Instytut Informatyki, Automatyki i Robotyki

Seria: PREPRINTY nr 91/2011

Planowanie ruchu układów robotycznych z deficytem napędów

(rozprawa doktorska)

Adam Ratajczak

Promotor: prof. dr hab. inż. Krzysztof Tchoń

Słowa kluczowe:
- planowanie ruchu;
- układ z deficytem napędów;
- algorytm egalitarny;
- algorytm z priorytetowaniem zadań;
- endogeniczna przestrzeń konfiguracyjna.

Wrocław, 2011-11-25

POLITECHNIKA WROCŁAWSKA
WYDZIAŁ ELEKTRONIKI

ROZPRAWA DOKTORSKA

Planowanie ruchu układów robotycznych
z deficytem napędów

Adam Ratajczak

Promotor:
prof. dr hab. inż. Krzysztof Tchoń

WROCŁAW 2011

Na ręce Pana Profesora dr hab. inż. ***Krzysztofa Tchonia*** *składam wyrazy wdzięczności za wiele inspiracji, poświęcony czas, włożony trud i wielkie zaangażowanie.*

W tym miejscu dziękuję także ***Joannie*** *za cierpliwość i cenne uwagi.*

Na końcu, chciałem serdecznie podziękować ***Rodzicom****, od których wszystko się zaczęło.*

Spis treści

Oznaczenia

x	wektor stanu
q	wektor współrzędnych uogólnionych
y	wektor współrzędnych zadaniowych
u	wektor sterowań
n	$\dim q$ – wymiar przestrzeni współrzędnych uogólnionych
m	$\dim u$ – wymiar przestrzeni sterowań
r	$\dim y$ – wymiar przestrzeni zadaniowej
z	liczba zadań
k	liczność szeregu funkcji bazowych
T	maksymalny czas ruchu
x_0	stan początkowy
y_d	zadany punkt w przestrzeni zadaniowej
$\varphi_{x_0,t}(u(\cdot))$	strumień w chwili t układu poddanego sterowaniu $u(\cdot)$ przy warunku początkowym x_0
$\mathbb{R}^n$	przestrzeń euklidesowa wymiaru n
$\mathcal{U}$	endogeniczna przestrzeń konfiguracyjna
$G(x)$	macierz sterowania afinicznego układu sterowania
$f(x)$	dryf afinicznego układu sterowania
$\mathsf{A}(t), \mathsf{B}(t), \mathsf{C}(t)$	macierze przybliżenia liniowego
$\Phi(t,s)$	macierz fundamentalna przybliżenia liniowego
λ	wektor mnożników Lagrange'a
$\mathcal{L}$	lagranżian
$K_{x_0,T}$	odwzorowanie końcowe
$J_{x_0,T}$	jakobian
$J^{\#}_{x_0,T}$	pseudoodwrotność jakobianu
$J^{*}_{x_0,T}$	jakobian dołączony
e	błąd
$\mathsf{G}_{x_0,t}(u(\cdot))$	macierz Grama afinicznego układu sterowania
$P_{x_0,t}(u(\cdot))$	projekcja na jądro $\ker J_{x_0,T}(u(\cdot))$
$\mathrm{id}_{\mathcal{U}}$	identyczność w $\mathcal{U}$
$\widetilde{\mathcal{U}}$	parametryczna endogeniczna przestrzeń konfiguracyjna
c	wektor współczynników sterowania
$\psi(t)$	funkcja bazowa
$\Psi(t)$	macierz funkcji bazowych
$\omega = \frac{2\pi}{T}$	pulsacja trygonometrycznych funkcji bazowych
$\widetilde{K}_{x_0,T}$	parametryczne odwzorowanie końcowe
$\widetilde{J}_{x_0,T}$	parametryczny jakobian
$\widetilde{J}^{\#}_{x_0,T}$	parametryczna pseudoodwrotność jakobianu
$\widetilde{e}$	parametryczny błąd

γ	wektor współczynników zbieżności algorytmu
E	macierz skalująca błąd w algorytmie egalitarnym
S_i	i-te zadanie
r_i	wymiar przestrzeni zadaniowej i-tego zadania
${}^{i}K_{x_0,T}$	odwzorowanie końcowe i-tego zadania
${}^{i}J_{x_0,T}$	jakobian i-tego zadania
${}^{i}J^{\#}_{x_0,T}$	pseudoodwrotność jakobianu i-tego zadania
${}^{i}e$	błąd i-tego zadania
$\sigma(t)$	macierz wag algorytmu minimalizacji funkcji sterujących
$\delta(t)$	macierz wag algorytmu minimalizacji wartości zmiennych stanu
$h(y)$	funkcja przeszkód
ϵ	odchyłka parametru układu rzeczywistego w stosunku do układu nominalnego
I_a	macierz jednostkowa wymiaru $a \times a$
$0_{a\times b}$	macierz zerowa wymiaru $a \times b$
$\overline{R}R$	pozioma kreska ponad symbolem przegubu oznacza przegub pasywny

Rozdział 1

Wstęp

1.1 Wprowadzenie

Jeśli zdefiniujemy pojęcie robot jako urządzenie mechaniczne zdolne do wykonywania automatycznych czynności wspomagających pracę ludzką, to pierwsze wystąpienie robota w literaturze pojawia się w VIII-VII wieku p.n.e. w „Iliadzie" Homera. Tam, grecki bóg Hefajstos konstruuje na własny użytek mechanicznych pomocników, wykonanych ze złota, którzy wspomagają niepełnosprawnego boskiego kowala. Kilka wieków później, około IV wieku p.n.e, grecki filozof Arystoteles w słowach swojej rozprawy „Polityka"

> *Jest zaś narzędziem ponad narzędzia każdy sługa. Gdyby bowiem każde narzędzie mogło spełniać swoje zadania według rozkazu albo i uprzedzając go, jak to podobno robiły posągi Dedala lub trójnogi Hefajstosa, które, jak mówi poeta, same się zjawiają na zebranie bogów, gdyby tak czółenka tkackie same tkały, a pałeczki od gitary same grały, to ani budowniczowie nie potrzebowaliby pomocników, ani panowie niewolników.*
>
> tłum. L. Piotrowicz

mówi, jak dobrze byłoby mieć w pobliżu kilka robotów. Około II-I wieku p.n.e. powstają w starożytnej Grecji pierwsze konstrukcje automatów, głównie zegarów napędzanych wodą lub powietrzem, poruszające figurkami, służące celom rozrywkowym. W tym samym czasie, również w antycznych Chinach powstają podobne konstrukcje. Jedna z pierwszych książek „Book of Ingenious Devices" opisująca wiele mechanicznych automatów powstała w 850r. w Persji i została napisana po arabsku. Niestety, wraz z upadkiem starożytnych cywilizacji wiele z tych pomysłów nie zostało rozwiniętych. Ponowne zainteresowanie tematem odżywa w okresie renesansu. Autorem pierwszego projektu robota był Leonardo da Vinci, którego szkice zostały odnalezione w latach pięćdziesiątych ubiegłego stulecia. Od tego czasu, w całej Europie zaczynają powstawać różne konstrukcje robotyczne, głównie w celach rozrywkowych. W okresie II wojny światowej inżynierowie automatycy skupiają się wokół zastosowań militarnych, powstają wówczas między innymi skomplikowane maszyny szyfrujące i deszyfrujące. Po wojnie, rozwój techniki sprzyja rozwojowi robotyki. W roku 1954 powstaje pierwsze programowalne ramię robotyczne, a rok później pierwszy, elektronicznie sterowany, robot mobilny. Dziś robotyka przenika wiele aspektów życia człowieka, od robotów przemysłowych wspomagających pracę fabryk, poprzez na przykład roboty inspekcyjne, społeczne czy rozrywkowe, aż po roboty eksplorujące kosmos.

Robotem trzeciej generacji nazywamy maszynę wyposażoną w zdolności interpretacji i wnioskowania niezbędnych do rozpoznania i wykonania zadania poprzez inteligentne powiązanie percepcji i działania [43]. Zgodnie z terminologią Międzynarodowej Federacji

Teorii Maszyn i Mechanizmów (ang. *International Federation for the Theory of Machines and Mechanisms - IFToMM*) [16], stopień swobody (ang. *Degree of Freedom - DOF*) układu mechanicznego to liczba niezależnych współrzędnych uogólnionych potrzebnych do pełnego opisania konfiguracji układu w każdej chwili czasu. Biorąc pod uwagę liczbę stopni swobody n, w odniesieniu do liczby napędów (sterowań) m układy robotyczne można podzielić na trzy grupy [4]

- układy w pełni napędzane, $n = m$,
- układy z redundancją napędów, $n < m$,
- układy z deficytem napędów, $n > m$.

Reprezentantem pierwszej grupy może być na przykład robotyczny manipulator przemysłowy. W skład drugiej grupy wchodzą między innymi wielonapędowe kołowe roboty mobilne [23]. Układy z deficytem napędów, należące do trzeciej grupy, stanowią obiekt badań w niniejszej rozprawie. Rozwiązanie zadań takich jak planowanie ruchu, planowanie trajektorii, czy śledzenie trajektorii w klasie układów z deficytem napędów jest trudne. Niemniej jednak, wykorzystując charakterystyczne konstrukcje tych układów oraz używając specyficznych algorytmów staje się to możliwe.

Przyjrzyjmy się bliżej klasie układów z deficytem napędów. Zgodnie z definicją zawartą w [29], układem z deficytem napędów (ang. *underacutated system*) nazywamy układ robotyczny, w którym wymiar przestrzeni stanu przewyższa wymiar przestrzeni sterowań. W świetle tej definicji bardzo dużo układów robotycznych zawiera się w grupie układów z deficytem napędów. Wymienić można tutaj dla przykładu dynamikę wszelkiego rodzaju platform mobilnych (kołowych, nawodnych, podwodnych, latających, balansujących), dynamikę manipulatorów z pasywnymi przegubami lub elastycznymi ramionami, roboty kroczące w fazie lotu, układy, gdzie przyczyną deficytu napędów jest uszkodzenie silnika, roboty kosmiczne oraz wiele innych. Zainteresowanie i praca nad algorytmami dedykowanymi takim układom przyczynia się do uzyskania rozwiązania problemów, takich jak na przykład stabilizacja i sterowanie statków nawodnych i podwodnych, helikopterów, poduszkowców, satelit i innych. Napędy zazwyczaj są stosunkowo ciężkie i drogie, jeśli możliwe jest sterowanie układem z mniejszą liczbą silników, to przynosi to wymierne korzyści. Specyficzne algorytmy sterowania mogą okazać się przydatne w przypadku awarii napędu. Dla przykładu, jeśli na statku kosmicznym doszło do awarii napędu wieloprzegubowego manipulatora w momencie kiedy ramie było rozłożone, to jedynym sposobem na zamknięcie ładowni było odrzucenie całego ramienia w przestrzeń kosmiczną. Jeśli taki uszkodzony manipulator potraktujemy jako układ z deficytem napędów, możemy użyć odpowiedniego algorytmu, przy pomocy którego uda się złożyć manipulator i bezpiecznie wrócić na Ziemię.

W chwili obecnej, nie istnieje uniwersalny algorytm sterowania dla klasy układów z deficytem napędów. Niemniej jednak, od lat powstają różnego rodzaju algorytmy dedykowane konkretnym układom. Problem stabilizacji odwróconego wahadła był w obszarze zainteresowań już na początku XX wieku [50]. Również współcześnie ten temat jest obiektem badań naukowych [10,29,32,47]. Algorytmy sterowania manipulatorami z pasywnymi przegubami przedstawione są między innymi w [14,30,39], a planowanie ruchu takich układów opisane jest w [25,33,34,37]. Przegląd układów manipulatorów z pasywnymi przegubami oraz dedykowanych algorytmów zawierają prace [9,48]. Planowanie ruchu platform mobilnych przedstawia wiele pozycji literaturowych, między innymi [11,31,38,42].

W niniejszej rozprawie będziemy starali się rozwiązać zadanie planowania ruchu układów robotycznych z deficytem napędów. Poprzez zadanie planowania ruchu rozumiemy

znalezienie funkcji sterujących, które podane na układ przeprowadzą go z określonego punktu początkowego do zadanego punktu końcowego, w określonym czasie. Algorytm rozwiązujący zadanie planowania ruchu, jest algorytmem działającym w otwartej pętli sprzężenia zwrotnego. Uzyskane funkcje sterujące podane na układ nie będą uwzględniały niedokładności modelowania ani zakłóceń występujących podczas ruchu. Niemniej jednak, otrzymywane rozwiązania mogą stanowić dobry materiał referencyjny dla algorytmów sterowania, które z kolei mogą pracować w zamkniętej pętli sprzężenia zwrotnego. W pracy wyprowadzimy algorytm planowania ruchu dla układów robotycznych z deficytem napędów. Ze względu na różnoraką specyfikę takich układów, bardzo często zachodzi potrzeba uzyskania konkretnej jakości rozwiązania zadania planowania ruchu. Z tego powodu wprowadzimy dwa algorytmy planowania ruchu z zadaniami dodatkowymi: algorytm egalitarny i algorytm z priorytetowaniem zadań. Ruch układu pod wpływem funkcji sterujących otrzymanych w wyniku działania takich algorytmów będzie nie tylko realizował zadanie planowania ruchu, lecz także uwzględniał zestaw zadań dodatkowych. Idee działania algorytmu egalitarnego i algorytmu z priorytetowaniem zadań są komplementarne. Pierwszy z nich traktuje wszystkie zadania składowe równorzędnie. Drugi, szereguje zadania składowe ze względu na ich stopień ważności. Podamy także definicje kilku zadań dodatkowych. Przedstawimy zadanie ograniczania energii sterowania, minimalizacji wartości zmiennych stanu, zadanie unikania przeszkód oraz kilka modyfikacji tych zadań.

Do konstrukcji algorytmu planowania ruchu będziemy korzystać z metody kontynuacji (homotopii). Przykłady zastosowań tej metody w robotyce są przedstawione w pracy [54], a algorytmy rozwiązujące problemy planowania ruchu układów robotycznych, korzystające z metody kontynuacji można znaleźć w pracach [7,41,53]. Z metody kontynuacji wywodzi się także idea endogenicznej przestrzeni konfiguracyjnej [55], która ujednolica metodologie dotyczące manipulatorów robotycznych i układów mobilnych. Oryginalnie, metoda ta była dedykowana algorytmom planowania ruchu kinematyki manipulatorów mobilnych, w których nie występuje dryf w układzie sterowania. Zastosowanie metody endogenicznej przestrzeni konfiguracyjnej do układów z dryfem zostało przedstawione w [57], a pierwszy wynik użycia tej metody do planowania manipulatorów z pasywnym przegubem zawiera [37]. Konstrukcja algorytmu z priorytetowaniem zadań łączy ze sobą metodę endogenicznej przestrzeni konfiguracyjnej z koncepcją algorytmu jakobianu pseudoodwrotnego z projekcją [6,27], która pierwotnie była przeznaczona dla holonomicznych manipulatorów redundantnych. Podobne podejścia stosowane są także w [1,26], gdzie redundancja manipulatorów jest wykorzystywana do realizacji zadań dodatkowych. Algorytm egalitarny również korzysta z metody endogenicznej przestrzeni konfiguracyjnej i jest rozszerzeniem oryginalnego algorytmu planowania ruchu przedstawionego w [55] do realizacji zadania planowania ruchu z zadaniami dodatkowymi. Jak wspomnieliśmy, zastosowanie metody endogenicznej przestrzeni konfiguracyjnej pozwala wykorzystywać koncepcje oryginalnie stworzone dla manipulatorów do zadań z innymi układami robotycznymi, w tym z układami robotycznymi z deficytem napędów. Fakt ten będziemy wykorzystywać przy definicji zadań dodatkowych. Zadanie dodatkowe dotyczące ograniczania zmiennych przegubowych przedstawione w [24] posłuży nam do zdefiniowania zadania dodatkowego ograniczającego energię sterowania i minimalizującego wartości zmiennych stanu układu z deficytem napędów. Zadanie planowania ruchu z ograniczeniami na zmienne stanu i sterowania rozwiązuje także algorytm jakobianu zaburzonego, którego zastosowanie do manipulatorów mobilnych zawiera [18], a zastosowanie do układów z deficytem napędów przedstawiono w pracach [33,34]. Inne zastosowanie algorytmu planowania ruchu do ograniczania poślizgu robota mobilnego zawarto w [61]. Do definicji zadania unikania przeszkód (osobliwości) wykorzystamy podejście pola potencjału [21], a także zadanie unikania przeszkód przez

manipulator stacjonarny zdefiniowane w [22] i zadanie unikania osobliwości manipulatora wprowadzone w [60]. Pierwsza próba użycia algorytmu planowania ruchu korzystającego z metody endogenicznej przestrzeni konfiguracyjnej z unikaniem przeszkód pojawiła się w [56]. Wyprowadzane algorytmy będą algorytmami planowania ruchu z uwzględnieniem stanu układu nominalnego. Dodatkowo, zbadamy podejście planowania ruchu z uwzględnieniem stanu układu rzeczywistego przedstawione w pracy [19]. Wykorzystanie tej koncepcji pozwoli na uzyskanie algorytmu planowania ruchu przy niepełnej znajomości modelu.

Przedmiotem pracy jest wyprowadzenie algorytmu planowania ruchu w oparciu o metodę endogenicznej przestrzeni konfiguracyjnej dla układów robotycznych z deficytem napędów. Algorytm umożliwia rozwiązanie zadania planowania ruchu z zestawem zadań dodatkowych. Jego efektywność zilustrujemy wynikami symulacji komputerowych. Głównym celem pracy jest wykazanie następującej tezy

Stosując metodę endogenicznej przestrzeni konfiguracyjnej można skonstruować algorytm planowania ruchu układów robotycznych z deficytem napędów, realizujący różne zadania dodatkowe.

Dalsza część rozdziału 1 zawiera wprowadzenie do układów z deficytem napędów. Omówimy różnicę pomiędzy układem holonomicznym a nieholonomicznym, objaśnimy mechanizm powstawania ograniczeń nieholonomicznych oraz omówimy ich wpływ na modelowanie układów. Często do układów z deficytem napędów stosuje się częściowo linearyzujące sprzężenie zwrotne, które opisujemy w podrozdziale 1.2.2. Następnie, przedstawiamy afiniczny układ sterowania z funkcją wyjścia, a pod koniec rozdziału przeprowadzamy krótką dyskusję na temat sterowalności układów nieliniowych.
Rozdział 2 przedstawia wyprowadzenie teoretyczne algorytmu planowania. Na początku wprowadzimy preliminaria dotyczące endogenicznej przestrzeni konfiguracyjnej. Dalej zdefiniujemy algorytm realizujący zadanie właściwego planowania ruchu. Następnie omówimy dwa sposoby dodania zadań dodatkowych do zadania planowania. Pierwszy z nich polega na rozszerzeniu jakobianu i funkcji błędu, a drugi opiera się na definicji pseudoodwrotności z projekcją. W pierwszym przypadku zadanie główne i zadania dodatkowe są traktowane równorzędnie. W drugim przypadku zadania są uszeregowane zgodnie z ich priorytetami. Dalsza część tego rozdziału zawiera definicje zadań dodatkowych. Zdefiniujemy następujące zadania dodatkowe: zadanie minimalizacji energii sterowania, zadanie minimalizacji wartości zmiennych stanu oraz zadanie unikania przeszkód. Rozdział kończy omówienie algorytmu planowania ruchu przy niepełnej znajomości modelu.
Aspekty implementacyjne algorytmu planowanie ruchu zostały omówione w rozdziale 3. W celu przeprowadzenia obliczeń numerycznych proponujemy reprezentację funkcji sterujących poprzez zestaw funkcji bazowych oraz definiujemy dyskretną wersję algorytmu. Ponadto, przedstawiamy sposób rozwiązania zadania planowania ruchu bez parametrycznej reprezentacji funkcji sterujących.
Rozdział 4 przedstawia symulacyjne wyniki badań efektywności algorytmu. Pierwszy podrozdział zawiera symulacje planowania ruchu statku wraz z dodatkowym zadaniem minimalizacji energii sterowania. Kolejne wyniki przedstawiają planowanie ruchu robota balansującego z równoczesną minimalizacją zmiennej stanu (zmiennej odpowiadającej odchyleniu korpusu robota od osi pionowej) i ograniczaniem energii sterowania. Trzecia symulacja dotyczy planowania ruchu poduszkowca omijającego przeszkody. W podrozdziale 4.2 dokonujemy porównania metody algorytmu egalitarnego i algorytmu z priorytetowaniem zadań. W podrozdziale 4.3, można znaleźć wyniki planowania ruchu przy niepełnej zna-

jomości modelu. Podrozdział 4.4 zawiera wyniki symulacji przy nieskończeniewymiarowej reprezentacji funkcji sterujących. Ostatni podrozdział zawiera symulacje wykorzystujące zadanie ograniczania zmiennych stanu do realizacji zadania reprodukcji trajektorii. Rozdział 5 został poświęcony analizie otrzymanych wyników i podsumowaniu pracy. Dbając o przejrzystość wywodów, część materiału została zawarta w dodatkach. W dodatku A przedstawiamy wyprowadzenia wybranych własności operatora projekcji. Dodatek B zawiera definicje skończeniewymiarowych odwzorowań końcowych, jakobianów, błędów oraz projekcji niezbędnych do implementacji numerycznej przedstawianych algorytmów. Dodatek C zawiera pięć modeli układów robotycznych z deficytem napędów. Znajdują się tam trzy układy platform mobilnych: statek–kontenerowiec, robot balansujący i poduszkowiec, a także dwa manipulatory z pasywnym przegubem: układ nieplanarnego i planarnego dwuwahadła.

Niniejsza rozprawa jest efektem badań finansowanych częściowo przez Fundację na rzecz Nauki Polskiej, Politechnikę Wrocławską w ramach grantów statutowych oraz Ministerstwo Nauki i Szkolnictwa Wyższego ze środków na naukę w latach 2010–2012 jako projekt promotorski.

1.2 Układy robotyczne z ograniczeniami

Dynamika układu robotycznego definiuje zależność pomiędzy siłami zewnętrznymi działającymi na układ a uogólnionymi współrzędnymi położenia, prędkości i przyspieszenia. Niech $q \in \mathbb{R}^n$ będzie wektorem współrzędnych uogólnionych. Następnie, wprowadźmy funkcję Lagrange'a $\mathcal{L}(q, \dot{q})$ [58] jako różnicę energii kinetycznej i potencjalnej układu

$$\mathcal{L}(q, \dot{q}) = K(q, \dot{q}) - V(q).$$

Zasada Najmniejszego Działania Hamiltona prowadzi do uzyskania równań dynamiki układu w postaci równań Eulera-Lagrange'a [58]

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial \mathcal{L}(q, \dot{q})}{\partial \dot{q}} - \frac{\partial \mathcal{L}(q, \dot{q})}{\partial q} = \mathcal{F}, \tag{1.1}$$

gdzie $\mathcal{F}$ jest wektorem sił niepotencjalnych działających na układ, w skład których wchodzą oddziaływania sterujące, opory ruchu itp. Równania (1.1) można zapisać w postaci układu równań różniczkowych drugiego rzędu

$$\frac{\partial^2 \mathcal{L}(q, \dot{q})}{\partial \dot{q}^2}\ddot{q} + \frac{\partial^2 \mathcal{L}(q, \dot{q})}{\partial q \partial \dot{q}}\dot{q} - \frac{\partial \mathcal{L}(q, \dot{q})}{\partial q} = \mathcal{F}. \tag{1.2}$$

Energia kinetyczna układu przyjmuje postać formy kwadratowej z dodatniookreśloną i symetryczną macierzą formy

$$K(q, \dot{q}) = \frac{1}{2}\dot{q}^{\mathsf{T}} M(q)\dot{q},$$

Uwzględniając postać energii kinetycznej w (1.2) otrzymujemy równania

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + D(q) = \mathcal{F}, \tag{1.3}$$

które definiują ogólną postać równań dynamiki układu robotycznego, w których

- $M(q)$ jest symetryczną i dodatniookreśloną macierzą inercji,
- $C(q, \dot{q})\dot{q}$ jest wektorem zawierającym oddziaływania odśrodkowe i Coriolisa,

- $D(q) = \frac{\partial V(q)}{\partial q}$ jest wektorem sił grawitacji.

Jak wspomnieliśmy wektor $\mathcal{F}$ zawiera siły działające na układ. Niech wektor $\mathcal{F} = B(q)\tau - f(q, \dot{q}, t)$, wówczas funkcja $f(q, \dot{q}, t)$ oznacza wszystkie siły uogólnione działające na układ poza oddziaływaniem sterującym. Wprowadzając wektor $F(q, \dot{q}) = C(q, \dot{q})\dot{q} + D(q) + f(q, \dot{q}, t)$ zawierający, w zależności od robota, siły odśrodkowe, Coriolisa, wyporu, hydro/aerodynamiczne, potencjalne, w tym grawitacji itp., ogólne równania dynamiki (1.3) przyjmują bardziej zwięzłą postać

$$M(q)\ddot{q} + F(q, \dot{q}) = B(q)\tau, \tag{1.4}$$

gdzie $B(q)$ jest macierzą sterowania, a $\tau \in \mathbb{R}^m$ jest wektorem sygnałów sterujących.

Ruch układu robotycznego opisanego przez (1.4) może odbywać się w dowolnych kierunkach. Często, ze względu na charakter i specyficzną budowę układu robotycznego, na ruch nakłada się ograniczenia, zwane w mechanice analitycznej więzami. Ograniczenia takie mogą zostać wyrażone zarówno przy pomocy współrzędnych uogólnionych, jak i przy pomocy kolejnych pochodnych tych współrzędnych [44]

$$h\left(t, q, \dot{q}, \ldots, \overset{(j)}{q}\right) = 0, \qquad j = 1, 2, \ldots. \tag{1.5}$$

W zależności od rzędu pochodnej występującej w ograniczeniach (1.5) wyróżniamy kilka postaci ograniczeń. Jeśli ograniczają one możliwe konfiguracje układu (1.4) wówczas nazywamy je ograniczeniami położeniowymi lub geometrycznymi. Takie ograniczenia można przedstawić w postaci

$$h_0(q) = 0, \qquad h\colon \mathbb{R}^n \to \mathbb{R}^l, \qquad l < n. \tag{1.6}$$

Postać ograniczeń (1.6) może zostać wykorzystana do wyeliminowania z (1.4) pewnych współrzędnych zależnych. Poza ograniczeniami położeniowymi wprowadźmy ograniczenia pierwszego i drugiego rzędu. Ograniczenia pierwszego rzędu to ograniczenia nałożone na współrzędne uogólnione oraz na prędkości uogólnione układu (1.4)

$$h_1(q, \dot{q}) = 0, \qquad h_1\colon \mathbb{R}^n \to \mathbb{R}^l, \qquad l < n,$$

które pochodzą od kinematyki układu. Ograniczenia drugiego rzędu to ograniczenia nałożone nie tylko na współrzędne uogólnione i prędkości uogólnione, ale także na uogólnione przyspieszenia układu (1.4)

$$h_2(q, \dot{q}, \ddot{q}) = 0, \qquad h_1\colon \mathbb{R}^n \to \mathbb{R}^l, \qquad l < n, \tag{1.7}$$

które mają swoje korzenie w dynamice układu. Jeśli ograniczenia pierwszego rzędu mogą zostać scałkowane (można je przedstawić za pomocą różniczki po czasie z funkcji współrzędnych uogólnionych q) to wówczas są to ograniczenia holonomiczne pierwszego rzędu. W przeciwnym razie ograniczenia są ograniczeniami nieholonomicznymi [13, 40]. Często ograniczenia nieholonomiczne pierwszego rzędu są przedstawiane w postaci ograniczeń fazowych Pfaffa

$$A(q)\dot{q} = 0, \qquad \text{rank}A(q) = l, \tag{1.8}$$

gdzie $A_{l\times n}(q)$ jest macierzą Pfaffa. Ograniczenia w postaci Pffaffa (1.8) są holonomiczne jeśli istnieje odwzorowanie $h(q)$, $h\colon \mathbb{R}^n \to \mathbb{R}^l$ takie, że (1.8) są równoważne $h(q) = 0$ [58]. Formalnie, ten warunek może być zapisany następująco: istnieje pewna macierz $Q(q)_{l\times l}$,

$\det Q(q) \neq 0$, taka że $Q(q)A(q) = \frac{\partial h(q)}{\partial q}$. Podobnie, ograniczenia drugiego rzędu są ograniczeniami nieholonomicznymi drugiego rzędu jeśli nie mogą zostać scałkowane (nie można ich przedstawić, za pomocą różniczki po czasie z funkcji współrzędnych uogólnionych q i prędkości uogólnionych $\dot{q}$). W przeciwnym przypadku będą ograniczeniami holonomicznymi [13, 40]. W celu określenia, czy ograniczenia nieholonomiczne w modelowanym układzie są pierwszego czy drugiego rzędu albo, być może, są ograniczeniami holonomicznymi, zostały sformułowane warunki częściowej i pełnej całkowalności ograniczeń. Warunki całkowalności dedykowane specyficznym układom można znaleźć w [29, 32, 59], natomiast w [39] przedstawiono warunki całkowalności ograniczeń nieholonomicznych niezależne od modelu. Wykorzystując wprowadzone definicje ograniczeń możemy zdefiniować dwa rodzaje układów. Jeżeli ograniczenia w układzie (1.4) są holonomiczne to układ jest ***układem holonomicznym***. Jeżeli ograniczenia są nieholonomiczne wówczas układ jest ***układem nieholonomicznym*** [10, 15].

Definicja 1.1 (Układ z deficytem napędów) *Układem z deficytem napędów będziemy nazywali nieholonomiczny układ robotyczny z ograniczeniami drugiego rzędu, w którym liczba napędów jest mniejsza od liczby stopni swobody.*

1.2.1 Modelowanie układów z deficytem napędów

Załóżmy, że w układzie występuje deficyt napędów, zatem $m < n$. Dodatkowo, podzielmy wektor współrzędnych $q = (q_a, q_b)$, $q_a \in \mathbb{R}^m$, $q_b \in \mathbb{R}^{n-m}$. Niech q_a będą aktywnymi stopniami swobody, a q_b pasywnymi. Wówczas, równanie dynamiki (1.4) możemy zapisać w postaci

$$\underbrace{\begin{bmatrix} M_{aa}(q) & M_{ab}(q) \\ M_{ab}^{\mathrm{T}}(q) & M_{bb}(q) \end{bmatrix}}_{M(q)} \begin{pmatrix} \ddot{q}_a \\ \ddot{q}_b \end{pmatrix} + \underbrace{\begin{pmatrix} F_a(q,\dot{q}) \\ F_b(q,\dot{q}) \end{pmatrix}}_{F(q,\dot{q})} = \underbrace{\begin{bmatrix} B_a(q) \\ 0_{(n-m)\times m} \end{bmatrix}}_{B(q)} \tau. \tag{1.9}$$

Ostatnie $l = n - m$ równań z (1.9) stanowi zestaw ograniczeń nieholonomicznych drugiego rzędu postaci (1.7), które powinne być spełnione podczas ruchu układu.

1.2.2 Częściowo linearyzujące sprzężenie zwrotne

Dla całej klasy układów robotycznych z deficytem napędów można zdefiniować sprzężenie zwrotne linearyzujące aktywne stopnie swobody (ang. *collocated*) [48]. Wynika to z założenia, że macierz inercji układu jest dodatnio określona i symetryczna. Ponadto, dla części układów możemy zdefiniować także sprzężenie zwrotne linearyzujące pasywne stopnie swobody (ang. *noncollocated*) [48].

Sprzężenie zwrotne linearyzujące aktywne stopnie swobody Częściowo linearyzujące sprzężenie zwrotne linearyzujące współrzędne q_a może być utożsamiane z linearyzacją wejściowo–wyjściową dynamiki (1.9), z funkcją wyjścia równą

$$y_a = q_a \in \mathbb{R}^m.$$

Funkcja wyjścia y_a jest powiązana z sygnałami wejściowymi poprzez aktywne współrzędne uogólnione. Rozważmy zatem linearyzację wejściowo–wyjściową dla układu z deficytem napędów. Przyjrzyjmy się ostatnim $n - m$ równaniom z (1.9)

$$\begin{bmatrix} M_{ab}^T(q) & M_{bb}(q) \end{bmatrix} \begin{pmatrix} \ddot{q}_a \\ \ddot{q}_b \end{pmatrix} + F_b(q,\dot{q}) = 0. \tag{1.10}$$

Można z nich wyznaczyć $\ddot{q}_b$, które po wstawieniu do m pierwszych równań (1.9) i zamianie $\ddot{q}_a = u_a$, gdzie u_a jest nowym sterowaniem, definiuje częściowo linearyzujące sprzężenie zwrotne

$$\tau = B_a^{-1}(q)\left(\left(M_{aa}(q) - M_{ab}(q)M_{bb}^{-1}(q)M_{ab}^{\mathsf{T}}(q)\right)u_a + F_a(q,\dot{q}) - M_{ab}(q)M_{bb}^{-1}(q)F_b(q,\dot{q})\right).$$

Stosując powyższe sprzężenie zwrotne do dynamiki (1.9) otrzymujemy częściowo zlinearyzowany układ

$$\begin{cases} \ddot{q}_a = u_a, \\ \ddot{q}_b = -M_{bb}^{-1}(q)(M_{ab}^{\mathsf{T}}(q)u_a + F_b(q,\dot{q})). \end{cases}$$

Sprzężenie zwrotne linearyzujące pasywne stopnie swobody Częściowo linearyzujące sprzężenie zwrotne linearyzujące współrzędne q_b polega na wyznaczeniu linearyzacji wejściowo–wyjściowej dynamiki (1.9) z funkcją wyjścia

$$y_b = q_b \in \mathbb{R}^{n-m}, \tag{1.11}$$

korespondującą z pasywnymi stopniami swobody, które nie są bezpośrednio zależne od funkcji sterującej. Takie sprzężenie zwrotne może zostać wyznaczone przy założeniu, że układ linearyzowany jest silnie sprzężony inercyjnie.

