917791

Jak widać, sieć w sporej mierze działa zadowalająco. Widać to po wysokich wartościach na przekątnej tabeli. Niestety, sieć mimo skuteczności ~ 75% pozostawia wiele do życzenia i ma spore problemy z rozpoznawaniem niektórych przedmiotów. Za przykład można

7. Implementacja węzłów ROS wykonujących zaplanowane scenariusze

wziąć telefon, który częściej był klasyfikowany jako pojemnik z lekami niż jako telefon. Nie byłem w stanie jednak wyjaśnić, dlaczego tak się stało.

Widoczne są również wyraźne problemy sieci z rozpoznawaniem talerza oraz ręki. Może to być spowodowane faktem, że te grupy obiektów zawierały dużo bardzo różnych wzorców. W grupie talerzy znajdowały się zarówno talerze puste, jak i pełne. Poza tym, miały one znacznie różniące się między sobą średnice podstawy. Dłoń nie miała jednego ustalonego ułożenia i siły nacisku, przez co również mogła być trudniejsza do rozpoznania.

Pozostałe przedmioty mają całkiem wysoką skuteczność rozpoznawania wynoszącą ~ 90%. Jest to wynik bardzo satysfakcyjny dla tych przedmiotów.

Otrzymane przeze mnie wyniki są znacznie gorsze, niż te otrzymane przez Yuana Liangqi, mimo iż klasyfikowałem mniejszą liczbę przedmiotów niż on. Miała na to wpływ przede wszystkim, opisywana wcześniej, dużo niższa rozdzielcość czujnika, jak również fakt, że moje przedmioty mogły posiadać różną wagę, nawet w tej samej klasie [29].

Wykrycie i rozpoznanie położonego na stoliku przedmiotu jest sygnalizowane przez wysłanie informacji na odpowiednim temacie (*predicted_item*). W zależności od wykrytego obiektu węzel może wysłać następujące wiadomości:

- *Nothing* - nic nie jest położone na stoliku,
- *Book* - książka,
- *Mug* - kubek (bez rozróżnienia na pełny i pusty),
- *Plate* - talerz (bez rozróżnienia na pełny i pusty),
- *Phone* - telefon,
- *Drugs* - pojemnik z lekami,
- *Hand* - ręka położona na stoliku,
- *Unknown* - węzel nie jest w stanie jednoznacznie rozpoznać przedmiotu.

Informacja o tym, że nic nie jest położone na stoliku jest ściśle powiązana z wykrywaniem czy coś zostało na stoliku położone, opisany w rozdziale 7.2.4. Jeśli granica czułości nie zostanie przekroczena to wysyłana jest informacja *Nothing*, w przeciwnym razie wysyłana jest odpowiednia, inna informacja o tym co jest położone na stoliku.

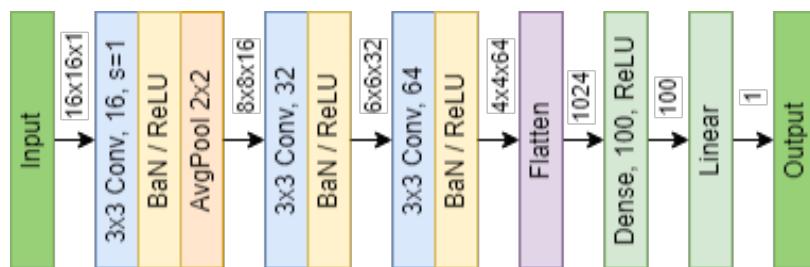
Aby ustalić, jaki dokładnie przedmiot został położony na stoliku musiałem wykonać odpowiednie progowanie. Sieć neuronowa zwraca prawdopodobieństwo, że przedmiot jest każdą z wymienionych rzeczy. Mówiąc dokładniej, wysyła ona 6 prawdopodobieństw, każde odnoszące się do innego przedmiotu. Jeśli któreś z tych prawdopodobieństw przekroczy próg, który w toku badań ustaliłem na poziomie 0,75, to wtedy węzel klasyfikuje obraz jako odpowiedni przedmiot. Jeśli sieć neuronowa nie jest w stanie z odpowiednią pewnością ustalić, jaki przedmiot znajduje się na stoliku, wtedy wysyła informację *Unknown*.

Uzupełniając wyniki, które zostały przedstawione w tabeli 7.2, sprawdziłem, jak w praktyce wygląda klasyfikacja obiektów kładzionych na rzeczywistym stoliku. Pierwsze kilka klasyfikacji od razu po położeniu przedmiotu są niejednoznaczne i często błędne. Po ~ 5s

wynik klasyfikacji się ustala i później już się nie zmienia (zazwyczaj). Jakość wyniku klasyfikacji (po ustaleniu się stanu) określiłbym jako bardzo zadowalającą, ale mimo wszystko czasem nie jest ona poprawna. Dlatego też zalecam ostrożność przy opieraniu się tylko na podstawie danych zwracanych przez węzeł.

7.2.8. Szacowanie wagi obiektu za pomocą sieci neuronowej

Nauczony doświadczeniami z projektowania sieci neuronowych dla rozpoznawania przedmiotu, postanowiłem spróbować zastosować tę technikę do poprawienia jakości estymacji wagi położonego przedmiotu. Dla ułatwienia prac postanowiłem jak najbardziej wykorzystać zaprojektowany już model sieci. Dlatego też, do sieci wykonanej na potrzeby klasyfikacji przedmiotów wprowadziłem poprawki, aby umożliwić jej wykorzystanie do szacowania wagi, w szczególności sieć została mocno zmniejszona. Okazało się w praktyce, że tak duża sieć mocno się przeucza na danych treningowych, przez co w praktyce na danych walidacyjnych i testowych nie radzi sobie najlepiej. Sieć, którą ostatecznie wykorzystałem do estymacji wagi jest widoczna na rysunku 7.11. Szacowanie wagi opisane w tym rozdziale jest alternatywnym podejściem do satysfakcji przypadku użycia opisanego przez UC5.



Rysunek 7.11. Architektura zaprojektowanej sieci CNN do estymowania wagi położonych obiektów

Same warstwy i ich znaczenie zostało opisane już w rozdziale 7.2.7, dlatego tutaj nie będę tego powtarzał. Zwróci jednak uwagę na zmiany, które wprowadziłem do modelu. Zostały usunięte 2 warstwy, dzięki czemu sama sieć została spłycona. Zmiany są widoczne również w końcowej części sieci, gdzie warstwa typu *Dense* z funkcją aktywacji *Softmax* została zamieniona na warstwę z liniową funkcją aktywacji. Jest to sposób aktywacji bardziej odpowiedni do wyznaczania pojedynczej wartości liczbowej na wyjściu. Widoczny jest również fakt, że sieć ta posiada tylko pojedyncze wyjście. Usunięta została również warstwa *Dropout*.

Do nauki sieci zostały wykorzystane obrazy z tego samego zbioru obrazów, co do nauki klasyfikatora przedmiotów, ale zmieniły się kryteria dotyczące obiektów wykorzystanych do uczenia. W tym przypadku użyłem tylko obrazów, które zostały zebrane i pierwotnie oznaczone jako przedmioty wewnętrz stolika. Nie brałem pod uwagę danych, które zostały oznaczone jako zebrane na krawędzi i częściowo poza stolikiem.

Obrazy wykorzystane do tego uczenia nie zostały również zmultiplikowane w celu zwiększenia liczby próbek testowych. Nie wykonałem tego, ponieważ jedną z obserwacji

z wykorzystania stolika jest fakt, że prawa i lewa strona stolika dają różne odczyty przy tym samym obciążeniu. Przy takim założeniu odwracanie obrazów, w celu nauczenia sieci na większej liczbie danych, zaprzecza całej idei poprawy odczytów.

Zmodyfikowana została natomiast metoda obliczania błędu przez sieć, a zarazem współczynnik optymalizacji. W rozdziale 7.2.7 błąd ten był zwykłym błędem średnio-kwadratowym. Tutaj błąd średniokwadratowy bardzo faworyzował dobrą estymację przy cięższych przedmiotach. Dlatego, podobnie jak w rozdziale 5.1.1, zmodyfikowałem sposób obliczania błędu, tak aby uwzględnił pierwotną wartość oczekiwana. Do obliczania błędu aproksymacji został wykorzystany błąd kwadratowy z procentowego odchylenia od wartości (6), gdzie e jest błędem aproksymacji, y_{true} faktyczną wagą przedmiotu, a y_{pred} przedyktowaną masą przedmiotu.

$$e = \left(100 * \frac{y_{true} - y_{pred}}{y_{true}} \right)^2 \quad (6)$$

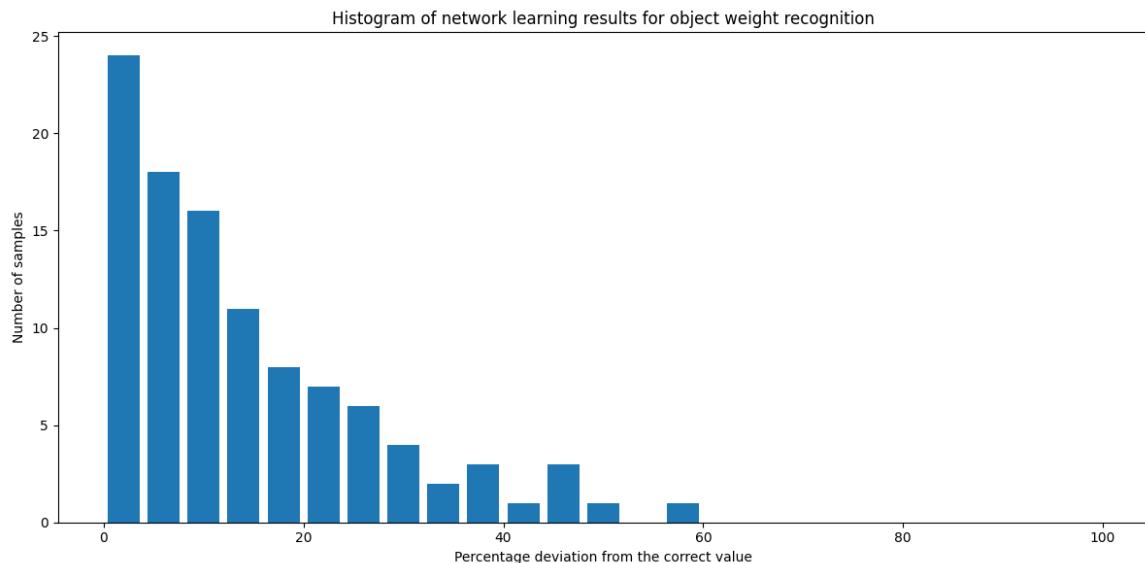
Stosując się do opisanych metod miałem do wykorzystania 1050 obrazów, na których uczyłem sieć. Zostały one podzielone na zbiory treningowy, walidacyjny oraz testowy w tej samej proporcji, co przy rozpoznawaniu obrazów, czyli 7 : 2 : 1. Aby sprawdzić jakość wytrenowanej w ten sposób sieci neuronowej wyznaczyłem kilka podstawowych wyznaczników statystycznych:

- maksymalne zanotowane odchyły: $-42,88\% / +59,21\%$,
- średni błąd bezwzględny: 53,75g,
- średni błąd względny: 11,66%,
- wariancja błędu: 272,48,
- odchylenie standardowe: 16,51.

Takie rozbieżności są w praktyce akceptowalne, mimo iż sama sieć nie nauczyła się na zebranych danych idealnie. Dla lepszego zobrazowania uczenia wykonałem histogram na podstawie przetworzenia danych testowych widoczny na rysunku 7.12. Niestety nie było to możliwe, aby utworzyć podobne porównanie dla metody szacowania wagi opisanej w rozdziale 7.2.6.

