

INSTYTUT PODSTAW KONSTRUKCJI MASZYN

Wydział Mechaniczny Technologiczny
Politechnika Śląska

PROJEKT INŻYNIERSKI

**Opracowanie intuicyjnego interfejsu
użytkownika do sterowaniem manipulatorem
robota mobilnego**

Tomasz ZWIERZYŃSKI

Kierunek studiów: Mechatronika

PROWADZĄCY PROJEKT

Dr hab. inż. Wojciech SKARKA, prof. Pol. Śl.

OPIEKUN

Dr inż. Piotr PRZYSTAŁKA

Gliwice, 2017

**Instytut Podstaw Konstrukcji Maszyn
Wydział Mechaniczny Technologiczny
POLITECHNIKA ŚLĄSKA**

OŚWIADCZENIE

Tomasz ZWIERZYŃSKI

Kierunek studiów: Mechatronika

Rok studiów: IV

Rok 2017

PROWADZĄCY PROJEKT:

Dr hab. inż. Wojciech SKARKA, prof. Pol. Śl.

OPIEKUN:

Dr inż. Piotr PRZYSTAŁKA

Oświadczam, że niniejszą pracę wykonałem/wykonałam* osobiście, pod kierunkiem PROWADZĄCEGO PROJEKT Dra hab. inż. Wojciecha SKARKI, prof. Pol. Śl. i OPIEKUNA Dra inż. Piotra PRZYSTAŁKI oraz, że przy jej realizacji nie naruszono praw osób trzecich, wynikających z przepisów prawa autorskiego.

Gliwice, dnia 201... roku

(własnoręczny podpis)

Tomasz ZWIERZYŃSKI

Wyrażam zgodę/nie wyrażam zgody* na udostępnienie mojego projektu inżynierskiego.

Gliwice, dnia 201... roku

(własnoręczny podpis)

Tomasz ZWIERZYŃSKI

.....
(poświadczenie wiarygodności podpisu przez Dziekanat)

*) niepotrzebne skreślić.

INSTYTUT MECHANIKI I INŻYNIERII OBliczeniowej
Wydział Mechaniczny Technologiczny
POLITECHNIKA ŚLĄSKA W GLIWICACH

PROJEKT INŻYNIERSKI

RMT6 – 2669/16/17

Student: Tomasz ZWIERZYŃSKI	Wydział: MT	Kierunek: MTA	Rodzaj studiów: s1	Semestr: VII	Specjalizacja:	Rok akademicki: 2016/17
--------------------------------	----------------	------------------	-----------------------	-----------------	----------------	----------------------------

Temat pracy:

**Opracowanie intuicyjnego interfejsu użytkownika do sterowania
manipulatorem robota mobilnego**

Zadania do wykonania:

1. Ustalenie specyfikacji projektu w porozumieniu z opiekunem projektu
2. Opracowanie środowiska symulacyjnego
3. Opracowanie algorytmu do sterowania ręcznego
4. Projekt i wykonanie interfejsu użytkownika
5. Opracowanie planu badań weryfikacyjnych
6. Przeprowadzenie badań weryfikacyjnych i analiza otrzymanych wyników

Prowadzący projekt/ Promotor: Dr hab. Inż. Wojciech Skarka, prof. Pol. Śl.	Podpis:	Dyrektor Instytutu: Prof. dr hab. inż. Wojciech Cholewa	Podpis:
Opiekun: Dr inż. Piotr Przystałka	Podpis:	Data wydania tematu: 07.10.2016	Planowany termin zakończenia: 09.01.2017

Spis treści

1 Wstęp	7
1.1 Definicja intuicji oraz intuicyjności	8
1.2 Cel projektu	8
1.3 Specyfikacja oraz ograniczenia	8
2 Przegląd narzędzi oraz zagadnień	10
2.1 3Dconnexion Space Navigator	10
2.2 V-REP	11
2.3 Język C++ i biblioteki Qt	11
2.4 Balsamiq Mockups	12
2.5 Kinematyka manipulatora	12
2.5.1 Zadanie proste kinematyki	13
2.5.2 Zadanie odwrotne kinematyki	14
2.6 Przegląd istniejących rozwiązań	15
3 Opracowanie środowiska symulacyjnego	18
3.1 Import elementów CAD	19
3.2 Model dynamiczny	20
3.3 Drzewo kinematyczne	21
3.4 Algorytmy sterowania	23
3.5 Skrypty sterujące	24
4 Projekt i implementacja interfejsu użytkownika	25
4.1 Koncepcje działania programu oraz interfejsu użytkownika	25
4.2 Wykonanie aplikacji	27
4.3 Instalacja oraz pierwsze uruchamianie	28
4.3.1 Opis okna głównego oraz instrukcja użytkowania	29
4.3.2 Okno wykresów	30
4.3.3 Architektura aplikacji	31
5 Sporządzenie planu i przeprowadzenie badań	33
5.1 Sceny testowe	33
5.1.1 Przenoszenie obiektów w widoku ogólnym	33
5.1.2 Przenoszenie obiektów z widokiem z kamery	33

5.1.3 Obsługa panelu operatorskiego	34
5.2 Testy weryfikacyjne	35
6 Wnioski i podsumowanie	37
7 Użyte oprogramowanie	39
8 Streszczenie	40

Rozdział 1

Wstęp

Współczesne roboty mobilne są w stanie wykonywać coraz bardziej skomplikowane zadania, zachowując przy tym większą autonomię. Oprócz coraz nowocześniejszych rozwiązań konstrukcyjnych, innym powodem tego stanu rzeczy jest rozwój oprogramowania oraz algorytmów sterowania [13]. To te elementy powodują, że roboty są coraz inteligentniejsze, potrafią tworzyć modele otoczenia w czasie rzeczywistym, podejmować kluczowe decyzje podczas wykonywania zleconego im zadania, a także komunikować się ze sobą i współpracować w grupie z innymi robotami. Zadanie jakie stoi przed programistami i projektantami układów i systemów sterowania polega na zaproponowaniu takich interfejsów operatorskich, które w zadowalający sposób pozwolą na nadzór i kontrolę nad podopiecznymi robotami, aby mogły one służyć zgodnie z ich przeznaczeniem.

Niniejsza praca jest fragmentem większego projektu grupowego mającego na celu zaprojektowanie oraz budowę robota mobilnego na odbywający się w Ameryce konkurs Łazików marsjańskich University Rover Challenge [6]. Do tych odbywających się co roku zawodów przystępują studenci z całego świata, mierząc się w różnych zadaniach polegających między innymi na interakcji z otoczeniem oraz manipulacji obiektami. Każda z drużyn posiada własną stację, z której steruje bezprzewodowo swoim łazikiem, mając dostęp jedynie do informacji, które otrzymują z urządzeń pokładowych. Zadania z grubsza dzielą się na zadania terenowe oraz zadania manipulacyjne, stąd każdy z robotów mobilnych posiada część jezdnią oraz osadzony na niej manipulator.

Projekt Łazika Marsjańskiego dzieli się na kilka podstawowych zadań projektowych. Są to:

- mechanika podwozia,
- mechanika manipulatora,
- sterowanie niskopoziomowe,
- sterowanie wysokopoziomowe oraz interfejs,
- układ zasilania.

Przedstawione w niniejszej pracy koncepcje są wstępem do realizacji zadania utworzenia kompletnego oraz intuicyjnego systemu sterowania pozwalającego na obsługę wszystkich instrumentów łazika, co pozwoli operatorowi na wygodne wykonywanie zadań konkursowych. Ze względu na obszerność zagadnienia sterowania, zakres pracy inżynierskiej ograniczony został tylko do zagadnienia intuicyjnego sterowania końcówką manipulatora oraz do zaprojektowania interfejsu obejmującego tylko to zadanie, z pominięciem innych zagadnień.

1.1. Definicja intuicji oraz intuicyjności

Na podstawie Słownika Języka Polskiego PWN [16], definiującego *intuicję* jako przeczucie i zdolność przewidywania, wnioskować można, że intuicyjność jest to cecha danego obiektu, polegająca na łatwości w przewidywaniu jego działania. Zatem intuicyjny interfejs użytkownika to taki, który nie wymaga szczegółowego objaśnienia, w jaki sposób trzeba z niego skorzystać, by osiągnąć za jego pomocą pożądane cele. Intuicyjne sterowanie oznacza taki system sterowania, który nie wymaga skomplikowanych objaśnień działania, tudzież nie wymaga długotrwałych szkoleń, by przygotować użytkownika do korzystania z danego systemu.

W odniesieniu do aplikacji komputerowych, intuicyjny interfejs graficzny oznacza taki układ elementów w oknie aplikacji, który minimalizuje czas potrzebny na nauczenie się przez użytkownika korzystania z tego interfejsu, co jest często do osiągnięcia poprzez stosowanie utartych i znanych wzorców projektowych znanych z innych aplikacji. Dotyczy to stosowania tzw. toolbarów, menu kontekstowych, skrótów klawiszowych itp. [19].

1.2. Cel projektu

Celem projektu jest opracowanie środowiska symulacyjnego czasu rzeczywistego, modelu robota mobilnego z manipulatorem w tym środowisku oraz wykonanie aplikacji sterującej tym modelem wraz z odpowiednim intuicyjnym interfejsem użytkownika oraz obsługą urządzeń wejścia do komunikacji z użytkownikiem. Następnie zaproponowanie oraz przeprowadzenie badań weryfikacyjnych, mających na celu sprawdzenie intuicyjności powstałego systemu sterowania w dobrze zdefiniowanych zadaniach.

1.3. Specyfikacja oraz ograniczenia

Ze względu na fakt, że niniejsza praca dotyczy jedynie zagadnień związanych z projektowaniem interfejsu oraz opracowywaniem symulacji, niektóre elementy wymagane do jej utworzenia, zostały wzięte z innych zadań Projektu Łazika Marsjańskiego. W momencie rozpoczęcia prac, nie było dostępnego modelu CAD mogącego służyć za podwozie w symulacji. Jednakże dostępny był wczesny model manipulatora, wraz z jego niektórymi parametrami, które zostaną opisane w dalszej części pracy. Poniżej przedstawione zostały założenia projektu podzielone na trzy grupy.

Założenia ogólne

Podstawowym założeniem jest wykorzystanie kinematyki odwrotnej w zagadnieniu sterowania manipulatorem oraz proste sterowanie kołami podwozia w celu przemieszczania całego układu. Oprócz tego założono także:

- wykorzystanie w sterowaniu myszy 3D,
- 64-bitowy system operacyjny Microsoft Windows.

Założenia dotyczące środowiska symulacyjnego

Założono wykorzystanie darmowego oprogramowania symulacyjnego V-REP, do przygotowywania środowiska symulacyjnego oraz sterowania obiekty symulacji w czasie rzeczywistym, ponadto założono także:

- zimportowanie i wykorzystanie modelu manipulatora dostępnego z Projektu Łazika Marsjańskiego,
- wykorzystanie przykładowego modelu podwozia jezdniego z dostępnych w oprogramowaniu V-REP,
- wykorzystanie przykładowego modelu chwytaka do manipulatora z dostępnych w oprogramowaniu V-REP,
- wykorzystanie dostępnych w oprogramowaniu V-REP narzędzi do obliczania kinematyki odwrotnej łańcucha kinematycznego manipulatora.

Założenia dotyczące aplikacji

Najważniejszym założeniem dotyczącym aplikacji sterującej jest intuicyjność, na skąda się kilka czynników:

- prosta instalacja,
- po uruchomieniu, aplikacja powinna samoczynnie uruchomić symulację oraz zainicjalizować obsługę myszy 3D,
- prosty, intuicyjny interfejs graficzny,
- przetwarzanie danych z myszy skutkujące intuicyjnym sterowaniem chwytakiem manipulatora w symulatorze,

Oprócz powyższego, założono także:

- język programowania C++,
- środowisko programistyczne Qt Creator,
- komunikacja z symulacją umożliwiająca wymianę informacji,
- możliwość dostosowywania prędkości kontrolera.

Rozdział 2

Przegląd narzędzi oraz zagadnień

Poniżej został przedstawiony krótki opis poszczególnych narzędzi, które zostały użyte oraz zagadnień, które musiały zostać poznane, aby możliwe było wykonanie pracy. Na koniec rozdziału przedstawiony jest krótki przegląd już istniejących systemów sterowania, mogących być uznane za intuicyjne.

2.1. 3Dconnexion Space Navigator

Mysz 3D firmy 3Dconnexion o nazwie Space Navigator [3], przedstawiona na Rys. 2.1, posiada sześć stopni swobody, co wykorzystywane jest w programach CAD do sterowania trójwymiarowym widokiem modelu, oraz dwa przyciski, do których przypisane są określone funkcje. W sterowaniu manipulatorem z zastosowaniem kinematyki odwrotnej (i przy założeniu co najmniej sześciu stopni swobody łańcucha kinematycznego manipulatora), mysz 3D może posłużyć użytkownikowi jako prosta metoda zadawania położenia oraz orientacji chwytaka lub innego urządzenia podłączonego do końcówki manipulatora. Pozwala to wprawnemu użytkownikowi wykonywać pewne sekwencje ruchów dużo szybciej, niż korzystając z konwencjonalnej klawiatury. Umieszczone na myszy 3D przyciski programowe służąć mogą do kontrolowania, bądź zmieniania stanu chwytaka lub innego urządzenia.

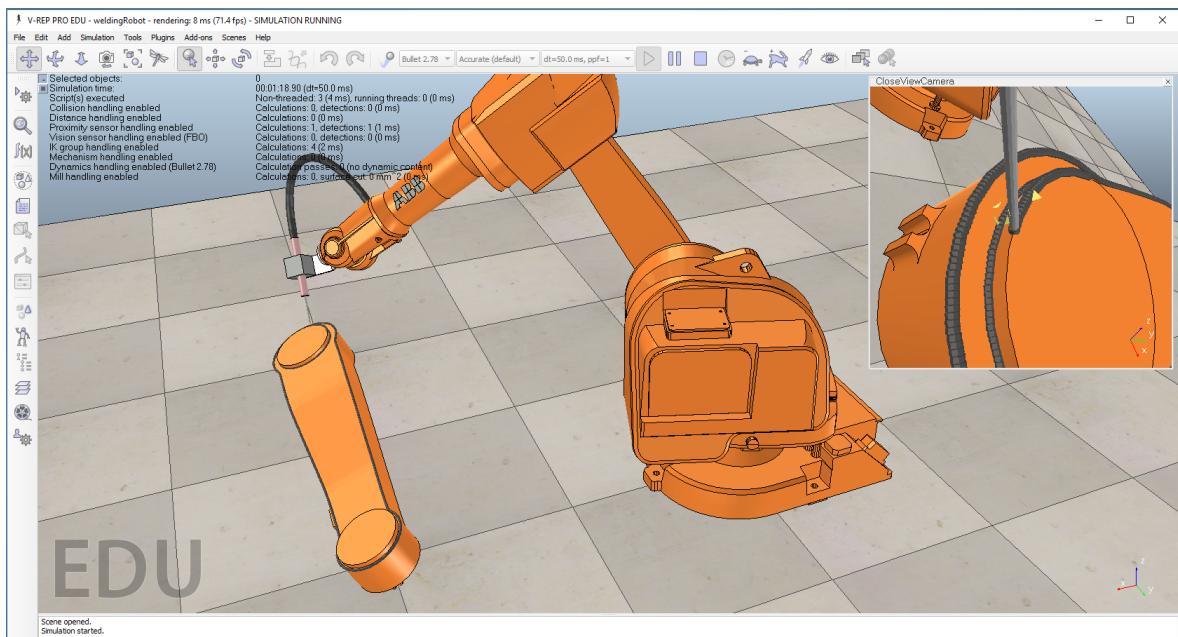


Rys. 2.1: Mysz 3D firmy 3Dconnexion

Producent myszy 3D udostępnia zestaw bibliotek programistycznych pozwalających na implementację obsługi sygnałów z myszy 3D w swojej aplikacji. Biblioteki są w postaci bibliotek statycznych, które należy załączyć podczas procesu konsolidacji (linkowania) przy budowaniu aplikacji.