Definicja 1.2 (Silne sprzężenie inercyjne [46]) *Układ (1.9) jest* ***silnie sprzężony inercyjnie*** *wtedy i tylko wtedy, gdy*

$$\mathrm{rank} M_{ab}(q) = n - m, \forall q \in \mathbb{R}^n.$$

Należy dodać, że własność silnego sprzężenia inercyjnego zakłada, że $m \geqslant n - m$, czyli liczba pasywnych zmiennych stanu nie może przewyższać liczby aktywnych zmiennych. Przy założeniu silnego sprzężenia inercyjnego możemy wyznaczyć pseudoodwrotność podmacierzy M_{ab}^{T}

$$\left(M_{ab}^{\mathsf{T}}\right)^{\#} = M_{ab}\left(M_{ab}^{\mathsf{T}}M_{ab}\right)^{-1}.$$

Korzystając z powyższej definicji możemy z (1.10) wyznaczyć $\ddot{q}_a$, następnie, podobnie jak poprzednio, wstawiając do (1.9) i zamieniając $\ddot{q}_b = u_b$, otrzymujemy częściowo linearyzujące sprzężenie zwrotne linearyzujące pasywne współrzędne uogólnione

$$\tau = B_a^{-1}(q)\left(\left(M_{ab}(q) - M_{aa}(q)(M_{ab}^{\mathsf{T}})^{\#}(q)M_{bb}(q)\right)u_b \right.$$
$$\left. + F_a(q,\dot{q}) - M_{aa}(q)(M_{ab}^{\mathsf{T}})^{\#}(q)F_b(q,\dot{q})\right).$$

Po zastosowaniu sterowania do układu (1.9) otrzymujemy układ

$$\begin{cases} \ddot{q}_a = -(M_{ab}^{\mathsf{T}})^{\#}(q)(M_{bb}(q)u_b + F_b(q,\dot{q})), \\ \ddot{q}_b = u_b, \end{cases}$$

w którym pasywne stopnie swobody zostały zlinearyzowane i odsprzężone od reszty układu, a pierwsze równanie, opisujące ruch aktywnych zmiennych, reprezentuje tutaj dynamikę wewnętrzną układu, korespondującą z równaniem wyjścia (1.11).

1.2.3 Afiniczny układ sterowania z funkcją wyjścia

Niezależnie od tego, czy dynamika układu robotycznego z deficytem napędów zostanie poddana częściowo linearyzującemu sprzężeniu zwrotnemu, czy też pozostanie w oryginalnej formie (1.9), można ją przekształcić w standardowy sposób do układu równań różniczkowych pierwszego rzędu. Dokonując zamiany zmiennych stanu $x = (x_1, x_2, x_3, x_4) = (q_a, q_b, \dot{q}_a, \dot{q}_b)$, $x \in \mathbb{R}^{2n}$, otrzymujemy afiniczny układ sterowania

$$\begin{cases} \dot{x}(t) = f(x(t)) + G(x(t))u(t) = f(x(t)) + \sum_{i=1}^{m} g_i(x(t))u_i(t), \\ y = k(x(t)), \end{cases} \tag{1.12}$$

który dodatkowo wyposażyliśmy w funkcję wyjścia. Dla dynamiki (1.9) dryf i macierz sterujących pól wektorowych w (1.12) są równe

$$f(x) = \begin{pmatrix} \begin{pmatrix} x_3 \\ x_4 \end{pmatrix} \\ M^{-1}(x)F(x) \end{pmatrix}, \qquad G(x) = \begin{bmatrix} 0_{n\times m} \\ M^{-1}(x)B(x) \end{bmatrix} = [g_1(x), \ldots, g_m(x)].$$

Stosując częściowo linearyzujące sprzężenie zwrotne, pola wektorowe w układzie (1.12) przyjmują postać

$$f(x) = \begin{pmatrix} \begin{pmatrix} x_3 \\ x_4 \end{pmatrix} \\ 0_{m\times 1} \\ -M_{bb}^{-1}(x)F_b(x) \end{pmatrix}, \qquad G(x) = \begin{bmatrix} 0_{n\times m} \\ I_m \\ -M_{bb}^{-1}(x)M_{ab}^{\mathsf{T}}(x) \end{bmatrix} = [g_1(x), \ldots, g_m(x)],$$

dla zlinearyzowanych współrzędnych q_a oraz

$$f(x) = \begin{pmatrix} \begin{pmatrix} x_3 \\ x_4 \end{pmatrix} \\ -(M_{ab}^{\mathsf{T}})^{\#}(x)F_b(x) \\ 0_{m\times 1} \end{pmatrix}, \qquad G(x) = \begin{bmatrix} 0_{n\times m} \\ -(M_{ab}^{\mathsf{T}})^{\#}(x)M_{ab}^{\mathsf{T}}(x) \\ I_m \end{bmatrix} = [g_1(x), \ldots, g_m(x)],$$

dla zlinearyzowanych współrzędnych q_b.

Układ postaci (1.12), opisujący dynamikę układu robotycznego z deficytem napędów będzie przedmiotem badań w niniejszej rozprawie.

1.2.4 Sterowalność układów z deficytem napędów

Przyjrzyjmy się zagadnieniu sterowalności układów robotycznych z deficytem napędów. Własności osiągalności i sterowalności są najważniejszymi własnościami układów sterowania. Osiągalność i sterowalność są równoważne dla układów liniowych. W układach nieliniowych, do których należą układy robotyczne z deficytem napędów, osiągalność nie zawsze implikuje sterowalność.

Niech $\mathcal{C}_{\text{traj}}(T)$ oznacza zbiór par $(x(t), u(t))$ będących trajektoriami sterowanymi układu (1.12) zdefiniowanymi na horyzoncie czasowym $[0, T]$. Ponadto, niech $\mathcal{R}_{x_0}(T)$ oznacza zbiór stanów osiągalnych, które mogą zostać osiągnięte przez trajektorie $\mathcal{C}_{\text{traj}}(T)$ układu (1.12) zapoczątkowane w x_0, zdefiniowany jako [5]

$$\mathcal{R}_{x_0}(T) = \{x(T) \mid (x(t), u(t)) \in \mathcal{C}_{\text{traj}}(T), x(0) = x_0\},$$
$$\mathcal{R}_{x_0}(\leqslant T) = \bigcup_{t\in[0,T]} \mathcal{R}_{x_0}(t).$$

Załóżmy, że prędkości w chwili $t = 0$ są zerowe, a więc stan $x_0 = (q_0, \dot{q}_0) = (q_0, 0)$ jest stanem spoczynkowym, ponadto załóżmy, że zachodzi zależność $f(x_0) = 0$.

Definicja 1.3 (Osiągalność [5]) *Układ (1.12) jest* osiągalny *ze stanu x_0 jeżeli, dla każdego zbioru sterowań $\mathcal{U} \ni u(\cdot)$ istnieje $T > 0$, takie że $\mathrm{int}(\mathcal{R}_{x_0}(\leqslant T)) \neq \emptyset$ dla każdego $t \in (0, T]$.*

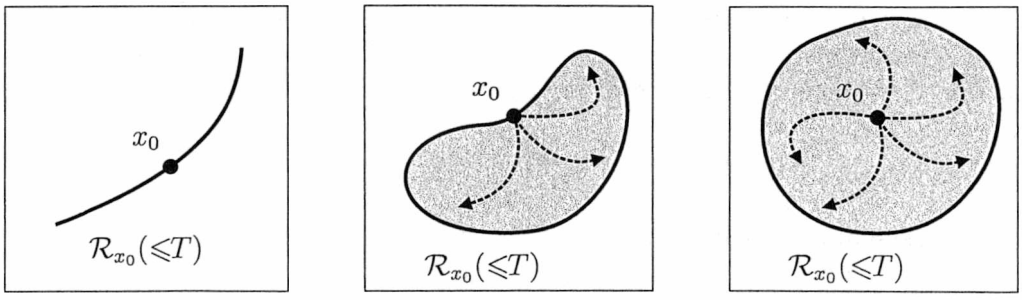
Definicja 1.4 (Sterowalność [5]) *Układ (1.12) jest* sterowalny *ze stanu x_0 jeżeli, dla każdego $x \in \mathbb{R}^{2n}$ istnieje $T > 0$ oraz $(x(t), u(t)) \in \mathcal{C}_{\mathrm{traj}}(T)$, takie że $x(0) = x_0$ oraz $x(T) = x$.*

Definicja 1.5 (Lokalna sterowalność w krótkim czasie [5]) *Układ (1.12) jest* lokalnie sterowalny w krótkim czasie *(ang.* Small–Time Local Controllability – STLC*) ze stanu x_0 jeżeli istnieje $T > 0$, takie że $x_0 \in \mathrm{int}(\mathcal{R}_{x_0}(\leqslant t))$ dla każdego $t \in (0, T]$*

Rysunek 1.1 przedstawia graficzną interpretację własności osiągalności i sterowalności układów. Dotychczas nie zostały opracowane warunki konieczne i wystarczające sterowalności układów nieliniowych. Istnieją jedynie warunki wystarczające sterowalności dedykowane specyficznym układom. Jedną z możliwości sprawdzenia, czy dany układ nieliniowy jest sterowalny, polega na zbadaniu sterowalności przybliżenia liniowego

$$\dot{z} = \mathsf{A}(t)z + \mathsf{B}(t)v,$$
$$\mathsf{A}(t) = \frac{\partial(f(x(t)) + G(x(t))u_0(t))}{\partial x}, \qquad \mathsf{B}(t) = G(x(t)),$$

tego układu, wokół trajektorii $x(t)$, wygenerowanej przez sterowanie $u_0(t)$ takiej, że $x(0) = x_0$ i $x(T) = x_d$. Jeżeli można wykazać, że przybliżenie liniowe jest sterowalne, to wówczas dla każdego stanu x bliskiego x_d istnieje sterowanie $u(t)$, które przeprowadza układ z x_0 do x w czasie T, zatem uzyskujemy lokalną sterowalność. Jeśli x_0 i u_0 odpowiadają punktowi równowagi $f(x_0) + G(x_0)u_0 = 0$, wówczas powyższe sprowadza się do wyznaczania sterowalności przybliżenia liniowego wokół punktu równowagi (x_0, u_0). Samo testowanie sterowalności układu liniowego opiera się na sprawdzeniu rzędu macierzy Kalmana, Hautusa lub Grama. W bezdryfowych układach sterowania, podobnie jak w układach liniowych, osiągalność implikuje sterowalność układu. Osiągalność, a zarazem sterowalność układu bezdryfowego testujemy warunkiem rzędu algebry Liego (ang. *Lie Algebra Rank Condition – LARC*). W ogólnych układach afinicznych występuje dryf i spełnienie warunku rzędu algebry Liego zapewnia jedynie osiągalność układu. Aby sprawdzić sterowalność układu afinicznego należy skorzystać z warunków wprowadzonych w [3, 52]. W tym celu, dla każdego nawiasu Liego v (pól $f, g_1, \ldots, g_m$) oznaczmy przez $\delta^0(v), \delta^1(v), \ldots, \delta^m(v)$



Rysunek 1.1 Interpretacja własności osiągalności i sterowalności. Od lewej: brak osiągalności, osiągalność bez sterowalności, lokalna sterowalność

liczbę wystąpień w v odpowiednio pól wektorowych $f(x), g_1(x), \ldots, g_m(x)$. Niech wektor $\theta = (\theta_0, \theta_1, \ldots, \theta_m)$ będzie dopuszczalnym wektorem wag, którego elementy są liczbami całkowitymi dodatnimi oraz $\theta_i \geqslant \theta_0$ dla każdego $i = 1, \ldots, m$. Stopień $\Theta(v)$ nawiasu Liego v zdefiniujemy jako $\Theta(v) = \sum_{i=0}^{m} \theta_i \delta^i(v)$. Wówczas, jeżeli dla każdego nawiasu Liego v, takiego że $\delta^0(v)$ jest nieparzyste, a $\delta^1(v), \ldots, \delta^m(v)$ są parzyste istnieje dopuszczalny wektor wag θ, taki że v jest Θ-neutralizowalny w punkcie x_0 (może zostać zapisany jako liniowa kombinacja nawiasów o niższym stopniu $\Theta(v)$), to system jest lokalnie sterowalny w krótkim czasie z punktu x_0. Istnieje wiele dedykowanych sposobów sprawdzania sterowalności. Na przykład, w pracy [51] przedstawiono test sterowalności dla układu afinicznego z dryfem, z jednym wejściem. Z kolei w [8] opracowano metodę sprawdzenia sterowalności dla układów z deficytem napędów, w których $n - m = 1$. Dotychczasowe wyniki dotyczące sterowalności układów nieliniowych zawsze zakładają, że stan początkowy jest stanem równowagi (zerowe prędkości) oraz że dryf zanika w $t = 0$. Kwestia sterowalności układów, w których nie zachodzą takie zależności jest wciąż otwarta.

Rozdział 2

Planowanie ruchu układów z deficytem napędów

Zadanie planowania ruchu polega na znalezieniu takich funkcji sterujących $u(t)$, które przeprowadzą układ (1.12) z pewnego położenia początkowego do zadanej pozycji końcowej w trakcie założonego horyzontu czasowego. Tak sformułowane zadanie jest równoważne zadaniu sterowania w otwartej pętli sprzężenia zwrotnego. Trajektorie układu znalezione przez algorytm planowania ruchu mogą następnie posłużyć jako funkcje referencyjne dla algorytmów sterowania w układzie zamkniętym.

W niniejszym rozdziale przedstawimy metodę endogenicznej przestrzeni konfiguracyjnej i algorytm planowania ruchu. Następnie omówimy dwie modyfikacje algorytmu planowania ruchu umożliwiające realizację dodatkowych zadań oraz podamy szczegółowe definicje zadań dodatkowych. W ostatnim podrozdziale zbadamy modyfikację algorytmu do rozwiązania zadania planowania ruchu przy niepełnej znajomości modelu.

2.1 Endogeniczna przestrzeń konfiguracyjna

Konfiguracją endogeniczną układu (1.12) nazwiemy dozwolone funkcje sterujące $u(\cdot) \in \mathcal{U}$ zdefiniowane na przedziale czasu $[0, T]$ [55]. Endogeniczna przestrzeń konfiguracyjna $\mathcal{U} = L_{2,m}[0, T]$ jest zbiorem funkcji całkowalnych z kwadratem tworzących przestrzeń Hilberta z iloczynem skalarnym

$$\langle u_1(\cdot), u_2(\cdot) \rangle_{\mathcal{U}} = \int_0^T u_1^{\mathsf{T}}(t) R(t) u_2(t)\, \mathrm{d}t, \qquad R(t) = R^{\mathsf{T}}(t) > 0$$

oraz odpowiadającą mu normą

$$\|u(\cdot)\|_{\mathcal{U}}^2 = \int_0^T u^{\mathsf{T}}(t) R(t) u(t)\, \mathrm{d}t.$$

2.1.1 Odwzorowanie końcowe i jakobian układu robotycznego z deficytem napędów

Zdefiniujmy $\varphi_{x_0,t}(u(\cdot))$ jako strumień układu (1.12) przy sterowaniu $u(\cdot)$, wyznaczony w chwili t i zainicjowany warunkiem początkowym $x(0) = x_0$. Tak więc, każdej konfiguracji endogenicznej $u(\cdot) \in \mathcal{U}$ odpowiada trajektoria stanu $x(t) = \varphi_{x_0,t}(u(\cdot))$ oraz odpowiadająca jej trajektoria wyjścia układu $y(t) = k(x(t))$.

Odwzorowanie końcowe

Odwzorowaniem końcowym nazwiemy funkcję przekształcającą przestrzeń endogeniczną w przestrzeń zadaniową układu. Wyznacza ono końcowe położenie wyjścia $y(T)$ układu robotycznego (1.12) w zależności od zadanej konfiguracji endogenicznej $u(\cdot) \in \mathcal{U}$

$$K_{x_0,T}\colon \mathcal{U} \to \mathbb{R}^r, \qquad K_{x_0,T}(u(\cdot)) = y(T) = k(\varphi_{x_0,T}(u(\cdot))), \tag{2.1}$$

Stosując analogię do robotów manipulacyjnych możemy zauważyć, że odwzorowanie końcowe odpowiada kinematyce manipulatora.

Jakobian

Podążając dalej za analogią zaczerpniętą z dziedziny manipulatorów robotycznych zdefiniujmy jakobian układu (1.12), który pozwala określić jak zmieni się wartość wyjścia $y(T)$ układu (1.12) w zależności od niewielkich przyrostów funkcji sterującej $u(t)$ więc $J_{x_0,T}(u(\cdot))\colon \mathcal{U} \to \mathbb{R}^r$. Podobnie, jak w przypadku manipulatorów, gdzie jakobian jest pochodną kinematyki, wyznaczymy jakobian różniczkując odwzorowanie końcowe (2.1)

$$J_{x_0,T}(u(\cdot))v(\cdot) = \mathcal{D}K_{x_0,T}(u(\cdot))v(\cdot) = \frac{\mathrm{d}}{\mathrm{d}\alpha}\bigg|_{\alpha=0} K_{x_0,T}(u(\cdot)+\alpha v(\cdot)) = \frac{\partial k(x(T))}{\partial x}\frac{\mathrm{d}}{\mathrm{d}\alpha}\bigg|_{\alpha=0} \varphi_{x_0,T}(u(\cdot)+\alpha v(\cdot)). \tag{2.2}$$

Korzystając z (1.12), wyliczmy pochodną strumienia

$$\begin{aligned}
&\frac{\mathrm{d}}{\mathrm{d}t}\frac{\mathrm{d}}{\mathrm{d}\alpha}\bigg|_{\alpha=0} \varphi_{x_0,t}(u(\cdot)+\alpha v(\cdot)) = \frac{\mathrm{d}}{\mathrm{d}\alpha}\bigg|_{\alpha=0} \frac{\mathrm{d}}{\mathrm{d}t}\varphi_{x_0,t}(u(\cdot)+\alpha v(\cdot)) = \\
&\frac{\mathrm{d}}{\mathrm{d}\alpha}\bigg|_{\alpha=0} \big(f(\varphi_{x_0,t}(u(\cdot)+\alpha v(\cdot))) + G(\varphi_{x_0,t}(u(\cdot)+\alpha v(\cdot)))(u(t)+\alpha v(t))\big) = \\
&\frac{\partial(f(x(t))+G(x(t))u(t))}{\partial x}\frac{\mathrm{d}}{\mathrm{d}\alpha}\bigg|_{\alpha=0} \varphi_{x_0,t}(u(\cdot)+\alpha v(\cdot)) + G(x(t))v(t).
\end{aligned} \tag{2.3}$$

Następnie, wprowadzając oznaczenie $\xi(t) = \frac{\mathrm{d}}{\mathrm{d}\alpha}\Big|_{\alpha=0} \varphi_{x_0,t}(u(\cdot)+\alpha v(\cdot))$ i warunek początkowy $\xi(0) = 0$, otrzymujemy wariacyjny układ stowarzyszony z układem (1.12) będący zarazem przybliżeniem liniowym układu (1.12) wzdłuż pary sterowanie–trajektoria $(u(t), x(t))$

$$\begin{cases} \dot{\xi} = \mathsf{A}(t)\xi + \mathsf{B}(t)v, \\ \zeta = \mathsf{C}(t)\xi. \end{cases} \tag{2.4}$$

Zgodnie z (2.3) macierze $\mathsf{A}(t)$ i $\mathsf{B}(t)$ są równe

$$\mathsf{A}(t) = \frac{\partial(f(x(t))+G(x(t))u(t))}{\partial x}, \qquad \mathsf{B}(t) = \frac{\partial(f(x(t))+G(x(t))u(t))}{\partial u} = G(x(t)),$$

a macierz $\mathsf{C}(t) = \frac{\partial k(x(t))}{\partial x}$. Rozwiązanie równania liniowego (2.4) przy warunku $\xi(0) = 0$ można wyrazić przez [45]

$$\xi(T) = \int_0^T \Phi(T,s)\mathsf{B}(s)v(s)\,\mathrm{d}s, \tag{2.5}$$

gdzie $\Phi(t,s)$ jest macierzą tranzycji [45] stanu układu (2.4) i może zostać wyznaczona z równania różniczkowego

$$\frac{\partial \Phi(t,s)}{\partial t} = \mathsf{A}(t)\Phi(t,s), \qquad \Phi(s,s) = I_{2n}.$$

Wstawiając rozwiązanie (2.5) do (2.2) można zapisać ostateczną postać jakobianu układu robotycznego z deficytem napędów i z funkcją wyjścia (1.12)

$$J_{x_0,T}(u(\cdot))v(\cdot) = \mathsf{C}(T)\int_0^T \Phi(T,s)\mathsf{B}(s)v(s)\,\mathrm{d}s. \tag{2.6}$$

Konfiguracja endogeniczna $u(\cdot) \in \mathcal{U}$ jest regularna, gdy jakobian $J_{x_0,T}(u(\cdot))\colon \mathcal{U} \to I\!R^r$ jest suriekcją. Pozostałe konfiguracje są konfiguracjami osobliwymi.

Pseudoodwrotność jakobianu

Jak wspomnieliśmy, jakobian przekształca prędkości $v(\cdot) \in \mathcal{U}$ z przestrzeni endogenicznej w prędkości $\eta \in I\!R^r$ w przestrzeni zadaniowej. Tak więc, słuszne jest równanie jakobianowe

$$J_{x_0,T}(u(\cdot))v(\cdot) = \mathsf{C}(T)\int_0^T \Phi(T,s)\mathsf{B}(s)v(s)\,\mathrm{d}s = \eta. \tag{2.7}$$

W regularnych konfiguracjach endogenicznych można rozwiązać powyższe równanie ze względu na zmienną $v(\cdot)$, co sprowadza się do wyznaczenia przekształcenia odwrotnego do $J_{x_0,T}(u(\cdot))v(\cdot)$. W tym celu użyjmy metody najmniejszych kwadratów polegającej na minimalizacji normy L_2.

$$\min_{v(\cdot)} \frac{1}{2}\int_0^T v^{\mathsf{T}}(t)v(t)\,\mathrm{d}t,$$

przy zachowanych ograniczeniach równościowych (2.7). Korzystając z metody mnożników Lagrange'a, wprowadźmy lagranżian

$$\mathcal{L}(v(\cdot),\lambda) = \frac{1}{2}\int_0^T v^{\mathsf{T}}(t)v(t)\,\mathrm{d}t + \lambda^{\mathsf{T}}\left(\mathsf{C}(T)\int_0^T \Phi(T,t)\mathsf{B}(t)v(t)\,\mathrm{d}t - \eta\right),$$

gdzie λ jest wektorem mnożników Lagrange'a. Przyrównując różniczkę lagranżianu do zera

$$\mathcal{D}\mathcal{L}(v(\cdot),\lambda)w(\cdot) = \frac{\mathrm{d}}{\mathrm{d}\alpha}\bigg|_{\alpha=0} \mathcal{L}(v(\cdot)+\alpha w(\cdot),\lambda) =$$
$$\frac{1}{2}\int_0^T v^{\mathsf{T}}(t)w(t)\,\mathrm{d}t + \lambda^{\mathsf{T}}\mathsf{C}(T)\int_0^T \Phi(T,t)\mathsf{B}(t)w(t)\,\mathrm{d}t = 0, \qquad \forall w(\cdot),$$

otrzymujemy relację

$$v(t) = -\mathsf{B}^{\mathsf{T}}(t)\Phi^{\mathsf{T}}(T,t)\mathsf{C}^{\mathsf{T}}(T)\lambda, \tag{2.8}$$

która wstawiona do (2.7) pozwala wyznaczyć mnożniki Lagrange'a

$$\lambda = -\left(\mathsf{C}(T)\int_0^T \Phi(T,s)\mathsf{B}(s)\mathsf{B}^{\mathsf{T}}(t)\Phi^{\mathsf{T}}(T,t)\,\mathrm{d}s\;\mathsf{C}^{\mathsf{T}}(T)\right)^{-1}\eta = -\mathsf{G}^{-1}_{x_0,T}(u(\cdot))\eta.$$

Wstawiając powyższy wynik do (2.8) otrzymujemy zależność

$$v(t) = \mathsf{B}^{\mathsf{T}}(t)\Phi^{\mathsf{T}}(T,t)\mathsf{C}^{\mathsf{T}}(T)\mathsf{G}^{-1}_{x_0,T}(u(\cdot))\eta,$$

która określa pseudoodwrotność (odwrotność Moore'a–Penrose'a) jakobianu

$$J^{\#}_{x_0,T}(u(\cdot))\colon \mathbb{R}^r \to \mathcal{U}, \qquad \left(J^{\#}_{x_0,T}(u(\cdot))\eta\right)(t) = \mathsf{B}^{\mathsf{T}}(t)\Phi^{\mathsf{T}}(T,t)\mathsf{C}^{\mathsf{T}}(T)\mathsf{G}^{-1}_{x_0,T}(u(\cdot))\eta, \quad (2.9)$$

gdzie

$$\mathsf{G}_{x_0,T}(u(\cdot)) = \mathsf{C}(T)\int_0^T \Phi(T,s)\mathsf{B}(s)\mathsf{B}^{\mathsf{T}}(s)\Phi^{\mathsf{T}}(T,s)\,\mathrm{d}s\;\mathsf{C}^{\mathsf{T}}(T)$$

jest macierzą Grama układu (2.4), która w zależności od układu, pełni rolę macierzy zręczności lub mobilności. Macierz Grama może posłużyć do wyznaczenia konfiguracji regularnych. Konfiguracja $u(\cdot) \in \mathcal{U}$ jest regularna wtedy i tylko wtedy, gdy macierz Grama jest pełnego rzędu $\operatorname{rank}\mathsf{G}_{x_0,T}(u(\cdot)) = r$. W regularnych konfiguracjach endogenicznych istnieje pseudoodwrotność (2.9) i układ sterowania (1.12) jest lokalnie sterowalny. Łatwo można pokazać, że pseudoodwrotność jakobianu spełnia zależność $J_{x_0,T}(u(\cdot))J^{\#}_{x_0,T}(u(\cdot)) = I_r$.

Jakobian dołączony

Do definicji algorytmów planowania ruchu niezbędny będzie także jakobian dołączony [55]

$$J^{*}_{x_0,T}(u(\cdot))\colon (\mathbb{R}^r)^* \to (\mathcal{U})^*.$$

Jakobian dołączony przekształca przestrzeń dualną do przestrzeni zadaniowej $(\mathbb{R}^r)^* \cong \mathbb{R}^r$ w dualną przestrzeń konfiguracyjną $(\mathcal{U})^* \cong \mathcal{U}$ zgodnie z wyrażeniem

$$\left\langle J^{*}_{x_0,T}(u(\cdot))\eta, v(\cdot)\right\rangle_{\mathcal{U}} = \eta^{\mathsf{T}} J_{x_0,T}(u(\cdot))v(\cdot), \quad (2.10)$$

gdzie $\eta \in \mathbb{R}^r$ i $v(\cdot) \in \mathcal{U}$. Możemy zatem, korzystając z (2.10), podać definicję jakobianu dołączonego

$$\left(J^{*}_{x_0,T}(u(\cdot))\eta\right)(t) = \mathsf{B}^{\mathsf{T}}(t)\Phi^{\mathsf{T}}(T,t)\mathsf{C}^{\mathsf{T}}(T)\eta. \quad (2.11)$$

Jakobian dołączony (2.11), podobnie jak pseudoodwrotność jakobianu (2.9) posłużą do wyprowadzenia algorytmu planowania ruchu.

Dzięki zastosowaniu metody endogenicznej przestrzeni konfiguracyjnej możliwe staje się zastosowanie podejścia znanego z manipulatorów robotycznych do innych układów, w tym układów robotycznych z deficytem napędów. W następnych rozdziałach pokażemy, jak skonstruować algorytmy planowania ruchu.

2.2 Zadanie planowania ruchu

Zadanie planowania ruchu polega na znalezieniu funkcji sterującej $u(\cdot) \in \mathcal{U}$, która przeprowadzi układ (1.12) z położenia $x(0) = x_0$ do zadanego punktu w przestrzeni zadaniowej $y(T) = y_d$, w zadanym czasie T. Tak postawione zadanie chcemy rozwiązać algorytmem jakobianowym wyprowadzonym przy użyciu metody kontynuacji [54,55]. W tym celu wybierzmy krzywą $u_\vartheta(\cdot)$ w endogenicznej przestrzeni konfiguracyjnej parametryzowaną przez $\vartheta \in \mathbb{R}$ i przechodzącą przez konfigurację (sterowanie) początkowe $u_0(\cdot) \in \mathcal{U}$. Następnie, wyznaczmy błąd wzdłuż tej krzywej

$$e(\vartheta) = K_{x_0,T}(u_\vartheta(\cdot)) - y_d. \quad (2.12)$$

Załóżmy, że błąd (2.12) będzie malał wykładniczo ze współczynnikiem $\gamma > 0$

$$\frac{\mathrm{d}e(\vartheta)}{\mathrm{d}\vartheta} = -\gamma e(\vartheta). \quad (2.13)$$

Podstawiając (2.12) do (2.13) otrzymujemy równanie Ważewskiego–Dawidenki [28, 55]

$$\frac{\mathrm{d}}{\mathrm{d}\vartheta}(K_{x_0,T}(u_\vartheta(\cdot)) - x_d) = J_{x_0,T}(u_\vartheta(\cdot))\frac{\mathrm{d}}{\mathrm{d}\vartheta}u_\vartheta(\cdot) = -\gamma e(\vartheta), \tag{2.14}$$

Do rozwiązania powyższego równania wykorzystamy odwrotność (2.9) jakobianu, a rozwiązanie uzyskamy jako granicę $\lim_{\vartheta\to\infty} u_\vartheta(\cdot)$ rozwiązania równania różniczkowego

$$\frac{\mathrm{d}u_\vartheta(t)}{\mathrm{d}\vartheta} = -\gamma\left(J^{\#}_{x_0,T}(u_\vartheta(\cdot))e(\vartheta)\right)(t). \tag{2.15}$$

Otrzymana trajektoria $u_{\vartheta\to\infty}(t)$ jest poszukiwaną funkcją sterującą rozwiązującą zadanie planowania ruchu.

Do wyznaczenia rozwiązania (2.14) można także użyć pseudoodwrotności z projekcją [55, 58], wówczas równanie (2.15) przyjmie postać

$$\frac{\mathrm{d}u_\vartheta(t)}{\mathrm{d}\vartheta} = -\gamma\left(J^{\#}_{x_0,T}(u_\vartheta(\cdot))e(\vartheta)\right)(t) + \left(P_{x_0,T}(u_\vartheta(\cdot))\mu_\vartheta(\cdot)\right)(t), \tag{2.16}$$

gdzie

$$P_{x_0,T}(u(\cdot)) = \mathrm{id}_{\mathcal{U}} - J^{\#}_{x_0,T}(u(\cdot))J_{x_0,T}(u(\cdot)) \tag{2.17}$$

jest rzutowaniem (projekcją) endogenicznej przestrzeni konfiguracyjnej $\mathcal{U}$ na jądro jakobianu $\ker J_{x_0,t}(u(\cdot))$, a $\mathrm{id}_{\mathcal{U}}$ jest odwzorowaniem identycznościowym. Symbol $\mu_\vartheta(\cdot) \in \mathcal{U}$ pojawiający się w (2.16) oznacza dowolny element w endogenicznej przestrzeni konfiguracyjnej i może zostać wykorzystany do rozwiązania zadań dodatkowych.

W pracy będziemy rozważać algorytmy planowania ruchu z zadaniami dodatkowymi. Takie algorytmy poza realizacją zadania głównego, którym jest właściwe planowanie ruchu, będą realizować zestaw zadań dodatkowych. Przez termin zadanie dodatkowe będziemy rozumieli zadanie, które będzie wykonywane obok właściwego planowania ruchu i będzie wpływać na jakość rozwiązania algorytmu planowania. Wśród zadań dodatkowych można wymienić między innymi: minimalizację energii sterowania, śledzenie trajektorii stanu czy wyjścia, unikanie przeszkód, itp. Wyprowadzimy dwa algorytmy planowania ruchu z zadaniami dodatkowymi: algorytm egalitarny i algorytm z priorytetowaniem zadań. Pierwszy z nich, algorytm egalitarny, będzie traktował wszystkie zadania równorzędnie. Odmiennie, algorytm z priorytetowaniem zadań będzie szeregował zadania ze względu na ich stopień ważności. Wprowadźmy terminologię systematyzującą zagadnienia związane z zadaniem głównym i zadaniami dodatkowymi. Niech $S = (S_1, S_2, \ldots, S_z)$ oznacza zbiór zadań, gdzie S_1 jest zadaniem właściwego planowania ruchu, a pozostałe $z-1$ zadań pełni rolę zadań dodatkowych.

Z wyprowadzeń algorytmu planowania ruchu (2.15) wynika, że dla zapisania algorytmu należy zdefiniować kolejno cztery elementy:

- odwzorowanie zadaniowe $K_{x_0,T}(u(\cdot))$,
- jakobian $J_{x_0,T}(u(\cdot))$,
- pseudoodwrotność jakobianu $J^{\#}_{x_0,T}(u(\cdot))$,
- błąd zadania e.