Wyniki estymacji wykonane przez sieć zdają się być lepsze od tych otrzymanych w rozdziale 7.2.6. Jednym, dostępnym do porównania tych metod, współczynnikiem jest jednak tylko maksymalne odchylenie pomiarów od faktycznej wartości, które na dodatek, w przypadku metody opartej na równaniu, zbierane było tylko organoleptycznie, przez obserwację działania w praktyce. W przypadku wykorzystania sieci odchylenie to zdaje się być lepsze, ale to dopiero faktyczne testy na fizycznym stoliku są w stanie zweryfikować obie metody.

Testy wytrenowanego modelu na fizycznym stoliku przebiegły całkiem dobrze. Wartości wagi przedmiotu, które odczytywałem ze stolika były całkiem blisko faktycznej wagi obiektu. Metoda ta w bezpośrednim porównaniu sprawdzała się dużo lepiej niż metoda



Rysunek 7.12. Histogram wyników uczenia sieci do rozpoznawania wagi przedmiotu

korzystająca tylko ze wzoru. Te względnie poprawne odczyty wagi są dostarczane do systemu już w kilka pomiarów po położeniu obiektu. Niestety, niesie to też za sobą konsekwencję tego, że z uwagi na brak implementacji dynamicznej charakterystyki Velostatu, po dłuższym czasie estymowana waga wynosi wyraźnie więcej niż przedmiot faktycznie waży. Między prawą i lewą stroną stolika wciąż jest widoczna delikatna różnica w odczytach wagi. Podsumowując, uważam, że ta metoda jest lepsza od tej prezentowanej w rozdziale 7.2.6. Przy korzystaniu z niej również zalecam pewien zapas zaufania, szczególnie przy większych obciążeniach ($> 2\text{kg}$), ponieważ takich danych sieć nie posiadała w trakcie uczenia.

7.3. Struktura intrasystemu zarządzającego scenariuszami

Intrasystem opisany w tym rozdziale miał za zadanie bezpośrednio obsłużyć scenariusze opisane w rozdziałach 6.1 oraz 6.2. Istniejący w tym systemie węzeł będzie wykorzystywał informacje wysyłane przez węzeł inteligentnego stolika. Węzeł ten, poza samymi informacjami ze stolika, musi wykorzystywać również inne informacje z systemu robota, jak również wysyłać do niego odpowiednie informacje i polecenia. Dla prezentowanych scenariuszy kluczowe jest wykorzystanie wymienionych informacji dostępnych w systemie robota:

- nawigacji po laboratorium,
- aktualnej mapy laboratorium,
- wiedzy o aktualnej lokalizacji robota oraz lokalizacji poszczególnych miejsc w laboratorium,
- rozpoznawania poleceń głosowych i możliwość naturalnej komunikacji z robotem [46].

7. Implementacja węzłów ROS wykonujących zaplanowane scenariusze

Przykładowy scenariusz zakłada wykorzystanie stolika w praktycznym zastosowaniu. Aby wykonać praktyczny scenariusz, oprócz projektowanego węzła inteligentnego stolika musiałem wykorzystać jeszcze inne węzły, rozwijane przez innych w laboratorium. Aby zebrać informacje o potrzebnych zasobach w jednym miejscu wykonałem odpowiednie schematy - diagram definicji bloków (7.13) oraz diagram wewnętrzny (7.14).

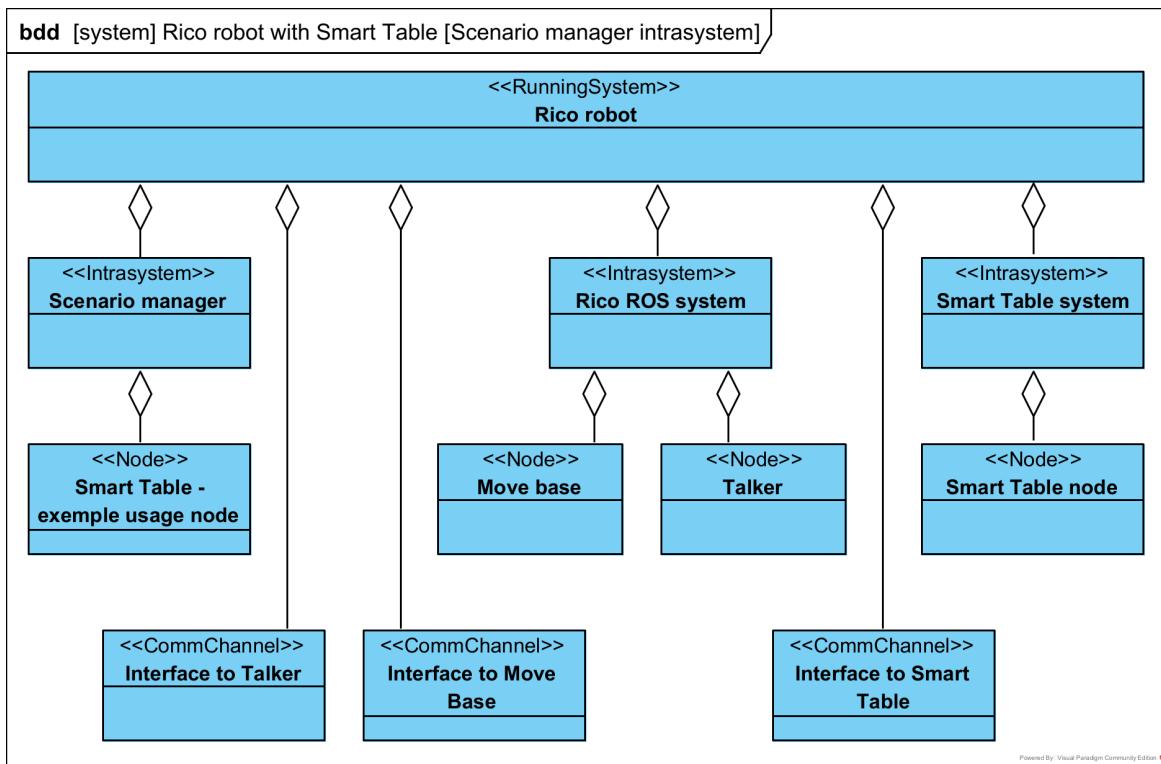
Na diagramie definicji bloków widocznym na rysunku 7.13 można wyróżnić następujące bloki funkcjonalne:

- *Rico robot* – cały system działającego robota Rico,
- *Scenario manager* – intrasystem zawierający elementy odpowiedzialne za zarządzanie wykonaniem scenariusza, opisywany w tym rozdziale,
- *Rico ROS system* – intrasystem zawierający pozostałe elementy ROSowe robota Rico,
- *Smart Table system* – intrasystem zawierający elementy odpowiedzialne za prawidłowe działanie inteligentnego stolika,
- *Smart Table – example usage node* – węzeł ROS obsługujący scenariusze wykonywane przez robota,
- *Move base* – węzeł odpowiadający za poruszanie się robota,
- *Talker* – węzeł odpowiadający za komunikację głosową robota z otoczeniem,
- *Smart Table node* – węzeł ROS inteligentnego stolika,
- *Interface to Talker* – interfejs komunikacyjny pomiędzy węzłem obsługującym scenariusze a węzłem odpowiadającym za komunikację głosową z robotem,
- *Interface to Move base* – interfejs komunikacyjny pomiędzy węzłem obsługującym scenariusze a węzłem odpowiedzialnym na jazdę i nawigację robota,
- *Interface to Smart Table* – interfejs komunikacyjny pomiędzy węzłem obsługującym scenariusze a inteligentnym stolikiem.

Zbiór tych bloków warto porównać z prezentowanym wcześniej rysunkiem 7.2, który prezentuje tylko intrasystem inteligentnego stolika. Oba te diagramy uzupełniają się wzajemnie.

Węzły *Move base* oraz *Talker* już wcześniej działały na robocie, jednak dopiero w tym momencie postanowiłem je wyodrębnić, ponieważ będą one bezpośrednio wykorzystywane przy wykonaniu scenariuszy. Jako, że oba były już wcześniej działające na robocie to zostały przedstawione jako węzły należące do intrasystemu *Rico ROS system*. Węzeł *Smart Table - example usage node*, będący częścią intrasystemu *Scenario manager*, został zaprojektowany przeze mnie i jego zadaniem jest obsługa wykonania scenariuszy postawionych przed robotem. Koordynuje on całe działanie robota i inteligentnego stolika. Z intrasystemu *Smart Table system* został wyszczególniony tylko węzeł *Smart Table node*, który przetwarza w sobie wszystkie informacje na temat czujnika taktylnego i przedmiotów znajdujących się na stoliku. Na diagramie widocznym na rysunku 7.13 widoczne są również trzy kanały komunikacji: *Interface to Talker*, *Interface to Move base* oraz *Inter-*

7. Implementacja węzłów ROS wykonujących zaplanowane scenariusze



Rysunek 7.13. Diagram definicji bloków dla intrasystemu zarządzającego wykonaniem scenariuszy

face to Smart Table. Wszystkie te kanały są osadzone w działającym systemie, ponieważ odpowiadają one za komunikację pomiędzy poszczególnymi intrasystemami.

Na diagramie bloków wewnętrznych przedstawionym na rysunku 7.14 widoczne są wszystkie wspomniane kanały komunikacji i sposób ich wykorzystywania przez węzeł przykładowego użycia inteligentnego stolika. Aby wyszczególnić jak dokładnie wygląda komunikacja pomiędzy poszczególnymi węzłami powstały dodatkowe diagramy opisujące dokładnie każdy z wykorzystywanych kanałów komunikacji.

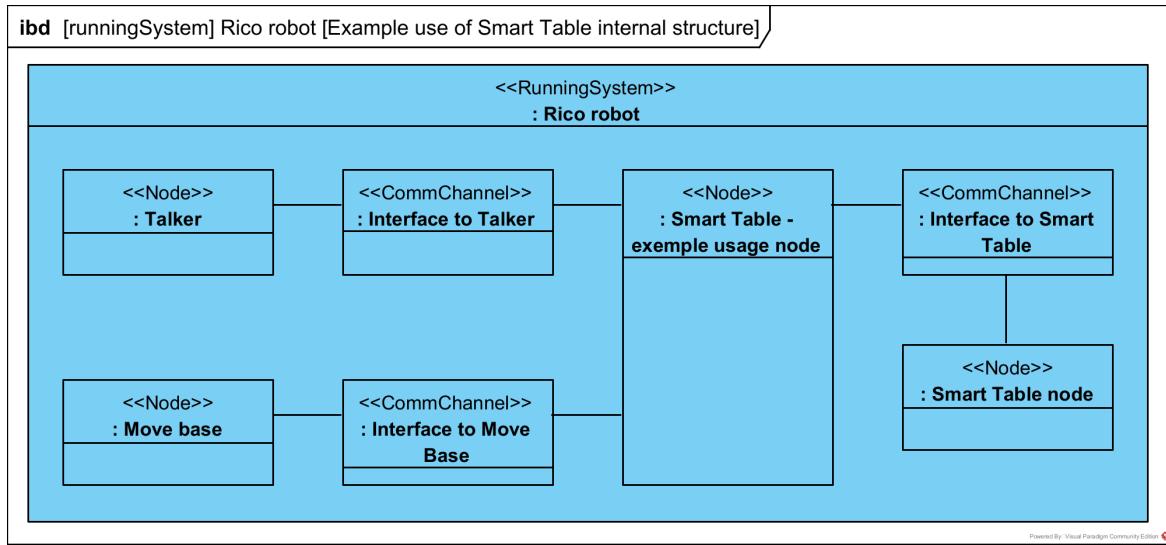
Interfejs do węzła inteligentnego stolika został przedstawiony na rysunku 7.15. Widoczne na nim są tylko cztery tematy:

- */is_placed* – sygnał informujący o tym, czy na stoliku jest aktualnie położony jakiś przedmiot,
- */location* – lokalizacja położonego na stoliku przedmiotu,
- */weight* – szacunkowa waga położonego na stoliku obiektu,
- */predicted_item* – informacja o tym, jaki przedmiot znajduje się na stoliku.