2.2. V-REP

V-REP jest to platforma symulacyjna pozwalająca na przygotowywanie scen oraz obiektów symulacji, głównie robotów i układów automatyki, które następnie mogą zostawać poddawane symulacjom, także w czasie rzeczywistym. Główną zaletą tego oprogramowania, oprócz wieloplatformowości jest wysoka programowalność, co jest odzwierciedlone przez dużą ilość sposobów pisania kodu. Każdy obiekt poddawany symulacji, może być sterowany poprzez wewnętrzne skrypty, a także przez zewnętrzne interfejsy programistyczne aplikacji (API) dla wielu języków programowania. Powoduje to dużą łatwość w integraniu symulacji z innymi aplikacjami i przede wszystkim umożliwia szybkie prototypowanie m.in. systemów sterowania [7].



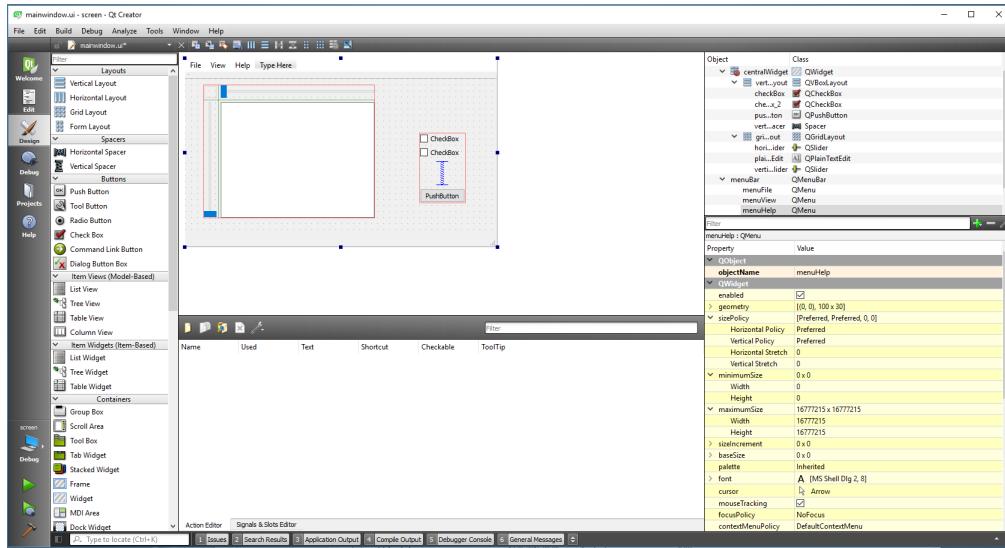
Rys. 2.2: Oprogramowania V-REP podczas przykładowej symulacji.

Podczas przeprowadzania symulacji można wykorzystywać jeden z czterech dostępnych silników fizycznych, wyświetlać wykresy parametrów dynamicznych dowolnego ciała w czasie oraz wyświetlać obraz z sensorów wizyjnych (jeżeli takie zostały umieszczone). Dodatkowo oprogramowanie umożliwia importowanie modeli CAD oraz narzędzia do tworzenia przybliżonych, prostszych brył służących do szybkich obliczeń dynamicznych [18]. Przykładowa scena symulacji spawania realizowanego przez robota przemysłowego przedstawiona została na Rys. 2.2.

2.3. Język C++ i biblioteki Qt

Z wielu języków programowania obecnie dostępnych, wybrany został język C++, ponieważ jest to jeden ze starszych języków programowania, co powoduje dużą ilość dostępnej pomocy w Internecie oraz literaturze. Ponadto biblioteka do obsługi komunikacji z symulatorem V-REP oraz przykłady jej zastosowania są dostępne właśnie w C++. Dużą zaletą tego języka programowania jest możliwość pisania kodu w różnych stylach programowania. Podejście obiektowe umożliwia pisanie obszernych aplikacji, w których zawarte są skomplikowane relacje między danymi (obiekty). Podejście struktu-

ralne, może zostać wykorzystane tam, gdzie trzeba w krótkim czasie utworzyć prosty program, który nie będzie miał niepotrzebnie skomplikowanego kodu źródłowego [11].



Rys. 2.3: Okno aplikacji QtCreator podczas tworzenia interfejsu graficznego.

Qt jest to zestaw wieloplatformowych bibliotek i narzędzi programistycznych dedykowanych m.in. dla języka C++, zawierających klasy do budowy graficznego interfejsu aplikacji komputerowych [2]. Dodatkowo istnieje darmowe zintegrowane środowisko programistyczne o nazwie QtCreator dedykowane dla Qt, pozwalające na graficzne tworzenie interfejsów. Qt wprowadza do języka C++ kilka dodatkowych słów kluczowych i makr, które redukują bolączki związane z czystym językiem C++ i zwiększają jego wysokopoziomowość (m.in. chodzi o skrupulatne zarządzanie pamięcią). Dodatkowo zaimplementowano system sygnałów i slotów, który pozwala na łatwą obsługę zdarzeń oraz bezproblemową komunikację między obiektami utworzonymi w pamięci. Wszystkie te dodatkowe elementy powodują konieczność skorzystania z narzędzia *qmake* przed klasyczną komilacją dostępnymi kompilatorami C++. Narzędzie te przygotowuje kod źródłowy, zamieniając graficznie zaprojektowaną aplikację oraz wszystkie dodatkowe słowa kluczowe na kod zrozumiały przez każdy kompilator C++ [17].

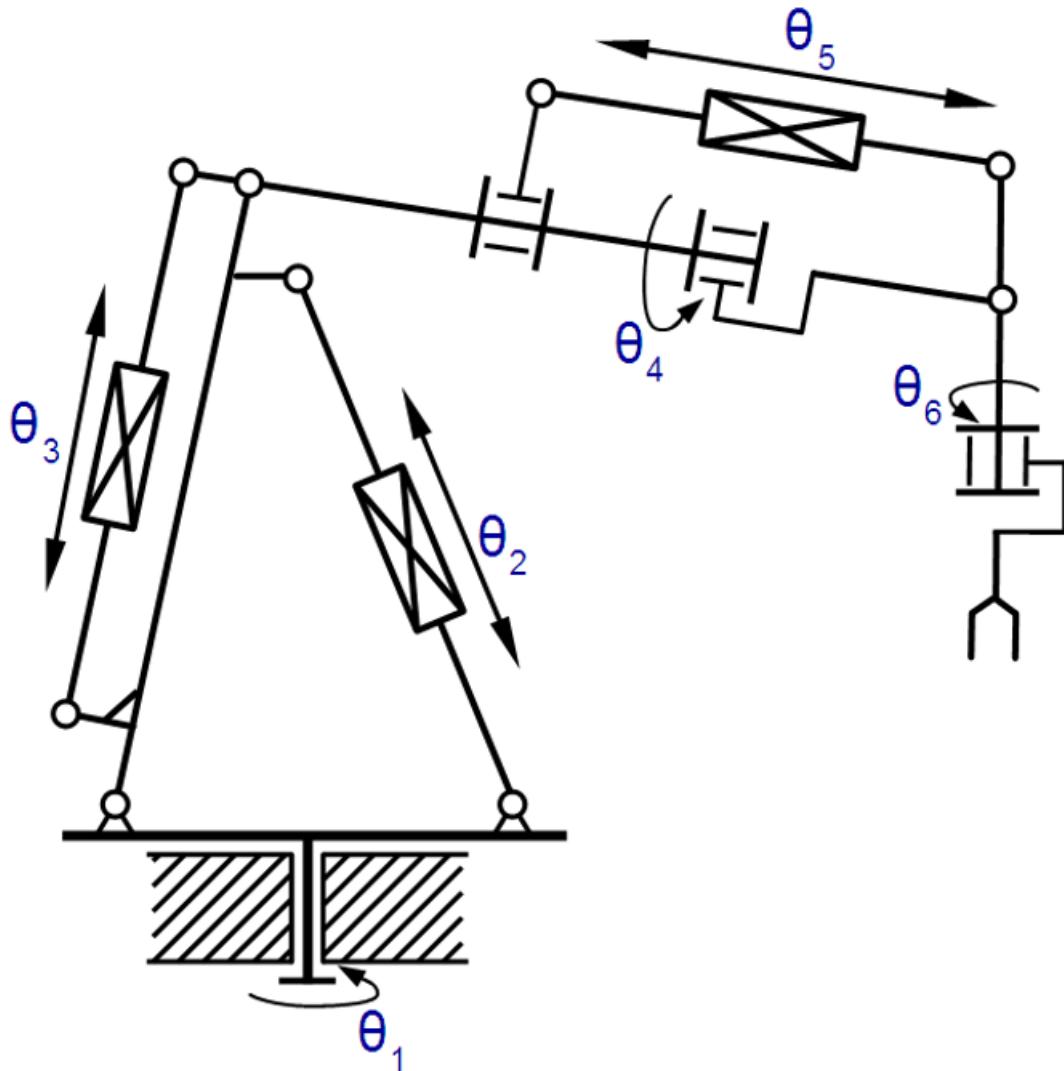
2.4. Balsamiq Mockups

Jest to prosta aplikacja służąca do tworzenia szkiców interfejsów graficznych. Pozwala na łatwą kontrolę nad różnymi koncepcjami. Została użyta przy projektowaniu intuicyjnego interfejsu aplikacji sterującej [1].

2.5. Kinematyka manipulatora

Manipulator, który będzie podlegać sterowaniu, tworzy łańcuch kinamtyczny o sześciu członach i bazie. Posiada sześć niezależnych par kinematycznych, a co za tym idzie, sześć stopni swobody. W łańcuchu znajdują się trzy aktuatorów obrotowych oraz trzy aktuatorów liniowych. Schemat kinematyczny tego manipulatora przedstawiony jest na Rys. 2.4. Kinematyka ta dla obliczeń może zostać uproszczona do

postaci przedstawionej na Rys. 2.5 poprzez zastosowanie odpowiedniej, jednoznacznej i geometrycznie wyrowadzonej funkcji konwertującej. Dla takiego łańcucha kinematycznego przypisuje się układy współrzędnych każdemu członowi, zgodnie z notacją Denavita-Hartenberga, a następnie opisuje się transformacje tych układów względem bazy otrzymując główną macierz transformacji manipulatora, zwaną macierzą Denavita-Hartenberga. Otrzymana macierz jest funkcją kątów przegubowych manipulatora[9].

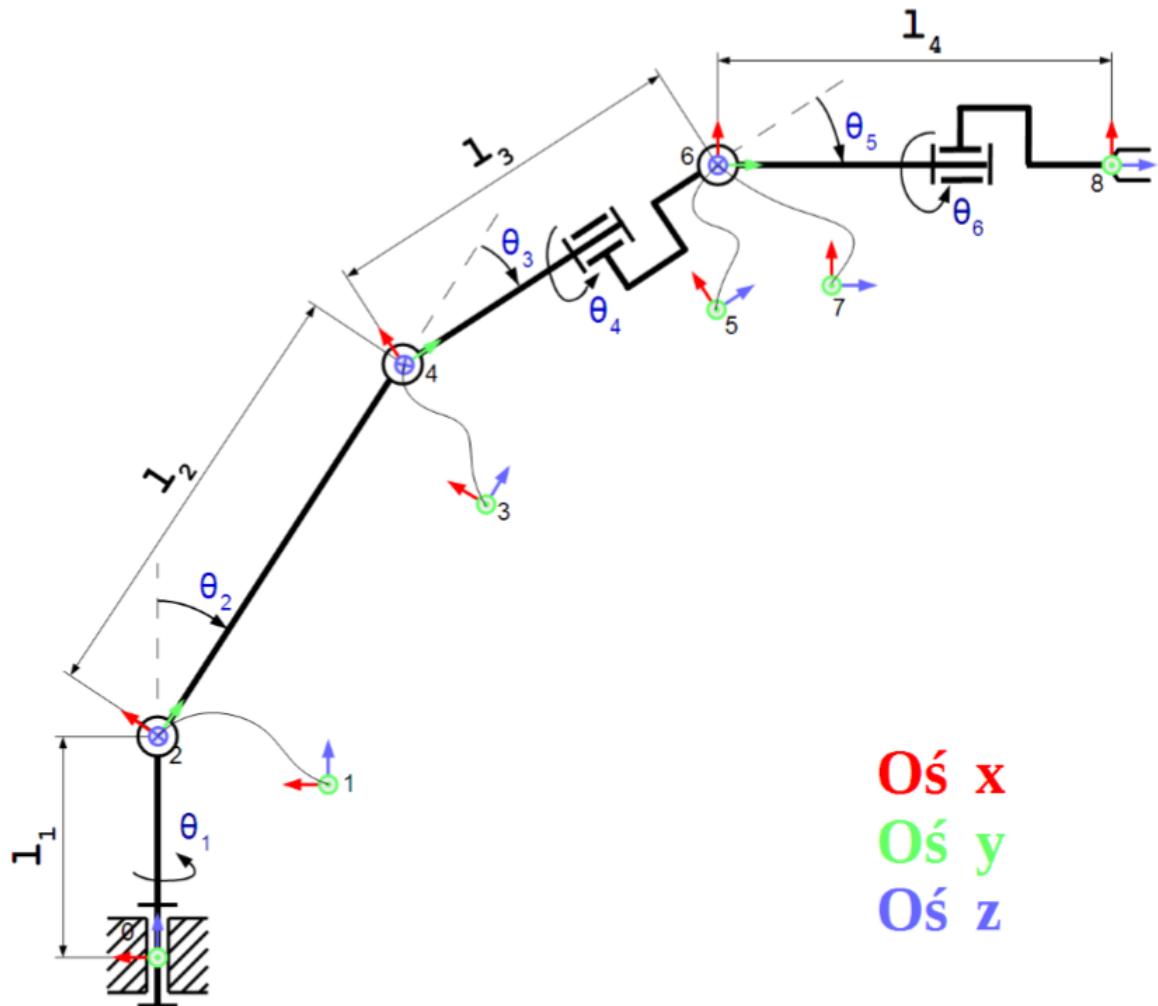


Rys. 2.4: Schemat kinematyczny manipulatora

Zagadnienie kinematyki manipulatora dzieli się na dwa zadania: zadanie proste kinematyki oraz zadanie odwrotne kinematyki.