Podobnie, aby zdefiniować zadanie dodatkowe będziemy potrzebowali wyżej wymienionych elementów. Wówczas każde zadanie można opisać następująco $S_i = ({}^iK_{x_0,T}, {}^iJ_{x_0,T}, {}^iJ^{\#}_{x_0,T}, {}^ie)$.

2.2.1 Egalitarny algorytm planowania ruchu

Rozważmy konstrukcję algorytmu realizującego zadanie planowania ruchu z zestawem zadań dodatkowych. Wszystkie zadania, zarówno zadanie główne (planowanie ruchu) jak i zadania dodatkowe będą wykonywane równorzędnie. Załóżmy, że dysponujemy zestawem $S = (S_1, S_2, \ldots, S_z)$ zadań, dla których zostały wyznaczone cztery wspomniane wcześniej elementy. Niech błąd dla całego zadania planowania będzie zdefiniowany przez

$$e(\vartheta) = \begin{pmatrix} {}^1e(\vartheta) \\ {}^2e(\vartheta) \\ \vdots \\ {}^ze(\vartheta) \end{pmatrix}.$$

Zakładając eksponencjalną zbieżność błędu do zera $\frac{\mathrm{d}}{\mathrm{d}\vartheta}e(\vartheta) = -\gamma e(\vartheta)$, $\gamma \in \mathbb{R}$, zapiszmy równanie Ważewskiego-Dawidenki

$$\mathbf{J}_{x_0,T}(u_\vartheta(\cdot))\frac{\mathrm{d}u_\vartheta(\cdot)}{\mathrm{d}\vartheta} = -\gamma e(\vartheta), \tag{2.18}$$

gdzie $\mathbf{J}_{x_0,T}(u(\cdot))$ jest operatorem blokowym złożonym z jakobianów dla odpowiednich zadań

$$\mathbf{J}_{x_0,T}(u(\cdot)) = \begin{bmatrix} {}^1J_{x_0,T}(u(\cdot)) \\ {}^2J_{x_0,T}(u(\cdot)) \\ \vdots \\ {}^zJ_{x_0,T}(u(\cdot)) \end{bmatrix} \tag{2.19}$$

Stosując prawostronną odwrotność jakobianu (2.19) możemy rozwiązać równanie (2.18) otrzymując układ dynamiczny

$$\frac{\mathrm{d}u_\vartheta(t)}{\mathrm{d}\vartheta} = -\gamma\left(\mathbf{J}^{\#}_{x_0,T}(u_\vartheta(\cdot))e(\vartheta)\right)(t), \tag{2.20}$$

gdzie $\mathbf{J}^{\#}_{x_0,T}(u_\vartheta(\cdot))$ jest odwrotnością (2.19) daną wzorem

$$\mathbf{J}^{\#}_{x_0,T}(u_\vartheta(\cdot))\eta = \mathbf{J}^{*}_{x_0,T}(u(\cdot))\mathsf{G}^{-1}_{x_0,T}(u(\cdot))\eta = \begin{bmatrix} {}^1J_{x_0,T}(u(\cdot)) \\ {}^2J_{x_0,T}(u(\cdot)) \\ \vdots \\ {}^zJ_{x_0,T}(u(\cdot)) \end{bmatrix}^{*} \mathsf{G}^{-1}_{x_0,T}(u(\cdot))\eta,$$

a $\mathsf{G}_{x_0,T}(u(\cdot)) = \mathbf{J}_{x_0,T}(u(\cdot))\mathbf{J}^{*}_{x_0,T}(u(\cdot))$ jest macierzą Grama i $\mathbf{J}^{*}_{x_0,T}(u(\cdot))$ oznacza jakobian dołączony (2.11) jakobianu blokowego (2.19). Aby umożliwić dobranie wpływu poszczególnych zadań dodatkowych na rozwiązanie algorytmu należy wprowadzić następujące skalowanie błędu

$$\frac{\mathrm{d}u_\vartheta(t)}{\mathrm{d}\vartheta} = -\gamma\left(\mathbf{J}^{\#}_{x_0,T}(u_\vartheta(\cdot))Ee(\vartheta)\right)(t), \tag{2.21}$$

gdzie $E = \mathrm{blokdiag}\{{}^i\varepsilon I_{r_i}\}$, $i = 1, 2, \ldots, z$ jest blokowodiagonalną macierzą wag ${}^i\varepsilon$, a $r_i = \dim({}^iK_{x_0,T})$ jest wymiarem przestrzeni zadaniowej i-tego zadania.

Funkcję sterującą realizującą zadanie główne równorzędnie z zadaniami dodatkowymi uzyskujemy jako granicę $\lim_{\vartheta\to\infty} u_\vartheta(\cdot)$ rozwiązania równania (2.20) lub (2.21).

2.2.2 Algorytm planowania ruchu z priorytetowaniem zadań

Załóżmy, że zadania w zbiorze S zostały uszeregowane ze względu na ich stopień ważności, w taki sposób, że zadanie S_i ma wyższy priorytet niż zadanie S_{i+1}. Wyprowadzimy algorytm jakobianowy pozwalający na realizowanie z zadań i respektującego ich priorytety. Niech $r_i \in \mathbb{R}$ oznacza wymiar przestrzeni zadaniowej i-tego zadania. Wówczas ${}^{i}e(\vartheta) \in \mathbb{R}^{r_i}$ jest błędem zdefiniowanym dla zadania o indeksie i. Na podstawie (2.13) i (2.14) możemy zapisać równanie dla i-tego zadania przy sterowaniu $u_\vartheta(\cdot)$

$$\frac{\mathrm{d}}{\mathrm{d}\vartheta}{}^{i}e(\vartheta) = {}^{i}J_{x_0,T}(u_\vartheta(\cdot))\frac{\mathrm{d}u_\vartheta(\cdot)}{\mathrm{d}\vartheta}, \qquad i = 1, 2, \ldots, z. \tag{2.22}$$

Równanie (2.22) może zostać rozwiązane algorytmem jakobianu pseudoodwrotnego z projekcją (2.16)

$$\frac{\mathrm{d}u_\vartheta(t)}{\mathrm{d}\vartheta} = -{}^{i}\gamma\left({}^{i}J^{\#}_{x_0,T}(u_\vartheta(\cdot)){}^{i}e(\vartheta)\right)(t) + \left({}^{i}P_{x_0,T}(u_\vartheta(\cdot)){}^{i}\mu_\vartheta(\cdot)\right)(t), \tag{2.23}$$

gdzie

$${}^{i}P_{x_0,T}(u_\vartheta(\cdot)) = \mathrm{id}_{\mathcal{U}} - {}^{i}J^{\#}_{x_0,T}(u_\vartheta(\cdot)){}^{i}J_{x_0,T}(u_\vartheta(\cdot))$$

jest rzutowaniem na jądro jakobianu, $\ker {}^{i}J_{x_0,T}(u_\vartheta(\cdot))$. Dla dwóch dowolnych zadań, załóżmy dla S_1 oraz S_2, funkcja sterująca $u_\vartheta(t)$ musi być taka sama, zatem z (2.23) wynika zależność

$$-{}^{1}\gamma\left({}^{1}J^{\#}_{x_0,T}(u_\vartheta(\cdot)){}^{1}e(\vartheta)\right)(t) + \left({}^{1}P_{x_0,T}(u_\vartheta(\cdot)){}^{1}\mu_\vartheta(\cdot)\right)(t) = -{}^{2}\gamma\left({}^{2}J^{\#}_{x_0,T}(u_\vartheta(\cdot)){}^{2}e(\vartheta)\right)(t) + \left({}^{2}P_{x_0,T}(u_\vartheta(\cdot)){}^{2}\mu_\vartheta(\cdot)\right)(t). \tag{2.24}$$

Korzystając z własności idempotentności i symetrii rzutowania oraz anihilacji ${}^{i}J^{\#}_{x_0,T}(u_\vartheta(\cdot))$

$${}^{i}P_{x_0,T}(u_\vartheta(\cdot)){}^{i}P_{x_0,T}(u_\vartheta(\cdot)) = {}^{i}P_{x_0,T}(u_\vartheta(\cdot)), \qquad {}^{i}P^{*}_{x_0,T}(u_\vartheta(\cdot)) = {}^{i}P_{x_0,T}(u_\vartheta(\cdot)),$$
$${}^{i}P_{x_0,T}(u_\vartheta(\cdot)){}^{i}J^{\#}_{x_0,T}(u_\vartheta(\cdot)) = 0,$$

(patrz dodatek A) oraz mnożąc obustronnie (2.24) przez ${}^{1}P_{x_0,T}(u_\vartheta(\cdot))$ otrzymujemy

$$-{}^{2}\gamma\left({}^{1}P_{x_0,T}(u_\vartheta(\cdot)){}^{2}J^{\#}_{x_0,T}(u_\vartheta(\cdot)){}^{i+1}e(\vartheta)\right)(t) + \left({}^{1}P_{x_0,T}(u_\vartheta(\cdot)){}^{2}P_{x_0,T}(u_\vartheta(\cdot)){}^{2}\mu_\vartheta(\cdot)\right)(t) = \left({}^{1}P_{x_0,T}(u_\vartheta(\cdot)){}^{1}\mu_\vartheta(\cdot)\right)(t). \tag{2.25}$$

Ostatecznie, wstawiając (2.25) do (2.23), otrzymujemy algorytm dla dwóch zadań

$$\frac{\mathrm{d}u_\vartheta(t)}{\mathrm{d}\vartheta} = -{}^{1}\gamma\left({}^{1}J^{\#}_{x_0,T}(u_\vartheta(\cdot)){}^{1}e(\vartheta)\right)(t) - {}^{2}\gamma\left({}^{1}P_{x_0,T}(u_\vartheta(\cdot)){}^{2}J^{\#}_{x_0,T}(u_\vartheta(\cdot)){}^{2}e(\vartheta)\right)(t) + \left({}^{1}P_{x_0,T}(u_\vartheta(\cdot)){}^{2}P_{x_0,T}(u_\vartheta(\cdot)){}^{2}\mu_\vartheta(\cdot)\right)(t).$$

Powtarzając powyższy tok rozumowania oraz kładąc ${}^{z}\mu_\vartheta(\cdot) = 0$, możemy napisać algorytm jakobianu pseudoodwrotnego z projekcją dla z zadań z priotytetowaniem [36]

$$\frac{\mathrm{d}u_\vartheta(t)}{\mathrm{d}\vartheta} = -\left(\sum_{i=1}^{z}{}^{i}\gamma\left(\prod_{j=0}^{i-1}{}^{j}P_{x_0,T}(u_\vartheta(\cdot))\right){}^{i}J^{\#}_{x_0,T}(u_\vartheta(\cdot)){}^{i}e(\vartheta)\right)(t), \tag{2.26}$$

gdzie ${}^{0}P_{x_0,T}(u_\vartheta(\cdot)) = \mathrm{id}_{\mathcal{U}}$. Podobnie, jak to miało miejsce w poprzednich rozdziałach, rozwiązanie zadania planowania za pomocą algorytmu z priorytetowaniem otrzymuje się wyznaczając granicę $\lim_{\vartheta\to\infty} u_\vartheta(\cdot)$ rozwiązania równania (2.26).

2.3 Różne zadania dodatkowe

W niniejszym podrozdziale przedstawimy kilka zadań dodatkowych, które mogą być wykonywane poza głównym zadaniem planowania ruchu. W zależności od użytego algorytmu zadania dodatkowe mogą być równorzędne lub mieć niższy priorytet od zadania głównego. W kolejnych podrozdziałach wyprowadzimy cztery kluczowe składniki $S_i = ({}^iK_{x_0,T}, {}^iJ_{x_0,T}, {}^iJ^{\#}_{x_0,T}, {}^ie)$ dla różnych zadań dodatkowych.

2.3.1 Zadanie minimalizacji energii sterowania

W celu zdefiniowania zadania minimalizacji energii sterowania wybierzmy odwzorowanie zadaniowe

$$K_{x_0,T}\colon \mathcal{U} \to \mathbb{R}, \qquad K_{x_0,T}(u(\cdot)) = \frac{1}{2}\int_0^T u^{\mathsf{T}}(t)\sigma(t)u(t)\,\mathrm{d}t, \tag{2.27}$$

które wyznacza całkowitą, ważoną energię sterowania zużytą w czasie trwania ruchu, a $\sigma(t) = \mathrm{diag}\{\sigma_1(t), \ldots, \sigma_m(t)\}$, $\sigma_i(t) > 0$ jest diagonalną macierzą wag. Mając zdefiniowane odwzorowanie zadaniowe możemy wyznaczyć jakobian $J_{x_0,T}(u(\cdot))\colon \mathcal{U} \to \mathbb{R}$, który w tym przypadku określa wartości zmian kryterium (2.27) w zależności od zmian sterowań $u(\cdot)$. Jakobian wyznaczymy różniczkując (2.27)

$$J_{x_0,T}(u(\cdot))v(\cdot) = \mathcal{D}K_{x_0,T}(u(\cdot))v(\cdot) = \frac{\mathrm{d}}{\mathrm{d}\alpha}\bigg|_{\alpha=0} K_{x_0,T}(u(\cdot)+\alpha v(\cdot)) = \int_0^T u^{\mathsf{T}}(t)\sigma(t)v(t)\,\mathrm{d}t. \tag{2.28}$$

Podobnie, jak w podrozdziale 2.1.1 korzystając z metody najmniejszych kwadratów i metody mnożników Lagrange'a, wyznaczmy przekształcenie odwrotne do (2.28). Równanie jakobianowe

$$J_{x_0,T}(u(\cdot))v(\cdot) = \int_0^T u^{\mathsf{T}}(t)\sigma(t)v(t)\,\mathrm{d}t = \eta, \qquad \eta \in \mathbb{R}, \tag{2.29}$$

definiuje ograniczenia równościowe, przy zachowaniu których będziemy minimalizować normę, $\min_{v(\cdot)} \frac{1}{2}\int_0^T v^{\mathsf{T}}(t)v(t)\,\mathrm{d}t$. Lagranżian dla zadania minimalizacji energii sterowania jest równy

$$\mathcal{L}(v(\cdot),\lambda) = \frac{1}{2}\int_0^T v^{\mathsf{T}}(t)v(t)\,\mathrm{d}t + \lambda\left(\int_0^T u^{\mathsf{T}}(t)\sigma(t)v(t)\,\mathrm{d}t - \eta\right). \tag{2.30}$$

Następnie, wyliczając różniczkę (2.30) i przyrównując ją do zera

$$\mathcal{D}\mathcal{L}(v(\cdot),\lambda)w(\cdot) = \frac{\mathrm{d}}{\mathrm{d}\alpha}\bigg|_{\alpha=0} \mathcal{L}(v(\cdot)+\alpha w(\cdot),\lambda) = \int_0^T v^{\mathsf{T}}(t)w(t)\,\mathrm{d}t + \lambda\int_0^T u^{\mathsf{T}}(t)\sigma(t)w(t)\,\mathrm{d}t = 0, \qquad \forall w(\cdot),$$

otrzymujemy zależność

$$v(t) = -\lambda\sigma(t)u(t), \tag{2.31}$$

przy pomocy której wyznaczamy mnożnik Lagrange'a z (2.29) równy

$$\lambda = -\frac{\eta}{\|\sigma(\cdot)u(\cdot)\|^2_{\mathcal{U}}}.$$

Ostatecznie, wstawiając mnożnik Lagrange'a do (2.31) otrzymujemy pseudoodwrotność jakobianu dla zadania minimalizacji energii sterowania

$$J^{\#}_{x_0,T}(u(\cdot))\colon \mathbb{R} \to \mathcal{U}, \qquad \left(J^{\#}_{x_0,T}(u(\cdot))\eta\right)(t) = \frac{\sigma(t)u(t)}{\|\sigma(\cdot)u(\cdot)\|^2_{\mathcal{U}}}\eta.$$

Ostatnim elementem jest błąd dla bieżącego zadania, który jest równy odwzorowaniu zadaniowemu

$$e = K_{x_0,T}(u(\cdot)) = \frac{1}{2}\int_0^T u^T(t)\sigma(t)u(t)\,\mathrm{d}t.$$

Zadanie przedstawione powyżej użyte w algorytmie planowania ruchu jako zadanie dodatkowe pozwoli znaleźć funkcje sterujące cechujące się minimalnym zużyciem energii sterowania.

2.3.2 Zadanie minimalizacji wartości zmiennych stanu

Zadanie dodatkowe minimalizujące wartości zmiennych stanu utrzymuje część lub wszystkie zmienne stanu w pobliżu wartości zerowej. Podobnie jak w definicji poprzedniego zadania, zapiszmy odwzorowanie zadaniowe

$$K_{x_0,T}\colon \mathcal{U} \to \mathbb{R}, \qquad K_{x_0,T}(u(\cdot)) = \frac{1}{2}\int_0^T x^{\mathsf{T}}(t)\delta(t)x(t)\,\mathrm{d}t, \tag{2.32}$$

gdzie $\delta(t) = \mathrm{diag}\{\delta_1(t),\ldots,\delta_{2n}(t)\}$, $\delta_i(t) > 0$ jest diagonalną macierzą wag, a $x(t) = \varphi_{x_0,t}(u(\cdot))$. Operator jakobianu $J_{x_0,T}(u(\cdot))\colon \mathcal{U} \to \mathbb{R}$, dla zadania minimalizacji zmiennych stanu uzyskamy różniczkując (2.32)

$$J_{x_0,T}(u(\cdot))v(\cdot) = \mathcal{D}K_{x_0,T}(u(\cdot))v(\cdot) = \frac{\mathrm{d}}{\mathrm{d}\alpha}\bigg|_{\alpha=0} K_{x_0,T}(u(\cdot)+\alpha v(\cdot)) =$$
$$\frac{\mathrm{d}}{\mathrm{d}\alpha}\bigg|_{\alpha=0} \frac{1}{2}\int_0^T \varphi^{\mathsf{T}}_{x_0,t}(u(\cdot)+\alpha v(\cdot))\delta(t)\varphi_{x_0,t}(u(\cdot)+\alpha v(\cdot))\,\mathrm{d}t.$$

Korzystając z obliczeń (2.3) wyznaczymy pochodną strumienia $\varphi_{x_0,t}(u(\cdot))$, postać jakobianu możemy wówczas zapisać jako

$$J_{x_0,T}(u(\cdot))v(\cdot) = \int_0^T x^{\mathsf{T}}(t)\delta(t)\int_0^t \Phi(t,s)\mathsf{B}(s)v(s)\,\mathrm{d}s\,\mathrm{d}t = \int_0^T x^{\mathsf{T}}(t)\delta(t)\mathbb{J}_{x_0,t}(u(\cdot))v(\cdot)\,\mathrm{d}t,$$

gdzie $\mathbb{J}_{x_0,t}(u(\cdot))v(\cdot)$ jest jakobianem (2.6) układu robotycznego z deficytem napędów (1.12) bez funkcji wyjścia ($\mathsf{C}(t) = I_{2n}$). Przystąpmy teraz do wyznaczenia pseudoodwrotności $J_{x_0,T}(u(\cdot))$. Chcąc zminimalizować $\min_{v(\cdot)} \frac{1}{2}\int_0^T v^{\mathsf{T}}(t)v(t)\,\mathrm{d}t$ przy ograniczeniach równościowych zdefiniowanych przez równanie jakobianowe

$$J_{x_0,T}(u(\cdot))v(\cdot) = \int_0^T x^{\mathsf{T}}(t)\delta(t)\int_0^t \Phi(t,s)\mathsf{B}(s)v(s)\,\mathrm{d}s\,\mathrm{d}t =$$
$$\int_0^T\int_s^T x^{\mathsf{T}}(t)\delta(t)\Phi(t,s)\,\mathrm{d}t\,\mathsf{B}(s)v(s)\,\mathrm{d}s = \eta, \qquad \eta \in \mathbb{R} \tag{2.33}$$

zapiszmy lagranżian

$$\mathcal{L}(v(\cdot),\lambda) = \frac{1}{2}\int_0^T v^{\mathsf{T}}(s)v(s)\,\mathrm{d}s + \lambda\left(\int_0^T\int_s^T x^{\mathsf{T}}(t)\delta(t)\Phi(t,s)\,\mathrm{d}t\,\mathsf{B}(s)v(s)\,\mathrm{d}s - \eta\right)$$

i wyznaczmy jego pochodną

$$\mathcal{D}\mathcal{L}(v(\cdot),\lambda)w(\cdot) = \frac{\mathrm{d}}{\mathrm{d}\alpha}\bigg|_{\alpha=0} \mathcal{L}(v(\cdot)+\alpha w(\cdot),\lambda) = \int_0^T v^{\mathsf{T}}(s)w(s)\,\mathrm{d}s + \lambda\int_0^T\int_s^T x^{\mathsf{T}}(t)\delta(t)\Phi(t,s)\,\mathrm{d}t\,\mathsf{B}(s)w(s)\,\mathrm{d}s.$$

Przyrównując różniczkę lagranżianu do zera

$$\int_0^T v^{\mathsf{T}}(s)w(s)\,\mathrm{d}s + \lambda\int_0^T\int_s^T x^{\mathsf{T}}(t)\delta(t)\Phi(t,s)\,\mathrm{d}t\,\mathsf{B}(s)w(s)\,\mathrm{d}s = \int_0^T\left(v^{\mathsf{T}}(s) + \lambda\int_s^T x^{\mathsf{T}}(t)\delta(t)\Phi(t,s)\,\mathrm{d}t\,\mathsf{B}(s)\right)w(s)\,\mathrm{d}s = 0, \qquad \forall w(\cdot),$$

otrzymujemy zależność

$$v(s) = -\lambda\mathsf{B}^{\mathsf{T}}(s)\int_s^T \Phi^{\mathsf{T}}(t,s)\delta(t)x(t)\,\mathrm{d}t = -\lambda\alpha_x(u(\cdot))(s). \tag{2.34}$$

Wstawiając (2.34) do (2.33) wyznaczamy mnożnik Lagrange'a równy $\lambda = -\frac{\eta}{\|\alpha_x(u(\cdot))(\cdot)\|^2_{\mathcal{U}}}$, przy pomocy którego wyznaczamy odwrotność jakobianu dla zadania minimalizacji wartości stanu

$$J^{\#}_{x_0,T}(u(\cdot))\colon \mathbb{R} \to \mathcal{U}, \qquad \left(J^{\#}_{x_0,T}(u(\cdot))\eta\right)(t) = \frac{\alpha_x(u(\cdot))(t)}{\|\alpha_x(u(\cdot))(\cdot)\|^2_{\mathcal{U}}}\eta.$$

Pozostaje teraz wprowadzić błąd, który dla zadania minimalizacji zmiennych stanu wybieramy jako

$$e = K_{x_0,T}(u(\cdot)) = \frac{1}{2}\int_0^T x^{\mathsf{T}}(t)\delta(t)x(t)\,\mathrm{d}t.$$

Uogólnienie zadania minimalizacji wartości zmiennych stanu na zadanie minimalizacji wartości zmiennych zadaniowych

Modyfikując elementy składowe w powyższych wyprowadzeniach można przekształcić algorytm minimalizujący wartości zmiennych stanu w algorytm minimalizujący wartości zmiennych zadaniowych. Odwzorowanie końcowe wybieramy jako

$$K_{x_0,T}\colon \mathcal{U} \to \mathbb{R}, \qquad K_{x_0,T}(u(\cdot)) = \frac{1}{2}\int_0^T y^{\mathsf{T}}(t)\delta(t)y(t)\,\mathrm{d}t = \frac{1}{2}\int_0^T k^{\mathsf{T}}(x(t))\delta(t)k(x(t))\,\mathrm{d}t,$$

$\delta(t) = \mathrm{diag}\{\delta_1(t),\ldots,\delta_r(t)\}$, $\delta_i(t) > 0$ jest diagonalną macierzą wag, a $x(t) = \varphi_{x_0,t}(u(\cdot))$. Jakobian, wyliczony poprzez zróżniczkowanie odwzorowania końcowego, przyjmuje postać

$$J_{x_0,T}(u(\cdot))v(\cdot) = \int_0^T y^{\mathsf{T}}(t)\delta(t)\mathsf{C}(t)\int_0^t \Phi(t,s)\mathsf{B}(s)v(s)\,\mathrm{d}s\,\mathrm{d}t = \int_0^T y^{\mathsf{T}}(t)\delta(t)\mathbb{J}_{x_0,t}(u(\cdot))v(\cdot)\,\mathrm{d}t,$$

gdzie $\mathbb{J}_{x_0,t}(u(\cdot))v(\cdot)$ jest jakobianem (2.6) układu robotycznego (1.12) z deficytem napędów, z funkcją wyjścia. Stosując podobne wyliczenia, jak dla zadania minimalizacji wartości zmiennych stanu, definiujemy pseudoodwrotność jakobianu

$$J^{\#}_{x_0,T}(u_\vartheta(\cdot))\colon \mathbb{R} \to \mathcal{U}, \qquad \left(J^{\#}_{x_0,T}(u_\vartheta(\cdot))\eta\right)(t) = \frac{\alpha_y(u(\cdot))(t)}{\|\alpha_y(u(\cdot))(\cdot)\|^2_{\mathcal{U}}}\eta,$$

gdzie $\alpha_y(u(\cdot))(s) = \mathsf{B}^{\mathsf{T}}(s)\int_s^T \Phi^{\mathsf{T}}(t,s)\mathsf{C}^{\mathsf{T}}(t)\delta(t)x(t)\,\mathrm{d}t$ oraz równanie błędu

$$e = K_{x_0,T}(u(\cdot)) = \frac{1}{2}\int_0^T y^{\mathsf{T}}(t)\delta(t)y(t)\,\mathrm{d}t.$$

Alternatywny wariant algorytmu minimalizacji wartości zmiennych stanu

W poprzednich wyprowadzeniach odwzorowanie końcowe przyjmuje wartości liczbowe, a odpowiadający mu jakobian i pseudoodwrotność jakobianu są wektorami. Przyjrzyjmy się teraz przypadkowi, w którym zdefiniujemy odwzorowanie końcowe jako wektor $K_{x_0,T}\colon \mathcal{U} \to I\!R^{2n}$, którego elementy będą odpowiadać kolejnym współrzędnym stanu

$$K_{x_0,T}(u(\cdot)) = \begin{pmatrix} K_{x_0,T,1}(u(\cdot)) \\ K_{x_0,T,2}(u(\cdot)) \\ \vdots \\ K_{x_0,T,2n}(u(\cdot)) \end{pmatrix} = \frac{1}{2} \begin{pmatrix} \int_0^T x_1^2(t)\ \mathrm{d}t \\ \int_0^T x_2^2(t)\ \mathrm{d}t \\ \vdots \\ \int_0^T x_{2n}^2(t)\ \mathrm{d}t \end{pmatrix}.$$

Wprowadzając macierz $\mathbf{M}_i = \mathbf{e}_i\mathbf{e}_i^{\mathsf{T}}$, gdzie $\mathbf{e}_i \in I\!R^{2n}$ jest i-tym wersorem, możemy zapisać $x_i^2 = x^{\mathsf{T}}\mathbf{M}_i x$. Dla tak określonego zadania jakobian przyjmie postać

$$J_{x_0,T}(u(\cdot))v(\cdot) = \int_0^T \begin{bmatrix} x^{\mathsf{T}}(t)\mathbf{M}_1 \\ x^{\mathsf{T}}(t)\mathbf{M}_2 \\ \vdots \\ x^{\mathsf{T}}(t)\mathbf{M}_{2n} \end{bmatrix} \mathbb{J}_{x_0,t}(u(\cdot))v(\cdot)\ \mathrm{d}t = \int_0^T \mathrm{diag}\{x_1(t), x_2(t), \ldots, x_{2n}(t)\}\mathbb{J}_{x_0,t}(u(\cdot))v(\cdot)\ \mathrm{d}t,$$

gdzie $\mathbb{J}_{x_0,t}(u(\cdot))v(\cdot)$ jest jakobianem (2.6) układu robotycznego z deficytem napędów (1.12) bez funkcji wyjścia ($\mathsf{C}(t) = I_{2n}$). Pseudoodwrotność jakobianu dla zmodyfikowanego zadania minimalizacji wartości zmiennych stanu definiujemy jako

$$J^{\#}_{x_0,T}(u_\vartheta(\cdot))\colon I\!R^n \to \mathcal{U}, \qquad \left(J^{\#}_{x_0,T}(u_\vartheta(\cdot))\eta\right)(t) = \frac{\beta(u(\cdot))(t)}{\|\beta(u(\cdot))(\cdot)\|^2_{\mathcal{U}}}\eta,$$

gdzie $\beta(u(\cdot))(s) = \mathsf{B}^{\mathsf{T}}(s)\int_s^T \Phi(t,s)\mathrm{diag}\{x_1(t), x_2(t), \ldots, x_{2n}(t)\}\ \mathrm{d}t$. Błąd definiujemy jak poprzednio, za pośrednictwem odwzorowania końcowego

$$e = K_{x_0,T}(u(\cdot)) = \begin{pmatrix} \int_0^T x_1^2(t)\ \mathrm{d}t \\ \int_0^T x_2^2(t)\ \mathrm{d}t \\ \vdots \\ \int_0^T x_{2n}^2(t)\ \mathrm{d}t \end{pmatrix}.$$

Tak zdefiniowane zadanie można również zmodyfikować, aby minimalizowało wartości zmiennych w przestrzeni zadaniowej.

Modyfikacja zadania minimalizacji wartości zmiennych stanu do realizacji zadania śledzenia trajektorii

Dokonując niewielkich modyfikacji w definicjach zadań dodatkowych przedstawionych w tym podrozdziale można uzyskać zadanie, które będzie starało się utrzymywać wybrane lub wszystkie zmienne stanu w okolicach zadanej trajektorii odniesienia. Przedstawmy teraz niezbędne przekształcenia poprzednich wyprowadzeń w celu uzyskania zadania śledzenia trajektorii odniesienia. Rozpoczniemy od zdefiniowania odwzorowania końcowego

$$K_{x_0,T}\colon \mathcal{U} \to I\!R, \qquad K_{x_0,T}(u(\cdot)) = \frac{1}{2}\int_0^T (x(t) - x_d(t))^{\mathsf{T}}\delta(t)(x(t) - x_d(t))\ \mathrm{d}t, \qquad (2.35)$$

gdzie $x(t) = \varphi_{x_0,t}(u(\cdot))$, $x_d(t)$ jest pożądaną trajektorią, a $\delta(t) = \mathrm{diag}\{\delta_1(t), \ldots, \delta_{2n}(t)\}$, $\delta_i(t) > 0$ jest diagonalną macierzą wag. Różniczkując (2.35) uzyskujemy jakobian

$$J_{x_0,T}(u(\cdot))v(\cdot) = \int_0^T (x(t) - x_d(t))^{\mathsf{T}}\delta(t)\int_0^t \Phi(t,s)\mathsf{B}(s)v(s)\,\mathrm{d}s\,\mathrm{d}t = \int_0^T (x(t) - x_d(t))^{\mathsf{T}}\delta(t)\mathbb{J}_{x_0,t}(u(\cdot))v(\cdot)\,\mathrm{d}t, \tag{2.36}$$

gdzie $\mathbb{J}_{x_0,t}(u(\cdot))v(\cdot)$ jest jakobianem (2.6) układu robotycznego z deficytem napędów (1.12) bez funkcji wyjścia ($\mathsf{C}(t) = I_{2n}$). Następnie, przeprowadzając wyliczenia podobne do (2.33)-(2.34) wyznaczamy pseudoodwrotność jakobianu (2.36)

$$J^{\#}_{x_0,T}(u_\vartheta(\cdot))\colon \mathbb{R} \to \mathcal{U}, \qquad \left(J^{\#}_{x_0,T}(u(\cdot))\eta\right)(t) = \frac{\alpha_{\mathrm{traj}}(u(\cdot))(t)}{\|\alpha_{\mathrm{traj}}(u(\cdot))(\cdot)\|^2_{\mathcal{U}}}\eta, \tag{2.37}$$

gdzie $\alpha_{\mathrm{traj}}(u(\cdot))(t) = \mathsf{B}^{\mathsf{T}}(t)\int_t^T \Phi^{\mathsf{T}}(s,t)\delta(t)(x(s) - x_d(s))\,\mathrm{d}s$. Ostatnim, czwartym elementem niezbędnym do zdefiniowania algorytmu śledzenia trajektorii jest błąd, który w tym przypadku definiujemy jako

$$e = K_{x_0,T}(u(\cdot)) = \frac{1}{2}\int_0^T (x(t) - x_d(t))^{\mathsf{T}}\delta(t)(x(t) - x_d(t))\,\mathrm{d}t. \tag{2.38}$$

Zamieniając definicję wyrażenia $(x(t) - x_d(t))$ na $(y(t) - y_d(t))$, modyfikując $\delta(t)$ i wstawiając odpowiednią macierz $C(t) = \frac{\partial k(x(t))}{\partial x}$ do jakobianu $\mathbb{J}_{x_0,t}(u(\cdot))v(\cdot)$, otrzymujemy zadanie śledzenia trajektorii we współrzędnych zadaniowych.