Sam węzeł inteligentnego stolika obsługuje więcej tematów, ale nie niosą one żadnej wartości przy wykonywaniu scenariuszy. Dlatego też nie są one wyszczególnione na diagramie. Na diagramie tym można również zauważyć, że wszystkie informacje przesyłane są tylko w jednym kierunku – do węzła wykorzystującego inteligentny stolik.

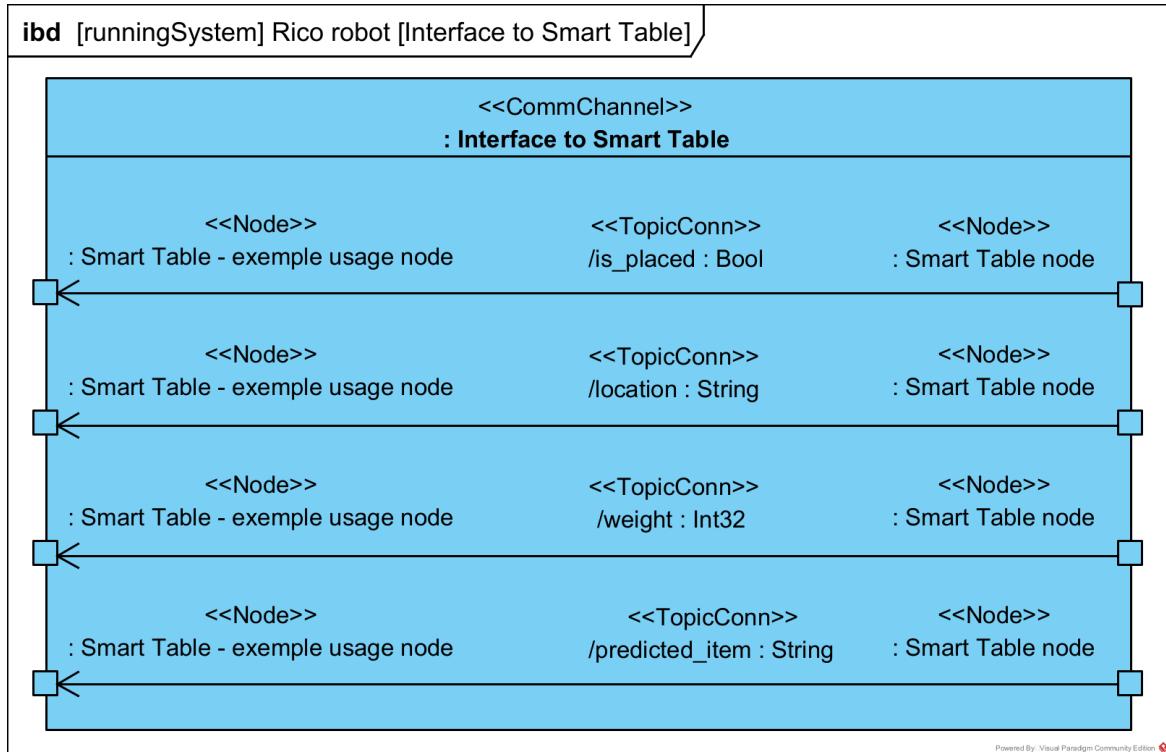
Interfejs *Interface to Move Base*, widoczny na rysunku 7.16, obsługuje już komunikację dwustronną. Węzeł *Move Base* przyjmuje informację o docelowej pozycji robota

7. Implementacja węzłów ROS wykonujących zaplanowane scenariusze



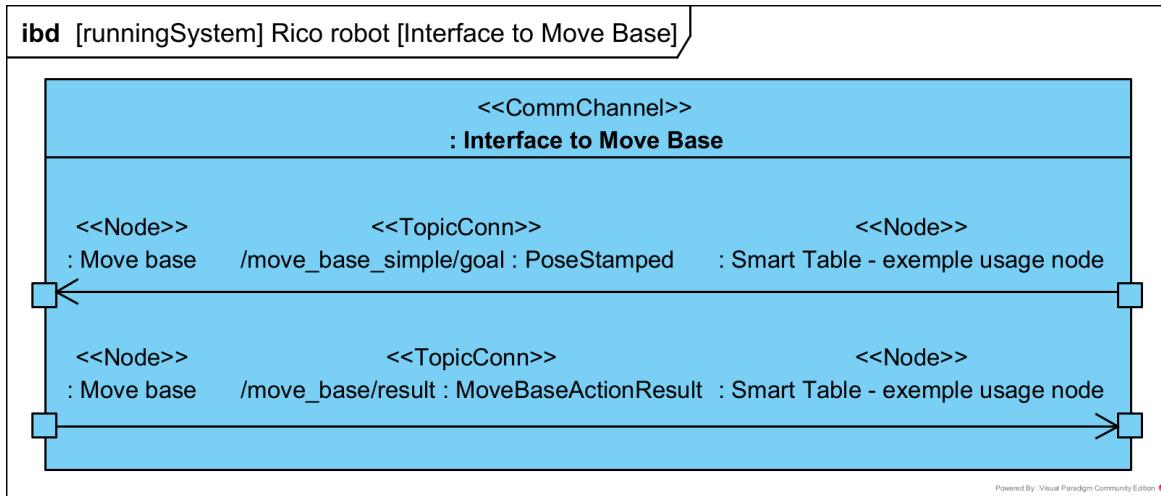
Rysunek 7.14. Diagram bloków wewnętrznych dla wykorzystywanego w scenariuszach węzła, który wykorzystywał dane z inteligenckiego stolika

w środowisku, w układzie kartezjańskim wraz z rotacją robota. Docelowa pozycja robota przesyłana jest na temacie `/move_base_simple/goal`. Węzeł ten zwraca natomiast status odpowiedni dla tego czy dojechał do zadanego miejsca, czy też nie udało mu się wykonać tego zadania. W przypadku nieudanego dojedania do wyznaczonego miejsca wysyłany jest również stosowny kod błędu, który określa z jakiego powodu robot nie mógł dojechać



Rysunek 7.15. Diagram bloków wewnętrznych obrazujący komunikację, pomiędzy węzłem przykładowego użycia inteligenckiego stolika a węzłem inteligenckiego stolika

7. Implementacja węzłów ROS wykonujących zaplanowane scenariusze



Rysunek 7.16. Diagram bloków wewnętrznych obrazujący komunikację, pomiędzy węzłem przykładowego użycia inteligentnego stolika a węzłem odpowiadającym za poruszanie

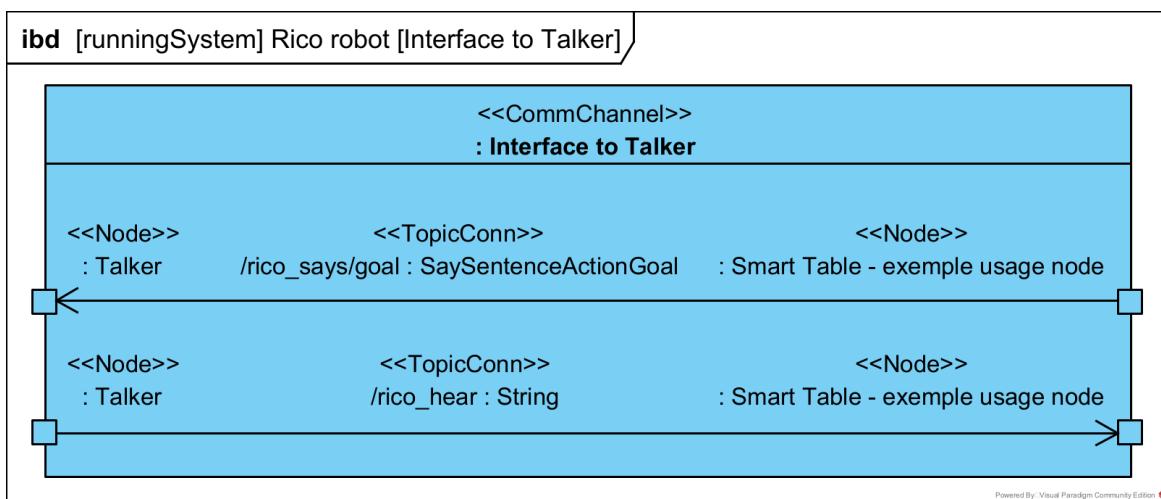
do ustalonego położenia. Status wysyłany jest na temacie */move_base/result*. Wysyłany jest on również tylko jeden raz, dlatego ważne jest, aby nasłuchiwać na tym temacie dopiero po wysłaniu polecenia ruchu i zinterpretować pierwszą otrzymaną wiadomość.

Ostatnim z wyszczególnionych na rysunku 7.14 kanałów komunikacji jest *Interface to Talker*. Interfejs ten jest stosunkowo prosty i składa się z dwóch tematów:

- */rico_says/goal* – tekst, który Rico ma wypowiedzieć,
- */rico_hear* – tekst, który Rico usłyszał.

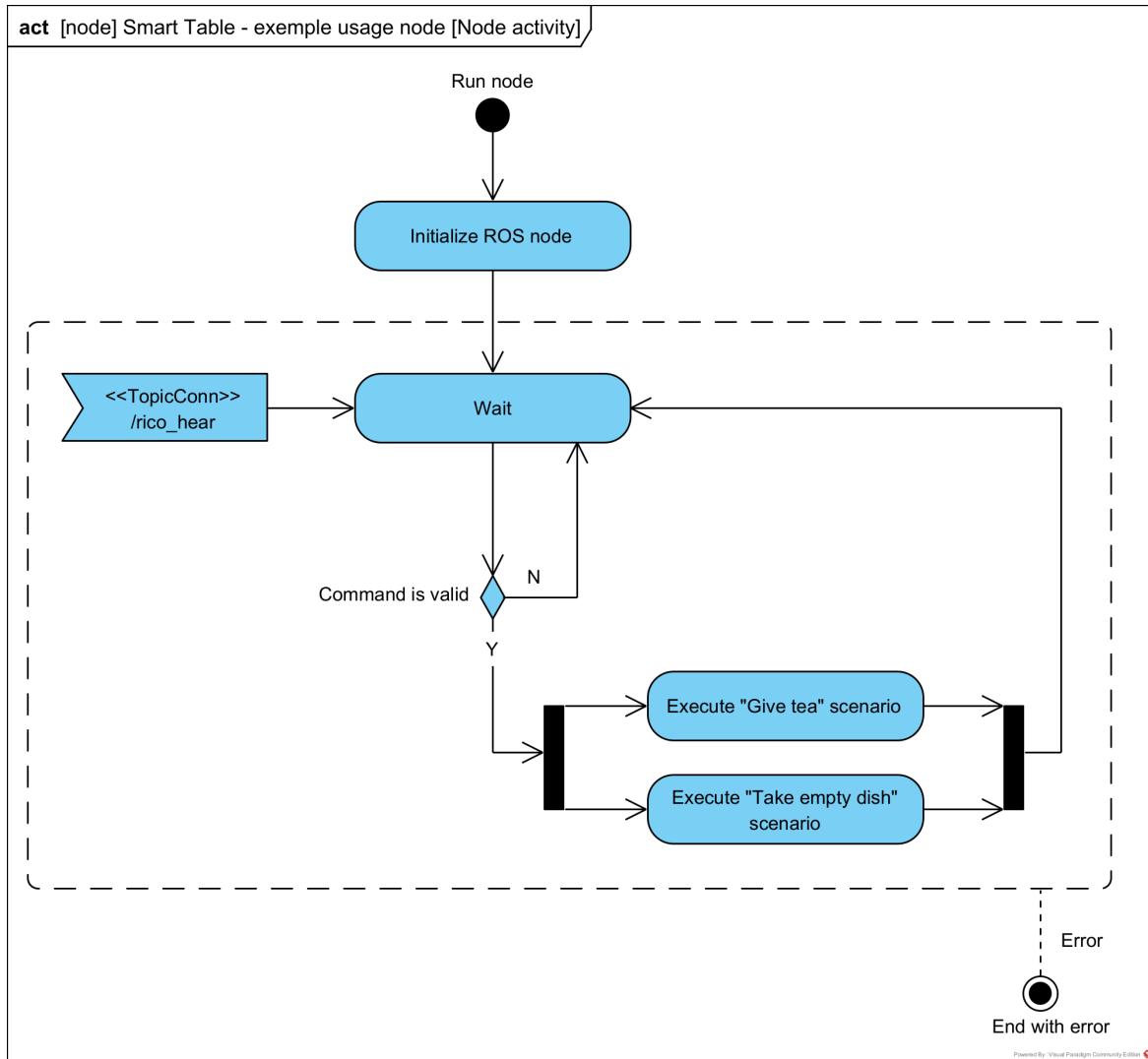
Diagram prezentujący ten interfejs widoczny jest na rysunku 7.17. Widoczne tam są również kierunki przepływu danych na wymienionych tematach.