2.5.1. Zadanie proste kinematyki

Zadanie proste kinematyki polega na znalezieniu w zdefiniowanym układzie współrzędnych, pozycji oraz orientacji wybranego punktu manipulatora, najczęściej punktu środka uchwytu chwytaka lub końcówki zamontowanego na manipulatorze narzędzia, na podstawie znajomości wartości kątów konfiguracyjnych. Jest to trywialne, jednoznaczne zadanie, którego rozwiązanie polega na obliczeniu



Rys. 2.5: Schemat kinematyczny przegubowy manipulatora

macierzy Denavita-Hartenberga.

2.5.2. Zadanie odwrotne kinematyki

Zadanie odwrotne kinematyki polega na znalezieniu kątów konfiguracyjnych manipulatora, na podstawie znajomości pozycji oraz orientacji wybranego punktu, najczęściej punktu środka uchwytu chwytaka lub końcówki zamontowanego na manipulatorze narzędzia. Choć z poznoru może nie wydawać się to trudne, w istocie jest to jedno z trudniejszych zagadnień robotyki. Problem stanowi fakt niejednoznaczności problemu (czasami dla jednego zestawu wejściowego można otrzymać kilka rozwiązań), a także skomplikowane wyrażenia trygonometryczne będące składowymi macierzy Denavita-Hartenberga, co skutecznie zmniejsza szansę na powodzenie metod analitycznych rozwiązania.

Prócz analitycznych metod rozwiązań, stosuje się numeryczne, przybliżone metody. Dwie z wielu takich metod zaimplementowano w module obliczeniowym oprogramowania V-REP, są to Pseudo-inverse oraz Damped Least Squares (DLS). Obie metody korzystają z Jacobianu manipulatora oraz pojęcia uogólnionej macierzy odwrotnej, lecz pojęcia te są zbyt obszerne by przedstawić je szerzej w tej pracy.



Rys. 2.6: Łazik *Centaur* oraz humanoid *Robonaut* w dwóch różnych wersjach. [8]

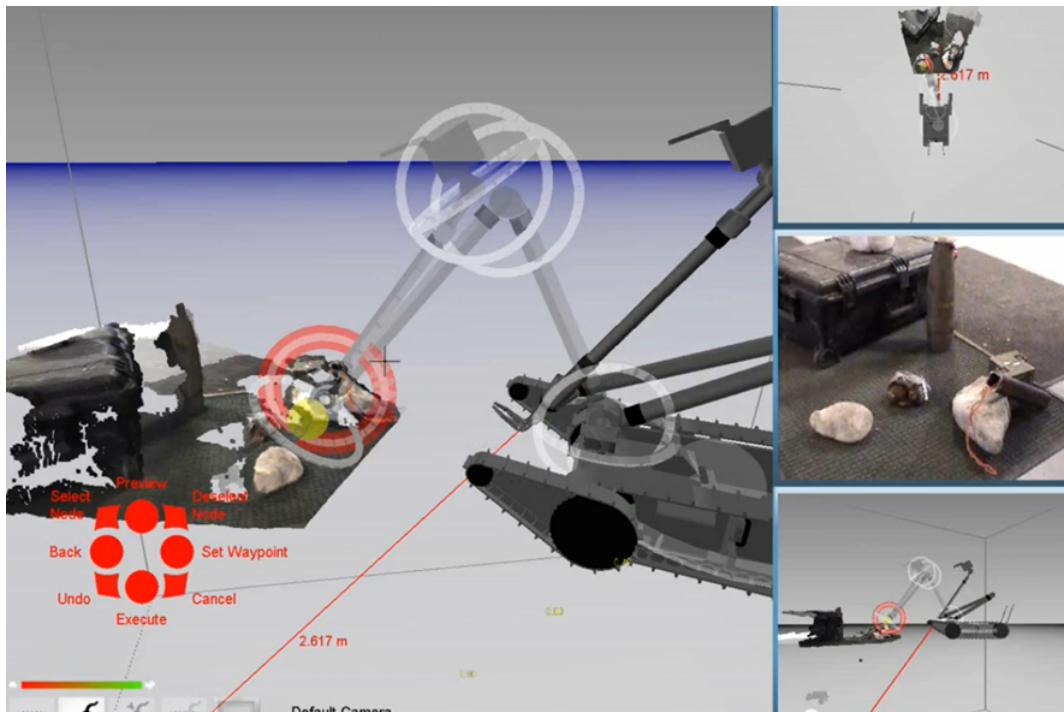
2.6. Przegląd istniejących rozwiązań

Niniejszy przegląd został przygotowany m.in. bazując na pracy Dr inż. Piotra Przystałki [15], który zauważa, że: "Współczesny stan techniki i nauki w zakresie rozwiązań konstrukcyjnych manipulatorów robotów mobilnych oraz ich systemów sterowania obecnie wyklucza możliwość uzyskania całkowitej autonomii działania urządzeń mechatronicznych tego typu." Mimo tego, trwają intensywne badania różnych instytucji naukowo-badawczych, przemysłowych i militarnych nad rozwojem w tym zakresie.

Prace o największym stopniu zaawansowania w zakresie systemów sterowania robotami mobilnymi oraz manipulatorami, prowadzone są przez agencje kosmiczne. W zagadnieniu podboju kosmosu, kwestia wysokiej autonomii jest jedną z najważniejszych, ponieważ w przestrzeni kosmicznej, pomoc ludzka w wykonywaniu przez roboty powierzonych im zadań jest niemożliwa. Przykładem mogą być prace Centrum Lotów Kosmicznych imienia Lyndona B. Johnsona przy Narodowej Agencji Aeronautyki i Przestrzeni Kosmicznej Stanów Zjednoczonych (NASA, JSC), które dotyczą badań nad połączeniem zdolności manipulacyjnych tułowia robota humanoidalnego *Robonaut* i zdolności mobilnych łazika *Centaur* [8]. Na Rys. 2.6 przedstawione są wspomniane układy.

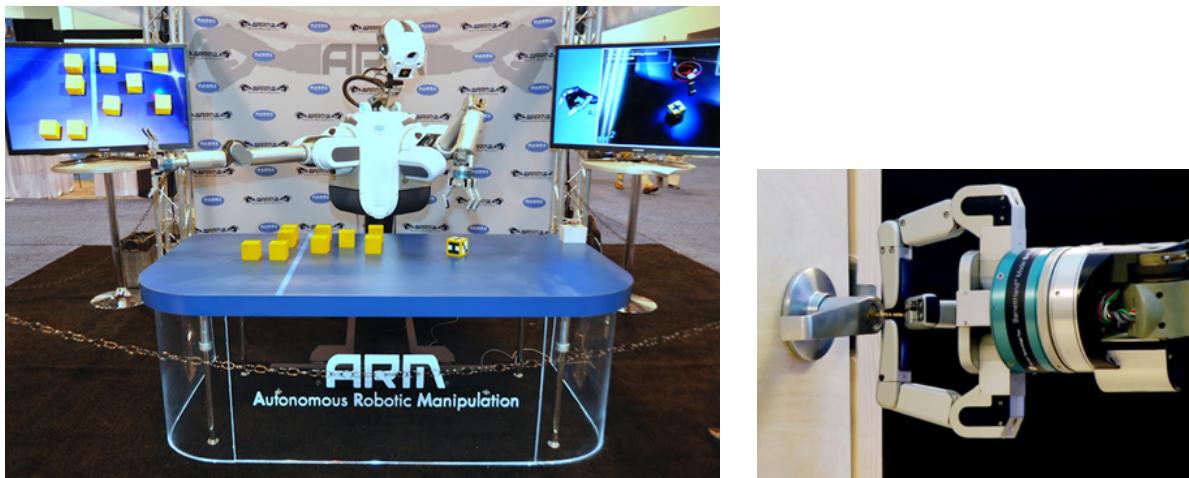
Jednym z najciekawszych ze znalezionych, istniejących już rozwiązań intuicyjnego sterowania, jest projekt firmy Autonomous Solutions Inc., która wspólnie z instytucjami EOD Technology Division (NAVEODTECHDIV), Navy SPAWAR oraz US Army TARDEC utworzyła system wizualizacji 3D przestrzeni roboczej robota mobilnego oraz manipulatora. Utworzono specjalną aplikację, widoczną na Rys. 2.7, która wyświetla model sceny, w środku której znajduje się robot i pozwala na sterowanie poprzez intuicyjny system oparty na podejściu "Kliknij i wykonaj działanie", polegający na tym, że operator podejmuje tylko decyzję o tym, co ma zostać wykonane, a robot wykonuje polecenie autonomicznie [14]. Jest to przykład daleko posuniętego intuicyjnego sterowania i intuicyjnego interfejsu użytkownika, przypominającego sterowanie w grach wideo, które cechują się przede wszystkim prostotą obsługi.

Razem w parze z intuicyjnością sterowania, zawsze musi koegzystować pewien stopień autonomiczność rozpatrywanego robota, aby swoją inteligencją, mógł on odciążyć operatora z nujących czynności. Przykładem może być program „Autonomous Robotic Manipulation (ARM)“ realizowany przez Agencję Zaawansowanych Obronnych Projektów Badawczych Departamentu Obrony Stanów Zjednoczonych (DARPA). Program ten obejmuje prace nad rozwojem i opracowywaniem rozwią-



Rys. 2.7: Widok okna aplikacji utworzonej przez firmę Autonomous Solutions Inc. [14]

zań sprzętowych i programistycznych umożliwiających uzyskanie manipulatorów o wysokim stopniu autonomii [10]. Głównym kierunkiem działań tego programu jest dążenie do utworzenia systemu sterowania pozwalającego na realizacje zadań w sposób autonomiczny. Na potrzeby tego zadania utworzono manipulator dwuramienny, a także udostępniono publicznie symulator tego manipulatora, umożliwiając tworzenie algorytmów sterowania tym manipulatorem innym osobom, także niezwiązanym z programem. Na Rys. 2.8 przedstawiono wspomniany wcześniej manipulator dwuramienny. Opracowane na potrzeby systemu algorytmy pozwalają na różnego rodzaju zadania manipulacyjne, począwszy od zszywania kartek zszywaczem, otwierania drzwi kluczem, poprzez załączenie i świecenie latarką w zadanym kierunku, a na wiercieniu otworów wiertarką skończywszy.



Rys. 2.8: Manipulator dwuramienny opracowany na potrzeby realizacji programu ARM [10]



Rys. 2.9: Wybrane funkcjonalności systemu sterowania firmy American Android Corp [12]

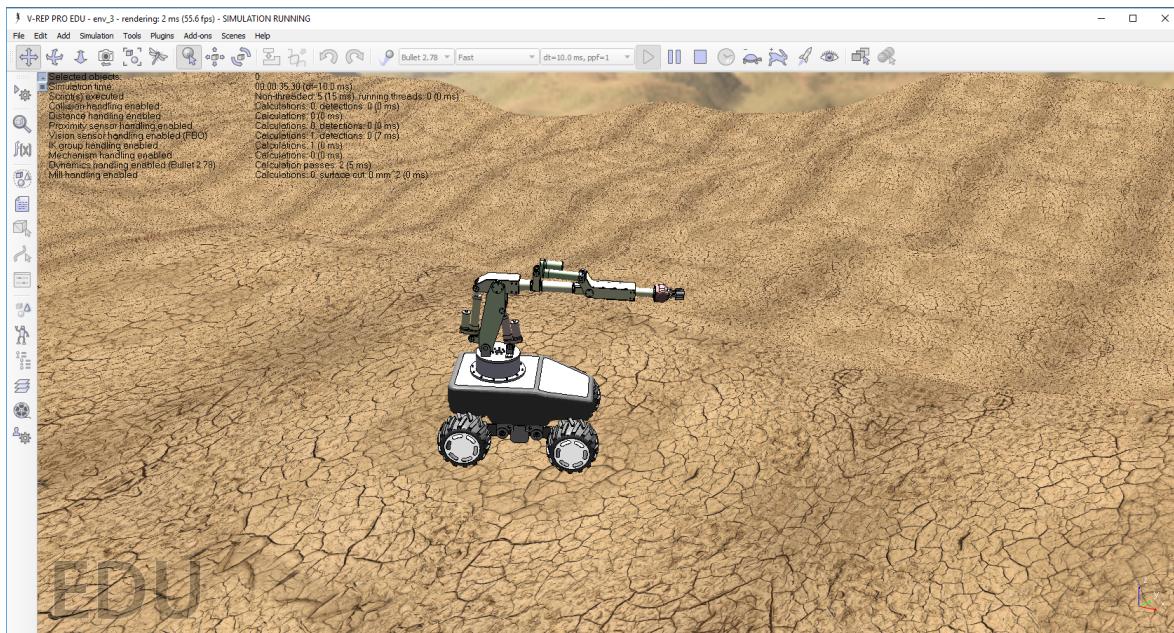
Z bardziej podobnych projektów do tego, przedstawionego w tej pracy, amerykańska firma American Android Corp zaoferowała oprogramowanie umożliwiające sterowanie manipulatorami o dużej liczbie stopni swobody. System ten pozwala na sterowanie zdalne, wyświetlając obraz z kamer zainstalowanych na manipulatorze [12]. Na Rys. 2.9 przedstawione są zastosowania tego systemu sterowania w manipulatorze dwuramiennym o 29 stopniach swobody. Intuicyjne sterowanie realizowane jest przez zastosowanie dwóch myszy 3D, a zadania realizowane przez ten system obejmują obsługę narzędzi, otwieranie drzwi czy przeszukiwanie plecaka.

Także w Polsce realizowane były działania na tle intuicyjnego systemu sterowania. W latach 2010-2012 realizowany był projekt o nazwie „Wielozadaniowe mobilne roboty wykorzystujące zaawansowane technologie”, który także wykorzystywał zadanie odwrotne kinematyki oraz kontroler 3D do wspomagania pracy operatora jednoramiennego robota transportowego [20].