2.3.3 Zadanie unikania przeszkód (osobliwości) w przestrzeni zadaniowej i w przestrzeni stanu

Do opisu zadania unikania przeszkód będzie nam potrzebna funkcja opisująca rozmieszczenie, rozmiar i kształt przeszkód. Zakładamy, że funkcja przeszkód będzie przyjmowała wartości liczbowe $h\colon \mathbb{R}^r \to \mathbb{R}$. Tę funkcję należy wybrać tak, aby jej wartość na obszarach wolnych od przeszkód była bliska zeru i przyjmowała bardzo duże wartości w miejscach występowania przeszkód oraz była funkcją dodatnią. Funkcję taką można utożsamiać z rozkładem natężenia pola grawitacyjnego (potencjalnego), a ruch ma się odbywać w sposób minimalizujący wypadkową siłę tego pola [21]. Alternatywnie, powyższą funkcję można traktować jako ukształtowanie terenu i zapewniać ruch minimalizujący wysokość nad poziomem morza. Przytoczone przykłady interpretacji przedstawione są dla płaszczyzny dwuwymiarowej, nic nie stoi jednak na przeszkodzie, aby definiowane tutaj zadanie unikania przeszkód było określone w przestrzeni wielowymiarowej.

Jedną z możliwych funkcji przeszkód zdefiniujemy w języku teorii pola grawitacyjnego (potencjalnego). Funkcja przeszkód będzie opisywać przeszkody okrągłe, czyli przeszkody, które mogą być utożsamiane z wielowymiarową kulą, której oddziaływanie na robota zwiększa się wraz ze zmniejszaniem odległości od jej środka. Niech o oznacza liczbę przeszkód, a $O_i = (m_i, y_i)$ oznacza i-tą przeszkodę, gdzie m_i jest wagą (promieniem) przeszkody a $y_i \in \mathbb{R}^r$ jest położeniem przeszkody we współrzędnych zadaniowych. Wówczas, funkcja przeszkód wyraża się przez

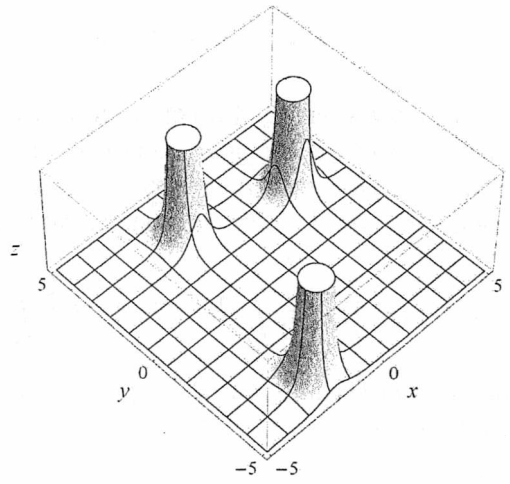
$$h(y) = \sum_{i=1}^{o} \frac{m_i}{\|y - y_i\|^2},$$

a jej przykładowy wykres, dla przestrzeni dwuwymiarowej, przedstawia rysunek 2.1. Często przydaje się możliwość ograniczenia obszaru ruchu. Chcemy, aby ruch odbywał się wewnątrz wielowymiarowego prostopadłościanu. Korzystając z funkcji eksponencjalnych, zdefiniujemy funkcję przeszkód, która będzie przyjmowała wartości bliskie zeru wewnątrz prostopadłościanu i przyjmowała duże wartości w pozostałych obszarach $\mathbb{R}^r$. Składając funkcje pojedynczych zmiennych w jedną wspólną funkcję, możemy utworzyć funkcję ograniczającą obszar ruchu do wielowymiarowego obszaru prostopadłościennego

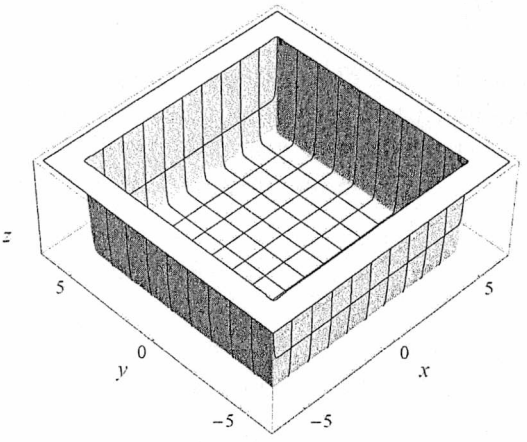
$$h(y) = \left\| \left(\exp((y_1 - y_{c_1})^2 - y_{r_1}^2), \ldots, \exp((y_r - y_{c_r})^2 - y_{r_r}^2) \right) \right\|^2,$$

gdzie $y_c \in \mathbb{R}^r$ są współrzędnymi środka prostopadłościanu, a $y_r \in \mathbb{R}^r$ są połowami długości boków prostopadłościanu wzdłuż każdej współrzędnej zadaniowej. Przykład funkcji ograniczającej ruch w przestrzeni R^2 ilustruje rysunek 2.2. Jeszcze inną funkcją przeszkód może być funkcja reprezentująca przeszkodę w kształcie podkowy, którą możemy zapisać jako

$$h(y) = \exp(-(\|y\|^2 - R^2)^2)\|y - Re_i\|^2,$$

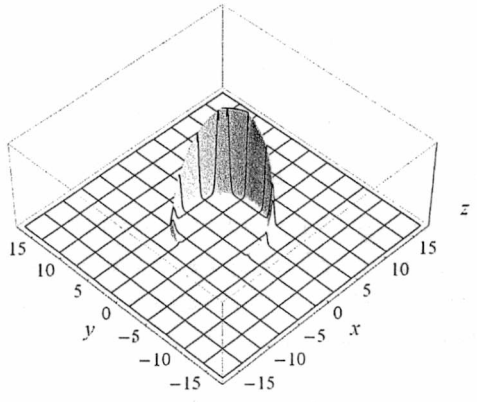


Rysunek 2.1 Przykładowa funkcja przeszkód okrągłych



Rysunek 2.2 Przykładowa funkcja przeszkód ograniczająca obszar ruchu

gdzie R jest promieniem przeszkody, a przy pomocy wersora e_i możemy zdefiniować, w którym miejscu przeszkoda ma być otwarta. Rysunek 2.3 pokazuje przykładowy wykres powyższej funkcji.



Rysunek 2.3 Przykładowa funkcja przeszkody w kształcie podkowy

Przyjrzyjmy się teraz, jak przy pomocy funkcji przeszkód $h(y)$ zdefiniować odwzorowanie końcowe, jakobian i inne elementy potrzebne do zapisania algorytmu unikania przeszkód. Jako odwzorowanie końcowe wybierzemy wartość funkcji przeszkód wzdłuż trajektorii $y(t) = k(x(t))$ wyznaczonej w całym okresie trwania ruchu

$$K_{x_0,T}\colon \mathcal{U} \to \mathbb{R}, \qquad K_{x_0,T}(u(\cdot)) = \frac{1}{2}\int_0^T h(y(t))\,\mathrm{d}t = \frac{1}{2}\int_0^T h(k(x(t)))\,\mathrm{d}t. \tag{2.39}$$

Korzystając z wyliczeń pochodnej strumienia (2.3), poprzez różniczkowanie (2.39) uzyskujemy jakobian $J_{x_0,T}(u(\cdot))\colon \mathcal{U} \to \mathbb{R}$

$$J_{x_0,T}(u(\cdot))v(\cdot) = \mathcal{D}K_{x_0,T}(u(\cdot))v(\cdot) = \frac{\mathrm{d}}{\mathrm{d}\alpha}\bigg|_{\alpha=0} K_{x_0,T}(u(\cdot)+\alpha v(\cdot)) = \frac{\mathrm{d}}{\mathrm{d}\alpha}\bigg|_{\alpha=0} \frac{1}{2}\int_0^T h(y(t))\,\mathrm{d}t =$$
$$\int_0^T \frac{\partial h(y(t))}{\partial y}\mathsf{C}(t)\int_0^t \Phi(t,s)\mathsf{B}(s)v(s)\,\mathrm{d}s\,\mathrm{d}t = \int_0^T \frac{\partial h(y(t))}{\partial y}\mathbb{J}_{x_0,t}(u(\cdot))v(\cdot)\,\mathrm{d}t, \tag{2.40}$$

gdzie $\mathbb{J}_{x_0,t}(u(\cdot))v(\cdot)$ jest jakobianem (2.6) układu robotycznego (1.12) z deficytem napędów i z funkcją wyjścia. Jakobian (2.40) definiuje zależność pomiędzy zmianami sterowania $u(\cdot) \in \mathcal{U}$ a zmianami wartości funkcji przeszkód wzdłuż trajektorii $y(t)$. Wyznaczmy pseudoodwrotność jakobianu dla zadania unikania przeszkód. Niech równanie jakobianowe

$$J_{x_0,T}(u(\cdot))v(\cdot) = \int_0^T \frac{\partial h(y(t))}{\partial y}\mathbb{J}_{x_0,t}(u(\cdot))v(\cdot)\,\mathrm{d}t = \eta, \qquad \eta \in \mathbb{R}, \tag{2.41}$$

będzie definiowało ograniczenia równościowe, przy respektowaniu których będziemy wyznaczać $\min\limits_{v(\cdot)} \frac{1}{2}\int_0^T v^{\mathsf{T}}(t)v(t)\,\mathrm{d}t$. W celu minimalizacji użyjemy metody mnożników Lagrange'a, co wiąże się ze zdefiniowaniem lagranżianu

$$\mathcal{L}(v(\cdot),\lambda) = \frac{1}{2}\int_0^T v^{\mathsf{T}}(s)v(s)\,\mathrm{d}s + \lambda\left(\int_0^T \frac{\partial h(y(t))}{\partial y}\mathsf{C}(t)\int_0^t \Phi(t,s)\mathsf{B}(s)v(s)\,\mathrm{d}s\,\mathrm{d}t - \eta\right),$$

i wyznaczeniu jego punktu stacjonarnego poprzez przyrównanie pochodnej do zera

$$\mathcal{D}\mathcal{L}(v(\cdot),\lambda)w(\cdot) = \frac{\mathrm{d}}{\mathrm{d}\alpha}\bigg|_{\alpha=0} \mathcal{L}(v(\cdot)+\alpha w(\cdot),\lambda) = \int_0^T v^{\mathsf{T}}(s)w(s)\,\mathrm{d}s + \lambda \int_0^T \frac{\partial h(y(t))}{\partial y}\mathsf{C}(t)\int_0^t \Phi(t,s)\mathsf{B}(s)w(s)\,\mathrm{d}s\,\mathrm{d}t = 0.$$

Stosując podobne przekształcenia jak w (2.33), dotyczące zmiany granic całkowania, uzyskujemy równość

$$v(s) = -\lambda \mathsf{B}^{\mathsf{T}}(s)\int_s^T \Phi^{\mathsf{T}}(t,s)\mathsf{C}^{\mathsf{T}}(t)\left(\frac{\partial h(y(t))}{\partial y}\right)^{\mathsf{T}}\mathrm{d}t = -\lambda\alpha(u(\cdot))(s), \tag{2.42}$$

która po podstawieniu do (2.41), pozwala wyznaczyć mnożnik Lagrange'a równy $\lambda = -\frac{\eta}{\|\alpha(u(\cdot))(\cdot)\|^2_{\mathcal{U}}}$. Eliminacja mnożników Lagrange'a λ w (2.42) prowadzi do uzyskania pseudoodwrotności jakobianu (2.40)

$$J^{\#}_{x_0,T}(u(\cdot))\colon \mathbb{R} \to \mathcal{U}, \qquad \left(J^{\#}_{x_0,T}(u(\cdot))\eta\right)(t) = \frac{\alpha(u(\cdot))(t)}{\|\alpha(y(\cdot))(\cdot)\|^2_{\mathcal{U}}}\eta.$$

Podobnie jak to miało miejsce w definicjach poprzednich zadań, także tutaj wybieramy funkcję błędu równą odwzorowaniu końcowemu

$$e = K_{x_0,T}(u(\cdot)) = \frac{1}{2}\int_0^T h(y(t))\,\mathrm{d}t = \frac{1}{2}\int_0^T h(k(x(t)))\,\mathrm{d}t.$$

Parametry, takie jak m_i, y_i, y_c, R, itp. pojawiające się w definicjach funkcji przeszkód $h(y)$ można uzależnić od czasu t, co spowoduje, że uzyskamy zadanie omijające ruchome przeszkody.

Jeśli w powyższych wyprowadzeniach założymy, że funkcja wyjścia układu (1.12) jest odwzorowaniem identycznościowym $y(t) = k(x(t)) = x(t)$ możemy zdefiniować funkcje przeszkód równe $h(y) = h(k(x)) = h(x)$, które mają zastosowanie w zadaniu omijania przeszkód w przestrzeni stanu. Jeśli teraz wybierzemy funkcję przeszkód $h(x)$, która będzie przyjmowała wartości zerowe w obszarach wolnych od osobliwości oraz duże wartości w miejscach występowania osobliwości, otrzymamy zadanie unikania osobliwości.

Przedstawioną tutaj ideologię można wykorzystać także do konstrukcji zadania, które będzie wymuszało, aby układ przejeżdżał przez wybrane punkty. Można to zrobić, definiując funkcję przeszkód w taki sposób, aby przyjmowała wartości bliskie zeru w wybranych punktach oraz odpowiednio wysokie wartości w pozostałym obszarze ruchu.

2.4 Planowanie przy niepełnej znajomości modelu

Algorytmy planowania ruchu robotów przedstawione w powyższych podrozdziałach działają w otwartej pętli sprzężenia zwrotnego. Ponadto, poszukiwane funkcje sterujące są wyznaczane na podstawie odpowiedzi nominalnego układu robotycznego. Zastosowanie wyznaczonych funkcji sterujących do układu rzeczywistego może skutkować stosunkowo dużą wartością błędu pomiędzy punktem zadanym y_d, a faktycznymi wartościami współrzędnych zadaniowych $y(T)$ w układzie rzeczywistym. Sposobem na poprawienie takiego stanu rzeczy jest użycie algorytmu planowania ruchu z uwzględnieniem stanu układu rzeczywistego [19]. Zaadaptowanie tej metody pozwoli na planowanie ruchu pomimo niepełnej znajomości parametrów modelu. Taka strategia będzie wyznaczała dla każdego ϑ poprawkę funkcji sterującej korzystając z wartości błędu zmierzonego na podstawie odpowiedzi układu rzeczywistego, oraz z pseudoodwrotności jakobianu wyznaczonej z modelu nominalnego.

Załóżmy, że układ (1.12) reprezentuje układ rzeczywisty, natomiast układ

$$\begin{cases} \dot{\widehat{x}}(t) = \widehat{f}(\widehat{x}(t)) + \widehat{G}(\widehat{x}(t))u(t), \\ \widehat{y} = k(\widehat{x}(t)). \end{cases} \tag{2.43}$$

będzie reprezentował układ nominalny, w którym wartości parametrów obiektu (takie jak masa, długość itp.) różnią się od parametrów w układzie rzeczywistym (1.12). W celu zmodyfikowania algorytmu (2.15) w taki sposób, aby mógł planować ruch przy niepełnej znajomości modelu zaproponujmy następującą formułę wyznaczenia sterowania

$$\frac{\mathrm{d}u_\vartheta(t)}{\mathrm{d}\vartheta} = -\gamma \left(\widehat{J}^{\#}_{x_0,T}(u_\vartheta(\cdot))e(\vartheta) \right)(t). \tag{2.44}$$

Jak wspomnieliśmy, wartość błędu będzie wyznaczana z układu rzeczywistego, a więc błąd $e(\vartheta)$ będzie zdefiniowany zgodnie z (2.12). Zmianie ulegnie natomiast definicja jakobianu

$$\left(\widehat{J}^{\#}_{x_0,T}(u(\cdot))\eta \right)(t) = \widehat{\mathsf{B}}^{\mathsf{T}}(t)\widehat{\Phi}^{\mathsf{T}}(T,t)\widehat{\mathsf{C}}^{\mathsf{T}}(T)\widehat{\mathsf{G}}^{-1}_{x_0,T}(u(\cdot))\eta,$$

gdzie

$$\widehat{\mathsf{G}}_{x_0,T}(u(\cdot)) = \widehat{\mathsf{C}}(T) \int_0^T \widehat{\Phi}(T,s)\widehat{\mathsf{B}}(s)\widehat{\mathsf{B}}^{\mathsf{T}}(s)\widehat{\Phi}^{\mathsf{T}}(T,s)\,\mathrm{d}s\mathsf{C}^{\mathsf{T}}(T)$$

jest macierzą Grama. Macierze $\widehat{\mathsf{A}}(t)$, $\widehat{\mathsf{B}}(t)$ i $\widehat{\mathsf{C}}(t)$ są macierzami przybliżenia liniowego układu nominalnego (2.43) wyznaczonymi wzdłuż pary sterowanie-trajektoria $(u(t), x(t))$ układu rzeczywistego (1.12) i są zdefiniowane następująco

$$\widehat{\mathsf{A}}(t) = \left. \frac{\partial(\widehat{f}(\widehat{x}(t)) + \widehat{G}(\widehat{x}(t))u(t))}{\partial \widehat{x}} \right|_{\widehat{x}(t)=x(t)},$$

$$\widehat{\mathsf{B}}(t) = \left. \widehat{G}(\widehat{x}(t)) \right|_{\widehat{x}(t)=x(t)}, \qquad \widehat{\mathsf{C}}(t) = \left. \frac{\partial k(\widehat{x}(t))}{\partial \widehat{x}} \right|_{\widehat{x}(t)=x(t)}.$$

Macierz $\widehat{\Phi}(t,s)$ jest macierzą tranzycji stanu przybliżenia liniowego układu (2.43) i można ją wyznaczyć z rozwiązania równania

$$\frac{\partial \widehat{\Phi}(t,s)}{\partial t} = \widehat{\mathsf{A}}(t)\widehat{\Phi}(t,s), \qquad \widehat{\Phi}(s,s) = I_{2n}.$$

Rozdział 3

Aspekty implementacyjne

W celu umożliwienia implementacji komputerowej algorytmu planowania ruchu układów robotycznych z deficytem napędów korzystającego z metody endogenicznej przestrzeni konfiguracyjnej niezbędne są dwie rzeczy: parametryczna reprezentacja funkcji sterujących oraz numeryczna metoda rozwiązania równania algorytmu. W pierwszej kolejności zajmiemy się reprezentacją konfiguracji endogenicznych. Dążąc do zredukowania czasu obliczeń wprowadzimy skończeniewymiarową, parametryczną reprezentację wykorzystującą bazę funkcji ortogonalnych. W dalszej części tego rozdziału omówimy dyskretną metodę uzyskania rozwiązywania algorytmu planowania ruchu. Pod koniec rozdziału przedstawimy podejście, które pozwoli uzyskać rozwiązanie zadania planowania ruchu bez konieczności wprowadzania parametrycznej reprezentacji funkcji sterujących.

3.1 Parametryczna reprezentacja konfiguracji endogenicznej

Jak wspomnieliśmy w rozdziale 2.1, konfiguracje endogeniczne $u(\cdot) \in \mathcal{U}$ są funkcjami klasy L_2 można je zatem przedstawić w postaci nieskończonego szeregu funkcji ortogonalnych. Z powodów obliczeniowych wprowadźmy parametryczną reprezentację funkcji sterujących w postaci skończonego (o liczności k) szeregu funkcji ortogonalnych

$$u_i(c,t) = \sum_{i=j}^{k} c_{i,j}\psi_j(t), \qquad i = 1,2,\ldots,m, \tag{3.1}$$

gdzie $c_{i,j} \in I\!R$ jest współczynnikiem stojącym przy j-tej funkcji bazowej w i-tym sterowaniu, a $\psi_j(t)$ jest j-tą funkcją bazową. Zestawiając współczynniki w wektor $c = (c_{1,1}, c_{1,2}, \ldots, c_{1,k}, c_{2,1}, \ldots, c_{m,k}) \in I\!R^{mk}$, a funkcje bazowe $\psi(t)$ w macierz

$$\Psi(t) = \begin{bmatrix} \Psi_1(t) & 0 & \cdots & 0 \\ 0 & \Psi_2(t) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \Psi_m(t) \end{bmatrix} =$$

$$\begin{bmatrix} [\psi_1(t), \psi_2(t), \ldots, \psi_k(t)] & [0,\ldots,0] & \cdots & [0,\ldots,0] \\ [0,\ldots,0] & [\psi_1(t), \psi_2(t), \ldots, \psi_k(t)] & \cdots & [0,\ldots,0] \\ \vdots & \vdots & \ddots & \vdots \\ [0,\ldots,0] & [0,\ldots,0] & \cdots & [\psi_1(t), \psi_2(t), \ldots, \psi_k(t)] \end{bmatrix},$$

reprezentację (3.1) można zapisać w postaci

$$u(c,t) = \Psi(t)c. \tag{3.2}$$

Parametryczna reprezentacja (3.1) konfiguracji endogenicznych definiuje skończeniewymiarową endogeniczną przestrzeń konfiguracyjną $\widetilde{\mathcal{U}} = \mathbb{R}^{km}$. W konsekwencji, wszystkie odwzorowania końcowe $\widetilde{K}_{x_0,T}\colon \widetilde{\mathcal{U}} \to \mathbb{R}^r$, jakobiany $\widetilde{J}_{x_0,T}(c) = J_{x_0,T}(u(c,\cdot))\colon \widetilde{\mathcal{U}} \to \mathbb{R}^r$ i odpowiadające im pseudoodwrotności jakobianów $\widetilde{J}^{\#}_{x_0,T}(c) = J^{\#}_{x_0,T}(u(c,\cdot))\colon \mathbb{R}^r \to \widetilde{\mathcal{U}}$ stają się skończeniewymiarowe. Wszystkie definicje skończeniewymiarowych elementów, zarówno dla właściwego zadania planowania ruchu jak i dla zadań dodatkowych opisanych w podrozdziale 2.3, zostały zebrane w dodatku B.

3.1.1 Ortogonalne funkcje bazowe

Do wyrażenia funkcji sterujących przy pomocy parametryzacji (3.1) można wykorzystać różne szeregi funkcji ortogonalnych. W [17] do rozwiązania zadania planowania ruchu manipulatora mobilnego zaproponowano reprezentację w postaci funkcji harmonicznych (szereg Fouriera), a także kilka szeregów funkcji wielomianowych, takich jak wielomiany Legendre'a, Gegenbauera, Czebyszewa, a także funkcje Haara. W niniejszej pracy, w symulacjach będziemy wykorzystywać dwie wybrane bazy funkcji ortogonalnych: bazę Fouriera i bazę Laguerre'a.

Baza Fouriera

Element o indeksie i bazy Fouriera ma postać

$$F_i(t) = \begin{cases} 1 & \text{dla } i = 0, \\ \sin\left(\frac{(i+1)}{2}\omega t\right) & \text{dla } i = 1,3,5,\ldots, \\ \cos\left(\frac{i}{2}\omega t\right) & \text{dla } i = 2,4,6,\ldots, \end{cases} \qquad \omega = \frac{2\pi}{T},$$

a dowolna funkcja może zostać przybliżona w tej bazie poprzez odpowiedni dobór współczynników $a = (a_0, \ldots, a_k)$

$$f_F(t) = \sum_{i=0}^{k} a_i F_i(t). \tag{3.3}$$

Baza Fouriera jest ortogonalna na przedziale $[0,T]$, a kilka kolejnych elementów bazy wykreślono na rysunku 3.1.

Baza Laguerre'a

Elementy bazy wielomianów Laguerre'a mogą zostać wyznaczone korzystając z formuły Rodriguesa [19]

$$l_i(t) = \frac{\exp(t)}{i!}\frac{\mathrm{d}^i}{\mathrm{d}t^i}\left(\exp(-t)t^i\right).$$

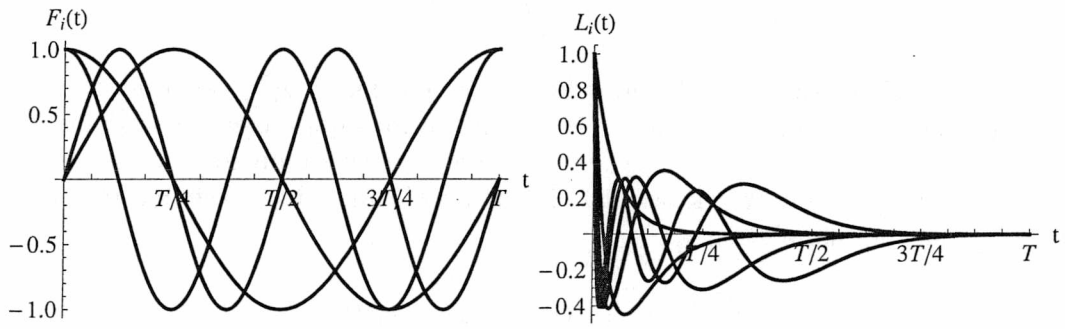
Aby zapewnić ortogonalność bazy Laguerre'a, niezbędne jest przemnożenie powyższych elementów przez pewien czynnik

$$L_i(t) = \exp(-t/2)l_i(t).$$

Ostatecznie, dowolna funkcja może zostać przybliżona przez skończony szereg wykorzystujący wielomiany Laguerre'a z odpowiednimi współczynnikami $a = (a_1, \ldots, a_k)$

$$f_L(t) = \sum_{i=1}^{k} a_i L_i(t). \tag{3.4}$$

Baza Laguerre'a ma większy obszar ortogonalności $[0, \infty)$ niż baza Fouriera. W razie potrzeby można przeskalować czas, aby zmienić tempo zbieżności elementów bazy. Kilka pierwszych elementów bazy Laguerre'a pokazano na rysunku 3.1.



Rysunek 3.1 Kilka pierwszych elementów bazy Fouriera – strona lewa, bazy Laguerre'a – strona prawa

3.2 Dyskretna wersja algorytmu planowania ruchu

Do wyznaczenia rozwiązania równania różniczkowego definiującego algorytm planowania ruchu (2.15) lub algorytmy z zadaniami dodatkowymi (2.20) i (2.26) można stosować różne schematy obliczeniowe, spośród których wykorzystamy metodę Eulera. W tym celu definiujemy równania różnicowe będące odpowiednikami ciągłych równań różniczkowych. Wersja dyskretna podstawowego algorytm planowania ruchu może zostać wyrażona przez

$$u_{\vartheta+1}(t) - u_\vartheta(t) = -\gamma \left(J^{\#}_{x_0,T}(u_\vartheta(\cdot))e(\vartheta) \right)(t), \qquad \vartheta = 1, 2, 3, \ldots. \tag{3.5}$$

Korzystając z parametrycznej reprezentacji (3.1), możemy uzyskać dyskretną i parametryczną wersję algorytmu planowania ruchu

$$c(\vartheta + 1) = c(\vartheta) - \gamma \widetilde{J}^{\#}_{x_0,T}(c(\vartheta))\widetilde{e}(\vartheta).$$

W podobny sposób, możemy zdefiniować dyskretne i parametryczne wersje wszystkich algorytmów planowania ruchu wyprowadzonych w rozdziale 2.

3.3 Nieparametryczna reprezentacja konfiguracji endogenicznej

Istnieje także możliwość uzyskania rozwiązania zadania planowania ruchu bez wprowadzania parametrycznej, skończeniewymiarowej reprezentacji funkcji sterujących. Takie podejście jest związane ze wzrostem czasu obliczeń, jednak z drugiej strony uzyskiwane rozwiązania (przebieg funkcji sterującej) nie są zależne od wyboru bazy.

Przyjrzyjmy się dokładniej algorytmowi planowaniu ruchu wyrażonego przez (2.15). W tym podejściu również będziemy korzystali z metody Eulera, co prowadzi do dyskretnej wersji algorytmu (3.5). Każda kolejna iteracja algorytmu wyznaczy kolejną funkcję sterującą według wzoru

$$u_{\vartheta+1}(t) = u_\vartheta(t) - \gamma \left(J^{\#}_{x_0,T}(u_\vartheta(\cdot))e(\vartheta) \right)(t).$$

Zgodnie z (2.9), do wyznaczenia pseudoodwrotności jakobianu $J^{\#}_{x_0,T}(u(\cdot))$ niezbędna jest macierz $\Phi(T,t)$. W zadaniu właściwego planowania ruchu zawsze pierwszy argument macierzy $\Phi(T,t)$ jest stały. Można to wykorzystać przy wyznaczaniu tej macierzy. W każdym kroku ϑ wyznaczamy trajektorie $x(t)$ i $\overline{\Phi}(t)$ rozwiązując dwa równania różniczkowe o wspólnej zmiennej niezależnej t

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} = f(x(t)) + G(x(t))u_\vartheta(t),$$
$$\frac{\mathrm{d}\overline{\Phi}(t)}{\mathrm{d}t} = -\overline{\Phi}(t)A(t),$$

gdzie $\overline{\Phi}(t) = \Phi(T,t)$ [45]. Następnie, uaktualniamy sterowanie

$$u_{\vartheta+1}(t) = u_\vartheta(t) - \gamma \left(J^{\#}_{x_0,T}(u_\vartheta(\cdot))e(\vartheta) \right)(t) =$$
$$u_\vartheta(t) - \gamma \mathsf{B}^{\mathsf{T}}(t)\overline{\Phi}^{\mathsf{T}}(t)\mathsf{C}^{\mathsf{T}}(T) \left(\mathsf{C}(T) \int_0^T \overline{\Phi}(s)\mathsf{B}(s)\mathsf{B}^{\mathsf{T}}(s)\overline{\Phi}^{\mathsf{T}}(s)\,\mathrm{d}s\mathsf{C}^{\mathsf{T}}(T) \right)^{-1} (k(x(T)) - y_d).$$

Ostatecznie, sprawdzamy warunek stopu $\|e(\vartheta)\| = \|k(x(T)) - y_d\| = \|k(\varphi_{x_0,T}(u(\cdot))) - y_d\| \leqslant e_{\max}$ i w przypadku jego niespełnienia rozpoczynamy całą procedurę od początku. Warunkiem początkowym algorytmu są początkowe funkcje sterujące $u_{\vartheta=0}(t)$.

Rozdział 4

Przykłady numeryczne

W niniejszym rozdziale przedstawimy symulacyjne rozwiązania zadania planowania ruchu układów robotycznych z deficytem napędów. Wszystkie wyniki zostały uzyskane przy pomocy programu MATHEMATICA. Pokażemy wyniki działania algorytmu z różnymi zadaniami dodatkowymi, dokonamy także porównania algorytmu egalitarnego z algorytmem z priorytetowaniem. Zaprezentujemy również wyniki symulacji algorytmu planowania ruchu przy niepełnej znajomości modelu. Ponadto, rozwiążemy zadanie planowanie ruchu przy nieparametrycznej reprezentacji funkcji sterujących, oraz wykorzystamy zadanie dodatkowe ograniczające wartości zmiennych stanu do uzyskania rozwiązania zadania reprodukcji trajektorii. Wszelkie modele użyte w bieżącym rozdziale zostały opisane w dodatku C. Dla każdej symulacji będziemy podawać parametry algorytmu, dla zaoszczędzenia miejsca wytłumaczmy teraz zastosowane oznaczenia

- $x_0 = x(0)$ – stan początkowy,
- y_d – zadana wartość wyjścia,
- T – czas trwania ruchu,
- γ – wektor współczynników zbieżności,
- E – macierz skalująca błąd przy algorytmie egalitarnym,
- $\Psi(t)$ – macierz funkcji bazowych reprezentacji sterowania,
- $c(0)$ – początkowy wektor współczynników sterowań.

Jakość rozwiązań symulacyjnych uzyskiwanych przez algorytmy planowania ruchu zależy w sposób istotny od doboru parametrów algorytmu. Poniżej opiszemy wpływ tych parametrów na pracę algorytmu i podamy kilka praktycznych wskazówek pomagających w doborze ich wartości.

Dobór liczby k elementów bazy funkcji ortogonalnych

W celu omówienia kwestii liczby elementów bazy wróćmy raz jeszcze do analogii pomiędzy podejściem endogenicznej przestrzeni konfiguracyjnej a metodologią zaczerpniętą z teorii manipulatorów stacjonarnych. Zgodnie z [27], redundancja manipulatora może zostać wykorzystana do realizacji zadań dodatkowych. Liczba km użytych funkcji bazowych w podejściu endogenicznym odgrywa rolę analogiczną do liczby zmiennych przegubowych w teorii manipulatorów. Zatem, aby realizować zadanie planowania ruchu z zadaniami

dodatkowymi, liczba km użytych funkcji bazowych musi być co najmniej równa sumie wymiarów przestrzeni zadaniowych wszystkich zadań. Taką zależność możemy opisać następująco

$$km \geqslant \sum_{i=1}^{z} r_i,$$

gdzie $r_i = \dim e(\vartheta)$ jest wymiarem przestrzeni zadaniowej i-tego zadania. Użycie większej liczby elementów bazy pozwala uzyskać mniejsze błędy zadań, oraz niejednokrotnie powoduje zmniejszenie liczby kroków potrzebnych do otrzymania rozwiązania. Wpływ liczby funkcji bazowych przedstawimy w symulacji zamieszczonej w podrozdziale 4.1.2. Zwiększenie liczby km pozwala często uniknąć problemów numerycznych podczas wyznaczania pseudoodwrotności jakobianu. Podejście przedstawione w podrozdziale 3.3, w którym nie parametryzujemy funkcji sterującej pozwala na uzyskanie „nieskończonej" redundancji, co powinno skutkować poprawą uzyskiwanych wyników.