Po dokładnym ustaleniu jakie dane mają przepływać z oraz do węzła wykorzystującego



Rysunek 7.17. Diagram bloków wewnętrznych obrazujący komunikację, pomiędzy węzłem przykładowego użycia inteligentnego stolika a węzłem odpowiadającym za komunikację głosową robota

7. Implementacja węzłów ROS wykonujących zaplanowane scenariusze



Rysunek 7.18. Diagram akcji obrazujący sposób działania węzła wykonującego zaplanowany scenariusz

inteligentny stolik zaprojektowałem również sposób działania tego węzła. Diagram akcji przedstawiający to działanie widoczny jest na rysunku 7.18.

Działanie węzła jest bardzo proste. Po uruchomieniu i poprawnym zainicjalizowaniu węzła oczekuje on na polecenia wydane przez użytkownika. Polecenie to jest wypowiadane przez użytkownika, a węzeł oczekuje stosownej informacji na temacie /rico_hear. Po otrzymaniu polecenia sprawdzane jest, czy węzeł jest w stanie je obsługiwać. Na ten moment węzeł obsługuje tylko 2 scenariusze: przywiezienie herbaty oraz odwiezienie pustego naczynia. Po stwierdzeniu, że węzeł jest w stanie obsługiwać żądanie uruchamiany jest odpowiedni scenariusz. Po jego wykonaniu węzeł ponownie wraca do początkowego stanu oczekiwania na kolejne polecenie. Same scenariusze zostały opisane w rozdziałach 6.1 dla scenariusza przywożenia herbaty oraz 6.2 dla odwiezienia naczynia. Na diagramie zaznaczone zostało również, że jeśli węzeł natrafi na krytyczny dla siebie błąd to zakończy on swoje zadanie ze stosowym błędem.

7.4. Problemy integracji węzłów z systemem ROS robota

W trakcie rozwijania oprogramowania natknąłem się na kilka wyraźnych ograniczeń systemowych, a dokładniej kompatybilności wykorzystywanych narzędzi. Ograniczenia te zostały narzucone przez dostępny sprzęt laboratoryjny oraz niezbędne w mojej pracy biblioteki. Komponenty, które nałożyły mi największe ograniczenia to:

- system ROS, w dystrybucji melodic,
- system operacyjny Linux Ubuntu 18.04,
- język skryptowy Python 2.7,
- biblioteka TensorFlow 2.12 (dostępna tylko w nowszych dystrybucjach języka Python np. Python 3.8).

Głównym ograniczeniem okazała się dystrybucja systemu ROS, na której działa robot Rico, dostępny w laboratorium. Działa on na systemie ROS w dystrybucji melodic. Nałożyło to spore ograniczenia na narzędzia, które mogły zostać przejęte mnie użyte. Dystrybucja ta dostępna jest tylko na starszych wersjach systemu Linux Ubuntu (17.10 i 18.04), które nie są już oficjalnie wspierane [85].

Dodatkowym, sporym utrudnieniem wprowadzanym przez dystrybucję melodic jest fakt, że wspiera ona Pythona tylko do wersji 2.7, który również jest przestarzały i od kilku lat nie rozwijany. Co za tym idzie, wiele bibliotek nie jest rozwijanych na tę wersję języka. Uniemożliwiło mi to wykorzystanie aktualnych narzędzi i bibliotek, które były mi niezbędne do wykonania zaplanowanych badań [57].

Do wykonania badań nad rozpoznawaniem przedmiotów niezbędna była dla mnie biblioteka TensorFlow w wersji 2.12 (wersje 2.13 oraz 2.14 zawierają krytyczne dla mnie błędy). Ta wersja oprogramowania do poprawnego działania wymaga użycia Pythona w wersji 3.7 lub nowszej. Powoduje to krytyczny w moim przypadku konflikt dotyczący wykorzystywanej na sprzęcie wersji Pythona [58].

Z uwagi na ograniczenia narzucone przez bibliotekę TensorFlow zdecydowałem się całość kodu źródłowego węzłów napisać w Pythonie 3.8 i na późniejszym etapie spróbować zintegrować go ze starszym systemem robota. Ostatecznie okazało się, że mój kod źródłowy uruchamiany przez Pythona 3.8 był w stanie bez większych problemów wykorzystać interfejs oferowany przez ROS melodic (w wersji Pythona 2.7). Ostatecznie wygląda na to, że węzeł wykorzystuje zarówno Pythona 3.8, jak i Pythona 2.7, ale nie jestem w stanie stwierdzić na jakiej dokładnie zasadzie działa ta współpraca.

8. Wykonanie zaplanowanych scenariuszy

8.1. Środowisko wykorzystane w scenariuszach

Scenariusze, które zostały zaplanowane w rozdziale 6 zostały również wykonane w rzeczywistym środowisku środowisku. Zostały one wykonane w laboratorium robotycznym na Politechnice Warszawskiej, na wydziale Elektroniki i Technik Informacyjnych. Laboratorium, w którym przeprowadzone zostały eksperymenty, zostało przedstawione na rysunku 8.1. Na fotografii zostały podpisane najważniejsze miejsca wykorzystywane w scenariuszu, czyli stolik, kuchnia oraz pozycja wyjściowa.



Rysunek 8.1. Laboratorium wykorzystane podczas wykonywania scenariuszy użycia inteligentnego stolika

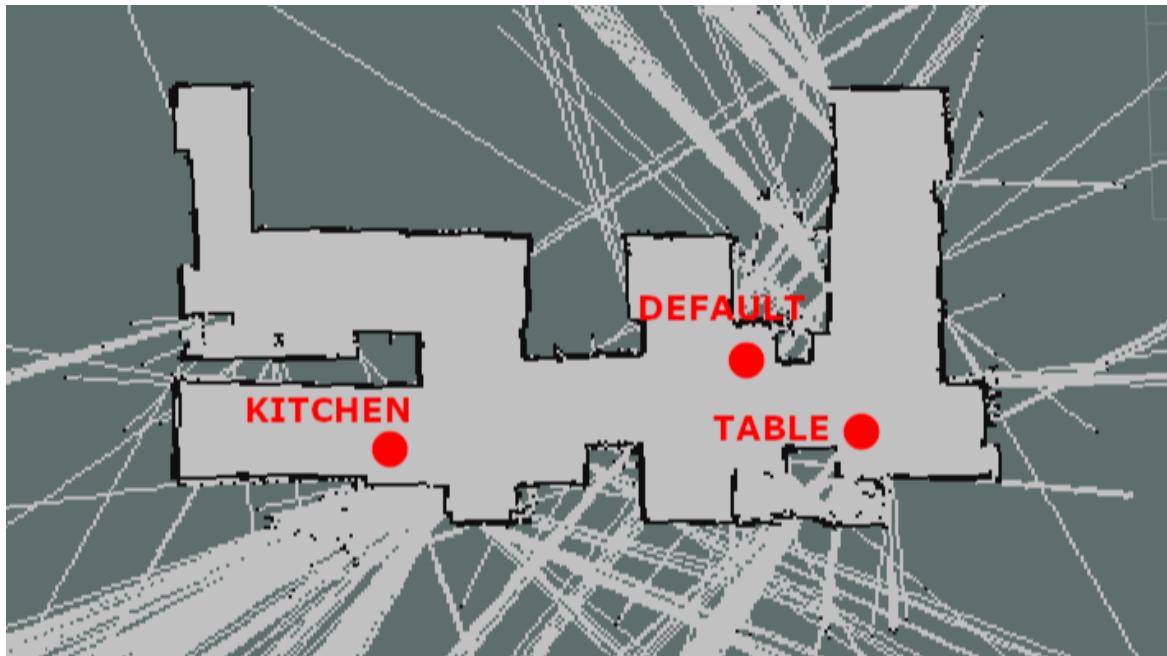
Po prawej stronie, za kolumną, widoczna jest pozycja domyślna i zarazem startowa robota. Niedaleko tej pozycji znajduje się także stacja dokująca robota. Miejsce to znajduje się mniej więcej na środku laboratorium, dzięki czemu robot jest w stanie usłyszeć potencjalne polecenia od każdej osoby, niezależnie od tego gdzie aktualnie się ona znajduje.

Po lewej stronie stronie widoczny jest wykorzystywany w scenariuszach stolik wraz z dwoma fotelami. Jest to podstawowe miejsce dla osoby, która prosi o wykonanie robota jakiejś czynności.

W oddali widoczna jest kuchnia. Kuchnia jest miejscem pracy opiekuna, który w scenariuszu przygotowuje podopiecznemu herbatę. Droga do kuchni nie jest mocno skomplikowana, ale samo zadanie nie ma na celu testować zdolności nawigacyjnych robota.

Pierwszym etapem, przed przystąpieniem do wykonania scenariusza, było sporządzenie mapy laboratorium. Mapa ta jest wgrana do systemu ROS, na którym działa Rico.

Mapa ta przedstawia fizyczne i stałe przeszkody, których robot nie jest w stanie przejechać. Mapa ta, oprócz podstawy przy wyznaczaniu ścieżki przejazdu, jest kluczowa przy wyznaczaniu położenia robota, ponieważ robot przy wyznaczaniu swojej pozycji w środowisku wspomaga się aktualnymi pomiarami ze skanera laserowego. Sama mapa została zaprezentowana na rysunku 8.2. Podobnie jak w przypadku zdjęcia laboratorium, na mapie zostały podpisane najważniejsze dla robota miejsca.