Rozdział 3

Opracowanie środowiska symulacyjnego

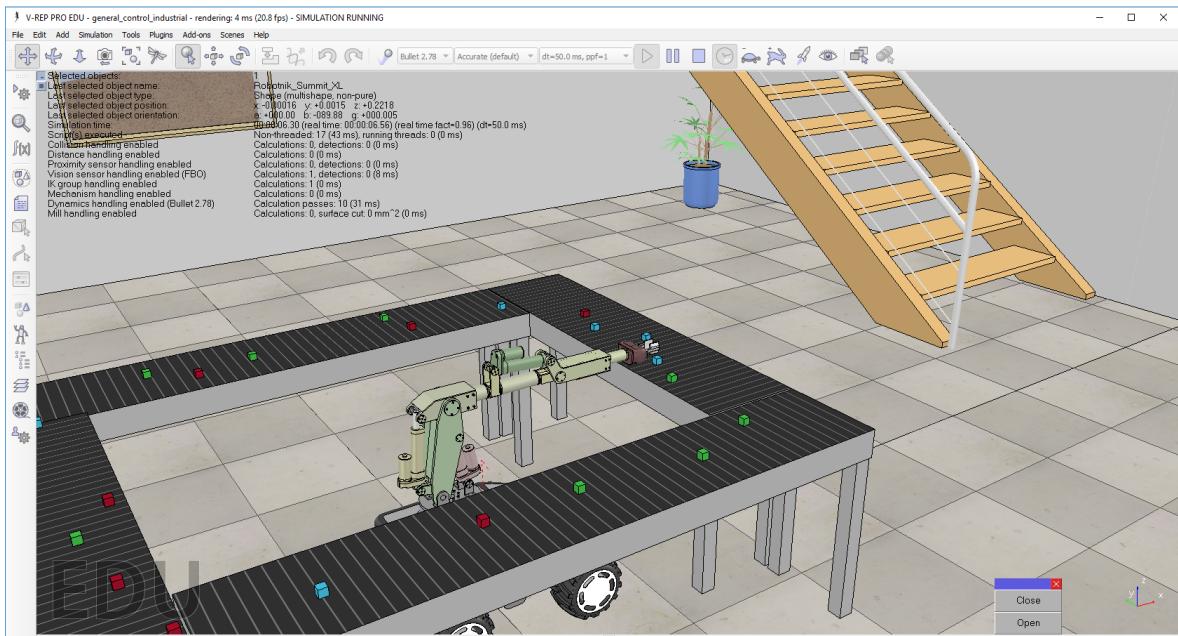
Utworzone środowisko symulacyjne opracowane w ramach niniejszej pracy składa się z dwóch składników: modelu robota mobilnego podlegającego sterowaniu oraz scenie z elementami której model będzie oddziaływał. Sceny reprezentują uproszczony fragment rzeczywistości, którego symulację należy przeprowadzić i mogą one być dowolnie tworzone, bądź modyfikowane w zależności od potrzeby. Z kolei, model robota mobilnego jest niezmienny w każdej scenie i reprezentować będzie rzeczywisty układ lub jego zachowanie. W tym rozdziale szczegółowo opisany zostanie utworzony model robota mobilnego z wyłączeniem szczegółów scen, ponieważ takowe zostaną ukazane w dalszej części pracy.



Rys. 3.1: Scena reprezentująca powierzchnię Marsa

Na Rys. 3.1 oraz 3.2 znajdują się przykładowe sceny z modelem robota mobilnego. Model robota mobilnego składa się z:

- wizualnych elementów podwozia, manipulatora i chwytaka,
- dynamicznych elementów podwozia, manipulatora i chwytaka,
- połączeń tych elementów w jeden łańcuch kinematyczny,



Rys. 3.2: Scena zawierająca elementy linii produkcyjnej

- skryptów obsługujących ruch podwozia i chwytaka,
- algorytmu sterującego realizującego kinematykę odwrotną manipulatora.

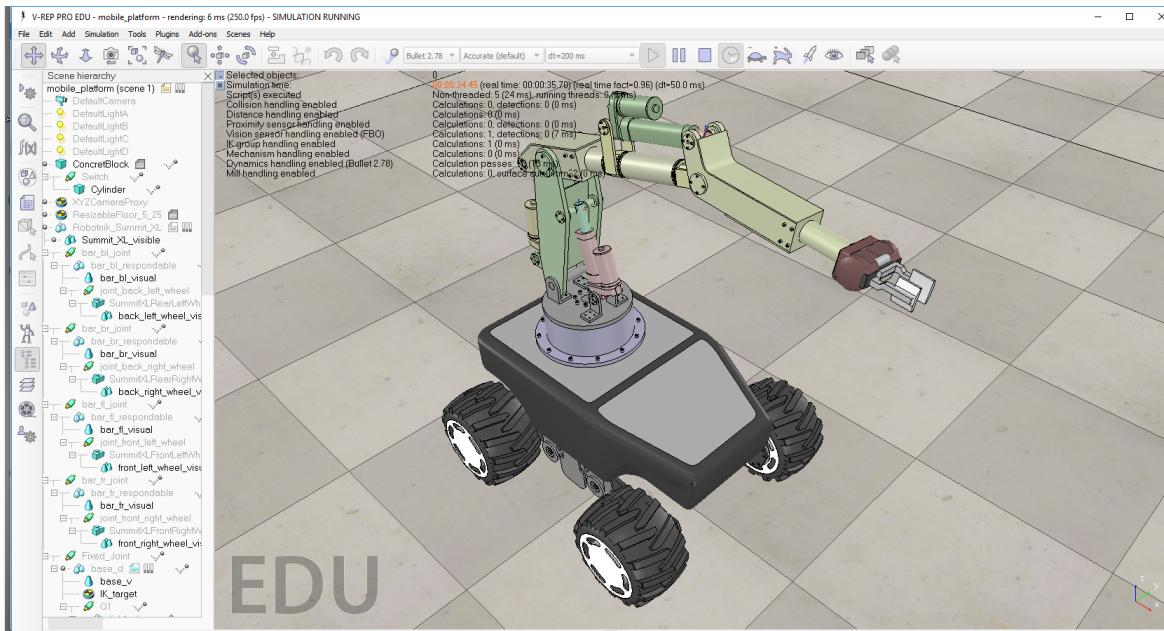
3.1. Import elementów CAD

Model CAD manipulatora, otrzymany od grupy Projektu Łazika Marsjańskiego zajmującej się modelowaniem CAD, który został wykorzystany, składał się oryginalnie z kilkunastu elementów związanych ze sobą w złożeniu. W oprogramowaniu CAD Autodesk Inventor elementy te zostały podzielone na grupy odpowiadające członom manipulatora mającym poruszać się względem siebie. Następnie grupy zostały scalone w jednolite bryły i eksportowane do formatu STL. W oprogramowaniu V-REP zostały zaimportowane i wstawione do sceny.

Warto dodać, że każda z części wyeksportowanych do pliku STL posiadała informację o pozycji układu współrzędnych, więc po wstawieniu wszystkich elementów do V-REPa człony manipulatora były ustawione tak samo jak w Inventorze, co jest bardzo ważne, przy definiowaniu parametrów dynamicznych.

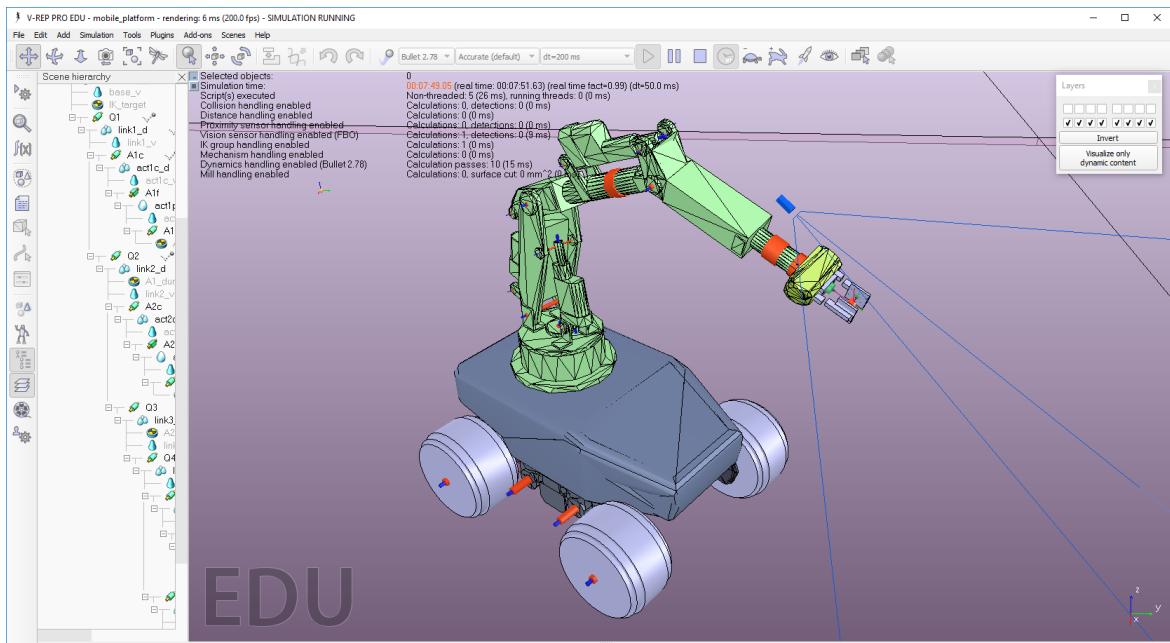
Z powodu braku modelu podwozia dostępnego w czasie przygotowywania modelu symulacyjnego, a także braku modelu chwytaka manipulatora, zdecydowano się na wykorzystanie bogatej biblioteki przykładów dostępnych darmowo w V-REPie. Wykorzystano model podwozia robota mobilnego Robotnik Summit XL [4], który został udostępniony dzięki uprzejmości firmy Robotnik oraz model chwytaka Baxter Gripper udostępnionego przez firmę Rethink Robotics [5]. Oba obiekty posiadają całkowicie przygotowaną część wizualną oraz dynamiczną modelu.

Na Rys. 3.3 przedstawiono wygląd wizualny modelu manipulatora wraz z podwoziem i chwytkiem.



Rys. 3.3: Widok modelu manipulatora wraz z podwoziem i chwytakiem w oprogramowaniu V-REP

3.2. Model dynamiczny



Rys. 3.4: Widok modelu dynamicznego manipulatora wraz z podwoziem i chwytakiem w oprogramowaniu V-REP

Aby zapewnić modelowi robota mobilnego interaktywność z otoczeniem, równolegle z elementami wizualnymi, istnieć muszą prostsze elementy dynamiczne. W zasadzie, poziom skomplikowania tych elementów może być dowolny, jednakże optymalnym rozwiązaniem jest stosowanie brył na tyle prostych, by obliczenia dynamiczne nie spowalniały nazbyt symulację, a także na tyle złożonych, by dobrze oddawały geometrię obiektu. Dobre w zastosowaniu są tzw. bryły wypukłe (ang. Convex

Shapes), które mają taką własność, że dla każdej pary punktów na brzegu tej bryły, odcinek łączący te punkty w całości należy do przestrzeni ograniczonej tą bryłą. Stosowanie tych brył jako obiektów dynamicznych znacznie przyspiesza obliczenia dynamiczne [18].

Do utworzenia elementów dynamicznych wykorzystano narzędzie dostępne w oprogramowaniu V-REP, które przekształca dowolną bryłę w zbiór brył wypukłych. Za pomocą tego narzędzia przekształcono wszystkie elementy wizualne modelu na elementy dynamiczne. Oba zbiory elementów zajmują te same pozycje w przestrzeni, a w celu ich odróżnienia, elementy dynamiczne przeniesione zostały do innej warstwy widoczności, dzięki temu nie muszą one być widoczne, gdy nie ma takiej potrzeby. Model dynamiczny przedstawiony jest na Rys. 3.4.

Kolejnym etapem przygotowywania modelu jest wprowadzenie parametrów dynamicznych elementów takich jak masy, momenty bezwładności oraz środki ciężkości. Korzystając z oryginalnego modelu CAD, będącego jeszcze przed procesem upraszczania, eksportowano wartości parametrów dla każdego członu, a następnie wprowadzono do uproszczonych elementów dynamicznych w V-REPie. Zachowując ten sam globalny układ współrzędnych zarówno w Inventorze jak i w V-REPie, nie było konieczności transformowania tensora bezwładności podczas jego przepisywania.

3.3. Drzewo kinematyczne

W momencie, gdy elementy wizualne oraz elementy dynamiczne były przygotowane, zbudowano z tych elementów drzewo kinematyczne całego robota mobilnego, które odzwierciedla rzeczywistą kinematykę układu. Struktura drzewa przedstawiona została na Rys. 3.5 Kompletne drzewo manipulatora składa się z:

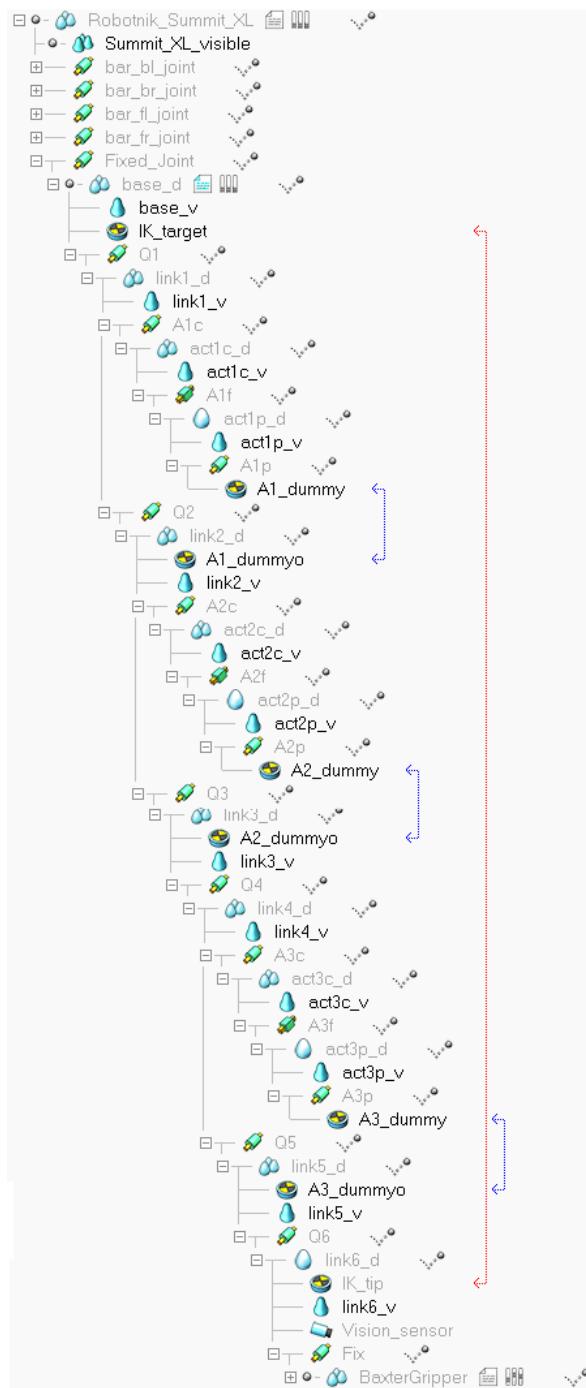
- par kinematycznych przegubowych (Q1 do Q6),
- par kinematycznych związanych z działaniem aktuatorów liniowych (A1 do A3),
- elementów dynamicznych (oznaczonych literką d),
- elementów wizualnych,
- punktów zorientowanych (ang. dummy).

Baza manipulatora połączona jest z podwoziem połączeniem sztywnym, tak samo jak chwytak podłączony jest do końcówki manipulatora.