Dobór współczynnika zbieżności γ

Zgodnie z formułą (2.13), współczynnik γ jest odpowiedzialny za szybkość zbieżności błędu e do zera. Nietrudno zauważyć, że przy metodzie Eulera rozwiązywania równania różniczkowego algorytmu, współczynnik γ wpływa także na długość kroku algorytmu. Zbyt niska wartość współczynnika powoduje zbyt wolną zbieżność algorytmu, który staje się niepraktyczny. Z drugiej strony, zbyt duża wartość sprawia, że otrzymanie rozwiązania z zadaną dokładnością będzie niemożliwe. Dobór wartości γ powinien być zatem uzależniony od dynamiki równania algorytmu (2.15), która zależy od dynamiki układu robotycznego i od wybranej odwrotności jakobianu. Badania symulacyjne pokazują [37], że możliwe jest również uzależnienie współczynnika od kroku $\gamma(\vartheta)$; umiejętne dobranie tej zależności pozwala uzyskać szybszą zbieżność algorytmu niż przy użyciu stałej wartości γ. Jeżeli potraktujemy parametr γ jako długość kroku w algorytmie gradientowym, to zwykle przyjmuje się wartości $0 < \gamma < 1$. Niemniej jednak, w zależności od konkretnego zadania, przyjęcie wartości $\gamma > 1$ niejednokrotnie przynosi lepsze rezultaty. W algorytmie z priorytetowaniem zadań współczynnik $\gamma = (\gamma_1, \ldots, \gamma_z)$ jest wektorem, w którym kolejne współrzędne γ_i odpowiadają kolejnym zadaniom S_i. W takim przypadku, dobór γ_i określa wpływ poszczególnych zadań składowych na rozwiązanie uzyskane za pomocą algorytmu z priorytetowaniem.

Dobór sterowania początkowego

Od przedstawianych tutaj algorytmów planowania ruchu możemy oczekiwać jedynie lokalnej zbieżności. Uzyskanie rozwiązania zadania zależy od doboru początkowego sterowania. W przypadku parametrycznej reprezentacji funkcji sterującej, sterowanie początkowe zadajemy poprzez parametry sterowania $u_{\vartheta=0}(t) = \Psi(t)c(\vartheta{=}0)$. Natomiast w przypadku, gdy nie korzystamy z parametrycznej reprezentacji, początkowe sterowanie definiujemy jako arbitralnie wybraną funkcję $u_{\vartheta=0}(t) = u_0(t)$. Badania symulacyjne wskazują, że nieumiejętne dobranie sterowania początkowego może być przyczyną nie uzyskania rozwiązania zadania planowania ruchu. Zwykle, wybranie początkowego sterowania w sposób, który umożliwi ruch układu w kierunku y_d od pierwszego kroku pozwala szybciej otrzymać zadowalające rozwiązanie. Niemniej jednak, jak pokażemy w symulacji (podrozdział 4.1.3), wybranie początkowego sterowania, które jest niezgodne z tą regułą także pozwala uzyskać rozwiązanie.

Czas trwania ruchu

Jeżeli stawiane zadanie planowania ruchu nie definiuje w jakim czasie ma zostać zrealizowany ruch układu, należy dobrać taki okres czasu T, który nie będzie wymuszał zbyt dużych lub zbyt małych prędkości układu. Horyzont czasowy T powinien zatem być uzależniony od dynamiki układu robotycznego i od zasięgu ruchu $\|k(x_0) - y_d\|$.

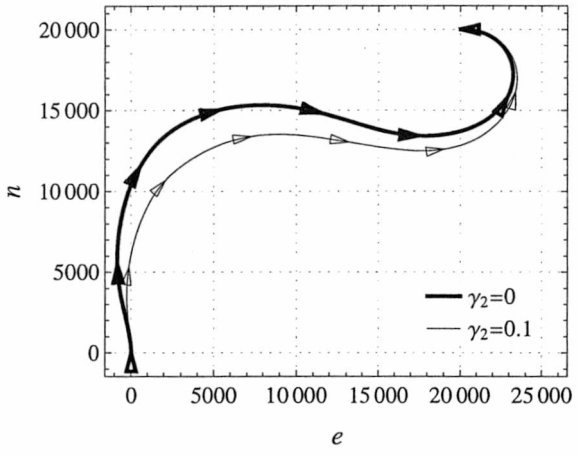
4.1 Symulacyjne wyniki planowania ruchu wybranych układów z różnymi zadaniami dodatkowymi

W pierwszym podrozdziale przeprowadzimy symulacje algorytmu planowania ruchu dla wybranych układów robotycznych z deficytem napędów z różnymi zadaniami dodatkowymi.

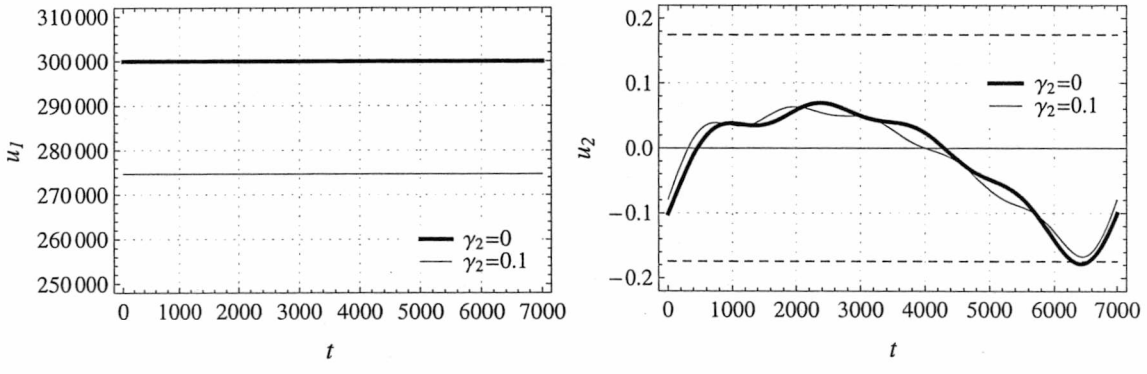
4.1.1 Minimalizacja energii sterowania przy planowaniu ruchu statku

Jako pierwszy wynik zaprezentujemy planowanie ruchu statku (patrz dodatek C.1). Przedstawimy wyniki działania dwóch algorytmów. Pierwszy składać się będzie tylko z głównego zadania (planowania ruchu), a drugi będzie zawierał zadanie główne, właściwe planowanie ruchu i zadanie dodatkowe, o niższym priorytecie, polegające na minimalizacji energii sterowania. Pozwoli to zaobserwować wpływ drugiego zadania na uzyskiwane wyniki. Planowanie odbywa się w przestrzeni zadaniowej $y = (n, e, \psi)$, a jako punkt końcowy wybraliśmy współrzędne $y_d = (20000, 20000, -\pi/2)$, które znajdują się stosunkowo daleko od zadanego punktu początkowego $x_0 = (n, e, \psi, u, v, r) = (0, 0, 0, 4, 0, 0)$ (oznaczenia wyjaśniono w dodatku C). Czas symulacji wynosi $T = 7000$, a współczynnik zbieżności pierwszego zadania ${}^1\gamma = 0.1$. Wyznaczane funkcje sterujące będą reprezentowane przez skończony szereg Fouriera (3.3) dla $k = 10$, a początkowy wektor współczynników sterowania jest równy $c(0) = (300000, 0_{1\times 10}, 0.01, 0_{1\times 10})$. Jak wspomnieliśmy, przeprowadzimy dwie symulacje, pierwszą bez drugiego zadania, gdzie ${}^2\gamma = 0$ i drugą z dodatkowym zadaniem minimalizacji energii sterowania, ze współczynnikiem ${}^2\gamma = 0.1$. Macierz wag wybraliśmy równą $\sigma = I_m$, co znaczy, że oba sterowania mają być minimalizowane w tym samym stopniu. Warunkiem stopu w obu algorytmach jest zmniejszenie się normy błędu głównego zadania poniżej wartości $\|{}^1e\| \leqslant 1$, przy czym należy zaznaczyć, że błąd orientacji spadał znacznie poniżej tej wartości. Ponadto, kąt skrętu steru statku nie powinien przekraczać wartości $|u_2| \leqslant 10°$ podczas trwania ruchu. Wyniki symulacji zostały pokazane na rysunkach 4.1–4.5. Na rysunku 4.1 przedstawiliśmy ścieżkę ruchu i zaznaczyliśmy sylwetkę statku w odstępie co 1000 jednostek czasu. Z rysunków 4.3 i 4.4 wynika, że algorytm w obu badanych przypadkach prawidłowo rozwiązał zadanie planowania ruchu. Na rysunku 4.2 zostały przedstawione funkcje sterujące. Algorytm uwzględniający drugie zadanie wyznaczył ruch charakteryzujący się mniejszą energią sterowania. Ponadto, sterowanie u_2 odpowiadające kątowi steru w przypadku z zadaniem dodatkowym nie przekracza wartości granicznych (linia przerywana), w przeciwieństwie do funkcji sterującej wyznaczonej przez algorytm przy ${}^2\gamma = 0$. Prędkość zbiegania się algorytmu rozwiązującego właściwe planowanie ruchu i algorytmu rozwiązującego zadanie planowania ruchu z zadaniem dodatkowym jest podobna co pokazuje rysunek 4.5. Inne wyniki działania algorytmu planowania ruchu statku mogą zostać znalezione w [35, 36]. Z przedstawionej symulacji wynika, że proponowany algorytm pomyślnie rozwiązuje zadanie planowania ru-

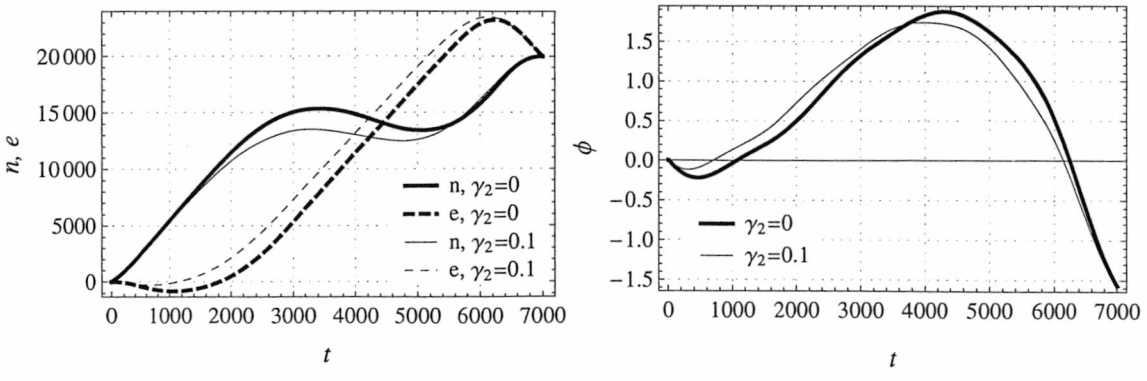
chu wraz z jednoczesnym minimalizowaniem energii sterowania. Ponadto pokazaliśmy, że zadany punkt docelowy $y_d = (20000, 20000, -\pi/2)$ może być znacznie odległy od punktu początkowego $y_0 = k(x_0) = (0, 0, 0)$, i czas ruchu może być bardzo długi.



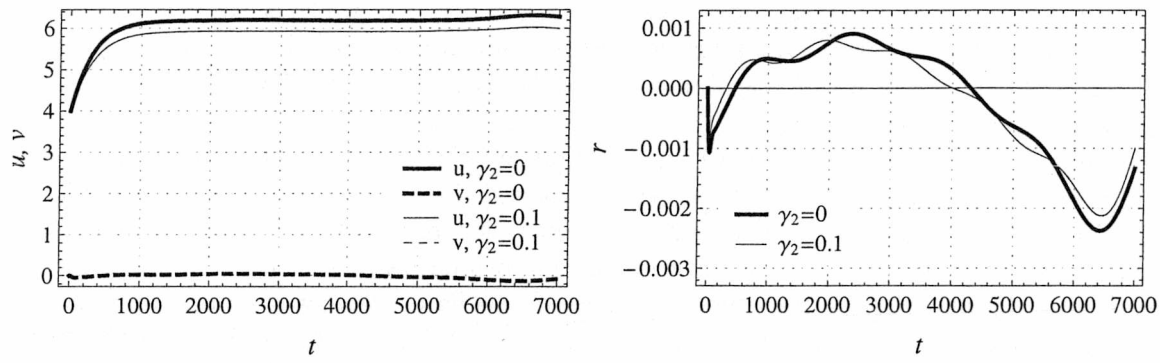
Rysunek 4.1 Ścieżka ruchu statku



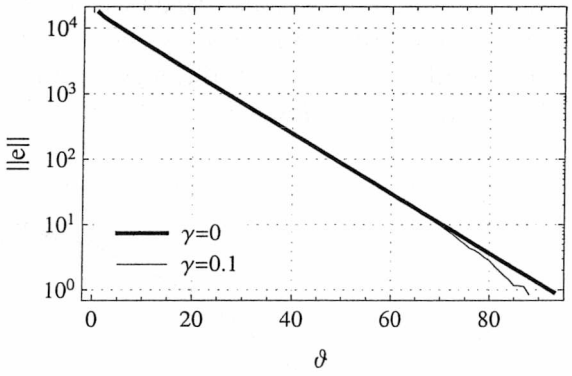
Rysunek 4.2 Przebiegi funkcji sterujących



Rysunek 4.3 Trajektorie zmiennych stanu (położenia)



Rysunek 4.4 Trajektorie zmiennych stanu (prędkości)

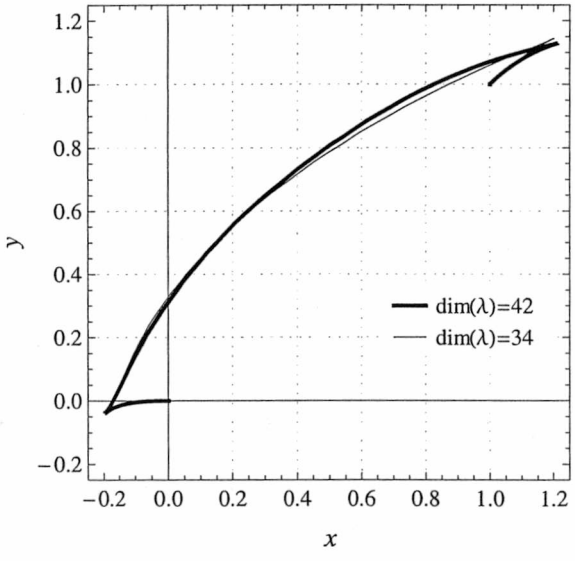


Rysunek 4.5 Zbieżność algorytmów (błąd właściwego planowania ruchu)

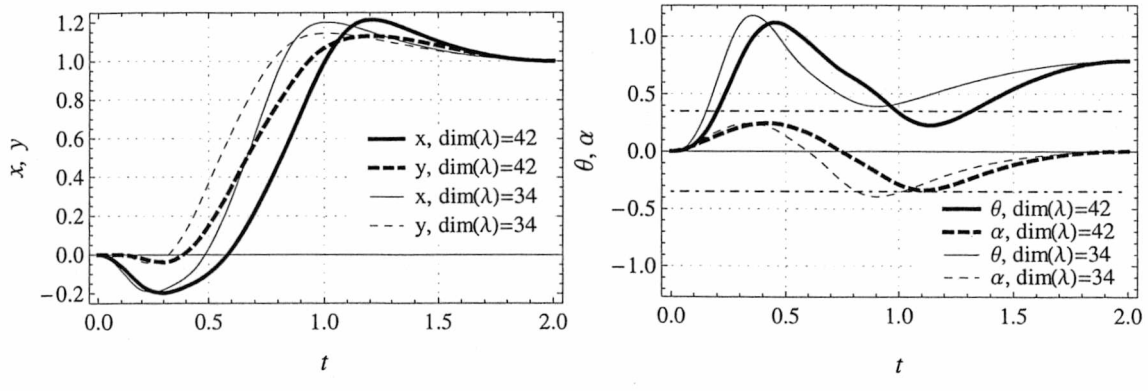
4.1.2 Minimalizacja wartości zmiennych stanu przy planowaniu ruchu robota balansującego

W kolejnej symulacji przedstawimy planowanie ruchu przy obecności zadania dodatkowego polegającego na minimalizacji wartości zmiennej stanu. Modelem testowym będzie robot balansujący opisany w dodatku C.2. Dla takiego robota zaproponujemy algorytm planowania ruchu utworzony z trzech zadań [38]. Pierwsze zadanie, główne, to zadanie planowania ruchu. Drugim zadaniem będzie utrzymywanie zmiennej odpowiedzialnej za wychylenie korpusu robota w pobliżu wartości zerowej, tak aby robot utrzymywał pozycję pionową. Trzecie zadanie będzie polegało na minimalizacji energii sterowania. Każde kolejne zadanie ma niższy priorytet od poprzedniego. Dodatkowo, pokażemy tutaj wpływ liczby funkcji bazowych reprezentujących sygnały sterujące na wartości błędów zadaniowych. W tym celu, przedstawimy wyniki symulacji dla dwóch różnych wymiarów wektora współczynników $\dim c = 34$ i $\dim c = 42$. Chcemy, aby algorytm wyznaczył sterowanie, które zapewni ruch robota od stanu spoczynkowego (zerowe prędkości i wahadło w pionie) zapoczątkowany w punkcie $(0,0)$ z orientacją $\theta = 0$ do stanu spoczynkowego w punkcie $x_d = (\eta_1, \eta_2, \eta_3, x, y, \theta, \alpha) = (0,0,0,1,1,\pi/4,0)$ (patrz dodatek C) w czasie $T = 2$. Obliczenia wykonywane są do momentu uzyskania wartości normy błędu pierwszego zadania $\|{}^1e\| \leqslant 10^{-5}$. Dodatkowo, żądamy aby kąt wychylenia korpusu robota balansującego nie przekraczał $|\alpha| < 20^\circ \approx 0.35$, a energia sterowania była możliwie najmniejsza. Do utworzenia algorytmu realizującego przedstawione zadania wybraliśmy macierz wag dla algorytmu minimalizacji zmiennych stanu $\delta = \mathrm{diag}\{0,0,0,0,0,0,1\}$, co oznacza minimalizację

tylko ostatniej zmiennej stanu α oraz macierz wag $\sigma = I_m$, prowadzącą do równorzędnego traktowania obu sterowań. Współczynnik zbieżności dla trzech kolejnych zadań wynosi $\gamma = (0.1, 0.1, 0.5)$. Każda z dwóch funkcji sterujących jest reprezentowana przez skończony szereg wielomianów Laguerre'a (3.4) przy $k = 17$ w pierwszym przypadku oraz $k = 21$ w drugim. Początkowe wartości wektora współczynników $c(0) = (0.1, 0_{1\times16}, 0.1, 0_{1\times16})$ i $c(0) = (0.1, 0_{1\times20}, 0.1, 0_{1\times20})$. Seria rysunków 4.6–4.10 przedstawia efekty symulacji. W obu badanych przypadkach uzyskane ścieżki ruchu robota są do siebie podobne (rysunek 4.6). Analizując trajektorie robota przedstawione na rysunkach 4.7 i 4.8 możemy zauważyć, że dla różnych ilości funkcji bazowych, zadanie właściwego planowania ruchu jest zrealizowane pomyślnie. Wszystkie współrzędne stanu osiągają zadane położenie x_d w zadanym czasie T. Istotne różnice pomiędzy rozpatrywanymi przykładami pojawiają się w jakości rozwiązania drugiego i trzeciego zadania dodatkowego. W przypadku z mniejszym wymiarem wektora współczynników sterowania, zmienna $\alpha(t)$ odpowiedzialna za wychylenie korpusu robota przekracza dozwoloną wartość (linia kreskowo-kropkowana na rysunku 4.7), w drugim przypadku przebieg $\alpha(t)$ zawiera się w ustalonych granicach. Ponadto, użyta energia sterowania jest mniejsza dla algorytmu z większą liczbą funkcji bazowych, co można zaobserwować na rysunkach 4.9 oraz 4.10. Można zatem wyciągnąć

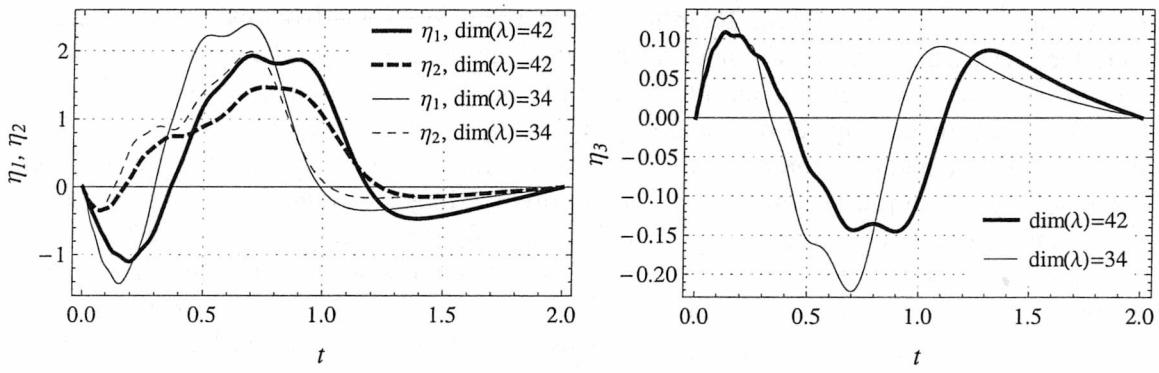


Rysunek 4.6 Ścieżka ruchu robota balansującego

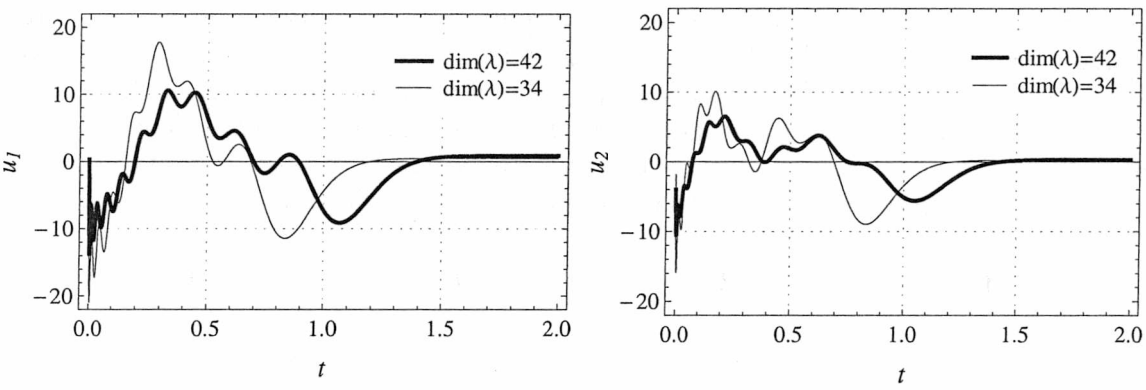


Rysunek 4.7 Trajektorie zmiennych stanu (położenia)

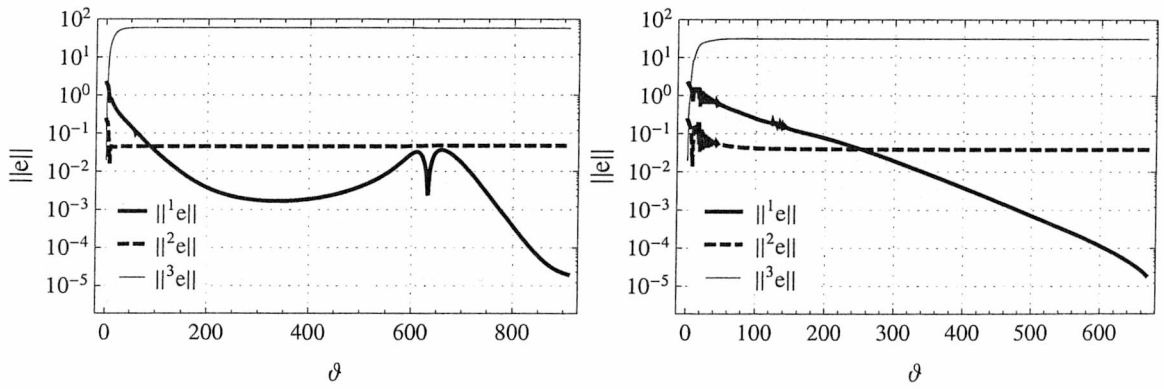
wniosek, że dzięki zwiększeniu wymiaru reprezentacji funkcji sterujących algorytm ma do dyspozycji bardziej różnorodne funkcje, co przekłada się na wzrost efektywności otrzymywanych rozwiązań. Dodatkowo, takie zwiększenie swobody zazwyczaj sprzyja zmniejszeniu liczby kroków algorytmu, co widać na rysunku 4.10. Mankamentem takiego rozwiązania jest zwiększenie nakładów obliczeniowych dla pojedynczego kroku algorytmu. Na uwagę zasługuje także fakt, że tutaj realizujemy aż trzy zadania jednocześnie.



Rysunek 4.8 Trajektorie zmiennych stanu (prędkości)



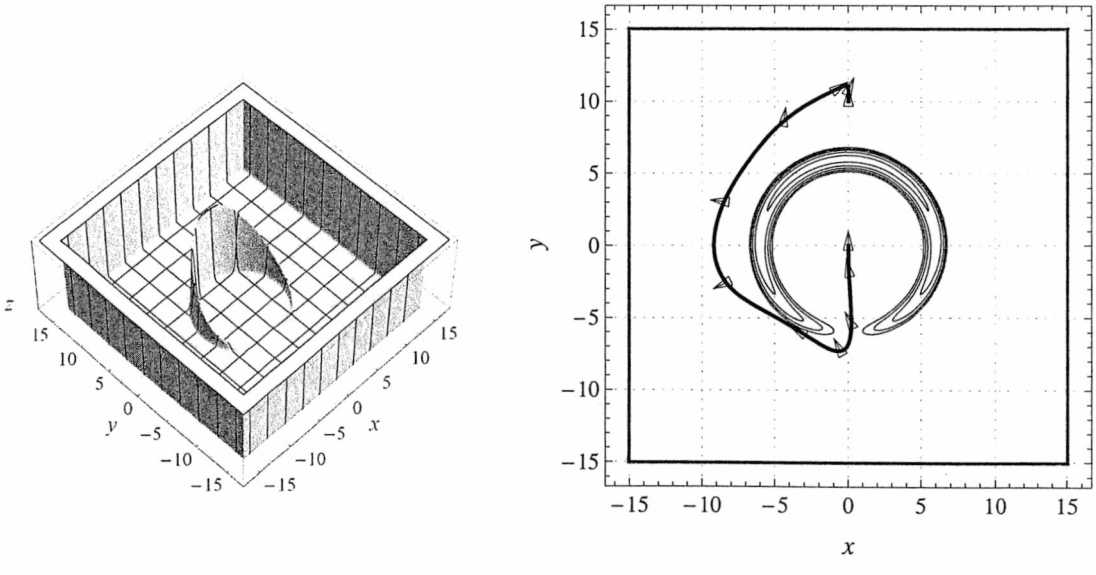
Rysunek 4.9 Przebiegi funkcji sterujących



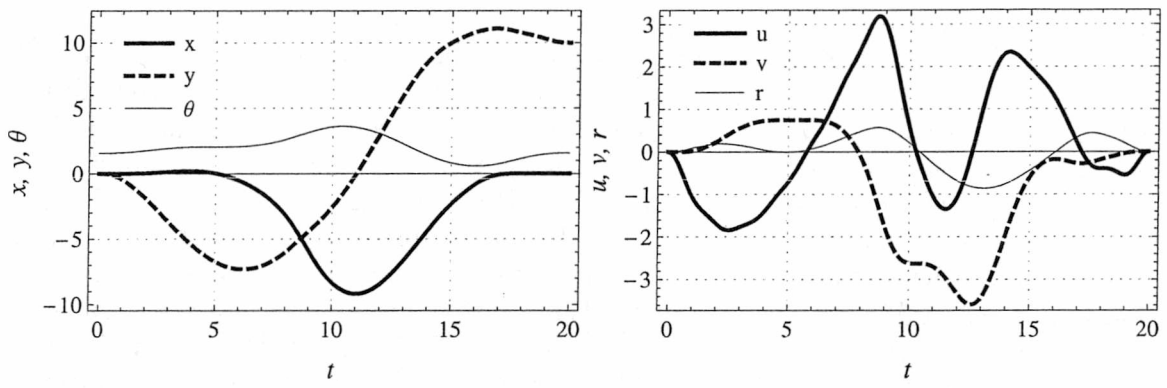
Rysunek 4.10 Zbieżność algorytmu (dim $c = 34$ – strona lewa, dim $c = 42$ – strona prawa)

4.1.3 Planowanie ruchu poduszkowca z unikaniem przeszkód

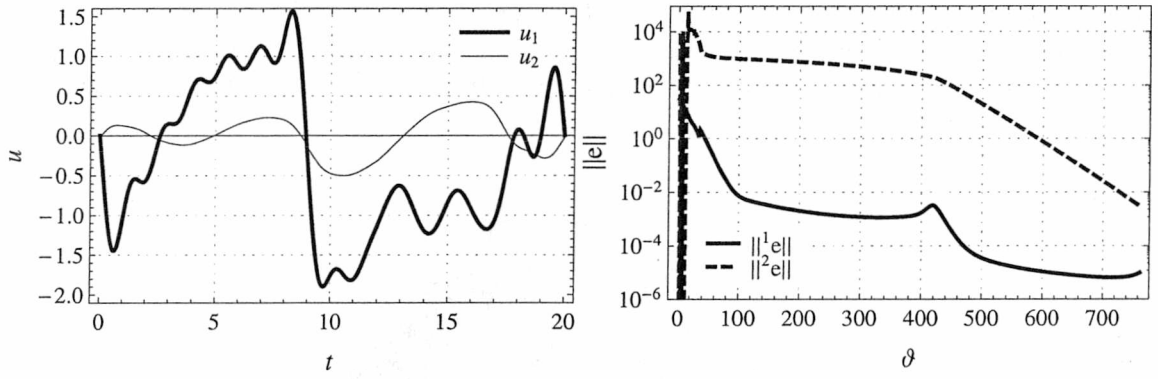
Ostatnim zadaniem analizowanym w tym rozdziale będzie planowanie ruchu modelu poduszkowca z unikaniem przeszkód. Szczegóły dotyczące modelu testowego zamieściliśmy w dodatku C.3. Zadanie będzie polegało na znalezieniu funkcji sterującej, która przeprowadzi robota z punktu początkowego do zadanego punktu końcowego, tak aby ścieżka ruchu nie kolidowała z przeszkodami. Do rozwiązania tego zadania użyjemy algorytmu z priorytetowaniem zadań. Rozmieszczenie przeszkód przedstawiliśmy na rysunku 4.11. Obszar ruchu jest obszarem kwadratowym ograniczonym funkcjami $x = y = 15$, $x = y = -15$, a na środku znajduje się przeszkoda w kształcie podkowy. W czasie $T = 0$ robot stoi w punkcie otoczonym przeszkodą i jest zwrócony tyłem do „wyjścia" podkowy. Punkt zadany, znajduje się przed robotem, lecz jest przesłonięty przeszkodą. Zarówno w punkcie początkowym jak i w zadanym założyliśmy zerowe prędkości. Możemy zatem napisać $x_0 = (x, y, \theta, u, v, r) = (0, 0, \pi/2, 0, 0, 0)$ oraz $x_d = (0, 10, \pi/2, 0, 0, 0)$, oznaczenia wg dodatku C. Czas ruchu wynosi $T = 20$. Funkcje sterujące reprezentujemy w postaci skończonego szeregu Fouriera (3.3) dla $k = 28$, a początkowe współczynniki sterowania $c(0) = (0.15, 0_{1\times 28}, 0.15, 0_{1\times 28})$. Należy zwrócić uwagę, że początkowe sterowanie odpowiada ruchowi w kierunku kolizyjnym. Nie jest tutaj konieczne „wskazanie" kierunku wyjazdu z przeszkody. Szybkość zbieżności obu błędów algorytmu reguluje współczynnik $\gamma = (0.1, 2)$, a obliczenia są prowadzone do momentu uzyskania wartości błędu pierwszego zadania $\|^1 e\| \leqslant 10^{-5}$. Główne zadanie, właściwe planowanie ruchu, jest realizowane z zadaną dokładnością, co można zaobserwować na rysunku 4.12. Wszystkie współrzędne stanu osiągają zadane wartości. Ścieżka ruchu została przedstawiona na rysunku 4.11; ruch poduszkowca odbywa się w sposób niekolidujący z przeszkodami. Sylwetka robota została zaznaczona w odstępach co 2 jednostki czasu. Rysunek 4.13 przedstawia przebieg funkcji sterujących i zbieżność algorytmu. Wartość odwzorowania końcowego zadania drugiego osiąga wartość rzędu 10^{-2}, tak więc uzyskany ruch jest bezkolizyjny.