Rysunek 8.2. Mapa laboratorium wykorzystane podczas wykonania scenariuszy

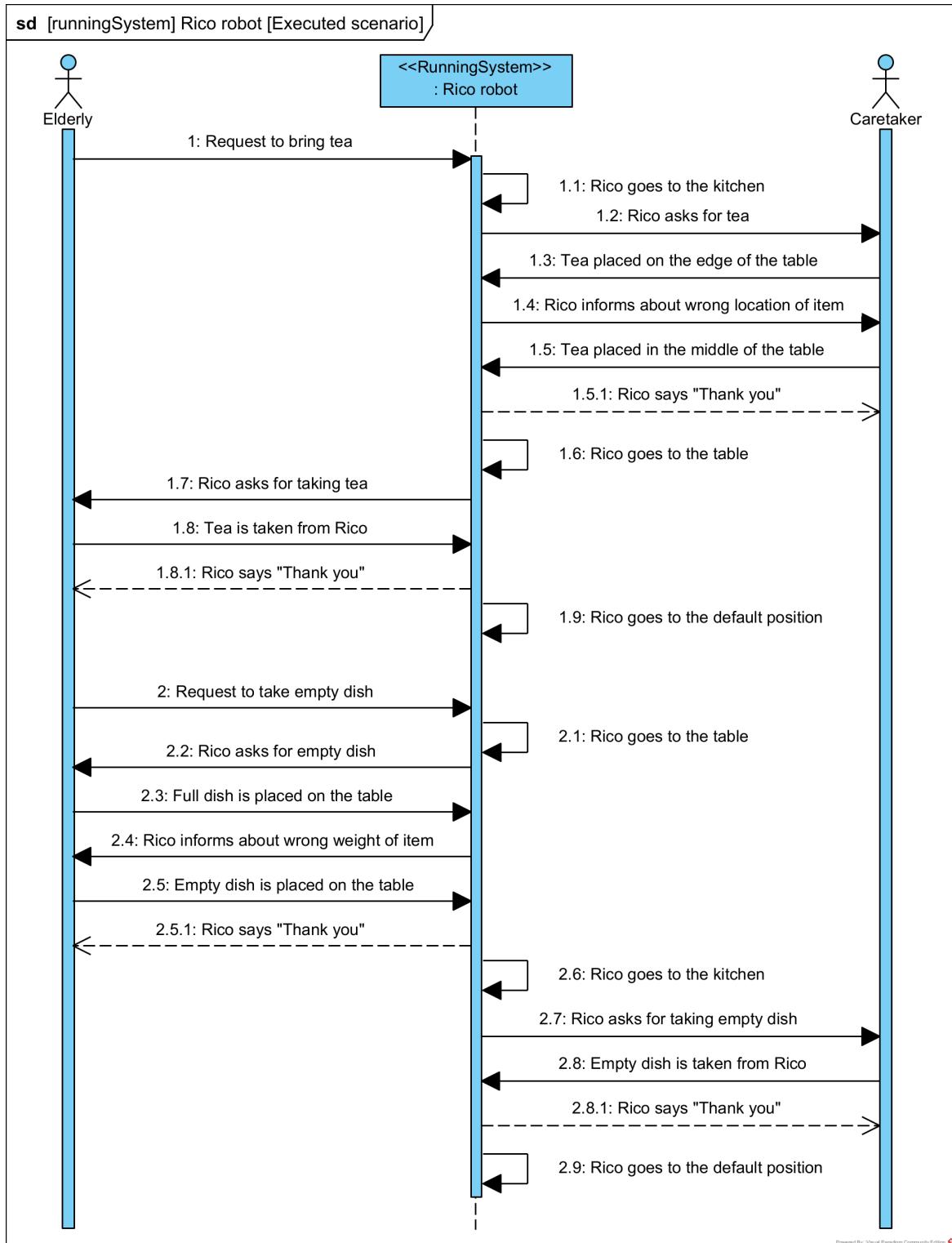
Widoczna na rysunku 8.2 mapa została wykonana przez Rico wcześniej, podczas sterowania ręcznie z zadajnika. Na mapie kolorem białym oznaczony jest obszar, po którym robot może swobodnie się poruszać i planować ścieżkę przejazdu. Obszar ten ograniczony jest czarnymi liniami, które oznaczają ścianę lub przedmiot (próg, stolik, szafa), przez który robot nie jest w stanie przejechać. Czerwonymi punktami zostały oznaczone najważniejsze dla robota miejsca wykorzystywane w scenariuszach.

Na mapie widoczne są również białe promienie wychodzące poza widoczne ściany laboratorium. Są to błędy, które wytwarzły się podczas zbierania danych z mapy w miejscowościach, gdzie robot z różnych powodów nie był w stanie poprawnie wyznaczyć odległości od ściany. Promienie te nie są czymś niepoającym, są one tylko błędem technologicznym tworzenia mapy. Robot w trakcie swojej normalnej pracy nie sugeruje się tymi błędami.

8.2. Wykonanie zaplanowanych scenariuszy

W laboratorium przeprowadzone zostały dwa scenariusze, opisane w rozdziałach 6.1 oraz 6.2. Scenariusze te zostały wykonane jeden po drugim, aby ukazać stabilność działania inteligentnego stolika. Dokładny przebieg obu scenariuszy został przedstawiony na diagramie sekwencji widocznym na rysunku 8.3.

8. Wykonanie zaplanowanych scenariuszy



Rysunek 8.3. Diagram sekwencji scenariusza faktycznego wykonanego na robocie Rico

Dla celów bezpieczeństwa zamiast przewożenia płynów przewożony był ryż. Podłoga w laboratorium jest nierówna, przez co ruchy robota czasami są skokowe i gwałtowne, a potencjalne rozłanie płynów może skutkować rozległą awarią czujnika taktylnego oraz robota.

Robot rozpoczyna oba scenariusze stojąc i czekając na odpowiednie polecenia w pozycji domyślnej, czyli dokładnie tam, gdzie jest on widoczny na zdjęciu 8.1. W obu scenariuszach pełnił zadania zarówno osoby starszej, jak i jej opiekuna, i przemieszczałem się pomiędzy stolikiem a kuchnią razem z robotem. Nagranie z wykonanych scenariuszy jest dostępne również na serwisie Vimeo¹².

W celu rozpoczęcia pierwszego scenariusza (6.1) wywołana została komenda przywiezienia herbaty. Rico po otrzymaniu i zrozumieniu danego polecenia udaje się do kuchni, aby odebrać od znajdującego się tam opiekuna herbatę. W kuchni opiekun kładzie kubek na krawędzi stolika. Sprawdzana jest w ten sposób możliwość robota do wykrycia faktu, że obiekt znajduje się na krawędzi stolika (wykorzystanie *UC2*). Robot zwraca się do opiekuna z prośbą o poprawienie lokalizacji obiektu. Po ponownym, tym razem poprawnym, położeniu kubka na stoliku przez opiekuna robot potwierdza to (wykorzystanie *UC4*) i kieruje się z kubkiem do podopiecznego znajdującego się przy stoliku. Przy stoliku kubek z herbatą jest odbierany, a robot po wykryciu i potwierdzeniu tej operacji (*UC4*) udaje się z powrotem do lokalizacji domyślnej, aby oczekwać na kolejne polecenia.

Po pomyslnym zakończeniu pierwszego zaplanowanego scenariusza od razu wywołany został drugi (6.2). W tym scenariuszu Rico proszony jest o odwiezienie kubka z powrotem do kuchni. Po wydaniu tego polecenia robot w pierwszej kolejności podjeżdża do osoby przy stoliku, aby odebrać kubek. Podopieczny kładzie na stoliku pełny kubek, ale robot wykrywa, że masa przedmiotu jest zbyt duża, aby mogło to być puste naczynie, dlatego prosi o upewnienie się, że prawidłowy przedmiot został położony na jego grzbicie (wykorzystanie *UC5*). Aktor ściąga kubek, wylewa płyn (w tym przypadku wysypuje ryż) i ponownie kładzie, tym razem pusty, kubek na stoliku robota. Rico po potwierdzeniu (*UC4*), że przedmiot znajduje się w akceptowalnym dla tego zadania przedziale wagi od razu udaje się do kuchni. W kuchni opiekun odbiera pusty kubek. Rico po wykryciu, że kubek został odebrany (*UC4*) kończy wykonywanie scenariusza i wraca z powrotem do pozycji domyślnej, gdzie znowu czeka na kolejne zadania.

Oba scenariusze przebiegły pomyślnie, mimo zdarzających się w ich trakcie jednostkowych błędów. Zdarzające się błędy obejmowały głównie zakres odczytu masy przedmiotu, który nie zawsze mieścił się w wyznaczonych w kodzie źródłowym zakresach, mimo iż w rzeczywistości przedmiot mieścił się w nich. Poza tym faktem stolik działał poprawnie i bez zarzutu.

¹² Nagranie z wykonania scenariuszy: <https://vimeo.com/903840750>

9. Wnioski

Podsumowując prace, które prowadziłem i opisałem w tym dokumencie, można dojść do wielu wniosków, zarówno tych pozytywnych, jak i negatywnych. Ze względu na szeroki zakres prowadzonych przeze mnie działań wnioski są zróżnicowane dla poszczególnych aspektów pracy. Dlatego dla każdego z poruszonych aspektów sformuowałem już wcześniej odpowiednie, szczegółowe wnioski. Ogólna moja ocena tego projektu jest taka, że jest on użyteczny i bardzo ułatwia komunikację z robotem, ale sam czujnik taktylny pozostawia wiele do życzenia.

Cele postawione przeze mnie na początku prac zostały w większości wykonane zadowalająco. Przegląd literatury zapewnił mi nowe światło na aktualnie rozwijane trendy w nauce i pogląd na to, co znajduje się na rynku konsumenckim. Pokazał mi również nowe, ciekawe zastosowania dla matrycowego czujnika taktylnego.

Stolik, jego budowa oraz montaż do robota zostały wykonane dobrze, ale sama konstrukcja mechaniczna stolika nie jest pozbawiona wad. Wynikają one z manualnego składania stolika oraz dynamicznego charakteru folii Velostat. Elektronika sterująca czujnikiem taktylnym działa bezproblemowo.

Oba napisane przeze mnie węzły ROS działają dobrze i wypełniają wszystkie zadania przed nimi stawiane. Ich plusem jest łatwy interfejs, proste użycie oraz niezależność od konkretnej dystrybucji ROS. Metody wykorzystane do określenia parametrów położonego obiektu są zadowalające, ale moje niewielkie praktyczne doświadczenie z nimi mogło ograniczyć zakres i użyteczność wykorzystania ich w pracy.

9.1. Podsumowanie założonych przypadków użycia inteligentnego stolika

Na początku pracy postawiłem założenia dotyczące przypadków użycia inteligentnego stolika, są one widoczne na rysunku 1.2. Na końcu pracy stosownym jest więc wykonać weryfikację, czy zaprojektowany przeze mnie system inteligentnego stolika jest w stanie je spełnić.

- *UC1 – Monitorowanie statusu przedmiotów położonych na stoliku.*

Ten przypadek wykorzystania nie został sprawdzony w praktycznym scenariuszu, ponieważ podczas ich przeprowadzania nie było zewnętrznego obserwatora, który mógłby to zrobić. Funkcja ta była sprawdzana doraźnie, podczas testów na robocie, do detekcji błędów, poprzez obserwowanie informacji przesyłanych na poszczególnych tematach. Nie został wykonany interfejs graficzny, który umożliwiłby w bardzo prosty sposób obserwować te dane, ale w razie potrzeby można je obserwować w programie RViz.

- *UC2 – Lokalizacja obiektów na stoliku.*

Lokalizacja obiektów na stoliku została sprawdzona zarówno podczas fizycznego scenariusza, jak i podczas testów jednostkowych. Opis projektowania sposobu lokalizacji obiektów na stoliku został przedstawiony w rozdziale 7.1. Lokalizacja obiektów na stoliku została sprawdzona zarówno podczas fizycznego scenariusza, jak i podczas testów jednostkowych. Opis projektowania sposobu lokalizacji obiektów na stoliku został przedstawiony w rozdziale 7.1.

zacji przedmiotu na stoliku jak i testów jednostkowych jest opisany w rozdziale 7.2.5. Lokalizacja obiektu została również sprawdzona podczas pierwszej części realizacji scenariusza na rzeczywistym robocie i przebiegła pomyślnie. Same wyniki lokalizacji obiektów były dobre i wykorzystane metody są bardzo zadowalające.

- *UC3 – Identyfikacja przedmiotów położonych na stoliku.*

Identyfikacja przedmiotów była testowana tylko podczas testów jednostkowych, które zostały opisane w rozdziale 7.2.7. Rozpoznawanie położonych przedmiotów okazało się być trudnym zadaniem z uwagi na bardzo niską rozdzielcość czujnika. Mimo to z zaprojektowane rozwiązanie było w stanie z dużym prawdopodobieństwem określić jaki przedmiot jest na nim położony.