Punkty zorientowane¹ mają w przestrzeni trójwymiarowej sześć stopni swobody i pełnią dwie funkcje w powyższym modelu. Pierwsza z nich, dotyczy rozwiązania zamkniętych pętli w łańcuchu kinematycznych, które występują w modelu. Gdy dwa punkty zorientowane zostaną ze sobą połączone, mogą utworzyć węzeł geometryczny. W drzewie kinematycznym sytuacja ta oznaczona jest niebieską strzałką. Oznacza to, że podczas symulacji te dwa punkty zorientowane będą zawsze w tym samym miejscu i będą miały tą samą orientację². Drugą funkcją, jaką pełnią punkty zorientowane jest utworzenie pary końcówka-cel (ang. tip-target), która służy do definiowania zadania kinematyki odwrotnej w module kinematyki odwrotnej V-REPa. W drzewie kinematycznym takie połączenie oznaczone jest czerwoną strzałką.

¹Zorientowane znaczy tutaj „posiadające swoją pozycję i orientację”

²Jest możliwa sytuacja, w której warunek ten nie będzie spełniony. Stanie się tak, gdy pary kinematyczne zostaną z jakiegoś powodu wyłamane podczas trwania symulacji.



Rys. 3.5: Drzewo kinematyczne modelu robota mobilnego

W modelu robota mobilnego, wykorzystywane są dwa rodzaje par kinematycznych V klasy: para obrotowa oraz para przesuwna. Kierunek działania połączenia definiuje się poprzez odpowiednie przestrzenne ustawienie wizualnego obiektu reprezentującego daną parę kinematyczną, a parametry działania danej pary w ustawieniach. Każda para kinematyczna posiada swój zakres ruchu (niektóre mogą być cykliczne³) oraz parametry dynamiczne, takie jak maksymalna siła (lub moment siły), którą może wygenerować para kinematyczna.

³cykliczne oznacza tutaj „mogące obracać się w nieskończoność”

3.4. Algorytmy sterowania

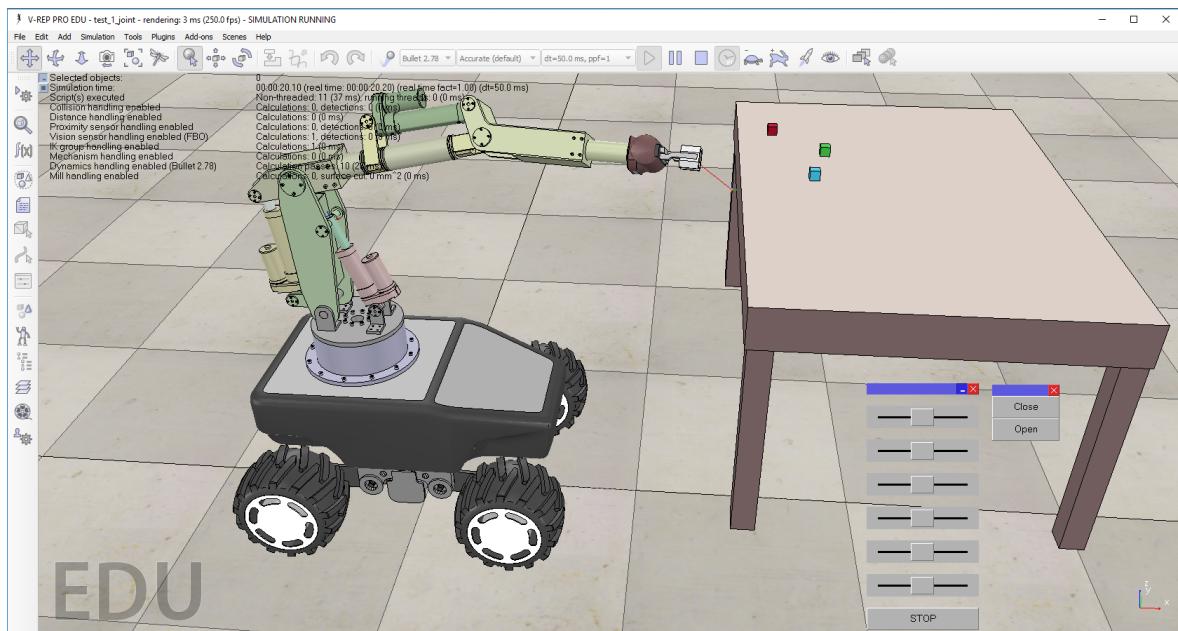
Występują trzy podukłady sterowania w przygotowanym modelu robota mobilnego:

- sterowanie kołami podwozia,
- sterowanie manipulatorem oparte na kinematyce odwrotnej,
- sterowanie zaciskami chwytaka.

Najprostsze z nich, sterowanie zaciskami chwytaka, opiera się na egzekwowaniu dwóch poleceń: otwórz i zamknij chwytak. Chwytak aktuowany jest poprzez pojedynczą, przesuwną parę kinematyczną, a aktuacja odbywa się przez ustawienie pożądanej wartości prędkości na parze kinematycznej. Kolejne, bardziej skomplikowane, choć wciąż proste, jest sterowanie kołami podwozia. Na każde koło przypada jedna cykliczna obrotowa para kinematyczna, co pozwala na sterowanie każdym kołem z osobna, co z kolei umożliwia sterowanie pojazdem po prostej oraz po łuku.

Algorytm sterowania manipulatorem oparty jest na module kinematyki odwrotnej wbudowanym w oprogramowaniu V-REP. Działanie tego modułu polega na minimalizacji odległości i odchylenia między punktami zorientowanymi (wspomnianymi w poprzedniej sekcji tekstu) w parze końcówka-cel poprzez zmianę pozycji konfiguracyjnej par kinematycznych zawartych między końówką a celem. Do tego celu wykorzystywany jest metoda Pseudoinverse lub Damped Least Squares (DLS). Obie metody są metodami iteracyjnymi i nie gwarantują znalezienia rozwiązania, mimo iż takie rozwiązanie może istnieć, lecz dają dobre rezultaty przy odpowiednich ustawieniach.

Zmiana pozycji lub orientacji punktu celu powoduje wzrost (ogólniej: zmianę) odległości między celem, a końówką, co powoduje ponowne rozwiązywanie kinematyki odwrotnej i ustawienie pozycji konfiguracyjnej par kinematycznych⁴.



Rys. 3.6: Widok modelu z prostą wersją interfejsu sterowania

⁴Ścisłe mówiąc zadanie kinematyki odwrotnej rozwiązywane jest wraz z każdym krokiem przebiegu symulacji.

W celach badawczych opisanych w dalszej części pracy utworzona także wersję sterowania manipulatorem opartą o kinematykę prostą - na każdy przegub podawana jest prędkość zależna od pozycji odpowiadającego suwaka na wewnętrznym interfejsie użytkownika, który przedstawiono na Rys. 3.6.

3.5. Skrypty sterujące

Poprawne działanie modelu robota mobilnego wymagało napisania kilku prostych skryptów wewnętrznych w języku LUA, są to skrypty do obsługi:

- serwera zdalnego sterowania oraz komunikacji z zewnętrzną aplikacją,
- sterowania podwoziem za pomocą klawiatury,
- sterowania chwytkiem oraz manipulatorem w trybie prostym.

Nie wszystkie skrypty koniecznie muszą występować w tej samej scenie, w zależności od potrzeby, odpowiednie skrypty mogą zostać deaktywowane.

Sterowanie za pomocą intuicyjnej aplikacji sterującej musi być poprzedzone uruchomieniem serwera zdalnego sterowania przez oprogramowanie V-REP, które odbywa się podczas symulacji poprzez wywołanie odpowiedniej funkcji z poziomu skryptu. Funkcja ta zwraca między innymi numer portu, przez który odbywać się będzie komunikacja, co także oznacza, że ten numer trzeba przekazać funkcji łączącej się z serwerem w zewnętrznej aplikacji. Nie ma możliwości, by V-REP nie był serwerem w tym typie komunikacji, co rodzi pewne problemy, gdyż z założień dotyczących intuicyjności, to aplikacja sterująca ma samoczynnie inicjować komunikację. Zostało to rozwiązane w następujący sposób:

- aplikacja sterująca uruchamia symulator przekazując mu ścieżkę dostępu,
- po wczytaniu symulatora, tenże uruchamia symulację,
- po uruchomieniu symulacji, wykonuje się skrypt uruchamiający serwer sterowania zewnętrznego,
- skrypt zapisuje za pomocą podprogramu sapi.exe potrzebne informacje do pliku znajdującego się w przekazanym wcześniej miejscu,
- aplikacja sterująca odczytuje plik z informacjami po upływie określonego czasu (by pozostawić szansę na wykonanie trzech poprzednich kroków),
- po odczytaniu pliku, aplikacja sterująca łączy się z serwerem sterowania zewnętrznego za pomocą zawartych w nim danych.

Teraz, gdy aplikacja jest podłączona, ma ona dostęp do wszystkich funkcji sterujących umożliwiających odczytywanie danych z symulacji, a także zmianę pozycji celu, powodując ruch manipulatorem.

Skrypt sterujący podwoziem wraz z każdym krokiem symulacji, sprawdza czy nie został naciśnięty jeden z klawiszy strzałek, następnie w zależności od wciśniętego klawisza ustawia niezerowe prędkości na odpowiednie pary kinematyczne związane z kołami podwozia, tak by powodować pożądany ruch robota mobilnego. Skrypty sterujące manipulatorem w trybie prostym oraz chwytkiem, odczytują stan kontrolek na interfejsie użytkownika i w zależności od jego stanu ustawiają prędkości na odpowiednich parach kinematycznych.

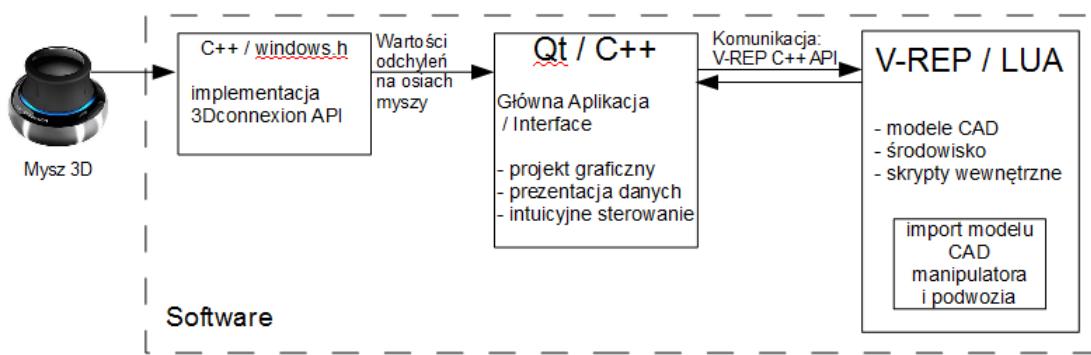
Rozdział 4

Projekt i implementacja interfejsu użytkownika

W tym rozdziale przedstawione zostaną projekty i koncepcje interfejsu użytkownika, ukazane zostanie rzeczywiste wykonanie interfejsu oraz wyjaśniona zostanie architektura aplikacji.

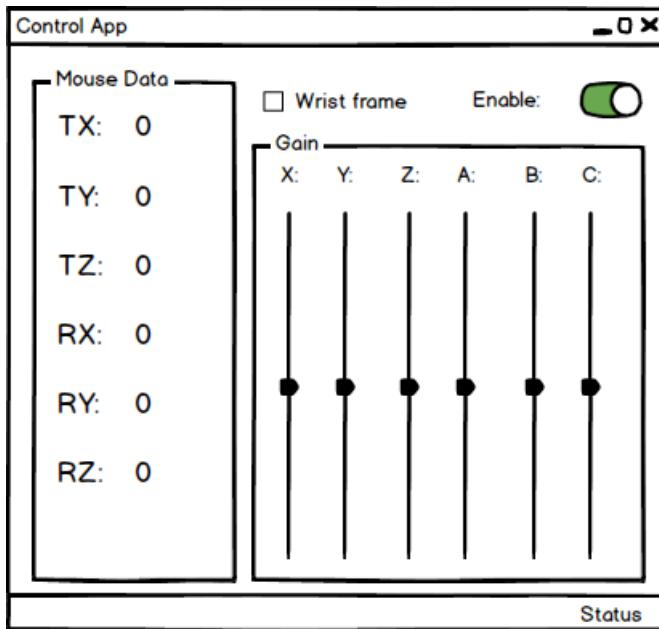
4.1. Koncepcje działania programu oraz interfejsu użytkownika

Na Rys. 4.1 przedstawiona została koncepcja działania aplikacji. Koncepcja zakłada modułowe rozwiązywanie problemu sterowania modelem robota mobilnego w symulatorze. Podstawową ideą jest podział zadań między trzema skomunikowanymi aplikacjami, co skutkuje lepszą przejrzystością całego systemu, a także powoduje możliwość szybkiej zamiany jednego elementu systemu na inny, bez ingerencji w pozostałe elementy (np. gdy wystąpi konieczność użycia innego urządzenia wejścia niż mysz 3D, wystarczy utworzyć nowy sterownik obsługujący dane urządzenie bez ingerencji w główną aplikację).



Rys. 4.1: Schemat koncepcji działania aplikacji

Sygnały odchyleń na poszczególnych osiach i stan przycisków myszy 3D odbierane będą w pętli przez program implementujący bibliotekę do obsługi myszy 3D udostępnioną przez producenta. Każde zmiana stanu urządzenia powodować będzie przesyłanie danych poprzez komunikację międzyprocesową do aplikacji głównej, która przetwarzając te dane będzie odpowiednio sterować elementami symulacji.