Rysunek 4.11 Rozmieszczenie przeszkód oraz ścieżka ruchu robota



Rysunek 4.12 Trajektorie zmiennych stanu



Rysunek 4.13 Przebiegi funkcji sterujących – strona lewa, zbieżność algorytmu – strona prawa

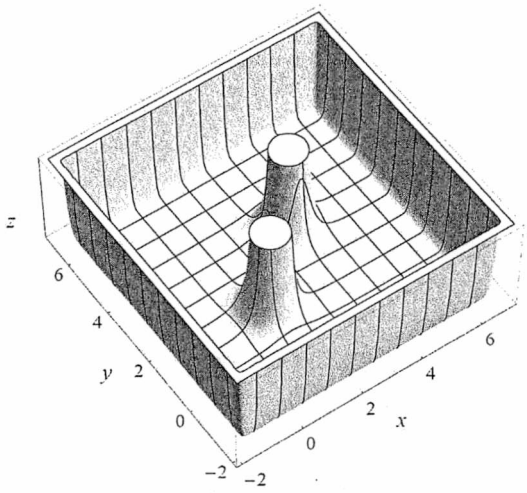
4.2 Porównanie algorytmu egalitarnego z algorytmem z priorytetowaniem

W bieżącym rozdziale przeprowadzimy porównanie efektywności dwóch algorytmów realizujących zadania dodatkowe. Porównamy algorytm egalitarny, wprowadzony w rozdziale 2.2.1, i algorytm z priorytetowaniem zadań, z rozdziału 2.2.2. Dla uproszczenia zapisów przyjmijmy oznaczenie $(\cdot)_p$ dla obiektów związanych z algorytmem z priorytetowaniem zadań, $(\cdot)_e$ dla obiektów związanych z algorytmem egalitarnym oraz $(\cdot)_b$ dla obiektów związanych z algorytmem bez żadnych dodatkowych zadań. Modelem testowym będzie poduszkowiec (dodatek C.3). Przeprowadzimy porównanie obu algorytmów z kilkoma zestawami zadań dodatkowych, które oznaczymy kolejno

- S_1 - właściwe planowanie ruchu (zadnie główne),
- S_2 - zadanie ograniczenia energii sterowania,
- S_3 - zadanie ograniczenia wybranej zmiennej stanu: prędkości poprzecznej ($v \to 0$),
- S_4 - zadanie omijania przeszkód w dwuwymiarowej przestrzeni położeń robota.

a symulacje będą przeprowadzone dla następujących zestawów zadań: tylko S_1, S_1 i S_2, S_1 i S_3 oraz S_1 i S_4. Większość współczynników symulacji pozostaje niezmienna dla wszystkich obliczeń. Punkt startowy wybraliśmy jako $x_0 = (0, 0, 0, 0, 0, 0)$, a punkt docelowy

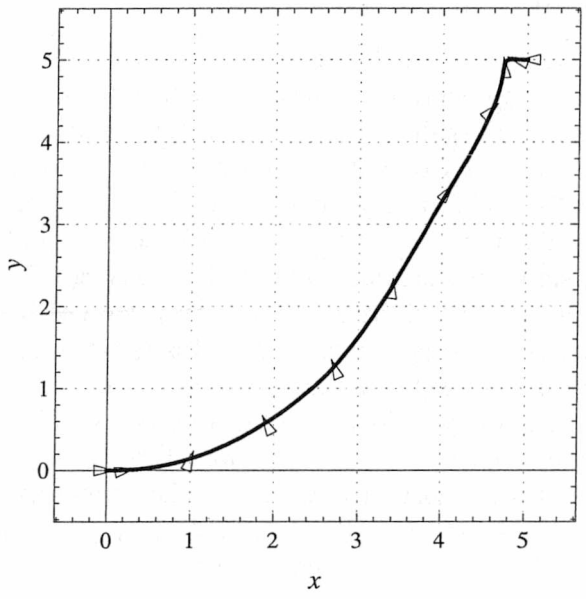
$y_d = x_d = (5, 5, \pi, 0, 0, 0)$ ma zostać osiągnięty w czasie ruchu $T = 5$. Funkcjami bazowymi są elementy szeregu Fouriera (3.3) dla $k = 6$, a początkowy wektor sterowań $c(0) = (0.1, 0_{1\times6}, 0.001, 0_{1\times6})$. W symulacji ograniczenia energii sterowania wybraliśmy macierz wag równą $\sigma = I_m$, czyli oba sterowania traktujemy równorzędnie. Macierz wag w algorytmie ograniczenia wartości zmiennej stanu wybrano tak, aby minimalizować zmienną $v(t)$ (prędkość poprzeczną, dodatek C), zatem $\delta = \mathrm{diag}\{0, 0, 0, 0, 1, 0\}$. W algorytmie omijania przeszkód obszar ruchu wraz z przeszkodami ilustruje rysunek 4.14. W zadaniach rozwiązywanych algorytmem egalitarnym współczynnik zbieżności jest równy $\gamma = 0.2$. Taką samą wartość wybraliśmy dla współczynnika zbieżności zadania głównego w algorytmie z priorytetowaniem ${}^1\gamma = 0.2$. Dla kolejnych zadań dodatkowych współczynniki przyjmują wartości ${}^2\gamma = 2$, ${}^3\gamma = 8$, ${}^4\gamma = 2$. W algorytmie egalitarnym wpływ poszczególnych zadań definiują macierze skalujące błąd, które dobieramy następująco $E = \mathrm{blokdiag}\{I_{r_1}, 0.0025I_{r_2}\}$ dla zestawu S_1 i S_2, $E = \mathrm{blokdiag}\{I_{r_1}, 0.1I_{r_3}\}$ dla zestawu S_1 i S_3 oraz $E = \mathrm{blokdiag}\{0.04I_{r_1}, 0.003I_{r_4}\}$ dla zestawu S_1 i S_4. Żądamy, aby błąd algorytmu właściwego planowania ruchu był poniżej wartości $\|{}^1e\| \leqslant 10^{-5}$. W kolejnych podrozdziałach przedstawimy wyniki poszczególnych symulacji, z których wyciągniemy szereg wniosków.



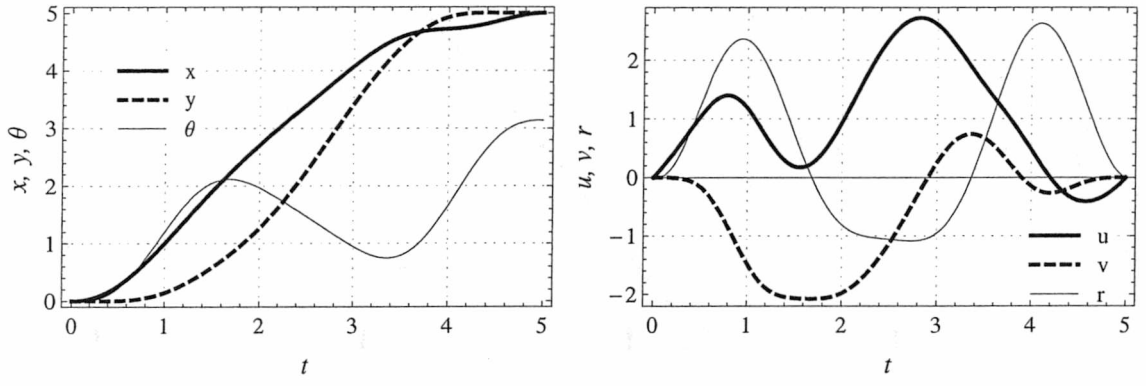
Rysunek 4.14 Rozmieszczenie przeszkód

4.2.1 Wyniki symulacji zadania S_1

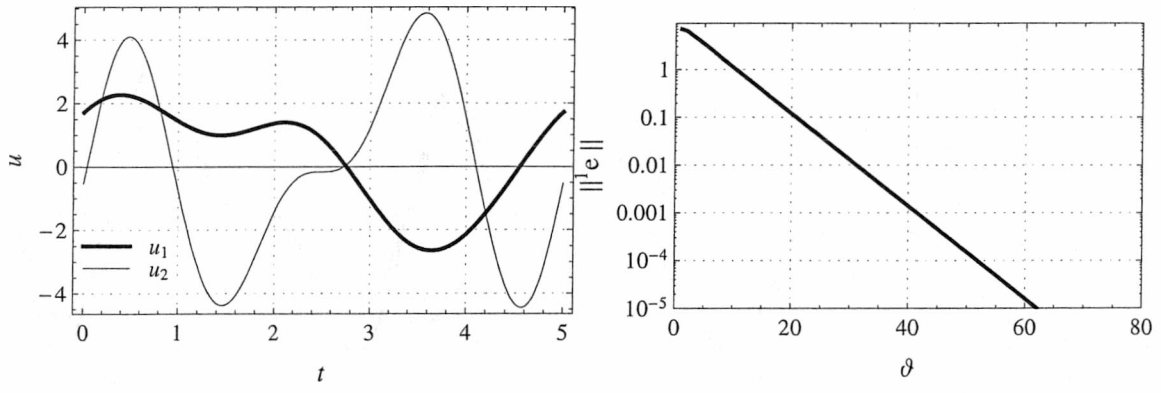
Seria rysunków 4.15–4.17 przedstawia wyniki symulacji planowania ruchu złożonego tylko z zadania właściwego planowania ruchu poduszkowca. Ścieżkę ruchu robota zamieściliśmy na rysunku 4.15, na którym także zaznaczyliśmy sylwetkę modelu w chwilach odległych o 0.5 jednostki czasu. Przebiegi funkcji sterujących zostały przedstawione na rysunku 4.17. Z rysunku 4.16 ilustrującego trajektorie stanu można wywnioskować, że zadanie planowania ruchu zostało rozwiązanie prawidłowo. Rysunek 4.17 przedstawia także zbieżność algorytmu. Można zauważyć, że algorytm potrzebuje nieco powyżej 60 kroków do rozwiązania tego zadania. Przedstawione tutaj wyniki będą stanowiły materiał do porównań.



Rysunek 4.15 Ścieżka ruchu poduszkowca (tylko S_1)



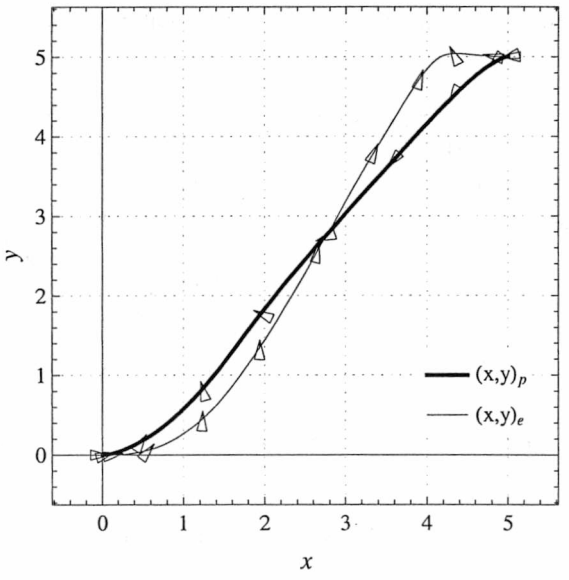
Rysunek 4.16 Trajektorie zmiennych stanu (tylko S_1)



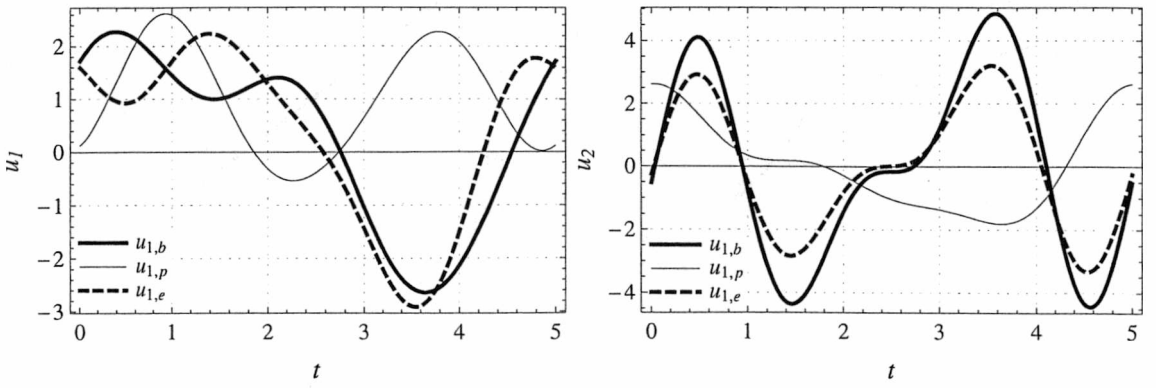
Rysunek 4.17 Przebiegi funkcji sterujących – strona lewa, zbieżność algorytmu – strona prawa (tylko S_1)

4.2.2 Wyniki symulacji zadań S_1 i S_2

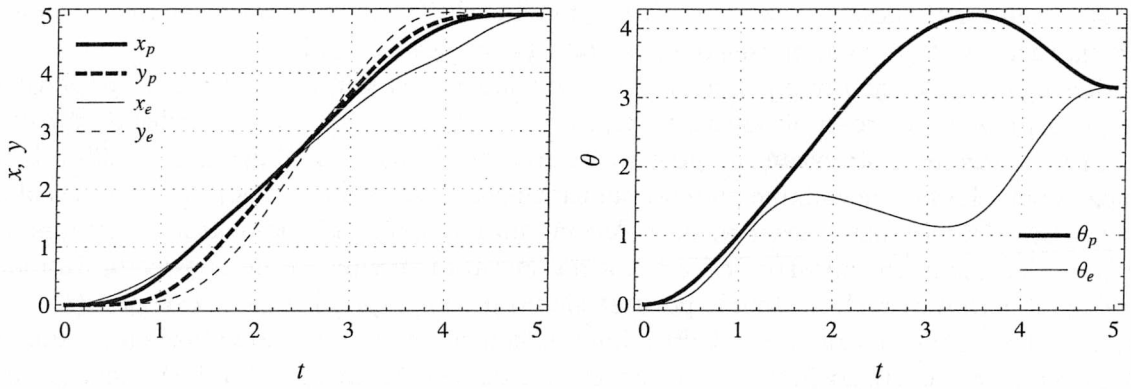
Kolejny zestaw składa się z dwóch zadań, zadania właściwego planowania ruchu i dodatkowego zadania minimalizacji energii sterowania. Będzie on rozwiązywany przy pomocy dwóch algorytmów, algorytmu egalitarnego i z priorytetowaniem zadań. Wyniki uzyskane w symulacjach ilustrują rysunki 4.18–4.22. Oba algorytmy poprawnie rozwiązują pierwsze zadanie, co można zaobserwować na rysunkach 4.20 i 4.21. Niemniej jednak, ścieżki ruchu robota nieco się różnią (rysunek 4.18). Istotne różnice w wynikach uzyskanych z obu algorytmów pojawiają się w rozwiązaniu zadania dodatkowego. Na rysunku 4.19, oprócz funkcji sterujących wyznaczonych przez oba algorytmy, zamieściliśmy także funkcje wyznaczone przez algorytm rozwiązujący zadanie S_1. Można zauważyć, że zarówno algorytm egalitarny, jak i algorytm z priorytetowaniem zadań wyznaczyły sterowanie o mniejszej energii niż wyznaczone w podrozdziale 4.2.1. Fakt ten można także zaobserwować na wykresie zbieżności błędu ^2e zamieszczonym na rysunku 4.22. Z tego samego rysunku można wywnioskować, że algorytm z priorytetowaniem uzyskał mniejszą energię sterowania niż algorytm egalitarny. Z drugiej strony, algorytm egalitarny potrzebował około 20 kroków mniej na poprawne rozwiązanie postawionego zadania.



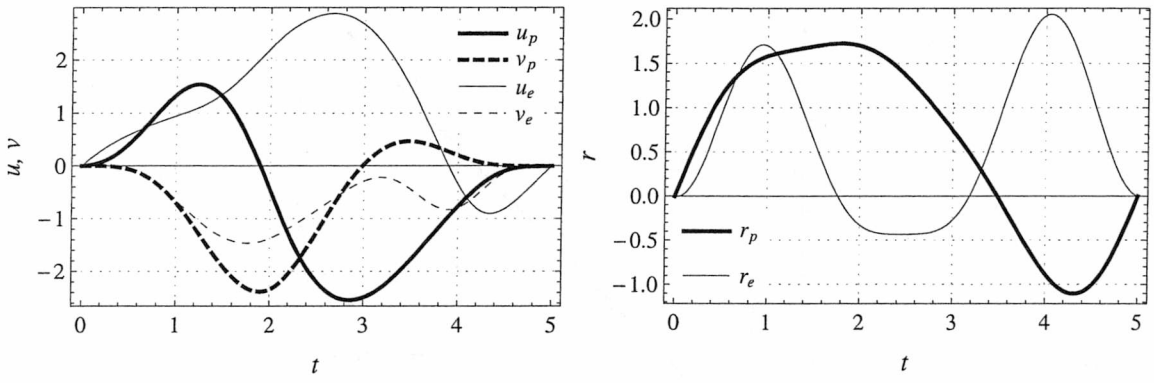
Rysunek 4.18 Ścieżka ruchu poduszkowca (S_1 i S_2)



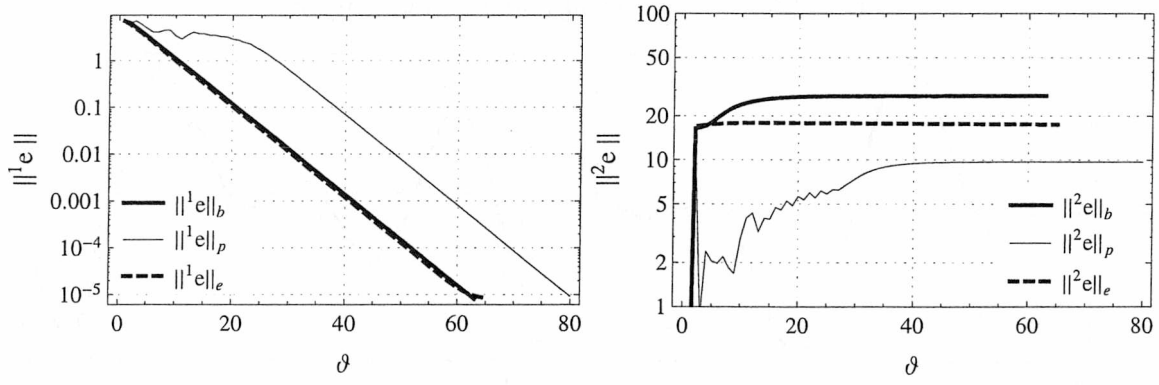
Rysunek 4.19 Przebiegi funkcji sterujących (S_1 i S_2)



Rysunek 4.20 Trajektorie zmiennych stanu – położenia (S_1 i S_2)



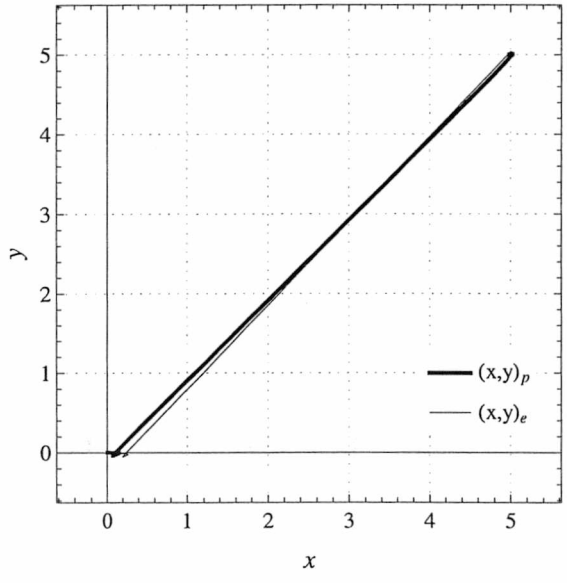
Rysunek 4.21 Trajektorie zmiennych stanu – prędkości (S_1 i S_2)



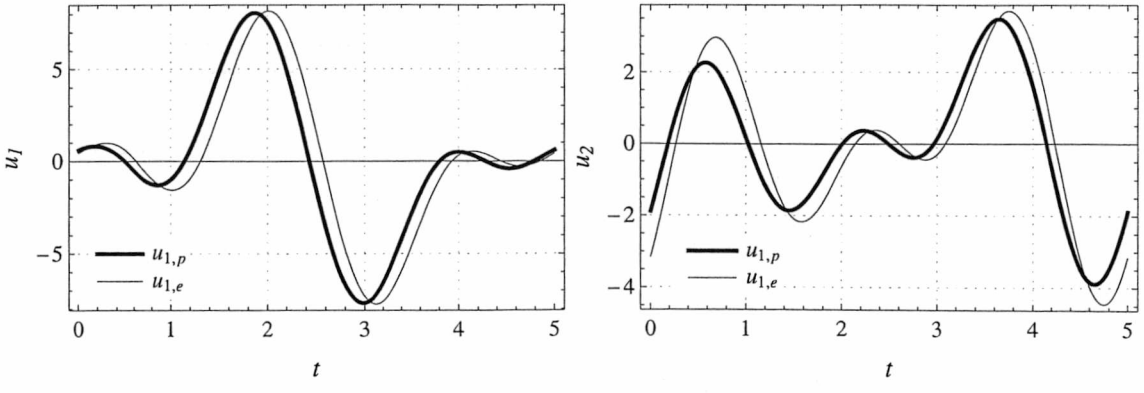
Rysunek 4.22 Zbieżność algorytmu (S_1 i S_2)

4.2.3 Wyniki symulacji zadań S_1 i S_3

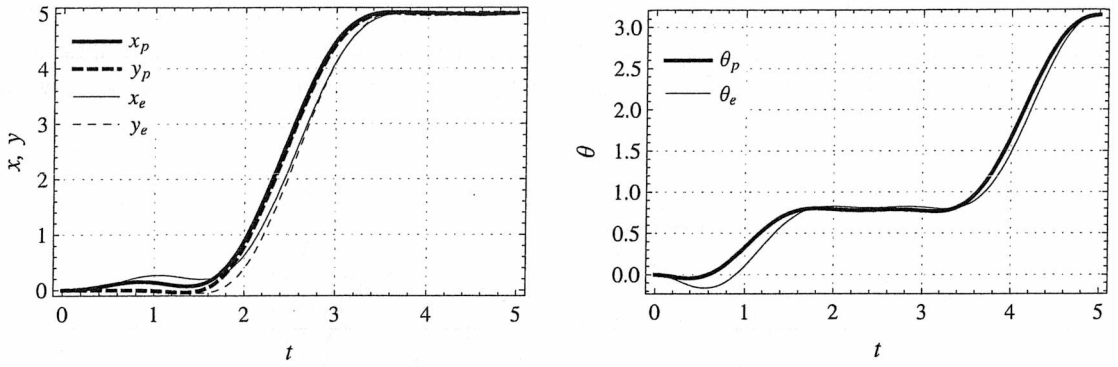
Zestaw zadań składający się z zadań S_1 i S_3 ma na celu, poza właściwym zadaniem planowania ruchu polegającym na osiągnięciu stanu x_d w czasie T, zminimalizowanie wybranej zmiennej stanu. W tym przypadku chcemy, aby zmienna stanu, $v(t)$ – prędkość poprzeczna, przyjmowała wartości bliskie zeru. W praktyce oznacza to, że poduszkowiec pod wpływem wyznaczonych sterowań nie powinien poruszać się bokiem. Z rysunków 4.25 i 4.26 można wnioskować, że zadanie planowania ruchu zostało rozwiązane poprawnie przez oba algorytmy. Obserwując ścieżki ruchu pokazane na rysunku 4.23 i trajektorie z rysunków 4.25 i 4.26 można wywnioskować, że robot na początku zmienia orientację wykorzystując prędkość obrotową $r(t)$, następnie porusza się po prostej z prędkością wzdłużną $u(t)$ i na końcu znowu zmienia orientację. Fakt minimalizacji zmiennej $v(t)$ widać także na rysunku 4.26. Podobne przebiegi funkcji sterujących przedstawione na rysunku 4.24 sugerują, że oba algorytmy wyznaczyły podobne ruchy. Tak postawione zadanie planowania okazało się stosunkowo trudne do rozwiązania przez oba algorytmy, widać to w niemonotonicznych przebiegach zbieżności algorytmów na rysunku 4.27. Przyczyną uzyskiwanych przebiegów zbieżności jest kwestia doboru współczynnika γ, którego wybór w tym miejscu był kom-



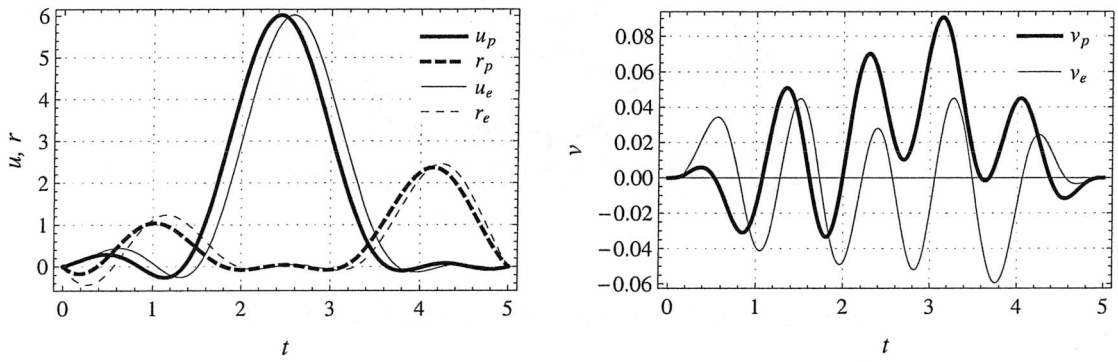
Rysunek 4.23 Ścieżka ruchu poduszkowca (S_1 i S_3)



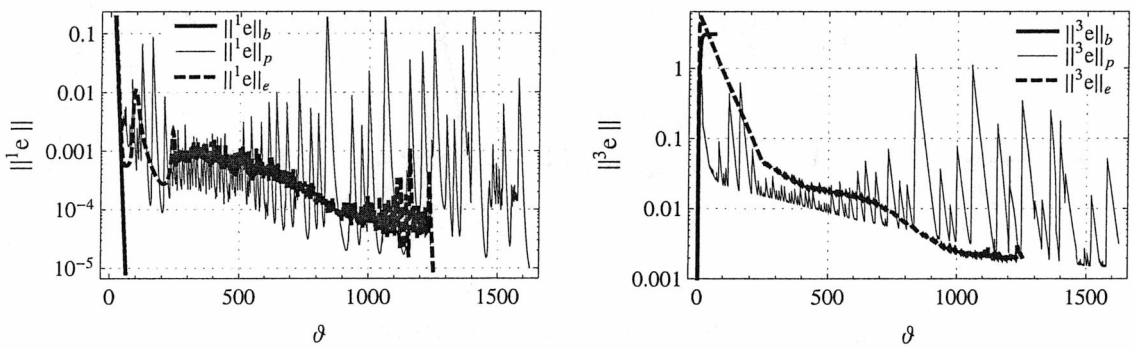
Rysunek 4.24 Przebiegi funkcji sterujących (S_1 i S_3)



Rysunek 4.25 Trajektorie zmiennych stanu – położenia (S_1 i S_3)



Rysunek 4.26 Trajektorie zmiennych stanu – prędkości (S_1 i S_3)



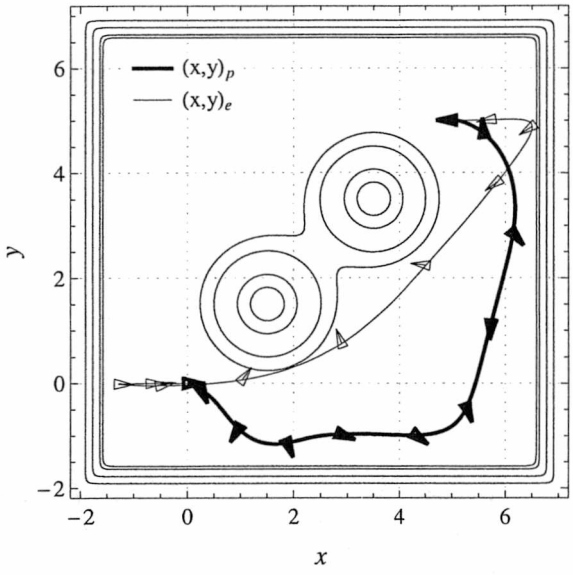
Rysunek 4.27 Zbieżność algorytmu (S_1 i S_3)

promisem pomiędzy długością kroku i szybkością zbieżności. Oba algorytmy osiągnęły podobny poziom błędu zadania $\|^3e\|$ i w obu przypadkach uzyskaliśmy rozwiązanie po ponad 1000 kroków algorytmu.

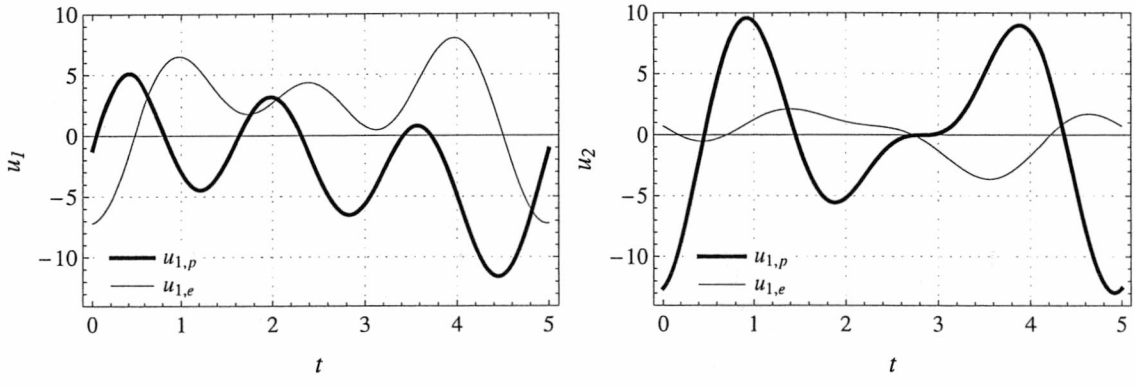
4.2.4 Wyniki symulacji zadań S_1 i S_4

Ostatnie dwie symulacje dotyczą zadania planowania ruchu z omijaniem przeszkód. Podobnie jak poprzednio, uzyskamy rozwiązanie zadania przy pomocy algorytmu egalitarnego i algorytmu z priorytetowaniem zadań. Obszar ruchu ograniczyliśmy kwadratem, którego boki stanowią funkcje $x = y = 6.5$ oraz $x = y = -1.5$, a wewnątrz umieściliśmy

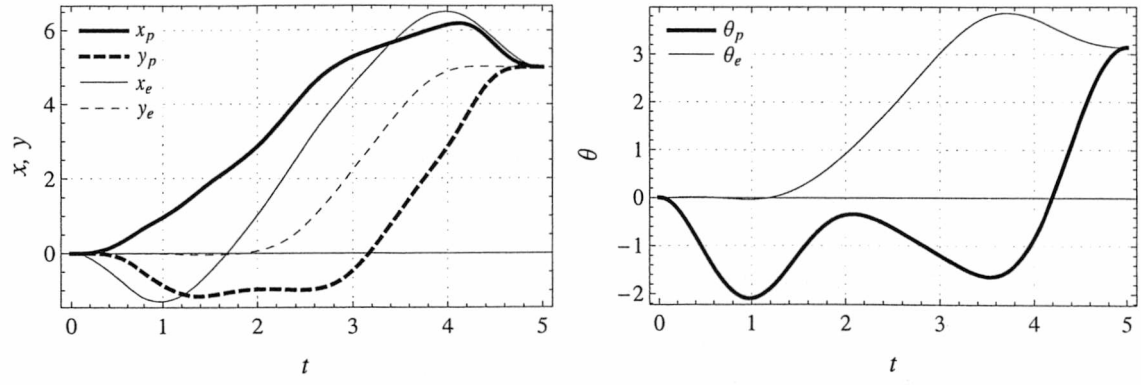
dwie koliste przeszkody o środkach w punktach (1.5, 1.5) i (3.5, 3.5). Rysunki 4.28–4.33 obrazują uzyskane wyniki symulacji. Ścieżka ruchu robota mobilnego jest przedstawiona na rysunku 4.28, zaznaczyliśmy tam także przeszkody oraz sylwetkę poduszkowca w od-



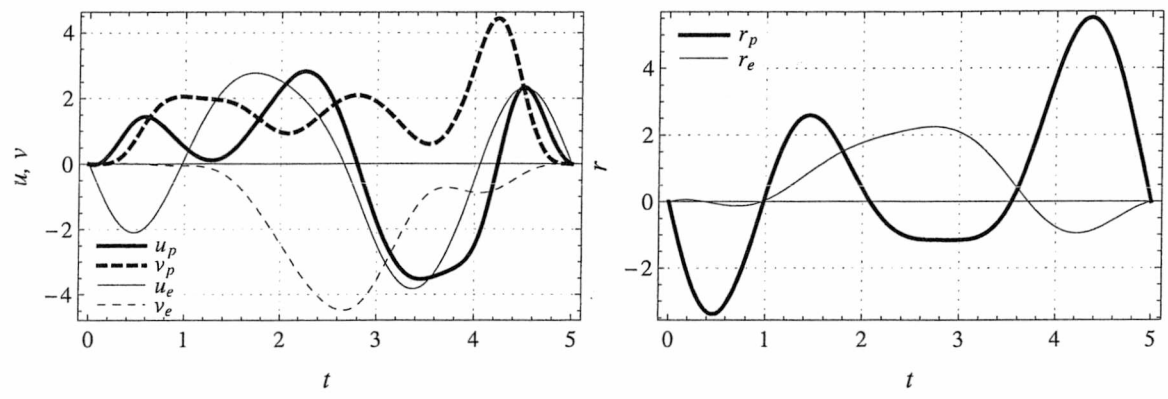
Rysunek 4.28 Ścieżka ruchu poduszkowca (S_1 i S_4)



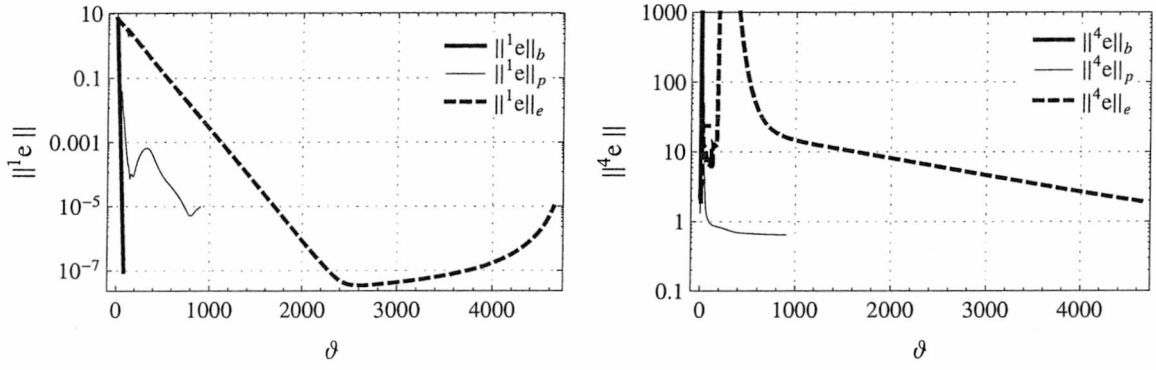
Rysunek 4.29 Przebiegi funkcji sterujących (S_1 i S_4)



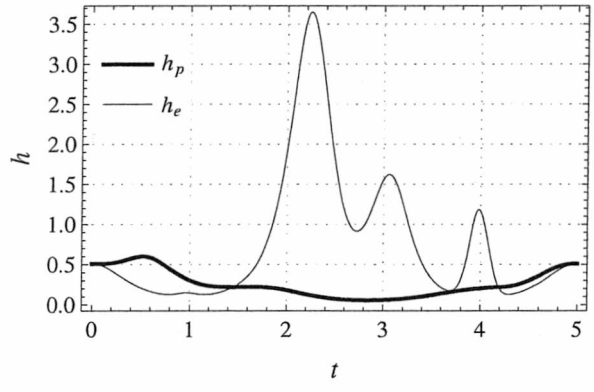
Rysunek 4.30 Trajektorie zmiennych stanu – położenia (S_1 i S_4)



Rysunek 4.31 Trajektorie zmiennych stanu – prędkości (S_1 i S_4)



Rysunek 4.32 Zbieżność algorytmu (S_1 i S_4)



Rysunek 4.33 Przebieg funkcji przeszkód $h(x)$ wzdłuż trajektorii $x(t)$

stępie co 0.5 jednostki czasu. Widać, że algorytmy wyznaczyły różne ścieżki. Kolejny rysunek 4.29 przedstawia funkcje sterujące. Z rysunków 4.30 i 4.31 wynika, że zadanie właściwego planowania ruchu zostało rozwiązane poprawnie. Na rysunku 4.33 przedstawiliśmy przebieg funkcji przeszkód $h(x)$ wzdłuż trajektorii $x(t)$. Na podstawie tego wykresu można stwierdzić, jak blisko przeszkód przebiega wyznaczona trajektoria; im większa wartość funkcji $h(x)$, tym robot znajduje się bliżej przeszkody. Wykorzystując analogię funkcji przeszkód do ukształtowania terenu wykres przedstawia wysokość wzdłuż trajektorii. W tym przypadku lepiej wypada algorytm z priorytetowaniem, niemniej jednak algorytm egalitarny także zapewnia omijanie przeszkód. Rysunek 4.32 prezentuje zbież-

ność algorytmów. W symulacji nie zatrzymywaliśmy pracy algorytmu, po zmniejszeniu się błędu zadania głównego poniżej zadanej wartości, ale pozwoliliśmy na dalszy spadek błędu zadania omijania przeszkód. W efekcie, zarówno algorytm egalitarny jak i algorytm z priorytetowaniem uzyskały zbliżone wartości odwzorowania końcowego zadania S_4.