- *UC4 – Zautomatyzowane potwierdzenie bądź zaprzeczenie wykonaj przez człowieka akcji.*

Potwierdzanie wykonania przez człowieka odpowiedniej akcji opierało się w bardzo dużej mierze na detekcji czy na stoliku coś zostało położone. Fragment opisujący badania prowadzone nad tym znajdują się w rozdziale 7.2.4. Oprócz samej detekcji czy obiekt znajduje się na stoliku do stwierdzania, że akcja wykonana przez człowieka jest nieprawidłowa były wykorzystane funkcje lokalizacji przedmiotu (rozdział 7.2.5) oraz pomiaru wagi przedmiotu (rozdziały 7.2.6 oraz 7.2.8). Sama automatyzacja zatwierdzania bądź negacji wykonania akcji przez człowieka przebiegła bardzo dobrze. Została ona również pomyślnie przetestowana w fizycznie wykonanych scenariuszach.

- *UC5 – Szacunkowa waga położonego przedmiotu.*

Szacowanie wagi położonego przedmiotu zostało opisane szczegółowo w rozdziałach 7.2.6 oraz 7.2.8. Oba sposoby estymacji wagi obiektu zostały przetestowane jednostkowo dając wyniki miejscami obarczone sporym błędem. Żadna z metod nie potrafiła poprawnie oszacować wagi wszystkich położonych przedmiotów. Mimo tych trudności metoda wykorzystująca sieci neuronowe została użyta i przetestowana podczas fizycznego scenariusza. Otrzymane wyniki zazwyczaj mieściły się w dużym przedziale tolerancji wagi, ale czasami jednak otrzymywałem błąd, nawet przy poprawnym obiekcie.

9.2. Wnioski dotyczące technologii wykonania czujnika taktylnego

O ile wykonany przeze mnie czujnik taktylny spełniał swoje zadanie prawidłowo, to jeśli chodzi o technologię jego wykonania, to widzę szerokie pole do poprawy. Zaletami, jakie oferuje wybrane przeze mnie rozwiązanie, są:

- niska cena materiałów,
- łatwość budowy,
- łatwość zbierania pomiarów,
- możliwość dopasowania do wybranej powierzchni,

9. Wnioski

- duża skalowalność rozwiązania.

Te cechy jednak nie zapewniły mi wygodnej i bezproblemowej pracy z czujnikiem. Do wad, które zauważałem podczas swojej pracy, można bez wątpienia zaliczyć:

- nieliniowość materiału Velostat,
- długie ustalenie się odpowiedzi Velostatu,
- różna charakterystyka materiału w zależności od tego, czy nacisk był dokладany czy zabierany,
- folia miedziana ma dużą podatność do tworzenia się na niej nierówności i fałdowania się,
- nierównomierne rozkładanie się nacisku przedmiotu na gumie.

Poza tym, na negatywne działanie czujnika wpływa również fakt, że z niewiadomego powodu jedna jego połowa jest bardziej czuła niż druga oraz czujnik przy krawędziach jest mniej czuły (z uwagi na użycie kleju).

Dodatkowo, szумy i niestałości pracy stolika przyczyniły się do niemożliwości wyznaczenia dobrych sposobów na wyznaczanie parametrów położonych przedmiotów. Odczyty były tak mocno zniekształcone, że nie można było z nich, z dobrą dokładnością, nic wyznaczyć i bezpośrednio przełożyły się na tak słabe wyznaczanie masy przedmiotu.

Porównując zalety i wady wybranego rozwiązania oraz dodając do tego swoje praktyczne doświadczenia podczas pracy ze stolikiem, mogę jednoznacznie stwierdzić, że Velostat jest bardzo niewdzięcznym materiałem. Jego niska cena i łatwość obsługi nie równoważą problemów, jakie powoduje on podczas korzystania z niego. Rozważałbym wykorzystanie innego materiału i sposobu obsługi intelligentnego stolika.

9.3. Wnioski dotyczące wykorzystanych metod detekcji parametrów położonego obiektu

Wykorzystane przeze mnie metody i sposoby rozpoznawania parametrów obiektu sprawdziły się dobrze. Szczególnie biorąc pod uwagę niewielką rozdzielcość dostępnych danych i duże szumy samego czujnika. Połączenie wykorzystania przeze mnie algorytmów (przy prostych problemach) oraz sieci neuronowych (w bardziej skomplikowanych przypadkach) dało dobre i oszczędne obliczeniowo rezultaty. Również przekształcenie pomiarów z czujników taktynnych na obraz okazało się dobrym podejściem, ponieważ pozwoliło w prosty sposób zaimplementować sieci neuronowe.

Warto również zauważyć, że gdybym od początku planował wykorzystać sieci neuronowe do szacowania wagi położonego na stoliku obiektu to praktycznie cały rozdział 5, traktujący o badaniu właściwości i zależności materiału Velostat, miałby dużo mniejsze znaczenie. Do trenowania sieci nie była mi potrzebna żadna ze znajdujących się tam informacji. Potrzebowałem znać tylko stan faktyczny obiektu (poprawne opisanie parametrów obiektu) oraz jego reprezentację na stoliku. Pozwoliłoby to podejść do problemu

jak do czarnej skrzynki, której sposobu działania nie znamy. Takie podejście jednak nie ujawniłoby faktycznego zachowania materiału i jego dynamicznych właściwości.

Wykorzystanie sieci neuronowych w identyfikacji i rozpoznaniu położonego na stoliku obiektu sprawdziło się bardzo dobrze, mimo iż nie zawsze zwracało zadowalające wyniki. Wykorzystanie sieci neuronowych w projekcie pozwala każdej osobie, bez głębokiej znajomości tematu, zaimplementować rozwiązanie dające zadowalające wyniki. Co prawda rozwiązania matematyczne i metodyczne są bardziej uniwersalne i dużo mniej obciążające dla systemu, ale wymagają wiedzy i czasu do wykonania ich implementacji. Sama implementacja sieci neuronowych jest dość prosta, tym bardziej, że w bibliotekach istnieje wiele gotowych do użycia sieci. Prawdopodobnie te właśnie powody są przyczyną takiego rozkwitu technologii opartych na tym właśnie fenomenie na rynku konsumenckim.

9.4. Wnioski dotyczące potencjału na rynku konsumenckim

Czujniki taktyльne na rynku konsumenckim mają większe zastosowanie niż to by się mogło wydawać. Wynika to z faktu, że rozwiązania te nie są widoczne na co dzień, są one wykorzystywane głównie w specjalistycznych i nietypowych zastosowaniach. Przeglądając rozwiązania dostępne na rynku widać, że na tego typu czujniki jest zastosowanie, ale warto oferować klientowi produkt przeznaczony do konkretnego wykorzystania.

Czy praca, którą tutaj opisałem może z powodzeniem zostać wykorzystana na rynku konsumenckim? Oczywiście, ale, jak napisałem wcześniej, wymagałoby to zaprezentowania faktycznego i praktycznego jego wykorzystania. Dodatkowo bardzo potrzebna byłaby dobra promocja swoich możliwości, opracowanie wygodnych narzędzi do obsługi czujnika oraz dobre finansowania na rozwój.

9.5. Wnioski dotyczące możliwości wykorzystania czujników taktylnych w robotyce

Wykorzystanie czujników taktylnych w robotyce w tym momencie opiera się głównie na wykorzystaniu ich w chwytakach, do estymacji miejsca kontaktu oraz do rozpoznawania obiektów kolizji. Ta praca poszerza wiedzę i możliwości wykorzystania czujników taktylnych w szczególności jeśli chodzi o rozpoznawanie obiektów. Pokazuje, że pomimo niskiej rozdzielczości jest to w pewnym stopniu możliwe.

Zastosowanie, w którym wykorzystałem inteligentny stolik potwierdziło również, że stolik ułatwia sposób współpracy człowieka z robotem. Zgodnie z założeniami, robot jest w tym momencie bardziej samodzielny i nie potrzebuje już jednoznacznego potwierdzenia od człowieka co do zmiany stanu położonego przedmiotu.

Obecnie w restauracjach i sklepach można spotkać roboty Bella służące zazwyczaj do przewożenia przedmiotów. Roboty te do potwierdzenia otrzymania przedmiotu potrzebują polecenia głosowego albo interakcji za pomocą tabletu. Dzięki wykorzystaniu inteligentnego stolika potwierdzanie to może być bardziej zautomatyzowane i mniej uciążliwe [86].

9.6. Możliwości rozwoju projektu inteligentnego stolika

Wykonany stolik w praktyce posiadał pewne wady, które wynikały z jego konstrukcji. Jedną z opcji rozwoju projektu jest więc wymiana stolika na taki, który jest wykonany w lepszej technologii, zapewniającej bardziej stabilne pomiary z czujników. Trzeba pamiętać jednak, że zmiana stolika jest dużą zmianą, która może wymusić ponowne wykonanie wielu prac opisanych w tym dokumencie. Zmianę taką trzeba wcześniej dobrze przemyśleć i oszacować czy potencjalne korzyści ze zmiany stolika są warte podjęcia prac.

Dodatkową rzeczą, którą można wykonać w tym projekcie, jest wykonanie interfejsu graficznego do obrazowania oraz obserwacji danych zbieranych i przetwarzanych przez inteligentny stolik. Będzie to w większym stopniu wypełniało zadania stawiane przez UC1. Dla użytkownika jest to wygodniejsza opcja nadzorowania pracy stolika niż logi tekstowe. Wspomniane zmiany są bardziej kosmetyczne niż fundamentalne. Są one do rozważenia dopiero w momencie wypuszczenia robota na rynek, gdzie dopracowany interfejs graficzny jest w stanie pozytywnie wpływać na percepcję rozwiązania z perspektywy potencjalnego klienta.

Jednym z przyszłościowych podejść jest również wykorzystanie stolika w bardziej skomplikowanych scenariuszach i zadaniach. W przypadku tej pracy przeprowadzone zostały tylko stosunkowo proste testy w niezbyt skomplikowanych scenariuszach. Przykładowymi dużymi scenariuszami może być wykorzystanie Rico w domu opieki bądź szpitalu, gdzie robot będzie miał kontakt z dużą liczbą ludzi. Bardzo wskazane jest, aby nowoprojektowane wówczas podsystemy robota brały pod uwagę możliwość wykorzystania i integracji inteligentnego stolika.

Alternatywnym podejściem do rozwinięcia projektu inteligentnego stolika jest wykorzystanie go na innych robotach lub nie tylko na robotach. Wykorzystanie robotów innego typu niż Rico daje całe nowe spektrum możliwości wykorzystania technologii czujników taktylnych. Niekoniecznie czujnik musi być wykorzystywany jako stolik, może zostać wykorzystany na przykład w chwytach bądź systemach zabezpieczeń.