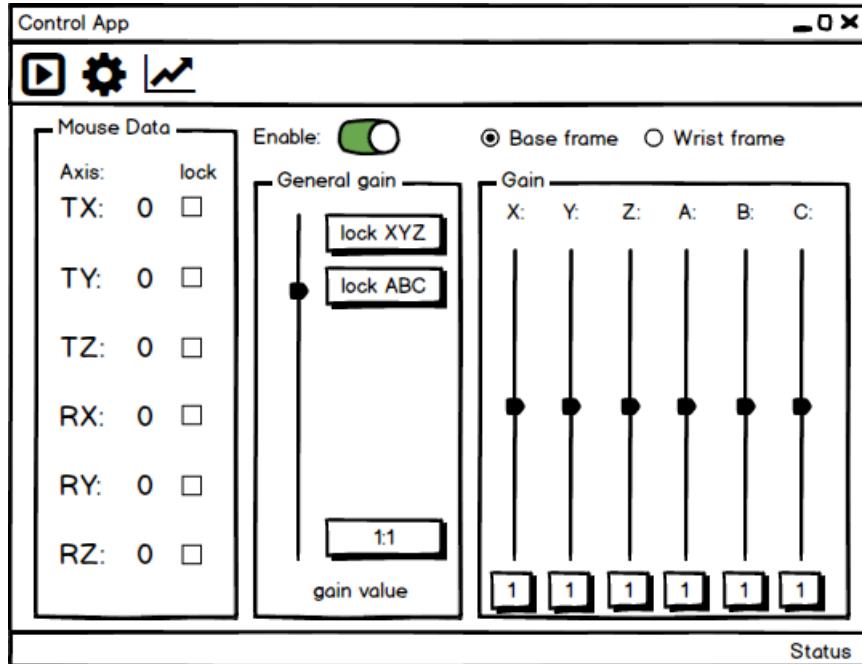


Rys. 4.2: Podstawowa koncepcja wyglądu okna głównego aplikacji

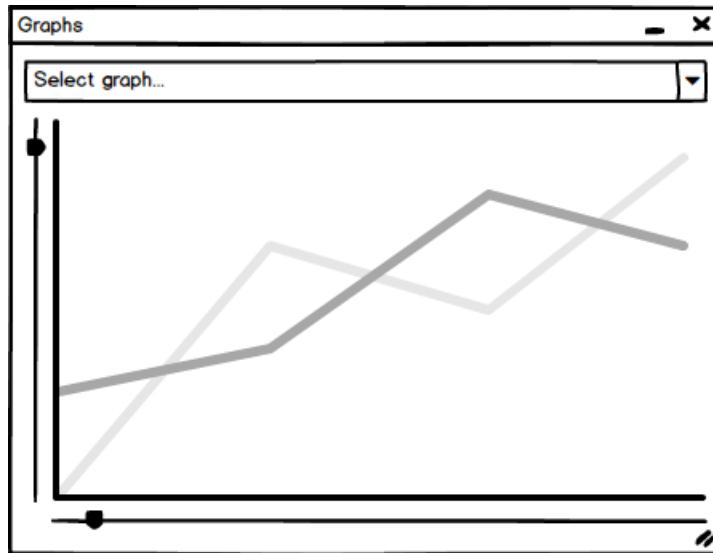
Zgodnie z przyjętymi założeniami, okno aplikacji powinno być jak najprostsze. Powinno umożliwiać wybór układu współrzędnych, względem których odbywa się sterowanie myszą 3D oraz zmianę prędkości działania myszy. Na Rys. 4.2 przedstawiony jest wygląd okna posiadającego wymienione cechy. Podczas projektowania tego okna, szybko pojawiły się pomysły zwiększające funkcjonalność interfejsu, a finalna koncepcja przedstawiona została na Rys. 4.3.

Etykiety w grupie Mouse Data pokazywać będą wartości odchyлеń przekazane przez sterownik do aplikacji, natomiast w grupie Gain znajdować się będą suwaki pozwalające dostosować prędkość wyjściową dla każdej osi z osobna, zwiększać wygodę operatora, który będzie mógł dostosować ustawienie do swoich preferencji. Oczywistą funkcją przełącznika Enable jest dezaktywacja sterowania, która może się przydać w niektórych sytuacjach. Można sobie wyobrazić sytuację, w której nastąpi potrzeba przemieszczenia się tylko w określonej liczbie osi, nie we wszystkich. Pomocna może się okazać funkcja ignorowania wartości odchylenia z wybranej osi myszy, co ułatwi operowanie manipulatorem w ciasnych przestrzeniach. Powyższa funkcja widoczna jest na Rys. 4.3, jako pola wyboru przy etykietach osi.

Na Rys. 4.3 przedstawione są także znajdujące się pod każdym suwakiem przyciski, które będą resetować je do pozycji domyślnej. Na górze okna aplikacji, będzie znajdować się pasek narzędzi umożliwiający szybki dostęp do ustawień i innych funkcji. W grupie General gain pojawi się także suwak pozwalający zmieniać prędkość myszy 3D we wszystkich osiach jednocześnie oraz przyciski pozwalające na blokowanie osi translacyjnych i osi rotacyjnych jednym kliknięciem. W aplikacji dostępne będą także wykresy przebiegów niektórych wielkości, których strumień docierać będzie z symulacji. Koncepcja wyglądu okienka z wykresami przedstawiona została na Rys. 4.4.



Rys. 4.3: Finalna koncepcja wyglądu okna głównego aplikacji



Rys. 4.4: Koncepcja wyglądu okna wyświetlającego wykresy

4.2. Wykonanie aplikacji

Na podstawie przedstawionej koncepcji, napisano aplikację z graficznym interfejsem użytkownika z wykorzystaniem biblioteki programistycznej Qt w wersji 5.7. Na Rys. 4.5 przedstawiony został zrzut ekranu głównego okna aplikacji podczas symulacji. Należy zwrócić uwagę, że większość czasu okno to i tak jest niewidoczne, gdyż schowane jest za oknem symulacji, która jest na pierwszym planie. W poniższych podrozdziałach przedstawiony został szczegółowy opis interfejsu oraz architektura działania aplikacji.



Rys. 4.5: Zrzut ekranu głównego okna aplikacji podczas działania

4.3. Instalacja oraz pierwsze uruchamianie

Przed instalacją samej aplikacji należy zainstalować następujące składniki:

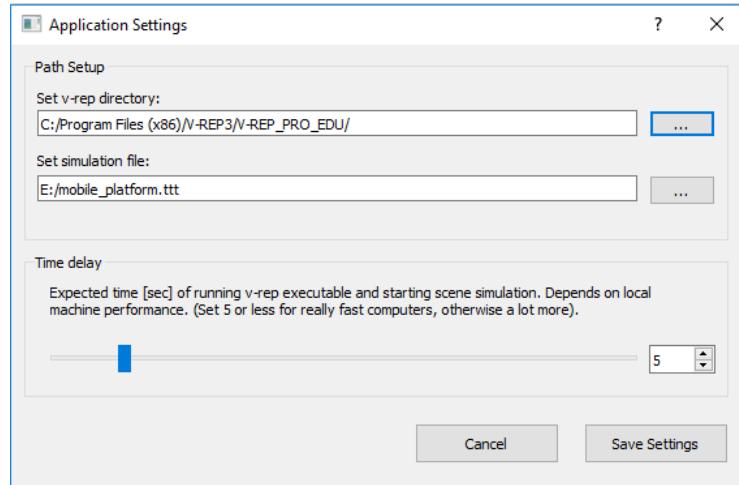
- sterowniki 3Dconnexion dostarczone przez producenta myszy 3D (wymagane do korzystania z myszy 3D w ogóle),
- pakiet Visual C++ Redistributable 2015 w wersji 64-bitowej (darmowo dostępne do pobrania ze strony firmy Microsoft),
- oprogramowanie V-REP (darmowo dostępne do скачnięcia ze strony Coppelia Robotics).

Instalacja polega na ręcznym przekopiowaniu plików:

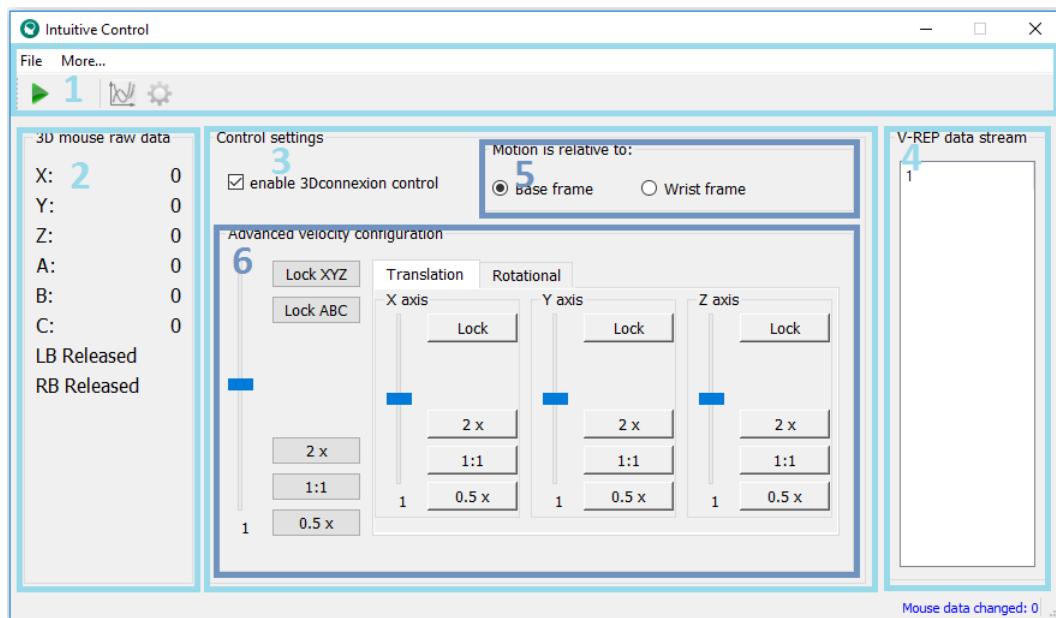
- wykonywalnych (Intuitive.exe, mapi.exe, sapi.exe, settings.exe),
- bibliotek dynamicznych (Qt5Core.dll, Qt5Gui.dll, Qt5Widgets.dll, platforms/qwindows.dll, imageformats/qjpeg.dll)

ze źródła do folderu znajdującego się na lokalnym dysku.

Podczas pierwszego uruchomienia aplikacja (Intuitive.exe) automatycznie wykryje brak pliku konfiguracyjnego i uruchomi konfigurację. W oknie konfiguracji, które przedstawione zostało na Rys. 4.6, należy podać za pomocą okna dialogowego lokalizację do zainstalowanego V-REPa (jest to miejsce, w którym znajduje się plik vrep.exe). Następnie należy wybrać plik z wybraną sceną, która zostanie wczytana przez symulator po jego uruchomieniu. Uwaga, ścieżki dostępu do pliku scen nie mogą zawierać spacji, gdyż powoduje to niemożliwość wczytania sceny. Ustawienia zatwierdza się przyciskiem Save Settings. Od tego momentu utworzony został plik konfiguracyjny settings.set, w którym zapamiętane są ustawienia, więc każde kolejne uruchomienie aplikacji nie będzie wymagało ponownego podawania ścieżek.



Rys. 4.6: Zrzut ekranu okna konfiguracji



Rys. 4.7: Obszary okna głównego aplikacji sterującej

4.3.1. Opis okna głównego oraz instrukcja użytkowania

Na Rys. 4.7 przedstawione jest okno główne aplikacji wraz ze zaznaczonymi obszarami opisanymi poniżej.

1: W tym miejscu znajduje się menu, w którym znajdują się przyciski do uruchomienia symulacji, szybkiego restartu aplikacji, uruchomienia ustawień czy wyświetlenia okna z wykresami. Najważniejsze funkcje zostały umieszczone jako skróty na pasku narzędzi znajdującym się poniżej menu.

2: W tym polu znajdują się dane odbierane przez aplikację dotyczące stanu myszy 3D. Pojawiają się tutaj wartości odchyлеń na poszczególnych osiach myszy, a także informacja o tym, w jakim stanie są przyciski myszy (odpuścizone, naciśnięte).

3: W centrum okna znajdują się wszystkie ustawienia razem z przyciskiem do aktywacji/dezaktywacji sterowania myszą 3D.

4: Podczas trwania symulacji, w tabelce znajdującej się w tym polu pojawiać się będą informacje płynące z V-REPa dotyczące symulacji i stanu robota mobilnego. Przed symulacją element ten jest nieaktywny.

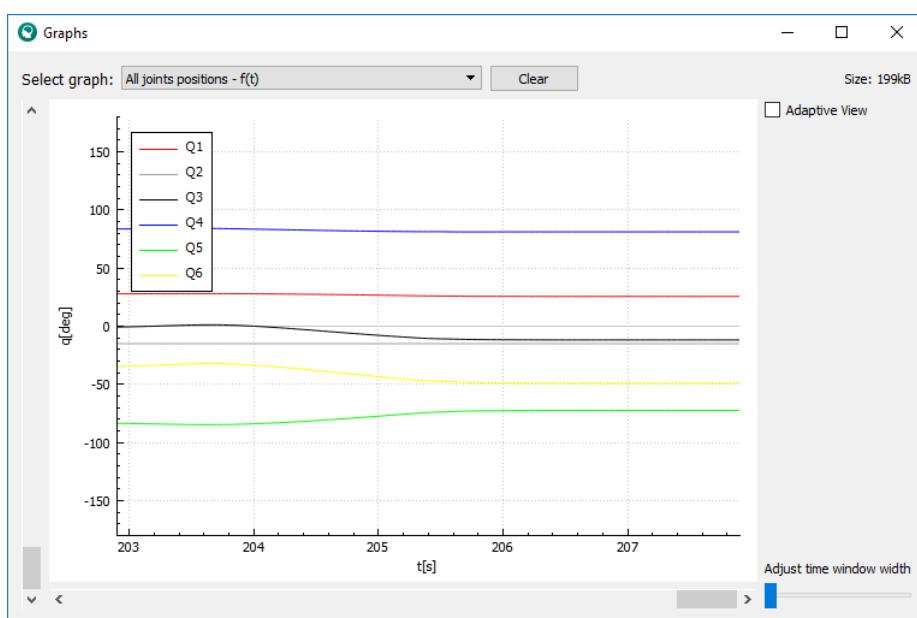
5: Przełączając przyciski w tej ramce, dokonywany jest wybór układu współrzędnych, w którym realizowane jest sterowanie.

6: W tej sekcji operator ma dostęp do zaawansowanej konfiguracji prędkości poszczególnych osi, a także ma możliwość blokowania niepotrzebnych w danym momencie przemieszczeń. Są tutaj także przyciski skrótów do szybkiego ustawienia suwaków na predefiniowane pozycje.

Aplikacja obsługuje także klawiaturę, co pozwala na zmianę relatywnego układu współrzędnych (klawisz 0), czy blokowanie osi (klawisze od 1 do 6) bez konieczności opuszczania symulacji.

Aby uruchomić symulację razem z wybraną wcześniej sceną należy nacisnąć zieloną strzałkę na pasku narzędzi. Po pewnym czasie uruchomiony zostanie symulator, a sterowanie myszą 3D jest już aktywne. W celu zmiany sceny należy uruchomić ustawienia, wybrać nową scenę, zapisać ustawienia i uruchomić ponownie aplikację sterującą.