4.2.5 Podsumowanie

W poprzednich podrozdziałach przedstawiliśmy rozwiązania trzech różnych zestawów zadań, z których każde zostało uzyskane algorytmem egalitarnym i algorytmem z priorytetowaniem zadań. W tabeli 4.1 zestawiliśmy wartości odwzorowań końcowych (błędów zadaniowych). Odwzorowanie końcowe może tutaj pełnić rolę miary jakości, im mniejsza jego wartość, tym lepiej algorytm rozwiązuje dane zadanie. Jak się można było spodziewać, algorytmy dedykowane poszczególnym zadaniom rozwiązały je zdecydowanie lepiej niż pozostałe. Korzystając z wartości zawartych w tabeli, podsumujmy porównanie algorytmu egalitarnego z algorytmem z priorytetowaniem. Zadanie planowania ruchu z ograniczeniem energii sterowania lepiej rozwiązał algorytm z priorytetowaniem; wartość odwzorowania jest mniejsza niż w przypadku algorytmu egalitarnego. Z drugiej strony, algorytm egalitarny potrzebował mniejszej liczby kroków do rozwiązania postawionego zadania. Rozwiązanie zadania planowania ruchu z jednoczesną minimalizacją zmiennej stanu wyznaczone przez algorytm egalitarny charakteryzuje się mniejszą wartością odwzorowania końcowego niż w przypadku algorytmu z priorytetowaniem. Dodatkowo, algorytm znalazł rozwiązanie w mniejszej liczbie kroków. Ostatnie zadanie, polegające na znalezieniu funkcji sterującej, która przeprowadzi poduszkowiec do punktu końcowego z omijaniem przeszkód, zdecydowanie lepiej rozwiązał algorytm z priorytetowaniem. Zarówno wartość odwzorowania końcowego jak i liczba potrzebnych kroków jest mniejsza niż w przypadku algorytmu egalitarnego. Reasumując, w powyższym porównaniu trudno o jednoznaczne wskazanie, który algorytm jest lepszy. Oba algorytmy poprawnie rozwiązują zadanie planowania ruchu, a różnice między uzyskiwanymi wynikami nie są duże. Istotną różnicę stanowi natomiast sposób działania obu algorytmów. Algorytm egalitarny traktuje wszystkie zadania jako równorzędne. z tego powodu możliwe są dwa wyniki: albo uzyskamy poprawne rozwiązanie wszystkich zadań składowych, albo nie uzyskamy go w ogóle. Nieco inaczej działa algorytm z priorytetowaniem zadań. Tutaj rozwiązanie zadania o niższym priorytecie nie wpływa na rozwiązanie zadania o wyższym. Można zatem uzyskać rozwiązanie jednego (lub kilku) zadań o wyższym priorytecie i jednocześnie nie rozwiązać zadań o niższym priorytecie. Wybór algorytmu powinien być uzależniony od konkretnego zestawu zadań, oraz od tego, czy uznajemy zadania za równorzędne, czy też uszeregowane zgodnie z priorytetami.

Tabela. 4.1 Zestawienie wartości odwzorowań końcowych dla porównywanych algorytmów

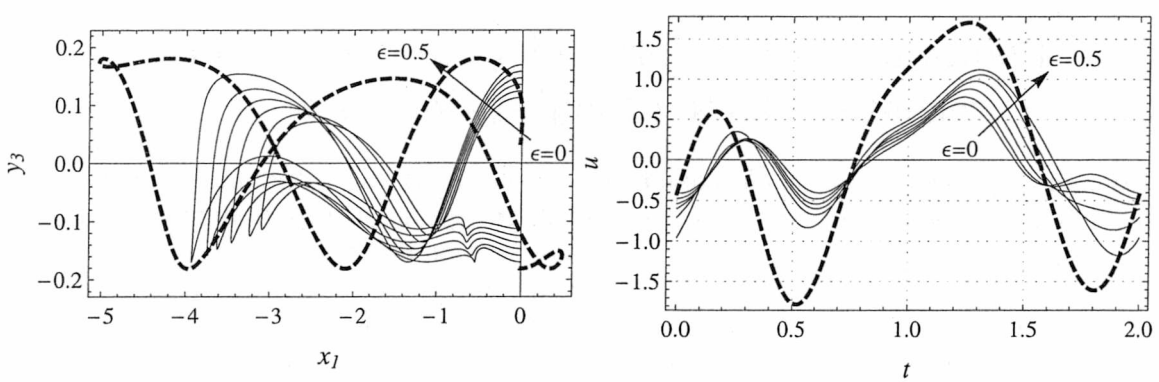
Algorytm		Odwzorowanie końcowe			$\theta_{\max}$
		${}^2K_{x_0,T}$	${}^3K_{x_0,T}$	${}^4K_{x_0,T}$	
S_1		27.3386	8.62341	23.8764	63
S_1 i S_2	Egalitarny	**16.7266**	6.91150	168.287	65
S_1 i S_2	Priorytety	**9.71420**	8.13275	11647.9	80
S_1 i S_3	Egalitarny	50.3382	**0.00205591**	30063.9	1252
S_1 i S_3	Priorytety	46.9494	**0.00315783**	19051.0	1626
S_1 i S_4	Egalitarny	57.2724	18.8452	**1.95060**	4641
S_1 i S_4	Priorytety	159.608	3.30677	**0.640784**	903

4.3 Planowanie przy niepełnej znajomości modelu

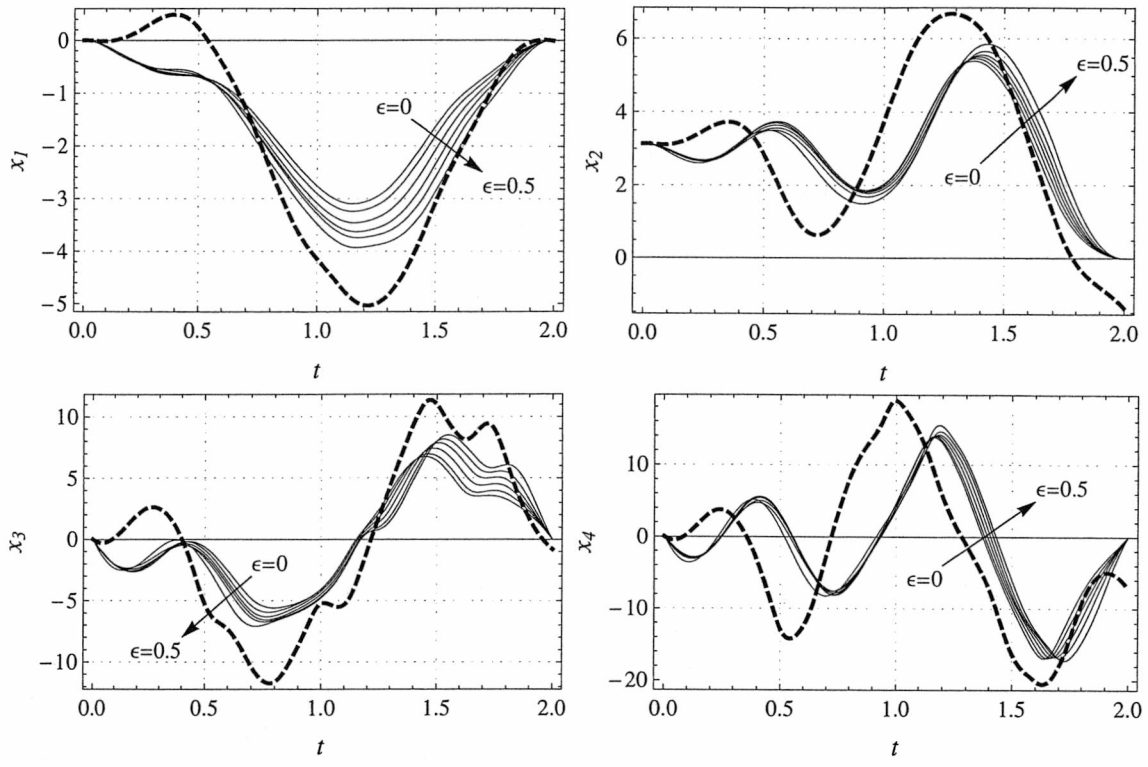
W niniejszym podrozdziale zajmiemy się badaniem algorytmu planowania zdefiniowanego równaniem (2.44). Taki sposób rozwiązania zadania planowania ruchu pozwala na uwzględnienie odchyłek pomiędzy rzeczywistymi parametrami obiektu a parametrami układu nominalnego. Jak zostało to szerzej opisane w rozdziale 2.4, algorytm wyznacza kolejne poprawki funkcji sterującej w oparciu o strukturę układu nominalnego oraz, w przeciwieństwie do pozostałych algorytmów przedstawianych w tej pracy, uwzględnia stan układu rzeczywistego. Wyniki symulacji, które przedstawiamy poniżej, prezentują rozwiązanie zadania planowania ruchu nieplanarnego dwuwahadła $R\overline{R}$ (wahadło Furuty) przy niepełnej znajomości modelu. Model tego manipulatora wyprowadziliśmy w dodatku C.4. Zakładamy, że wartość każdego parametru modelu zostaje zaburzona w następujący sposób

$$\hat{a} = a + \epsilon a.$$

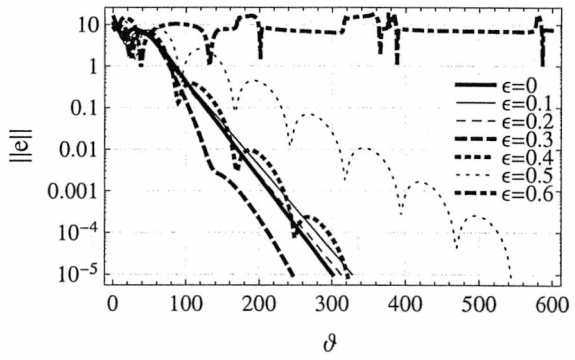
W symulacjach przyjęliśmy, że współczynnik ϵ przyjmuje wartości $\epsilon \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6\}$. Wraz ze wzrostem współczynnika ϵ, układ nominalny coraz bardziej różni się od układu rzeczywistego. Zadanie planowania ruchu polega na znalezieniu funkcji sterującej, która wykona manewr „podrzutu" wahadła ze stabilnego punktu równowagi $x_0 = (0, \pi, 0, 0)$ do niestabilnego punktu równowagi $y_d = x_d = (0, 0, 0, 0)$. Punkt x_d chcemy osiągnąć w czasie ruchu $T = 2$ z dokładnością 10^{-5}. Funkcje sterujące reprezentujemy przy pomocy skończonego szeregu Fouriera (3.3) przy $k = 8$, początkowy wektor sterowań $c(0) = (0.1, 0_{1\times 8})$, a współczynnik zbieżności $\gamma = 0.1$. Wyniki symulacji przedstawiono na rysunkach 4.34–4.36. Na rysunkach 4.34 i 4.35 linią ciągłą zaznaczyliśmy elementy związane z wartościami parametru $0 \leqslant \epsilon \leqslant 0.5$, natomiast linią przerywaną zaznaczony jest przypadek $\epsilon = 0.6$. Widać (rysunki 4.34 i 4.35), że do wartości $\epsilon = 0.5$ zadanie planowania ruchu jest rozwiązywane prawidłowo, dopiero po przekroczeniu tej wartości otrzymanie poprawnego rozwiązania staje się niemożliwe. Fakt ten można także zauważyć na rysunku 4.36. Tylko przypadek $\epsilon = 0.6$ nie uzyskał zadanej wartości błędu $\|e\|$. Przedstawione wyniki pokazują efektywność algorytmu planowania ruchu z uwzględnieniem stanu układu rzeczywistego, który poradził sobie z zadaniem planowania ruchu przy niepełnej znajomości modelu. Nawet w przypadku, gdy wszystkie wartości parametrów układu nominalnego znacznie (o 50%) różniły się od parametrów rzeczywistych, algorytm rozwiązał zadanie planowania ruchu z zadaną dokładnością.



Rysunek 4.34 Rzut ścieżki ruchu końca efektora na siatkę walca o promieniu l_1 – strona lewa, przebiegi funkcji sterującej – strona prawa



Rysunek 4.35 Trajektorie zmiennych stanu

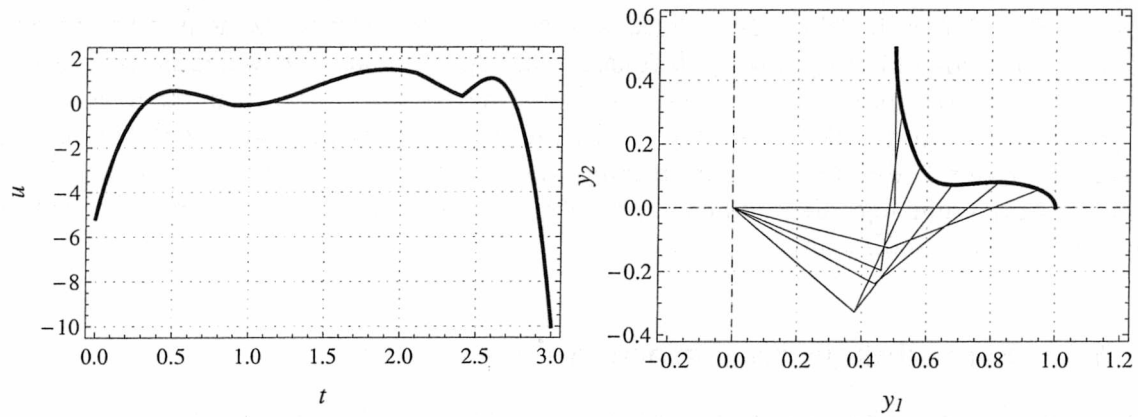


Rysunek 4.36 Zbieżność algorytmu

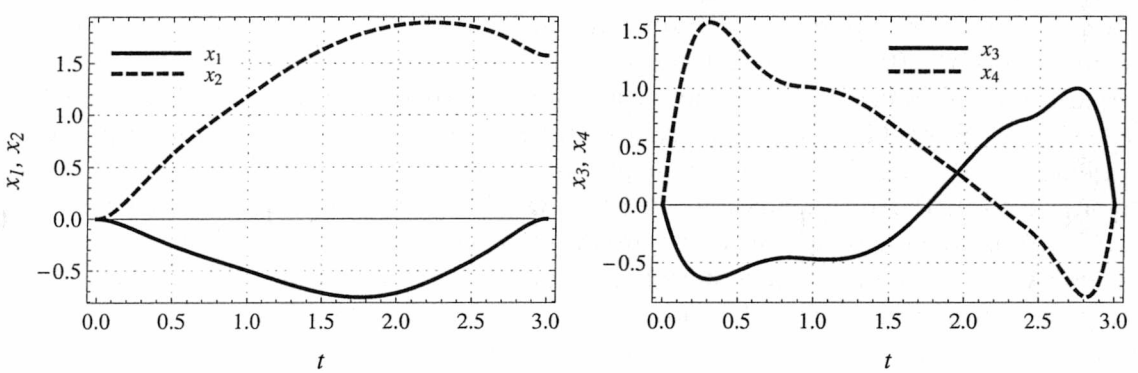
4.4 Planowanie ruchu przy nieparametrycznej reprezentacji konfiguracji endogenicznej

Zajmijmy się teraz rozwiązaniem zadania planowania ruchu przy nieparametrycznej reprezentacji konfiguracji endogenicznych. Oznacza to, że będziemy bezpośrednio rozwiązywać równanie (2.15) korzystając z postaci odwrotności jakobianu (2.9), bez wprowadzania parametrycznej reprezentacji funkcji (3.1). Praca algorytmu polega na iteracyjnym rozwiązywaniu równania (3.5), do momentu uzyskania zadanej wartości błędu planowania ruchu. Jako model testowy wybraliśmy planarne dwuwahadło, na ruch którego nie wpływa siła grawitacji (patrz dodatek C.5). Chcemy zaplanować ruch pomiędzy punktem spoczynkowym $x_0 = (0, 0, 0, 0)$, a punktem spoczynkowym $y_d = x_d = (0, \pi/2, 0, 0)$. Ruch ma trwać $T = 3$, a punkt docelowy x_d ma zostać osiągnięty z dokładnością 10^{-5}. Współczyn-

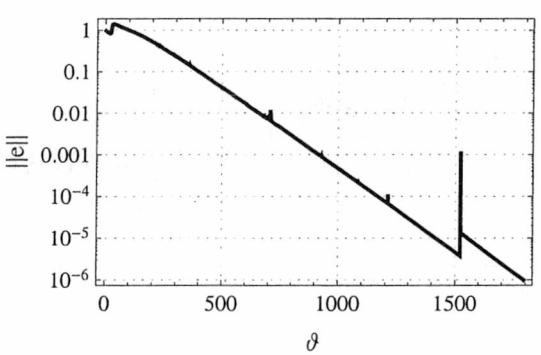
nik zbieżności jest równy $\gamma = 0.01$, a początkową funkcję sterującą wybraliśmy równą $u_0(t) = -0.1$. Funkcję sterującą uzyskaną w wyniku zastosowania algorytmu przedstawia rysunek 4.37. W odróżnieniu od poprzednich wyników, gdzie funkcje sterujące były reprezentowane przez bazy ortogonalne, przebieg z rysunku przyjmuje odmienny kształt. Ścieżkę ruchu końca efektora w przestrzeni zadaniowej pokazano na rysunku 4.37. Sylwetka manipulatora na tym rysunku została wyznaczona w odstępie co 0.5 jednostki czasu. Wykresy trajektorii zmiennych stanu 4.38 potwierdzają poprawność rozwiązania zadania planowania ruchu, a zbieżność algorytmu ilustruje rysunek 4.39.



Rysunek 4.37 Przebieg funkcji sterującej – strona lewa, ścieżka ruchu końca efektora wraz z sylwetką manipulatora – strona prawa



Rysunek 4.38 Trajektorie zmiennych stanu



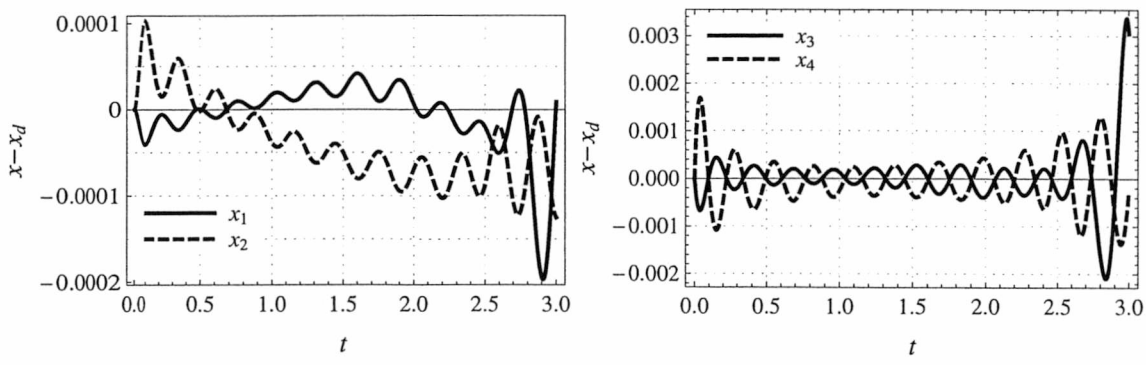
Rysunek 4.39 Zbieżność algorytmu

4.5 Wykorzystanie zadania ograniczania stanu do reprodukcji trajektorii

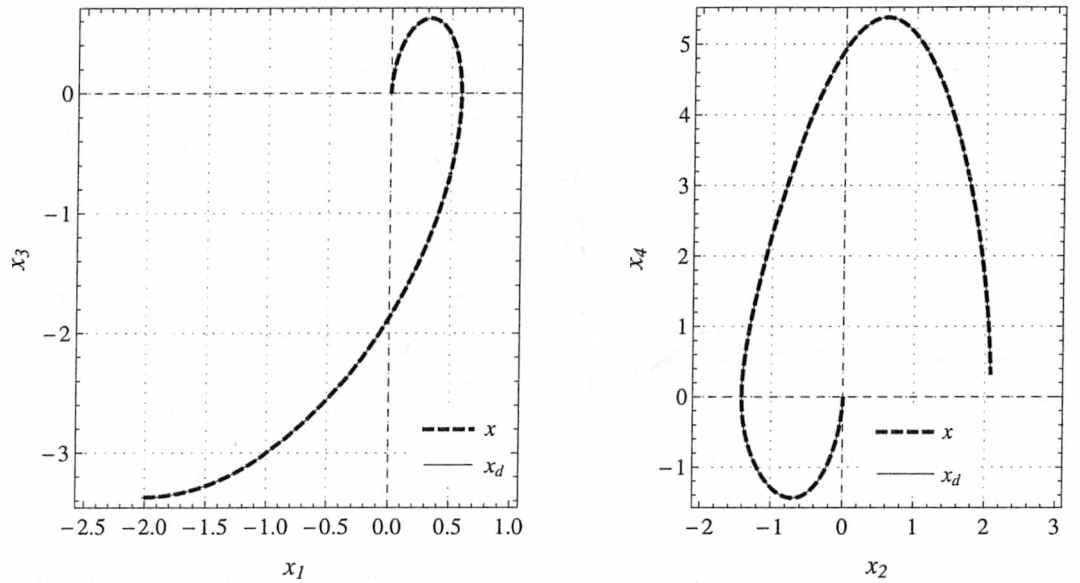
Ostatnie wyniki symulacji będą dotyczyły zadania reprodukcji trajektorii. Przez zadanie reprodukcji trajektorii rozumiemy znalezienie funkcji sterującej, która będzie wpływać na układ robotyczny w taki sposób, aby robot początkowo znajdujący się na zadanej trajektorii, poruszał się wzdłuż tej trajektorii przez zadany czas T. Zadaniem głównym dotychczasowych algorytmów było zadanie planowania ruchu. Żądaliśmy, aby w końcowej chwili T układ znajdował się w zadanym położeniu y_d. W tym podrozdziale, zadaniem głównym i jedynym będzie zadanie odtwarzania trajektorii wprowadzone pod koniec rozdziału 2.3.2. Poszukiwaną funkcję sterującą, realizującą zadanie odtwarzania trajektorii, otrzymamy wyznaczając granicę $\lim_{\vartheta\to\infty} u_\vartheta(\cdot)$ rozwiązania równania różniczkowego (2.15) z pseudoodwrotnością jakobianu (2.37) i błędem (2.38). Tak zdefiniowane zadanie rozwiążemy dla dwuwahadła z pasywnym drugim przegubem, opisanego w dodatku C.5. Przeprowadzimy symulacje dla zadania reprodukcji trajektorii, zarówno we współrzędnych stanu, jak i we współrzędnych zadaniowych.

4.5.1 Reprodukcja trajektorii zmiennych stanu

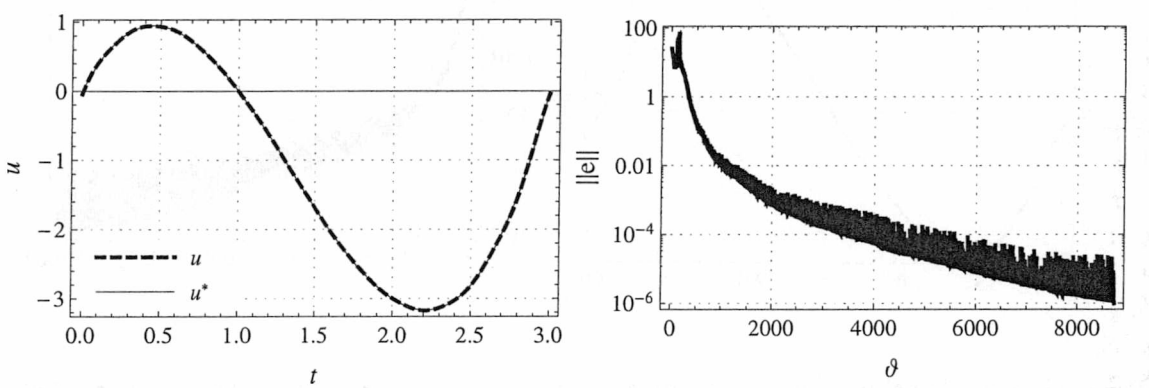
Aby uzyskać trajektorię osiągalną przez układ dwuwahadła, podamy na jego wejście pewną funkcję sterującą $u^*(t)$. Następnie, trajektoria $x(t)$ uzyskana z układu będzie trajektorią zadaną dla naszego zadania. Jako funkcję sterującą wybierzmy wielomian $u^*(t) = 1.5(3t - 4t^2 + t^3)$ określony na czasie trwania ruchu $T = 3$. Poszukiwane funkcje będą reprezentowane poprzez skończony szereg Fouriera (3.3) dla $k = 20$, a początkowy wektor sterowań $c(0) = (1, 0_{1\times 20})$. Współczynnik zbieżności jest równy $\gamma = 0.05$, a obliczenia będą prowadzone do momentu uzyskania wartości błędu reprodukcji trajektorii $\|e\| \leqslant 10^{-6}$. Rysunki 4.40–4.42 pokazują wyniki symulacji. Chwilowy błąd reprodukcji trajektorii $x(t) - x_d(t)$ w najgorszym przypadku nieznacznie przekracza wartość 0.003 (rysunek 4.40). Portrety fazowe ruchu przedstawiliśmy na rysunku 4.41, zaznaczyliśmy tam zarówno ruch zadany, jak i otrzymany za pomocą algorytmu. Jak się można było spodziewać, otrzymana funkcja sterująca przybliża zadaną funkcję $u^*(t)$ (rysunek 4.42). Przyczyną niemonotonicznej zbieżności algorytmu jest dobór parametru γ, którego wartość zapewnia tutaj wystarczająco mały krok przy akceptowalnej prędkości zbieżności. Rozwiązanie zadania uzyskujemy po niespełna 9000 krokach algorytmu.



Rysunek 4.40 Przebiegi błędu reprodukcji trajektorii we współrzędnych stanu



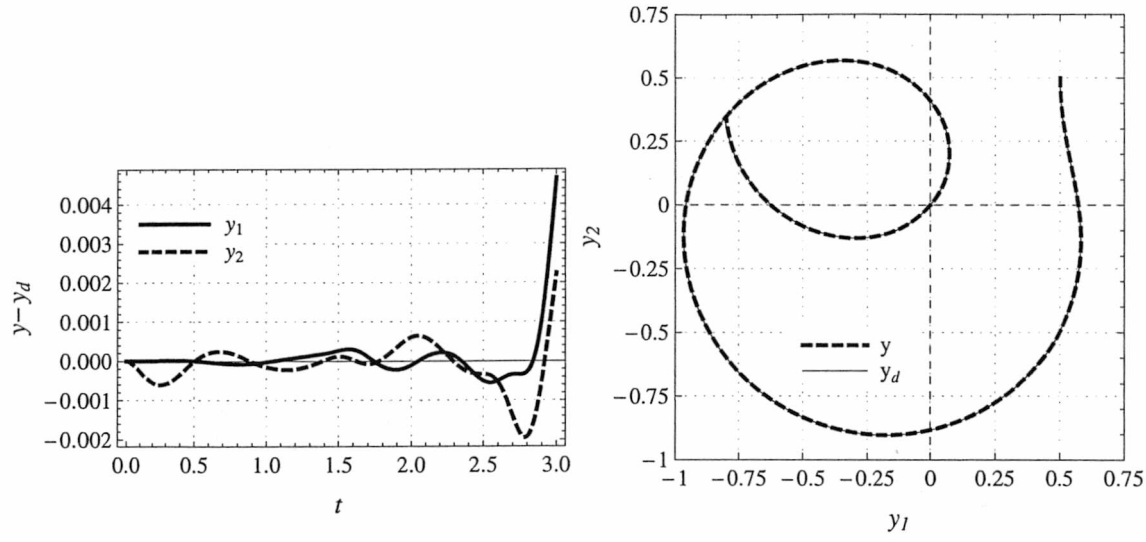
Rysunek 4.41 Portrety fazowe ruchu



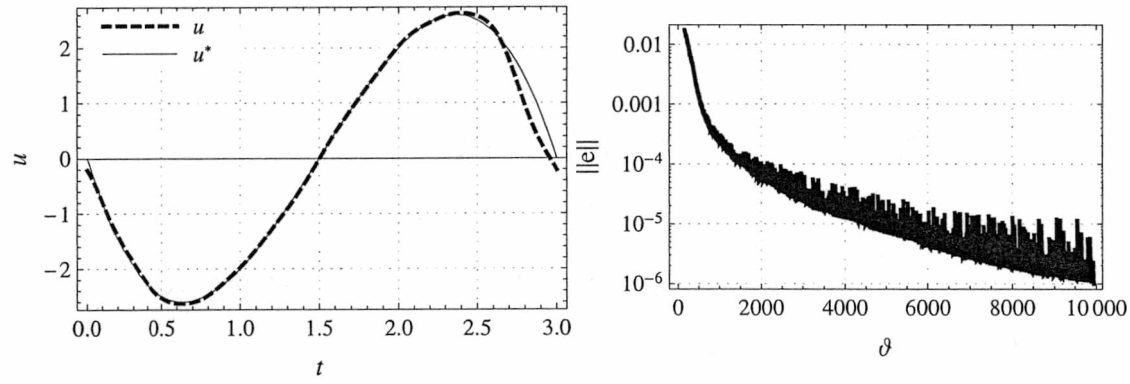
Rysunek 4.42 Przebieg funkcji sterującej – strona lewa, zbieżność algorytmu – strona prawa

4.5.2 Reprodukcja trajektorii zmiennych zadaniowych

W tym podrozdziale rozwiążemy zadanie podobne do poprzedniego, lecz trajektoria zadana będzie określona w przestrzeni zadaniowej. Podobnie jak poprzednio, osiągalną (dopuszczalną) trajektorię zadaną otrzymamy poprzez podanie sterowania $u^*(t)$ na układ dwuwahadła, a następnie wyznaczymy przebieg funkcji wyjścia $y(t)$, która będzie trajektorią zadaną $y_d(t)$. Wybierzmy $u^*(t) = -2(4.5t - 4.5t^2 + t^3)$. Funkcje sterujące będą reprezentowane dokładnie tak, jak w poprzednim podrozdziale, a początkowy wektor sterowań $c(0) = (0, -0.1, 0_{1\times 19})$. Pozostałe parametry algorytmu są identyczne z poprzednią symulacją. Błąd reprodukcji trajektorii przedstawiono na rysunku 4.43. Widać, że nie przekracza on wartości 0.005. Ponadto na rysunku 4.43 przedstawiliśmy ścieżkę ruchu końca efektora w przestrzeni zadaniowej. Funkcję sterującą i zbieżność algorytmu prezentuje rysunek 4.44. Uzyskana funkcja sterująca przybliża funkcję $u^*(t)$. Zadaną dokładność rozwiązania uzyskujemy po około 10000 krokach.