Bibliografia

- [1] I. Asimov, "Runaround", w *I, Robot*, Doubleday, 1950.
- [2] T. Indeka, "Sztuczna skóra zastosowana w robotach asystujących", prac. inż. Wydział Elektroniki i Technik Informacyjnych, Politechnika Warszawska, 2021.
- [3] *Film prezentujący działanie wykonanego prototypu sztucznej skóry w ramach pracy inżynierskiej podczas testów na robocie*, Dostęp zdalny (27.09.2023): <https://vimeo.com/manage/videos/562491170>, 2021.
- [4] I. Orha i S. Oniga, "Assistance and telepresence robots: a solution for elderly people", *Carpathian Journal of Electronic and Computer Engineering ISSN 1844 - 9689*, t. 5, s. 87–90, list. 2012.
- [5] D. Feil-Seifer i M. Mataric, "Defining socially assistive robotics", w *9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005.*, 2005, s. 465–468.
- [6] J. Dave, "Assistive Robotics", Dostęp zdalny (7.01.2024): <https://web.stanford.edu/class/engr110/2014/05a-Jaffe2.pdf>, 2014.
- [7] G. G. Kost, *Podstawy Budowy Robotów*. Wydawnictwo Politechniki Śląskiej, 1996.
- [8] M. V. Ravinder S. Dahiya, *Robotic Tactile Sensing: Technologies and System*. Springer Dordrecht, 2012.
- [9] R. S. Dahiya, G. Metta, M. Valle i G. Sandini, "Tactile Sensing—From Humans to Humanoids", *IEEE Transactions on Robotics*, t. 26, nr. 1, s. 1–20, 2010.
- [10] Strona internetowa IEEE Xplore, Dostęp zdalny (18.10.2023): <https://ieeexplore.ieee.org/Xplore/home.jsp>, 2023.
- [11] U. Martinez-Hernandez i T. Assaf, "Soft Tactile Sensor With Multimodal Data Processing for Texture Recognition", *IEEE Sensors Letters*, t. 7, nr. 8, s. 1–4, 2023.
- [12] R. Nakashima i H. Takahashi, "Multi-Axial Tactile Sensor Using Standing Lig Cantilevers on Polyimide Film", w *2022 IEEE 35th International Conference on Micro Electro Mechanical Systems Conference (MEMS)*, 2022, s. 688–690.
- [13] J. Wagner, H. Winger, C. Cherif i F. Ellinger, "Smart Glove With Fully Integrated Textile Sensors and Wireless Sensor Frontend for the Tactile Internet", *IEEE Sensors Letters*, t. 7, nr. 2, s. 1–4, 2023.
- [14] M. A. Hari, S. C. Karumuthil, S. Varghese i L. Rajan, "Performance Enhancement of Flexible and Self-Powered PVDF-ZnO Based Tactile Sensors", *IEEE Sensors Journal*, t. 22, nr. 10, s. 9336–9343, 2022.
- [15] F. Durini, G. Terruso, J. D'Abbraccio i in., "Soft large area FBG tactile sensors for exteroception and proprioception in a collaborative robotic manipulator", w *2021 Smart Systems Integration (SSI)*, 2021, s. 1–4.
- [16] S. A. Hassan i C. M. Oddo, "Tactile sensors for Material recognition in Social and Collaborative Robots: A brief review", w *2022 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*, 2022, s. 1–5.

9. Bibliografia

- [17] B. M. Rocha Lima, T. E. Alves de Oliveira i V. P. da Fonseca, "Classification of Textures using a Tactile-Enabled Finger in Dynamic Exploration Tasks", w *2021 IEEE Sensors*, 2021, s. 1–4.
- [18] W. Luo, V. Sharma i D. J. Young, "A Comparative Evaluation of a Wearable MEMS Tactile Sensors Array and a Photoplethysmography Sensor for Atrial Fibrillation Detection Under Sitting Condition", w *2021 21st International Conference on Solid-State Sensors, Actuators and Microsystems (Transducers)*, 2021, s. 1436–1439.
- [19] R. Kaneta, T. Hasegawa, T. Abe i M. Sohgawa, "Sensitivity Enhancement of MEMS Tactile Sensor by Redesign of Microcantilever and Strain Gauge", w *2021 IEEE Sensors*, 2021, s. 1–4.
- [20] C. Hou, K. Wang, L. Lou, S. Zhang, H. Liu i L. Sun, "Wide Range Bridge Type 3D Tactile Sensor with Variable Sensitivity Through Replaceable Elastic Layer Attaching on PDMS Cap", w *2021 21st International Conference on Solid-State Sensors, Actuators and Microsystems (Transducers)*, 2021, s. 214–217.
- [21] K. He, X. Shi, D. Yu i X. Guo, "Vision-Based High-Resolution Tactile Sensor By Using Visual Light Ring", *IEEE Sensors Letters*, t. 6, nr. 4, s. 1–4, 2022.
- [22] O. C. Kara, N. Ikoma i F. Alambeigi, "HySenSe: A Hyper-Sensitive and High-Fidelity Vision-Based Tactile Sensor", w *2022 IEEE Sensors*, 2022, s. 1–4. DOI: 10.1109/SENSORS52175.2022.9967133.
- [23] F. R. Hogan, M. Jenkin, S. Rezaei-Shoshtari, Y. Girdhar, D. Meger i G. Dudek, "Seeing Through your Skin: Recognizing Objects with a Novel Visuotactile Sensor", w *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2021, s. 1217–1226.
- [24] S. Pohtongkam i J. Srinonchat, "Object Recognition Using Glove Tactile Sensor", w *2022 International Electrical Engineering Congress (iEECON)*, 2022, s. 1–4.
- [25] S. Suprapto, A. Setiawan, H. Zakaria, W. Adiprawita i B. Supartono, "Low-Cost Pressure Sensor Matrix Using Velostat", w *2017 5th International Conference on Instrumentation, Communications, Information Technology, and Biomedical Engineering (ICICI-BME)*, 2017, s. 137–140.
- [26] L. Shu, T. Hua, Y. Wang, Q. Li, D. D. F. Feng i X. Tao, "In-Shoe Plantar Pressure Measurement and Analysis System Based on Fabric Pressure Sensing Array", *IEEE transactions on information technology in biomedicine : a publication of the IEEE Engineering in Medicine and Biology Society*, t. 14, s. 767–75, maj 2010.
- [27] T. Z. Jia, Z. Ahmad, M. A. A. Saidi i M. Z. Baharom, "Development and effect of the body balance to the pressure sensor monitoring", w *Innovative Manufacturing, Mechatronics & Materials Forum 2022 (iM3F 2022)*, t. 2022, 2022, s. 115–119.
- [28] M. W. Wani, Y. P. Gururaj, V. P, S. A. Karre, R. Reddy i S. Azeemuddin, "VelGmat: Low Cost Gait Mat For Stance Phase Calculation", w *2022 IEEE Sensors*, 2022, s. 1–4.
- [29] L. Yuan, H. Qu i J. Li, "Velostat Sensor Array for Object Recognition", *IEEE Sensors Journal*, t. PP, s. 1–1, grud. 2021.

- [30] M. D. Gupta, L. Lakshmanan, A. Fatema i A. M. Hussain, "Flexible Writing Pad based on a Piezoresistive Thin Film Sensor Matrix", w *2023 IEEE Applied Sensing Conference (APSCON)*, 2023, s. 1–3.
- [31] C. Chen, H. Shi, I. González-Afanador, N. Sepúlveda i X. Tan, "Rapid Fabrication of Flexible Pressure Sensor Array", *IEEE Sensors Letters*, t. 7, nr. 6, s. 1–4, 2023.
- [32] M. Kanimozhi, I. Yamuna, B. Srimathi i R. S. Kumaran, "Development of Health Monitoring Vest using Velostat", w *2021 International Conference on System, Computation, Automation and Networking (ICSCAN)*, 2021, s. 1–5.
- [33] I. Kuriakose, S. Chauhan, A. Fatema i A. M. Hussain, "Wearable Pressure Sensor Suit for Real-Time Detection of Incorrect Exercise Techniques", w *2022 IEEE Sensors*, 2022, s. 1–4.
- [34] *Strona internetowa opisująca sposoby badania jakości spawów*, Dostęp zdalny (20.12.2023): <https://landreko.eu/kielce-ladreko/>, 2016.
- [35] *Strona internetowa firmy Novel*, Dostęp zdalny (8.10.2023): <https://novel.de/>, 2023.
- [36] L. Liu i X. Zhang, "A Focused Review on the Flexible Wearable Sensors for Sports: From Kinematics to Physiologies", *Micromachines*, t. 13, s. 1356, sierp. 2022.
- [37] J. Erp, I. Saturday i C. Jansen, "Application of tactile displays in sports: where to, how and when to move", *Proceedings of the Conference on EuroHaptics, Paris, France*, sty. 2006.
- [38] *Strona internetowa firmy PPS*, Dostęp zdalny (20.12.2023): <https://pressureprofile.com/>, 2023.
- [39] *Strona internetowa firmy Tekscan*, Dostęp zdalny (20.12.2023): <https://www.tekscan.com/>, 2023.
- [40] M. Dittmann, S. Arpagaus, V. Hungerbühler, M. Weishaupt i S. Latif, "Feel the Force"—Prevalence of Subjectively Assessed Saddle Fit Problems in Swiss Riding Horses and Their Association With Saddle Pressure Measurements and Back Pain", *Journal of Equine Veterinary Science*, t. 99, s. 103 388, sty. 2021.
- [41] A. Stotz, K. Hollander, C. Heidt, S. Sehner i A. Zech, "Clinical Assessment of the Medial Longitudinal Arch in Children: Rater Agreement and Relationship to Objective Foot Arch Measurements", *SN Comprehensive Clinical Medicine*, t. 2, grud. 2020.
- [42] D. A. Wichelhaus, C. Harms, J. Neumann i in., "Parameters influencing hand grip strength measured with the manugraphy system", *BMC Musculoskeletal Disorders*, t. 19, lut. 2018.
- [43] *Strona internetowa robota Tiago*, Dostęp zdalny (27.09.2023): <https://pal-robotics.com/robots/tiago/>, 2016.
- [44] *Strona internetowa poświęcona używaniu robota Tiago w systemie ROS*, Dostęp zdalny (27.09.2023): <http://wiki.ros.org/Robots/TIAGo>, 2016.
- [45] J. Sikora, "Komunikacja głosowa z robotem społecznym Rico", prac. mag., Wydział Elektroniki i Technik Informacyjnych, Politechnika Warszawska, 2021.

9. Bibliografia

- [46] S. Stankevich, "System konwersacyjny dla robota usługowego", prac. inż. Wydział Elektroniki i Technik Informacyjnych, Politechnika Warszawska, 2023.
- [47] T. Winiarski, W. Dudek, M. Stefańczyk, Ł. Zieliński, D. Giełdowski i D. Seredyński, "An intent-based approach for creating assistive robots' control systems", Dostęp zdalny (27.09.2023): <https://arxiv.org/abs/2005.12106>, 2020.
- [48] A. Micor, "Wykrywanie hazardów temperaturowych z wykorzystaniem robota społecznego TIAGo", prac. inż. Wydział Elektroniki i Technik Informacyjnych, Politechnika Warszawska, 2023.
- [49] K. Bugała, "Teleoperacja robotem asystującym", prac. inż. Wydział Elektroniki i Technik Informacyjnych, Politechnika Warszawska, 2023.
- [50] W. Dudek i T. Winiarski, "Scheduling of a Robot's Tasks With the TaskER Framework", *IEEE Access*, t. PP, s. 1–1, sierp. 2020.
- [51] *Strona internetowa opisująca znaczenie nazywania robota dla użytkownika*, Dostęp zdalny (20.12.2023): <https://www.socinvestigation.com/how-are-robots-named-a-guide-to-naming-robots-androids/>, 2023.
- [52] *Strona internetowa zawierająca informację o zakupionej folii Velostat*, Dostęp zdalny (29.09.2023): <https://www.adafruit.com/product/1361>, 2013.
- [53] A. Dzedzickis, E. Sutinys, V. Bučinskas i in., "Polyethylene-Carbon Composite (Velostat®) Based Tactile Sensor", *Polymers*, t. 12, s. 2905, grud. 2020.
- [54] *Film Marco Repsa o projekcie sztucznej skóry wykorzystującej Velostat*, Dostęp zdalny (21.11.2023): https://www.youtube.com/watch?v=4JBSHqUcaG4&t=401s&ab_channel=MarcoReps, 2017.
- [55] *Strona internetowa zawierająca informację o mikrokontrolerach z rodziny STM32*, Dostęp zdalny (28.01.2024): <https://www.st.com/en/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus.html>, 2021.
- [56] *Strona internetowa mikrokontrolera STM32F103RB*, Dostęp zdalny (28.01.2024): <https://www.st.com/en/microcontrollers-microprocessors/stm32f103rb.html>, 2007.
- [57] *Strona internetowa języka Python*, Dostęp zdalny (27.11.2023): <https://www.python.org/>, 2023.
- [58] *Strona internetowa narzędzia TensorFlow*, Dostęp zdalny (27.11.2023): <https://www.tensorflow.org/>, 2023.
- [59] *Strona internetowa biblioteki Keras*, Dostęp zdalny (27.12.2023): <https://keras.io/>, 2023.
- [60] *Strona internetowa systemu ROS™*, Dostęp zdalny (28.01.2024): <http://ros.org/>, 2018.
- [61] *Strona internetowa symulatora Gazebo*, Dostęp zdalny (28.01.2024): <http://gazebosim.org/>, 2019.
- [62] *Strona internetowa narzędzia RViz*, Dostęp zdalny (28.01.2024): <https://wiki.ros.org/rviz>, 2018.