4.3.2. Okno wykresów



Rys. 4.8: Okno rysujące wykresy

Okno z wykresami przedstawione na Rys. 4.8, rysuje przebiegi wybranych wielkości odczytywanych z symulatora. Do wyboru są następujące opcje:

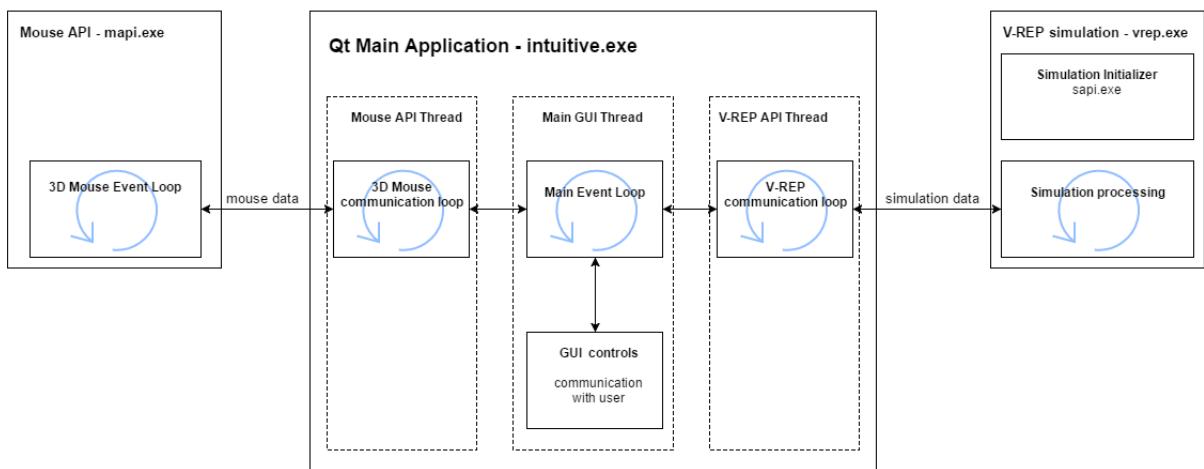
- pozycja kątowa przegubów,
- pozycja aktuatorów liniowych,
- siły występujące na aktuatorach,
- pozycja końcówki i celu w rzucie na płaszczyzne XY, XY oraz YZ.

Znajdujące się w oknie wykresów paski przewijania służą do dostosowywania skali wykresu. Pole wyboru będące w prawym górnym rogu pozwala na zmianę trybu wyświetlania na tryb adaptacyjny,

który zawsze wyświetla cały przebieg od początku, w przeciwnym razie, suwak poniżej pozwala dostosować szerokość widocznego okna czasowego na wykresie. Wielkość okna może być modyfikowana w zależności od preferencji.

4.3.3. Architektura aplikacji

Na Rys. 4.9 przedstawiony został ogólny schemat działania aplikacji z uwzględnieniem podziału na podprogramy i wątki. Użycie wątków jest konieczne ze względu na występowanie funkcji oczekujących zarówno po stronie obsługi sygnałów z myszy 3D jak i po stronie obsługi komunikacji z symulatorem. Komunikacja wewnętrzna między wątkami odbywa się za pomocą mechanizmu sygnałów i slotów zaimplementowanych w Qt.



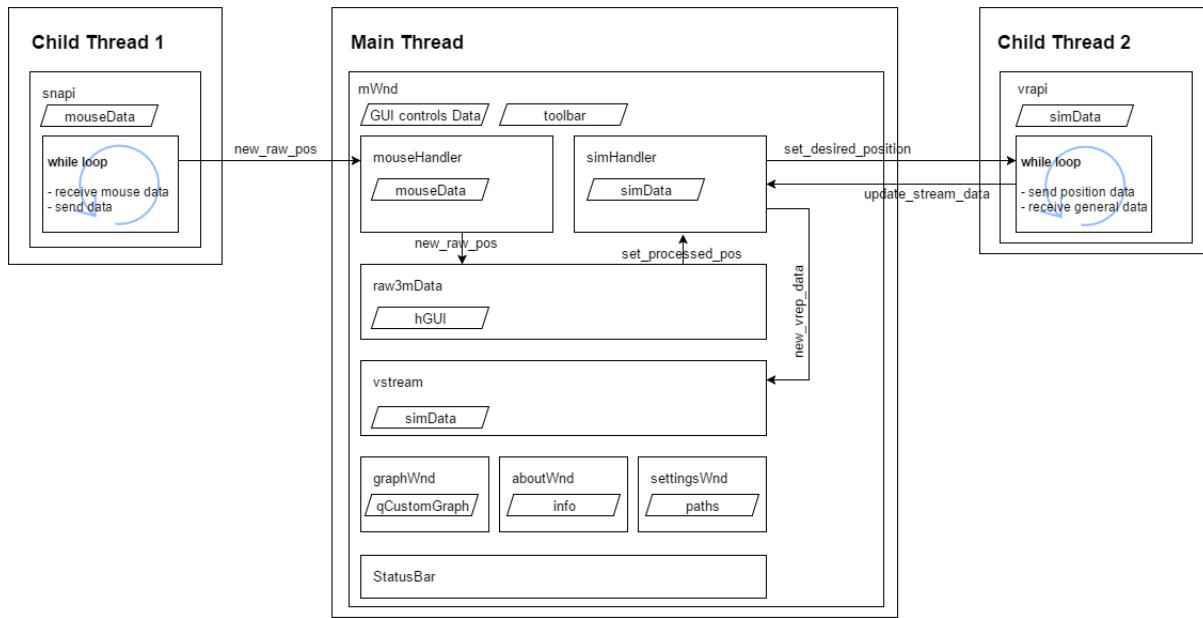
Rys. 4.9: Schemat ogólnego działania aplikacji

Program mapi.exe reaguje na wydarzenia zmiany stanu myszy dostarczane przez system operacyjny i wysyła do aplikacji głównej Intuitive.exe odpowiednie dane za pomocą systemu komunikacji międzyprocesowej, metodą wykorzystującą tzw. potoki nazwane (ang. named pipes). Funkcje obsługujące tą metodę dostępne są w systemach Windows po dołączeniu pliku nagłówkowego "windows.h". Po stronie głównej aplikacji, wątek obsługujący mysz 3D, w pętli nasłuchiwanie nowych danych, a gdy takowe się pojawią, emituje sygnał odbierany przez inne obiekty.

Symulator vrep.exe skomunikowany jest z wątkiem obsługującym symulację za pomocą interfejsu programistycznego aplikacji (API) dostarczonego przez producenta oprogramowania V-REP. Wątek ten, także w pętli wysyła i odbiera dane, przekazując najaktualniejsze informacje dotyczące elementów symulacji do struktury odpowiedzialnej za przechowywanie tych danych.

Na Rys. 4.10 przedstawiony jest schemat podziału aplikacji na klasy i struktury danych, a także oznaczone są kluczowe sygnały przekazujące dane między obiektami danych klas. Na schemacie obiekt danej klasy reprezentowany jest przez ramkę o prostokątnym kształcie, a dane składowe reprezentowane są przez ramkę o kształcie równoległoboku. Obiekt klasy *snapi* obsługujący dane z myszy 3D oraz obiekt klasy *vrapi* obsługujący komunikację z symulatorem działają, we wspomnianych wcześniej, osobnych wątkach. Oba obiekty posiadają odpowiadające ich przeznaczeniu dane.

Klasa *mWnd* zajmuje się obsługą głównej pętli wydarzeń i dba o interfejs aplikacji. Znajdują się w niej pozostałe opisane poniżej klasy oraz wszystkie elementy interfejsu takie jak przyciski,



Rys. 4.10: Schemat ogólny działania aplikacji

etykiety, przełączniki itp. Klasy *mouseHandler* oraz *simHandler* odpowiedzialne są za uruchamianie i terminację wątków potomnych, a także stanowią interfejs komunikacji innych klas z tymi wątkami. Klasa *raw3mData* zajmuje się przetwarzaniem danych z myszy 3D, czyli mnożeniem wartości przez wzmacnienia ustawione przez użytkownika i dbaniem o to, aby zablokowane osie nie powodowały ruchu. Po przetwarzaniu, klasa wysyła nowe dane do klasy *mouseHandler*, która zgłasza nowe wartości do wysłania do symulatora. Klasa *vstream* odpowiedzialna jest za obróbkę informacji odebranych z symulatora oraz wyświetlanie ich w tabeli. Klasa *graphWnd*, *aboutWnd* oraz *settingsWnd* dotyczą dodatkowych okien, których wyświetlenie może zostać wywołane w czasie działania aplikacji.

Rozdział 5

Sporządzenie planu i przeprowadzenie badań

Badania będą polegały na wykonywaniu przez wybranych uczestników zestawu trzech zadań związanych z koniecznością wykonania pewnych zadań manipulacyjnych opisanych w kolejnej sekcji. Każde zadanie będzie wykonywane przez każdego użytkownika dwa razy, pierwszy raz z użyciem sterowania prostego, a drugi raz z użyciem intuicyjnego sterowania z zastosowaniem kinematyki odwrotnej i myszy 3D. Podczas wykonywania zadania odmierzany będzie czas od momentu rozpoczęcia do momentu zakończenia. Następnie czasy wynikowe zestawione zostaną w tabeli, która ułatwi porównywanie wyników i wyciąganie wniosków. Maksymalny czas wykonania pojedynczego zadania został ustalony na 5 minut, po upłynięciu tego czasu zadanie zostaje uznane za niezaliczone.

5.1. Sceny testowe

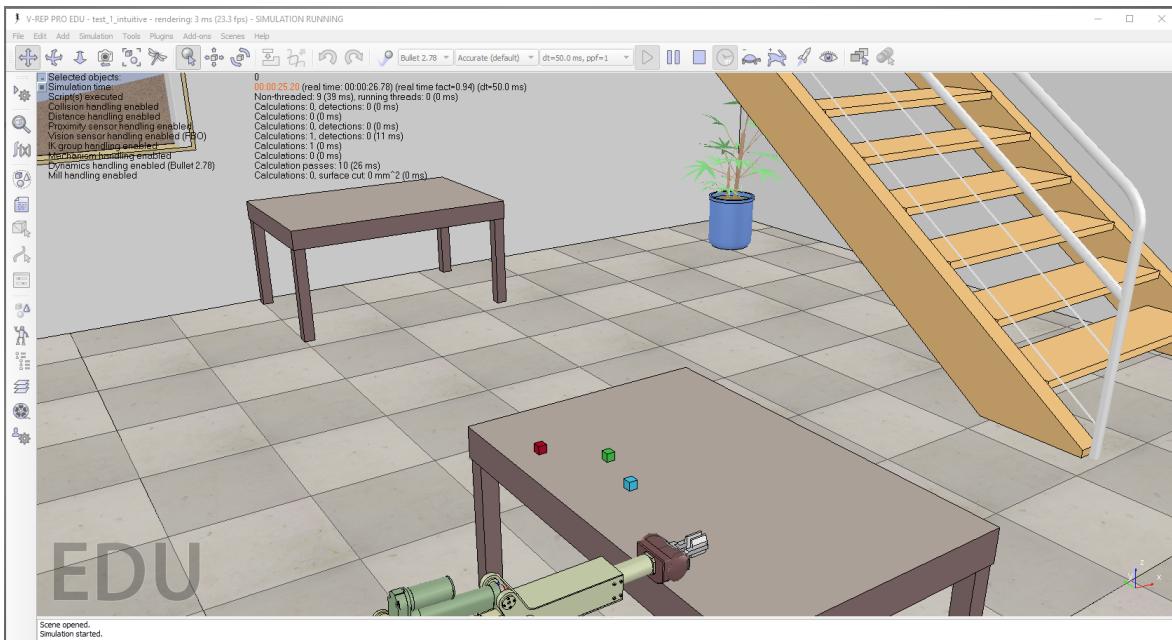
Do każdego zadania przygotowane zostały osobne sceny. Poniżej krótko opisana została każda z nich.

5.1.1. Przenoszenie obiektów w widoku ogólnym

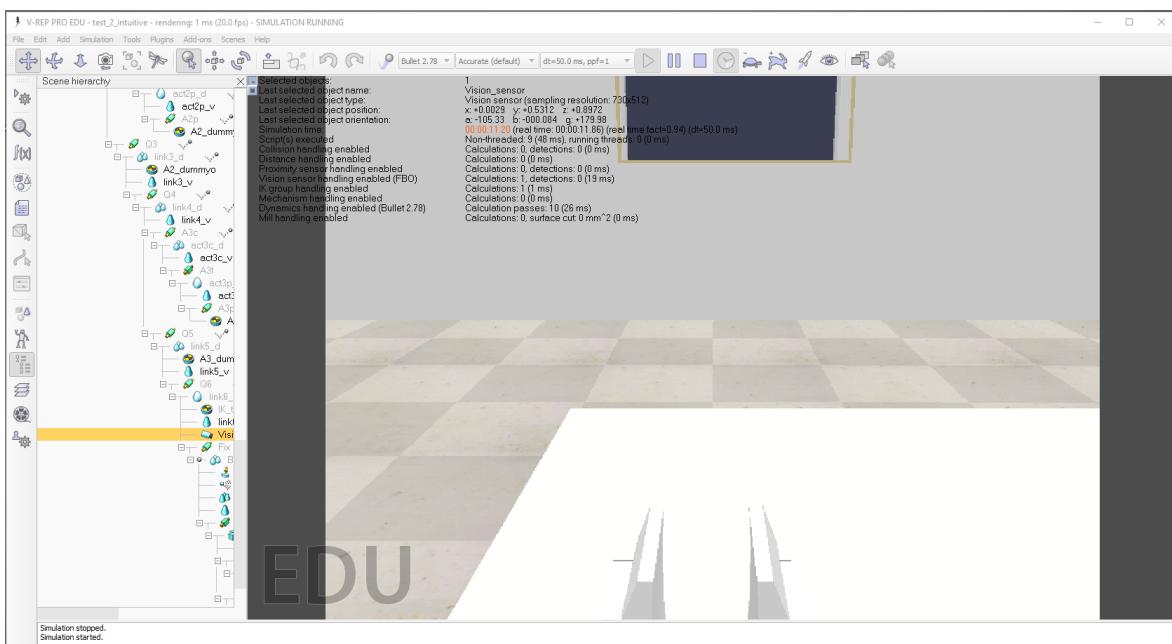
Zadanie pierwsze polega na przeprowadzaniu czynności manipulacyjnych. Należy za pomocą chwytaka manipulatora podnieść i przenieść trzy prostopadłościenne kostki z jednego stołu na drugi, znajdujący się w pewnej odległości. Odległość ta jest na tyle duża, że oprócz sterowania manipulatorem, wymaga także skorzystania z możliwości lokomocyjnych całego robota mobilnego. W tym zadaniu, dozwolony jest widok z dowolnego punktu. Na rys. 5.1 przedstawiono wygląd sceny testowej.

5.1.2. Przenoszenie obiektów z widokiem z kamery

Zadanie drugie podobne jest do zadania pierwszego, także polega na przeprowadzeniu czynności manipulacyjnych, tym razem jednak, umożliwione jest korzystanie wyłącznie z kamery umieszczonej na manipulatorze. Ma to symulować rzeczywistą sytuację, gdzie operator łazika nie ma dostępu do dowolnego widoku ogólnego, a jedynie widok z kamer pokładowych. Jednakże stolik, na który położyć trzeba prostopadłościenne elementy stoi dużo bliżej stolika z ich pozycją startową. Na Rys. 5.2 przedstawiono widok, który widzi uczestnik, po rozpoczęciu testu.



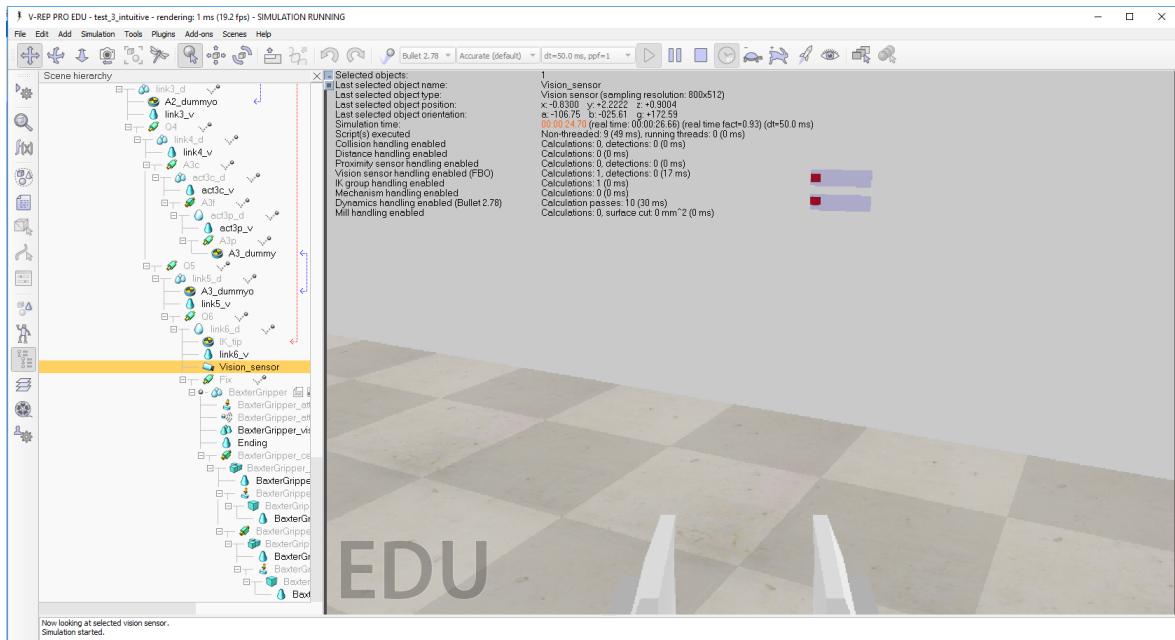
Rys. 5.1: Widok sceny zadania 1



Rys. 5.2: Widok po rozpoczęciu zadania 2

5.1.3. Obsługa panelu operatorskiego

Zadanie trzecie polega na przełączeniu dwóch suwaków znajdujących się na ścianie pomieszczenia, reprezentujących potencjometry liniowe. Zadanie to ma symulować jedno z zadań, które pojawia się na zawodach łażików marsjańskich, gdzie za pomocą manipulatora umieszczonego na łażiku należy wykonać określone czynności na panelu operatorskim. Oczywiście jedyny dostępny dla uczestnika badań widok to widok z kamery umieszczonej na manipulatorze. Na Rys. 5.3 przedstawiono to co uczestnik widzi w czasie wykonywania tego zadania.



Rys. 5.3: Widok na panel operatorski w zadaniu 3

Tab. 5.1: Czasy osiągnięte przez uczestników badań.

Zadanie: Uczestnik:	Sterowanie					
	intuicyjne			konwencjonalne		
1	2	3	1	2	3	
1	4m 03s	2m 02s	2m 14s	>5m	4m 16s	3m 28s
2	4m 52s	3m 10s	2m 24s	>5m	3m 27s	4m 13s
3	>5m	>5m	>5m	>5m	>5m	>5m
4	>5m	4m 10s	3m 47s	>5m	>5m	>5m
5	3m 58s	1m 58s	1m 30s	>5m	3m 22s	2m 42s

5.2. Testy weryfikacyjne

Do badań przystąpiło pięć osób o różnych doświadczeniach związanych z obsługą komputera i urządzeń wejścia. Uczestnik numer 1 to student mechatroniki, posiada dobrze opanowaną mysz 3D, ponieważ korzysta z niej podczas projektowania w oprogramowaniu CAD. Uczestnik nr 2 to uczeń szkoły podstawowej, nie znający jeszcze pojęć z dziedziny robotyki, ale mający duże doświadczenie w grach komputerowych. Uczestnik nr 3 to kobieta w średnim wieku, nie mająca wiele styczności z komputerami w ogóle. Uczestnik nr 4 to mężczyzna w średnim wieku, także nie mający wiele styczności z komputerami. Uczestnik nr 5 to autor niniejszej pracy. Na Rys. 5.4 znajduje się zdjęcie uczestnika nr 2 podczas wykonywania zadania drugiego z wykorzystaniem intuicyjnego sterowania.

Dla każdego uczestnika, przed przystąpieniem do właściwych zadań przysługuje bliżej nie określony czas na oswojenie się z systemem sterowania.

Po przeprowadzeniu badań oraz po analizie wyników przedstawionych w Tab. 5.1 można wysunąć kilka wniosków wymienionych poniżej.



Rys. 5.4: Uczestnik nr 2 podczas wykonywania drugiego zadania

1. Wykonanie pierwszego zadania z wykorzystaniem konwencjonalnego sterowania w określonym limicie czasu okazało się praktycznie niemożliwe, żaden z uczestników nie zdołał wykonać tego zadania.
2. Mimo utrudnienia, polegającego na dostępie do widoku tylko z kamery manipulatora, nie okazało się to dużą przeszkodą w wykonywaniu zadania drugiego i trzeciego.
3. Uczestnicy nie mający styczności na co dzień z komputerami mieli duże problemy z wykonywaniem zadań. Uczestnik nr 3, nie był w stanie w ogóle dokończyć zadania, bez znaczenia z zastosowaniem jakiego sterowania.
4. Okazało się, że doświadczenie w graniu w gry wideo okazało się pomocne w opanowywaniu ruchów potrzebnych do zadowalającego sterowania myszą 3D.
5. Ogólnie rzecz biorąc, uczestnicy mający wcześniej do czynienia z myszą 3D nie mieli problemów, by wykorzystać ją do realizowania zaplanowanych przez siebie ruchów. Natomiast uczestnicy nie mający wcześniej styczności, nie mieli szans nabraci przez krótki okres treningowy wyczucia pozwalającego na wygodne sterowanie.
6. Na podstawie wyników można wywnioskować, że sterowanie za pomocą myszy 3D pomaga zredukować czas potrzebny na wykonanie określonego zadania, przy sterowaniu manipulatorem. Należy zauważyć jednak, że potrzebny jest okres czasu pozwalający na przyzwyczajenie się do tego urządzenia.

Rozdział 6

Wnioski i podsumowanie

Jak pokazano w niniejszym raporcie, cel projektu został osiągnięty, ponieważ utworzono model symulacyjny robota mobilnego wraz z manipulatorem, środowisko z którym ten model może oddziaływać podczas symulacji, a także napisano aplikację sterującą z intuicyjnym, graficznym interfejsem użytkownika. Wykorzystano zadanie kinematyki odwrotnej w algorytmie sterowania i przeprowadzono badania dotyczące sprawowania się całego systemu podczas praktycznego użycia.

Podczas prac nasunęły się następujące wnioski:

1. Ilość pracy potrzebnej do utworzenia sensownego systemu sterowania jest ogromna. Praktycznie niemożliwe wydaje się utworzenie profesjonalnego, intuicyjnego systemu sterowania podczas pracy w pojedynkę. Zadaniem tym powinni zajmować się całe zespoły inżynierów, programistów, konstruktorów oraz testerów.
2. Utworzone w ramach środowiska symulacyjnego sceny, w których można testować sterowanie, w pewien przybliżony sposób odzwierciedlają rzeczywistość. Można za ich pomocą sprawdzać zachowanie się systemu sterowania w różnych sytuacjach. Ponadto oprogramowanie V-REP okazało się bardzo przyjemnym w użytkowaniu narzędziem, dającym możliwość tworzenia i programowania modeli w łatwy sposób.
3. Napisana z wykorzystaniem bibliotek Qt aplikacja spełnia założenia i działa bardzo sprawnie. Pozwala na sterowanie modelem manipulatora robota mobilnego za pomocą myszy 3D, a także udostępnia narzędzia do śledzenia przebiegu niektórych parametrów obiektów symulacji. Skorzystanie z biblioteki programistycznej Qt okazało się bardzo dobrym pomysłem, gdyż biblioteka ta zawiera wszystkie potrzebne elementy by w szybki sposób przygotować interfejs oraz napisać właściwy kod.
4. Podczas testowania utworzonego układu sterowania, okazało się, że przyjęta kinematyka manipulatora jest niezadowalająca. Po pierwsze, przyjęte zakresy ruchów poszczególnych przegubów, wynikające z zastosowania aktuatorów liniowych w łańcuchu kinematycznym, powodują powstanie ciasnej, nieoptymalnej przestrzeni roboczej, która znacznie utrudnia wykonywanie czynności manipulacyjnych. Po drugie, brak możliwości obracania przedostatnim przegubem w zakresie pełnego kąta, co spowodowane jest zastosowaniem aktuatora liniowego, powoduje duże problemy w osiąganiu pożąanej orientacji chwytaka, a czasami wręcz jest to niemożliwe.

Utworzony system sterowania, wraz ze środowiskiem symulacyjnym będzie stale ulepszany i połuży w pierwszej kolejności za platformę testową kolejnych algorytmów sterowania, a potem jako

platformę treningową dla operatora rzeczywistego robota mobilnego, który powstanie na konkurs ła-
zików marsjańskich. Samo użycie myszy 3D także zostanie zaimplementowane w układzie sterowania
rzeczywistego robota.

Rozdział 7

Użyte oprogramowanie

Podczas tworzenia niniejszej pracy wykorzystano:

- oprogramowanie V-REP PRO EDU V3.3.2 (darmowa licencja edukacyjna),
- środowisko programistyczne QtCreator w wersji 3.6.1 (licencja open-source LGPL v.3),
- biblioteka programistyczna Qt w wersji 5.7 (licencja open-source LGPL v.3),
- kompilator MSVC2015 64-bit (darmowa licencja edukacyjna),
- oprogramowanie Balsamiq Mockups w wersji 3.5.5 (30-dniowa licencja typu trial),
- edytor TeX TeXnicCenter w wersji 2.02 (licencja open-source LGPL v.3),

Rozdział 8

Streszczenie

Założeniem niniejszej pracy pt. "Opracowanie intuicyjnego interfejsu użytkownika do sterowaniem manipulatorem robota mobilnego" było zaprojektowanie oraz utworzenie aplikacji z graficznym interfejsem użytkownika, wykorzystującej mysz 3D oraz kinematykę odwrotną do sterowania modelem manipulatora robota mobilnego, utworzonego w oprogramowaniu symulacyjnym.

W ramach wykonywania tego zadania, wykorzystano darmowe oprogramowanie symulacyjne V-REP do utworzenia środowiska symulacyjnego oraz modelu robota mobilnego z manipulatorem. Zaprojektowano wygląd interfejsu aplikacji oraz sposób jej działania, a następnie utworzono aplikację sterującą, wykorzystując w tym celu bibliotekę programistyczną Qt. Jako mysz 3D, wykorzystano urządzenie firmy 3Dconnexion i zaimplementowano sterowniki obsługujące sygnały z tego urządzenia.

Działający system sterowania, pozwalający na intuicyjne sterowanie modelem manipulatora robota mobilnego, wykorzystujące mysz 3D oraz zadanie kinematyki odwrotnej, poddano badaniom weryfikacyjnym. Grupa uczestników złożona z ludzi, mających różne doświadczenia w obsłudze komputera, wykonała po trzy zadania polegające na wykorzystaniu manipulatora robota mobilnego do zadań manipulacyjnych. Wykorzystywano w tym celu dwa systemy sterowania: intuicyjne sterowanie kontrolerem 3D oraz konwencjonalne sterowanie z wykorzystaniem kinematyki prostej. Mierzono czas wykonywania każdego zadania. Rezultaty badań wskazują na przewagę stosowania sterowania intuicyjnego nad sterowaniem konwencjonalnym, co odzwierciedlane jest poprzez niższe czasy wykonywania zadania metodą intuicyjną.

Bibliografia

- [1] Strona internetowa aplikacji Balsamiq Mockups. <https://balsamiq.com/products/mockups/>.
- [2] Strona internetowa biblioteki Qt. <https://www.qt.io/>.
- [3] Strona internetowa dotycząca myszy 3D Space Navigator. <http://www.3dconnexion.pl/products/spacemouse/spacenavigator.html>.
- [4] Strona internetowa dotycząca robota Robotnik Summit XL. <http://www.robotnik.eu/mobile-robots/summit-xl-hl/>.
- [5] Strona internetowa firmy Rethink Robotics. <http://www.rethinkrobotics.com/>.
- [6] Strona internetowa konkursu „The University Rover Challenge”. <http://urc.marssociety.org/>.
- [7] Strona internetowa opisująca API programistyczne oprogramowania V-REP. <http://www.coppeliarobotics.com/helpFiles/en/apisOverview.htm>.
- [8] Reed B. Matthies L. Lavery D. Korsmeyer D. Ambrose R., Wilcox B. Robotics, tele-robotics and autonomous systems. technology area 04. National Aeronautics and Space Administration, 2012.
- [9] John J. Craig. *Wprowadzenie do robotyki, mechanika i sterowanie*. Wydawnictwo Naukowo-Techniczne, 1995.
- [10] Guizzo E. Darpa seeking to revolutionize robotic manipulation. <http://spectrum.ieee.org/automaton/robotics/robotics-software/darpa-arm-program>, 2010.
- [11] Jerzy Grębosz. *Symfonia C++ standard*. Edition 2000, 2008.
- [12] Franken G. H. Handelman, D. A., H. Komsuoglu. Agile and dexterous robot for inspection and eod operations. *Proceedings of SPIE*, wolumen 7692, 2010.
- [13] A. Mazur R. Muszyński K. Tchoń I. Dulęba, R. Hossa. *Manipulatory i Roboty Mobilne: modele, planowanie ruchu, sterowanie*. Akademicka Oficyna Wydawnicza PLJ, 2000.
- [14] Berkemeier M. Johnston J., Alberts J., Edwards J. Manipulator autonomy for eod robots. *Proceedings of the Army Science Conference*, 2008.
- [15] Piotr Przystałka. Autonomia manipulatorów. Opracowanie własne, Politechnika Śląska, 2013.
- [16] PWN. Słownik języka polskiego pwn, 2012.
- [17] Qt. Dokumentacja dla qt 5. <https://doc.qt.io/qt-5/>.
- [18] Coppelia Robotics. Dokumentacja dla v-rep 3. <http://www.coppeliarobotics.com/helpFiles/index.html>.
- [19] Jenifer Tidwell. *Projektowanie interfejsów. Sprawdzone wzorce projektowe*. Helion S.A, 2012.

- [20] M. Adamczyk M. Januszka D. Pająk W. Panfil P. Przystałka M. Targosz R. Wiglenda M. Wyleżoł W. Moczułski, W. Skarka. Rozwiązań konstrukcyjnych wielozadaniowych robotów mobilnych wykorzystujących zaawansowane technologie. problemy eksploatacji 3/2011. s. 131–137.