- [63] *Strona internetowa języka SysML*, Dostęp zdalny (17.01.2024): <https://sysml.org/>, 2024.
- [64] *Strona internetowa języka UML*, Dostęp zdalny (17.01.2024): <https://www.uml.org/>, 2024.
- [65] T. Winiarski, "MeROS: SysML-Based Metamodel for ROS-Based Systems", *IEEE Access*, t. 11, s. 82 802–82 815, 2023, ISSN: 2169-3536. adr.: <http://dx.doi.org/10.1109/ACCESS.2023.3301727>.
- [66] *Strona internetowa programu Visual Paradigm*, Dostęp zdalny (17.01.2024): <https://www.visual-paradigm.com/>, 2024.
- [67] *Przepis na makówki*, Dostęp zdalny (20.12.2023): <https://domowe-wypieki.pl/przepisy/desery/602-makowki>, 2013.
- [68] T. Winiarski, M. Węgierek i M. Bogusz, "Rozbudowa stanowisk badawczo-dydaktycznych robotów mobilnych i manipulacyjnych", KNR BIONIK, Wydział Elektroniki i Technik Informacyjnych, Politechnika Warszawska, spraw. tech., 2019.
- [69] *Strona internetowa układu scalonego ULN2803*, Dostęp zdalny (13.01.2021): <https://www.ti.com/product/ULN2803A>, 2006.
- [70] *Dokumentacja tranzystora DMN3404L*, Dostęp zdalny (28.09.2023): <https://www.diodes.com/assets/Datasheets/DMN3404L.pdf>, 2006.
- [71] R. Barba, Á. De Madrid i J. G. Boticario, "Development of an Inexpensive Sensor Network for Recognition of Sitting Posture", *International Journal of Distributed Sensor Networks*, t. 2015, s. 1–13, sierp. 2015.
- [72] *Strona internetowa opisująca funkcję fmincon Matlaba*, Dostęp zdalny (2.10.2023): <https://www.mathworks.com/help/optim/ug/fmincon.html>, 2020.
- [73] A.-R. A. Laaraibi, G. Jodin, D. Hoareau, N. Bideau i F. Razan, "Flexible Dynamic Pressure Sensor for Insole Based on Inverse Viscoelastic Model", *IEEE Sensors Journal*, t. 23, nr. 7, s. 7634–7643, 2023.
- [74] A. Fatema, S. Poondla, R. B. Mishra i A. M. Hussain, "A Low-Cost Pressure Sensor Matrix for Activity Monitoring in Stroke Patients Using Artificial Intelligence", *IEEE Sensors Journal*, t. 21, nr. 7, s. 9546–9552, 2021.
- [75] Z. Del Prete, L. Monteleone i R. Steindler, "A novel pressure array sensor based on contact resistance variation: Metrological properties", *Review of Scientific Instruments*, t. 72, s. 1548–1553, lut. 2001.
- [76] X. Lin i B.-C. Seet, "A Linear Wide-Range Textile Pressure Sensor Integrally Embedded in Regular Fabric", *IEEE Sensors Journal*, t. 15, nr. 10, s. 5384–5385, 2015.
- [77] L. Lakshmanan, M. D. Gupta, A. Fatema i A. M. Hussain, "Characterisation and Quantification of Crosstalk on a Velostat-Based Flexible Pressure Sensing Matrix", w *2023 IEEE International Conference on Flexible and Printable Sensors and Systems (FLEPS)*, 2023, s. 1–4.
- [78] J. Nagi, F. Ducatelle, G. A. Di Caro i in., "Max-pooling convolutional neural networks for vision-based hand gesture recognition", w *2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, 2011, s. 342–347.

9. Bibliografia

- [79] I. Goodfellow, Y. Bengio i A. Courville, *Deep Learning*. MIT Press, 2016, <http://www.deeplearningbook.org>.
- [80] T. Cody, E. Lanus, D. D. Doyle i L. Freeman, "Systematic Training and Testing for Machine Learning Using Combinatorial Interaction Testing", w *2022 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 2022, s. 102–109.
- [81] A. Shepard i N. Naheed, "Application of Data Transformation Techniques and Train-Test split", w *2021 International Conference on Computational Science and Computational Intelligence (CSCI)*, 2021, s. 58–63.
- [82] T. Takase, S. Oyama i M. Kurihara, "Evaluation of Stratified Validation in Neural Network Training with Imbalanced Data", w *2019 IEEE International Conference on Big Data and Smart Computing (BigComp)*, 2019, s. 1–4.
- [83] *Blog internetowy opisujący problem podziału danych na zbiory*, Dostęp zdalny (30.12.2023): <https://www.v7labs.com/blog/train-validation-test-set>, 2021.
- [84] H. Liu, J. Greco, X. Song, J. Bimbo, L. Seneviratne i K. Althoefer, "Tactile image based contact shape recognition using neural network", w *2012 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, 2012, s. 138–143.
- [85] *Strona internetowa dotycząca dystrybucji melodic systemu ROS*, Dostęp zdalny (27.11.2023): <https://wiki.ros.org/melodic>, 2018.
- [86] *Strona internetowa robota Bella*, Dostęp zdalny (7.01.2024): <https://www.pudurobotics.com/products/bellabot>, 2023.

Spis rysunków

1.1	Prototyp sztucznej skóry wykonany na potrzeby pracy inżynierskiej [2]	7
1.2	Diagram przypadków użycia inteligentnego stolika i danych przez niego zbieranych	8
1.3	Plan rozszerzenia robota Rico o inteligentny stolik	10
1.4	Podział robotów na podklasy zaproponowany przez Ioana Orha [4]	11
2.1	Przykładowe podejście do rozpoznawania faktury materiału [11]	15
2.2	Przykładowe podejście do pomiaru siły tnącej [12]	15
2.3	Przykładowe podejście do wykrywania i rozpoznawania przedmiotów wspomaganej przez system wizyjny [23]	16
2.4	Przykładowe podejście do rozpoznawania przedmiotów chwyconych dlonią wykorzystując przygotowaną rękawicę [24]	16
2.5	Prototyp czujnika taktylnego wykonanego przez Yuana Liangqi [29]	17
2.6	Rozwiązań firmy Novel do badania rozkładu nacisku stopy podczas chodzenia [35]	19
3.1	Robot TIAGo [43]	20
3.2	Robot Rico - z modyfikacjami wykonanymi w laboratorium	22
4.1	Zastana wersja stolika na robocie	26
4.2	Zasadnicza struktura sztucznej skóry	27
4.3	Zaprojektowany i wykonany stolik pokryty sztuczną skórą	28
4.4	Schemat elektroniczny przedstawiający zasadę działania i odczytywania wartości rezystancji z pól czujnika taktylnego	30
4.5	Zaprojektowana i wykonana elektronika do kontroli stolika	30
4.6	Diagram ramki wysyłanej do komputera	31
4.7	Diagram automatu skońzonego sterownika sztucznej skóry	32
4.8	Wykonana przeze mnie wersja stolika na robocie	33
5.1	Stanowisko badawcze do pomiarów zależności pomiędzy naciskiem a rezystancją	35
5.2	Rezystancja pola czujnika dla różnych pól powierzchni badanego czujnika (średnia 4 z 6)	37
5.3	Prezentacja problemu kompensacji - faktyczne odczyty czujnika taktylnego przy braku nacisku	42
5.4	Stanowisko badawcze do badania faktycznej wagi przedmiotów na stoliku . .	44
5.5	Podgląd w czasie rzeczywistym na wartości wagi obliczane z wykorzystaniem kolejnych metod kompensacji szumów	44
5.6	Porównanie wszystkich pomiarów uzyskanych dla wszystkich metod kompensacji	45
5.7	Porównanie pomiędzy ostatnimi dwoma wybranymi metodami kompensacji	46

5.8 Zależność pomiędzy przedmiotami o różnych średnicach podstawy, przy wykorzystaniu wybranej metody kompensacji	47
5.9 Aproksymacja linią prostą wykonana na wybranej metodzie kompensacji w porównaniu do realnej wagi przedmiotu	48
6.1 Podział działającego systemu robota Rico na koncepcyjne intrasystemy	50
6.2 Diagram sekwencji dla przykładowego scenariusza przywożenia herbaty dla osoby starszej	51
6.3 Diagram sekwencji dla przykładowego scenariusza odwożenia pustego naczynia	52
7.1 Struktura hierarchiczna kodów źródłowych inteligentnego stolika	55
7.2 Diagram definicji bloków dla intrasystemu inteligentnego stolika	57
7.3 Diagram bloków wewnętrznych realizacji węzła inteligentnego stolika wewnątrz systemu Rico	57
7.4 Diagram bloków wewnętrznych obrazujący komunikację pomiędzy węzłem inteligentnego stolika a czujnikiem taktylnym	58
7.5 Diagram bloków wewnętrznych obrazujący komunikację, w postaci tematów ROS, pomiędzy węzłem inteligentnego stolika a pozostałą częścią systemu .	59
7.6 Diagram aktywności obrazujący sposób działania węzła inteligentnego stolika	60
7.7 Diagram sekwencji kalibracji pomiarów z czujnika taktylnego	61
7.8 Program do podglądu nacisku wywieranego przez obiekt na stolik	62
7.9 Interpretacja oraz wstępna obróbka obrazu (kubka) przez węzeł inteligentnego stolika	63
7.10 Architektura zaprojektowanej sieci CNN do klasyfikacji położonych obiektów	67
7.11 Architektura zaprojektowanej sieci CNN do estymowania wagi położonych obiektów	71
7.12 Histogram wyników uczenia sieci do rozpoznawania wagi przedmiotu	73
7.13 Diagram definicji bloków dla intrasystemu zarządzającego wykonaniem scenariuszy	75
7.14 Diagram bloków wewnętrznych dla wykorzystywanego w scenariuszach węzła, który wykorzystywał dane z inteligentnego stolika	76
7.15 Diagram bloków wewnętrznych obrazujący komunikację, pomiędzy węzłem przykładowego użycia inteligentnego stolika a węzłem inteligentnego stolika .	76
7.16 Diagram bloków wewnętrznych obrazujący komunikację, pomiędzy węzłem przykładowego użycia inteligentnego stolika a węzłem odpowiadającym za poruszanie	77
7.17 Diagram bloków wewnętrznych obrazujący komunikację, pomiędzy węzłem przykładowego użycia inteligentnego stolika a węzłem odpowiadającym za komunikację głosową robota	77
7.18 Diagram akcji obrazujący sposób działania węzła wykonującego zaplanowany scenariusz	78

8.1 Laboratorium wykorzystane podczas wykonywania scenariuszy użycia inteligentnego stolika	80
8.2 Mapa laboratorium wykorzystane podczas wykonania scenariuszy	81
8.3 Diagram sekwencji scenariusza faktycznie wykonanego na robocie Rico	82

Spis tabel

5.1 Rezystancja pola czujnika dla różnych pól powierzchni badanego czujnika (średnia 4 z 6)	37
5.2 Wyniki aproksymacji pomiarów przy założeniu, że $x_0 = 0$	38
5.3 Wybrane punkty obciążenia Velostatu do porównania moich wyników z wynikami innych badaczy	39
5.4 Porównanie odpowiedzi rezystancyjnej materiału Velostat z wynikami innych badaczy	40
7.1 Macierz błędów dla wykrywania czy obiekt znajduje się wewnątrz stolika . . .	65
7.2 Tabela wyników klasyfikacji dla wybranego modelu sieci	69