Rysunek 4.43 Przebieg błędu reprodukcji trajektorii we współrzędnych zadaniowych – strona lewa, ścieżka ruchu końca efektora – strona prawa



Rysunek 4.44 Przebieg funkcji sterującej – strona lewa, zbieżność algorytmu – strona prawa

Zarówno w przestrzeni stanu, jak i w przestrzeni zadaniowej, uzyskaliśmy pozytywne wyniki działania algorytmu reprodukcji trajektorii. Otrzymane funkcje sterujące prowadzą robota wzdłuż zadanej trajektorii podczas trwania ruchu. Wartości błędów zadania reprodukcji przyjmują akceptowalne wartości, które mogą zostać jeszcze bardziej zmniejszone przez rozszerzenie bazy funkcji ortogonalnych. Przebieg poszukiwanej funkcji $u^*(t)$ jest przybliżany przez otrzymaną funkcję sterującą przy pomocy wybranej bazy. Wyznaczenie pojedynczego kroku algorytmu reprodukcji trajektorii wymaga większych nakładów obliczeniowych niż ma to miejsce w algorytmie planowania ruchu, z tego względu rozwiązanie zadania reprodukcji jest bardziej pracochłonne.

Rozdział 5

Podsumowanie

W pracy rozważyliśmy zagadnienie planowania ruchu układów robotycznych z deficytem napędów. Tytułem wstępu wprowadziliśmy pojęcia związane z modelowaniem tego typu układów. Następnie, przedstawiliśmy wyprowadzenie algorytmu planowania ruchu skonstruowanego w oparciu o metodę endogenicznej przestrzeni konfiguracyjnej. Do budowy algorytmu niezbędnym było zdefiniowanie odwzorowania końcowego i jakobianu układu robotycznego z deficytem napędów. W dalszej części zaproponowaliśmy dwa algorytmy realizujące, wraz z właściwym zadaniem planowania ruchu, zestaw zadań dodatkowych: algorytm egalitarny i algorytm z priorytetowaniem zadań. Algorytmy te różnią się wpływem zadań dodatkowych na uzyskiwane rozwiązanie. W algorytmie egalitarnym zadania dodatkowe są równorzędne z zadaniem właściwego planowania ruchu, natomiast w algorytmie z priorytetowaniem zadań zadania dodatkowe są uszeregowane według priorytetów. W rozprawie zostały rozważone trzy typy zadań dodatkowych: zadanie minimalizacji energii sterowania, zadanie minimalizacji zmiennych stanu i zadanie unikania kolizji z przeszkodami. Przy definiowaniu zadań dodatkowych podajemy różne możliwości modyfikacji ich definicji, co pozwala na rozwiązanie zróżnicowanych zadań planowania ruchu. Oryginalny algorytm planowania ruchu wyprowadzony w pracy jest algorytmem planowania ruchu z uwzględnieniem stanu układu nominalnego, który pracuje w otwartej pętli sprzężenia zwrotnego. Sposobem na uniknięcie tej niedogodności może być algorytm planowania ruchu z uwzględnieniem stanu układu rzeczywistego [19]. W pracy zbadaliśmy działanie takiego algorytmu co pozwoliło rozwiązać zadanie planowania ruchu przy niepełnej znajomości modelu. Wyprowadzenia teoretyczne przeprowadzamy w funkcyjnej przestrzeni endogenicznej. Na potrzeby obliczeń numerycznych została wprowadzona parametryczna reprezentacja funkcji sterujących. W rozprawie proponujemy także rozwiązanie zadania planowania ruchu przy nieparametrycznej reprezentacji. Wydaje się, że jak dotąd nie próbowano takiego podejścia.

Wśród zalet przedstawionych algorytmów planowania ruchu należy wymieć to, że w klasie układów afinicznych, algorytm jest niezależny od typu modelu. Jak wskazywaliśmy we Wstępie, często spotyka się w literaturze algorytmy dedykowane specyficznym układom z deficytem napędów. Wyprowadzenie prezentowanych algorytmów nie opiera się na szczególnych własnościach modelu, z wyjątkiem założenia nieosobliwości konfiguracji endogenicznych, tj. sterowalności przybliżenia liniowego układu wokół trajektorii, co w przypadku braku kryteriów sterowalności wydaje się być sensownym założeniem. Korzystanie z metody endogenicznej przestrzeni konfiguracyjnej umożliwia nam stosowanie metodologii wyprowadzonych dla manipulatorów stacjonarnych do rozwiązywania problemów dotyczących innych modeli robotycznych. Wreszcie, uwzględnienie zadań dodatkowych, które są realizowane wraz z właściwym zadaniem planowania ruchu umożliwia

uzyskiwanie pożądanych jakości rozwiązań. Poza zaprezentowanymi zadaniami dodatkowymi można definiować inne, jak na przykład zadanie minimalizujące poślizg w kołowych robotach mobilnych poprzez minimalizację normy $\int_0^T \|A(q)\dot{q}\|^2 \, \mathrm{d}t$, gdzie $A(q)$ oznacza macierz Pfaffa i wiele innych. Dodanie dodatkowych zadań minimalizujących sterowanie oraz zmienne stanu sprawia, że możliwa staje się implementacja algorytmu na układach rzeczywistych. Można bowiem uwzględnić w algorytmie skończone zakresy ruchu i skończone wartości prędkości poruszania się układów. Ponadto, użycie algorytmu planującego ruch przy niepełnej znajomości modelu umożliwia uzyskanie zadowalających wyników bez konieczności dokładnej identyfikacji parametrów robota.

Oczywiście, prezentowane algorytmy nie są wolne od wad. Zasadniczym problemem jest ich lokalność. Uzyskanie poprawnego rozwiązania w znacznym stopniu zależy od wyboru wartości początkowych wektora c_0, w przypadku algorytmów z bazą w endogenicznej przestrzeni konfiguracyjnej oraz od początkowej funkcji sterującej u_0 w algorytmach z nieparametryczną reprezentacją sterowań. Uzyskanie rozwiązania zależy także od wyboru zadań dodatkowych. W przypadku algorytmu egalitarnego wpływ zadań dodatkowych na rozwiązanie zadania głównego jest duży. Inaczej wygląda sytuacja w przypadku algorytmu z priorytetowaniem. Tutaj możliwe jest uzyskanie rozwiązania zadania lub zadań o wyższych priorytetach kosztem zadań o niższych priorytetach.

Efektywność wyprowadzanych w pracy algorytmów poddaliśmy badaniom symulacyjnym. Pierwsze trzy symulacje prezentują w działaniu kolejno trzy rodzaje zadań dodatkowych. W uzyskiwanych wynikach można zauważyć, jak zadania dodatkowe modyfikują rozwiązanie zadania planowania ruchu. Następna grupa symulacji dostarczyła materiał do porównania działania algorytmu egalitarnego i algorytmu z priorytetowaniem zadań. Analizując otrzymane wyniki, trudno jednoznacznie wskazać, który z dwóch algorytmów daje lepsze rozwiązanie. W ogólności, wyniki otrzymywane przy zastosowaniu obu algorytmów są zbliżone. Różnicę między algorytmem egalitarnym a algorytmem z priorytetowaniem zadań widać w sposobie działania algorytmów. W algorytmie egalitarnym błędy zadań składowych charakteryzują się podobną zbieżnością. Oznacza to, że algorytm poszukuje funkcję sterującą realizującą zadanie właściwego planowania ruchu i jednocześnie modyfikuje tę funkcję w sposób uwzględniający zadania dodatkowe. W algorytmie z priorytetami rozwiązanie głównego zadania, właściwego planowania ruchu, jest uzyskiwane po stosunkowo małej liczbie kroków algorytmu. Następnie, w kolejnych krokach, funkcja sterująca jest modyfikowana w celu rozwiązania zadań dodatkowych o niższych priorytetach. W kolejnych wynikach symulacji przedstawiamy działanie algorytmu planowania ruchu przy niepełnej znajomości modelu. Obliczenia numeryczne pokazują, że możliwe jest uzyskanie poprawnych rozwiązań, nawet gdy parametry modelu różnią się o 50% od parametrów rzeczywistych. W symulacjach przeprowadzanych przy nieparametrycznej reprezentacji przestrzeni endogenicznej pokazujemy, że możliwe jest rozwiązanie zadania planowania ruchu bez użycia funkcji bazowych. Dzięki temu, charakter otrzymywanych funkcji sterujących jest niezależny od wybory bazy. Z drugiej strony, takie rozwiązanie komplikuje obliczenia. W przypadku parametrycznej reprezentacji należy w każdym kroku algorytmu wyznaczyć odpowiedź układu robotycznego na bieżące sterowanie, a następnie wyznaczyć odwrotność jakobianu, który w tym przypadku jest macierzą. W przypadku nieparametrycznej reprezentacji, oprócz wyznaczenia odpowiedzi układu należy również wyznaczyć trajektorię macierzy tranzycji $\Phi_{2n\times 2n}(T,t)$, a przy wyznaczaniu pseudoodwrotności trzeba całkować funkcje macierzowe. Ostatnie symulacje przedstawiają wyniki pracy algorytmu zadania reprodukcji trajektorii. Jeśli skonstruujemy algorytm planowania ruchu (2.15), z pseudoodwrotnością jakobianu (2.37) i błędem zadania (2.38) z podrozdziału 2.3.2 otrzymamy algorytm reprodukcji trajektorii. Obserwując otrzymane wyniki można

Dodatek A

Wybrane własności

Przytoczmy aksjomaty odwrotności Moore'a-Penrose'a [2], które zostaną wykorzystane w dowodach. Niech A będzie danym operatorem, A^+ jego odwrotnością, a A^* operatorem sprzężonym. Wówczas zachodzą cztery własności

$$AA^+A = A, \qquad A^+AA^+ = A^+, \qquad (A^+A)^* = A^+A, \qquad (AA^+)^* = AA^+.$$

Rzutowanie P na jądro jakobianu J jest zdefiniowane jako

$$P = \mathrm{id}_{\mathcal{U}} - J^{\#}J, \tag{A.1}$$

gdzie $\mathrm{id}_{\mathcal{U}}$ jest identycznością w $\mathcal{U}$, a $J^{\#}$ oznacza pseudoodwrotność J.

A.1 Idempotentność rzutowania P

Własność idempotentności definiujemy następująco

$$PP = P. \tag{A.2}$$

Przyjrzyjmy się lewej stronie równania (A.2)

$$PP = \left(\mathrm{id}_{\mathcal{U}} - J^{\#}J\right)\left(\mathrm{id}_{\mathcal{U}} - J^{\#}J\right) = \mathrm{id}_{\mathcal{U}} - 2J^{\#}J + J^{\#}\underbrace{JJ^{\#}J}_{J} = \mathrm{id}_{\mathcal{U}} - J^{\#}J = P,$$

co potwierdza słuszność (A.2).

A.2 Symetria rzutowania P

W przestrzeni Hilberta $\mathcal{U}$ zachodzi równość $\mathcal{U}^* = \mathcal{U}$, gdzie $\mathcal{U}^*$ jest przestrzenią dualną do $\mathcal{U}$. Symetria rzutowania wyraża się przez

$$P^* = P.$$

Rozwijając lewą stronę powyższej zależności

$$P^* = \mathrm{id}_{\mathcal{U}^*} - \left(J^{\#}J\right)^* = \mathrm{id}_{\mathcal{U}} - J^{\#}J = P$$

stwierdzamy, że rzutowanie (A.1) jest symetryczne.

A.3 Anihilacja $J^{\#}$

Sprawdźmy, czy słuszna jest równość

$$PJ^{\#} = 0, \tag{A.3}$$

Rozwijając

$$PJ^{\#} = \left(\mathrm{id}_{\mathcal{U}} - J^{\#}J\right) J^{\#} = J^{\#} - \underbrace{J^{\#}JJ^{\#}}_{J^{\#}} = 0,$$

udowadniamy równość (A.3).

Dodatek B

Parametryczne definicje funkcji $\widetilde{K}_{x_0,T}(c)$, $\widetilde{J}_{x_0,T}(c)$, $\widetilde{J}^{\#}_{x_0,T}(c)$, $\widetilde{e}$

W tym dodatku zebraliśmy parametryczne reprezentacje elementów składowych algorytmów dla zadania właściwego planowania ruchu i innych zadań dodatkowych wprowadzonych w rozdziale 2.3. Na końcu definiujemy także parametryczną wersję projekcji (2.17).

Stosując parametryczną reprezentację funkcji sterujących (3.2) uzyskujemy skończeniewymiarową endogeniczną przestrzeń konfiguracyjną $\widetilde{\mathcal{U}}$. W konsekwencji, otrzymujemy parametryczne wersje elementów potrzebnych do zdefiniowania algorytmu. Niezależnie od typu zadania, skończeniewymiarowa odwrotność jakobianu jest odwrotnością Moore'a-Penrose'a, którą w obszarach wolnych od osobliwości definiujemy wzorem

$$\widetilde{J}^{\#}_{x_0,T}(c) = J^{\#}_{x_0,T}(u(c,\cdot)) = \widetilde{J}^{\mathsf{T}}_{x_0,T}(c)\left(\widetilde{J}_{x_0,T}(c)\widetilde{J}^{\mathsf{T}}_{x_0,T}(c)\right)^{-1}.$$

B.1 Właściwe planowanie ruchu

Odwzorowanie końcowe

$$\widetilde{K}_{x_0,T}(c) = K_{x_0,T}(u(c,\cdot)) = k(\varphi_{x_0,T}(u(c,\cdot))) = y(T)$$

Jakobian

$$\widetilde{J}_{x_0,T}(c) = J_{x_0,T}(u(c,\cdot)) = \mathsf{C}(T)\int_0^T \Phi(T,s)\mathsf{B}(s)\Psi(s)\,\mathrm{d}s \tag{B.1}$$

Błąd

$$\widetilde{e} = \widetilde{K}_{x_0,T}(c) - y_d$$

B.2 Zadanie minimalizacji energii sterowania

Odwzorowanie końcowe

$$\widetilde{K}_{x_0,T}(c) = K_{x_0,T}(u(c,\cdot)) = \frac{1}{2}\int_0^T (\Psi(t)c)^{\mathsf{T}}\sigma(t)\Psi(t)c\,\mathrm{d}t = \frac{1}{2}c^{\mathsf{T}}\underbrace{\int_0^T \Psi^{\mathsf{T}}(t)\sigma(t)\Psi(t)\,\mathrm{d}t}_{Q(T)}c = \frac{1}{2}c^{\mathsf{T}}Q(T)c,$$

Jakobian

$$\widetilde{J}_{x_0,T}(c) = J_{x_0,T}(u(c,\cdot)) = c^{\mathsf{T}}Q(T),$$

Błąd

$$\widetilde{e} = \widetilde{K}_{x_0,T}(c)$$

B.3 Zadanie minimalizacji wartości zmiennych stanu (wyjścia)

Odwzorowanie końcowe

$$\widetilde{K}_{x_0,T}(c) = K_{x_0,T}(u(c,\cdot)) = \frac{1}{2}\int_0^T x^{\mathsf{T}}(t)\delta(t)x(t)\,\mathrm{d}t = \frac{1}{2}\int_0^T \varphi^{\mathsf{T}}_{x_0,t}(u(c,\cdot))\delta(t)\varphi_{x_0,t}(u(c,\cdot))\,\mathrm{d}t$$

Jakobian

$$\widetilde{J}_{x_0,T}(c) = J_{x_0,T}(u(c,\cdot)) = \int_0^T x^{\mathsf{T}}(t)\delta(t)\int_0^t \Phi(t,s)\mathsf{B}(s)\Psi(s)\,\mathrm{d}s\,\mathrm{d}t = \int_0^T x^{\mathsf{T}}(t)\delta(t)\widetilde{\mathbb{J}}_{x_0,t}(c)\,\mathrm{d}t$$

gdzie $\widetilde{\mathbb{J}}_{x_0,t}(c)$ jest parametryczną wersją jakobianu (2.6) układu robotycznego z deficytem napędów bez funkcji wyjścia (1.12) zdefiniowaną przez (B.1) przy założeniu $\mathsf{C}(t) = I_{2n}$.

Błąd

$$\widetilde{e} = \widetilde{K}_{x_0,T}(c)$$

B.4 Zadanie unikania przeszkód (osobliwości) w przestrzeni zadaniowej i w przestrzeni stanu

Odwzorowanie końcowe

$$\widetilde{K}_{x_0,T}(c) = K_{x_0,T}(u(c,\cdot)) = \frac{1}{2}\int_0^T h(y(t))\,\mathrm{d}t = \frac{1}{2}\int_0^T h(k(x(t)))\,\mathrm{d}t = \frac{1}{2}\int_0^T h(k(\varphi_{x_0,t}(u(c,\cdot))))\,\mathrm{d}t.$$

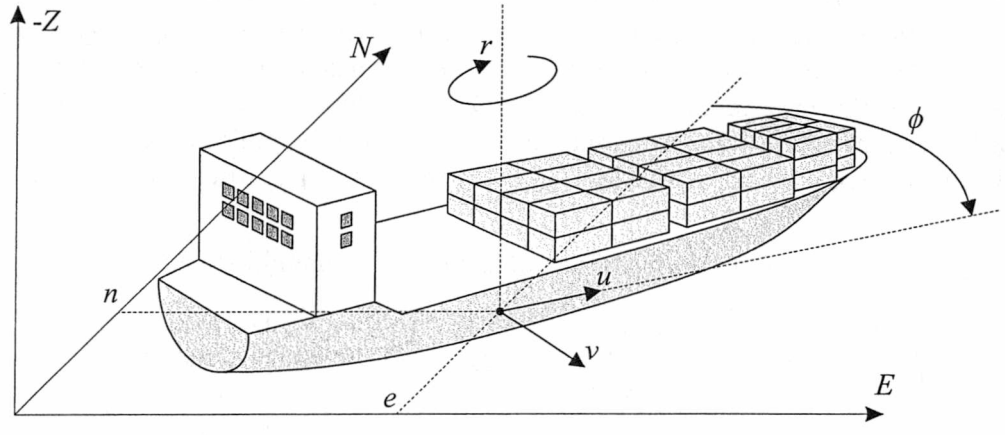
Dodatek C

Modele układów z deficytem napędów

W bieżącym dodatku wprowadzimy modele układów robotycznych z deficytem napędów używanych w pracy. Wszystkie modele są układami nieholonomicznymi. Pojawiające się w nich ograniczenia nieholonomiczne mają swoje źródło w dynamice tych układów, zatem takie ograniczenia są reprezentowane przez równania różniczkowe drugiego rzędu. Wśród przedstawionych układów znajdują się trzy układy mobilne: statek, robot balansujący i poduszkowiec oraz dwa manipulatory stacjonarne z pasywnym przegubem.

C.1 Statek (kontenerowiec)

Jako pierwszy układ robotyczny z deficytem napędów rozważmy model statku kontenerowca [11]. Schematyczna ilustracja modelu jest przedstawiona na rysunku C.1. Oznaczmy



Rysunek C.1 Statek (kontenerowiec)

przez $\eta = (n, e, \phi) \in I\!R^3$ wektor położenia i orientacji, gdzie n i e oznaczają pozycję „Północ-Wschód", a ϕ jest azymutem mierzonym od północy. Niech wektor $\nu = (u, v, r) \in I\!R^3$ będzie wektorem prędkości w ciele, gdzie u jest prędkością wzdłużną, v jest prędkością poprzeczną, a r jest prędkością obrotową wokół osi pionowej. Statek jest wyposażony w ster, który umożliwia zmianę orientacji statku i w pędnik zapewniający zmianę prędkości statku. Zapiszmy kinematykę statku

$$\dot{\eta} = R(\phi)\nu, \qquad R(\phi) = \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

gdzie $R(\phi)$ jest macierzą rotacji wokół osi Z. Dynamikę układu znormalizujemy zgodnie z systemem „Prime" opisanego przez *SNAME* - ang. *The Society of Naval Architects and Marine*. To podejście używa aktualnej prędkości statku $U = \sqrt{u^2 + v^2}$, długości statku L, jednostki czasu L/U, oraz jednostki masy $1/2\rho L^3$ (ρ oznacza gęstość wody) jako zmiennych normalizujących [11]. Dynamikę wyrażoną w układzie ciała możemy zapisać jako

$$M(\nu)\dot{\nu} + N(\nu)\nu = B(\nu)u, \tag{C.1}$$

gdzie $M(\nu)$ jest symetryczną i dodatniookreśloną macierzą inercji, a $N(\nu)\nu$ może zawierać elementy związane z siłami odśrodkowymi, Coriolisa, efektami hydrodynamicznymi, itp. $B(\nu)$ jest macierzą sterowania, a $u \in \mathbb{R}^2$ jest wektorem sterowania, gdzie u_1 jest siłą ciągu pędnika i u_2 jest kątem skrętu steru. Zgodnie z [11], zdefiniujmy macierze $M(\nu)$, $N(\nu)$ i $B(\nu)$ jako

$$M(\nu) = \begin{bmatrix} \frac{L(m'+m'_x)}{U^2} & 0 & 0 \\ 0 & \frac{L(m'+m'_y)}{U^2} & 0 \\ 0 & 0 & \frac{L^2(I'_z+J'_z)}{U^2} \end{bmatrix},$$

$$N(\nu) = \begin{bmatrix} \frac{-X'_{uu}|u/U|}{U} & 0 & \frac{-L(m'+m'_y)v/U}{U} \\ 0 & \frac{-Y'_v}{U} & \frac{-L(Y'_r+(m'+m'_x)u/U)}{U} \\ 0 & \frac{-N'_v}{U} & \frac{-LN'_r}{U} \end{bmatrix},$$

$$B(\nu) = \begin{bmatrix} \frac{2(1-t_d)}{\rho U^2 L^2} & 0 \\ 0 & Y'_\delta \\ 0 & N'_\delta \end{bmatrix}.$$

Numeryczne wartości współczynników pojawiających się w powyższych równaniach zostały zawarte w tabeli C.1. Elementy m', m'_x i m'_y są bezwymiarowe i mają związek z masą statku i z efektem masy związanej. Symbole I'_z, J'_z oznaczają znormalizowane momenty bezwładności statku oraz momenty masy związanej. Wielkości X'_{uu}, Y'_v, Y'_r, Y'_δ, N'_v, N'_r i N'_δ są bezwymiarowymi współczynnikami efektów hydrodynamicznych. t_d jest współczynnikiem siły ciągu pędnika, ρ oznacza gęstość wody, a L jest długością statku.

Tabela. C.1 Wartości współczynników dynamiki statku

$m' = 0.00792$	$m'_x = 0.000238$	$m'_y = 0.007049$
$I'_z = 0.000456$	$J'_z = 0.000419$	$X'_{uu} = -0.0004226$
$Y'_v = -0.0116$	$Y'_r = 0.00242$	$Y'_\delta = -0.002578$
$N'_v = -0.0038545$	$N'_r = -0.00222$	$N'_\delta = 0.00126$
$t_d = 0.175$	$\rho = 1000\,[\mathrm{kg/m^3}]$	$L = 175\,[\mathrm{m}]$

Definiując nowy wektor stanu $x = (x_1, x_2) = (\eta, \nu) \in \mathbb{R}^6$, dynamikę statku (C.1) możemy wyrazić układem równań

$$\begin{cases} \dot{x}_1 = R(x_{13})x_2, \\ \dot{x}_2 = M^{-1}(\nu)(B(x_2)u - N(x_2)x_2), \end{cases} \tag{C.2}$$

gdzie x_{13} oznacza trzecią współrzędną x_1 (kąt orientacji ϕ). Funkcję wyjścia wybierzemy następująco

$$y = k(x) = x_1 = \eta, \tag{C.3}$$

gdzie $y \in \mathbb{R}^3$ jest wektorem współrzędnych zadaniowych. Równania (C.2) i (C.3) definiują afiniczny układ sterowania z dryfem i funkcją wyjścia (1.12)

$$\begin{cases} \dot{x} = f(x) + G(x)u, \\ y = h(x) = x_1, \end{cases} \tag{C.4}$$

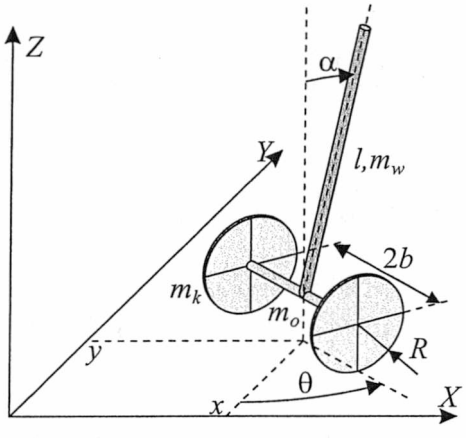
gdzie

$$f(x) = \begin{pmatrix} R(x_{13})x_2 \\ M^{-1}(x_2)N(x_2)x_2 \end{pmatrix}, \quad G(x) = \begin{bmatrix} 0_{3\times 2} \\ M^{-1}(x_2)B(x_2) \end{bmatrix}.$$

Układ robotyczny (C.4) jest obiektem badań symulacyjnych algorytmu planowania ruchu z ograniczaniem funkcji sterującej przedstawionych w rozdziale 4.1.1. Inne wyniki zastosowania algorytmu planowania ruchu do przedstawionego modelu statku można znaleźć w [35, 36].

C.2 Robot balansujący

Drugim robotem z deficytem napędów jest robot balansujący. Konstrukcja robota składa się z dwóch, niezależnie napędzanych, ułożonych współosiowo kół dołączonych do korpusu oraz odwróconego wahadła połączonego z tym korpusem. Schemat robota zawiera rysunek C.2. Zgodnie z oznaczeniami na rysunku C.2 x i y oznaczają pozycję punktu odniesienia



Rysunek C.2 Robot balansujący

umieszczonego w środku osi kół, θ jest orientacją robota, α jest kątem wychylenia wahadła, $2b$ jest rozstawem kół, l oznacza długość wahadła, a R reprezentuje promień koła. Symbole m_o, m_w, m_k odnoszą się odpowiednio do masy osi, wahadła (korpusu) i koła.

Korzystając z prac [20,38] wyprowadźmy kinematykę i dynamikę robota balansującego. Zdefiniujmy wektory $q = (x, y, \theta, \alpha) \in \mathbb{R}^4$ i $\eta = (\eta_1, \eta_2, \eta_3) \in \mathbb{R}^3$, gdzie η_1 i η_2 są prędkościami liniowymi kół, a η_3 odpowiada liniowej prędkości wychylania się wahadła. Kinematykę wyrażającą toczenie się kół opisuje układ

$$\dot{q} = F(q)\eta, \tag{C.5}$$

gdzie

$$F(q) = \begin{bmatrix} \cos q_3 & \cos q_3 & \cos q_3 \\ \sin q_3 & \sin q_3 & \sin q_3 \\ -\frac{1}{b} & \frac{1}{b} & 0 \\ 0 & 0 & \frac{1}{R} \end{bmatrix}.$$

Dynamikę robota balansującego możemy wyrazić przez

$$M(q)\dot{\eta} + N(q,\eta) = B\tau, \tag{C.6}$$

gdzie $M(q) = [M_{ij}]$ jest symetryczną i dodatniookreśloną macierzą inercji, której elementy są zdefiniowane następująco

$$\begin{aligned}
M_{11} &= M_{22} = \frac{4m_a}{3} + m_p\left(1 + \frac{l^2\sin^2\alpha}{b^2}\right) + m_w\left(8 + \frac{R^2}{b^2}\right),\\
M_{12} &= m_p + \frac{2m_a}{3} - \frac{m_p l^2\sin^2\alpha + m_w R^2}{b^2},\\
M_{13} &= M_{23} = m_p + 4m_w + m_a + \frac{lm_p\cos\alpha}{R},\\
M_{33} &= m_p + 4m_w + m_a + \frac{2lm_p\cos\alpha}{R} + \frac{l^2 m_p}{R^2}.
\end{aligned}$$

Wektor $N(q,\eta) \in \mathbb{R}^3$ zawiera składniki związane z siłami odśrodkowymi, Coriolisa oraz grawitacji, a jego elementy definiujemy jako

$$\begin{aligned}
N_1 &= -\frac{lm_p\eta_3^2\sin\alpha}{R^2} - \frac{lm_p(2\eta_2+\eta_3)(\eta_2-\eta_1)\sin\alpha}{b^2} - \frac{2l^2 m_p(\eta_2-\eta_1)\eta_3\sin\alpha\cos\alpha}{b^2 R},\\
N_2 &= -\frac{lm_p\eta_3^2\sin\alpha}{R^2} + \frac{lm_p(2\eta_1+\eta_3)(\eta_2-\eta_1)\sin\alpha}{b^2} + \frac{2l^2 m_p(\eta_2-\eta_1)\eta_3\sin\alpha\cos\alpha}{b^2 R},\\
N_3 &= -\frac{lm_p(\eta_2-\eta_1)^2(R+l\cos\alpha)\sin\alpha}{b^2 R} - \frac{lm_p\eta_3^2\sin\alpha}{R^2} - \frac{lm_p g\sin\alpha}{R}.
\end{aligned}$$

Macierz

$$B = \begin{bmatrix} B_{1\ (2\times2)} \\ B_{2\ (1\times2)} \end{bmatrix} = \frac{2k_m}{R}\begin{bmatrix} I_2 \\ 0_{1\times2} \end{bmatrix}, \tag{C.7}$$

jest macierzą sterowania, której podmacierz B_1 jest odwracalna. Oznaczenie k_m pojawiające się w (C.7) jest stałą silników, a wektor sygnałów sterujących $\tau \in \mathbb{R}^2$ w (C.6) odnosi się do prądów w silnikach.

C.2.1 Częściowo linearyzujące sprzężenie zwrotne robota balansującego

Zgodnie z wywodem w rozdziale 1.2.2, częściowo linearyzujące sprzężenie zwrotne dla robota balansującego przyjmuje postać

$$\tau = B_1^{-1}\left(\left(\begin{bmatrix} M_{11} & M_{12} \\ M_{12} & M_{22} \end{bmatrix} - \begin{pmatrix} M_{13} \\ M_{23} \end{pmatrix} M_{33}^{-1}\begin{pmatrix} M_{13} \\ M_{23} \end{pmatrix}^{\mathsf{T}}\right)u - \begin{pmatrix} M_{13} \\ M_{23} \end{pmatrix} M_{33}^{-1}N_3 + \begin{pmatrix} N_1 \\ N_2 \end{pmatrix}\right).$$

Po jego zastosowaniu do (C.6) dynamikę robota można wyrazić jako

$$\begin{cases} \dot{\eta}_1 = u_1 \\ \dot{\eta}_2 = u_2 \\ \dot{\eta}_3 = -M_{33}^{-1}(M_{13},\ M_{23})u - M_{33}^{-1}N_3. \end{cases} \tag{C.8}$$

Nowy wektor sterowania $u \in \mathbb{R}^2$ ma teraz znaczenie liniowych przyspieszeń obu kół.

Łącząc równania (C.5) i (C.8) otrzymujemy afiniczny układ sterowania z dryfem

$$\dot{x} = f(x) + G(x)u,$$

gdzie $x = (\eta, q) \in \mathbb{R}^7$ jest nowym wektorem stanu, a elementy $f(x)$ i $G(x)$ są następujące

$$f(x) = \left(0,\, 0,\, -M_{33}^{-1}N_3,\, F(q)\eta\right)^{\mathsf{T}},$$

$$G(x) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -M_{33}^{-1}M_{13} & -M_{33}^{-1}M_{23} \\ 0_{4\times 2} & \end{bmatrix}.$$

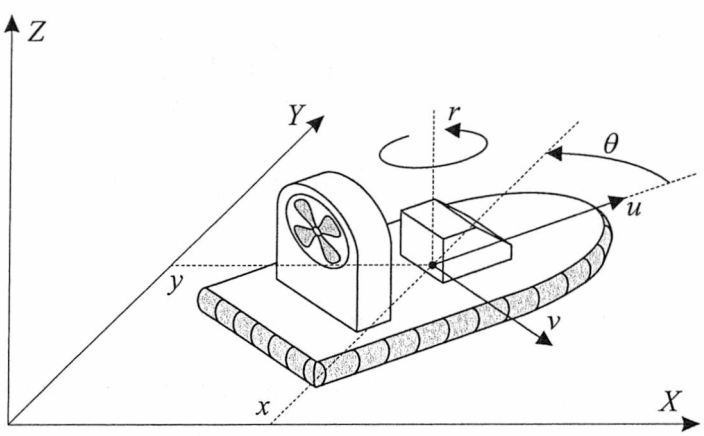
Na potrzeby obliczeń numerycznych przyjęliśmy parametry robota balansującego przedstawione w tabeli C.2. W wynikach symulacyjnych w rozdziale 4.1.2 przeprowadzamy obliczenia algorytmu planowania ruchu dla całego stanu x. Nie ma zatem potrzeby definiowania funkcji wyjścia, która byłaby w tym przypadku identycznościowa $y = k(x) = x$. Algorytm zaproponowany w wyżej wymienionym rozdziale rozwiązuje zadanie planowania ruchu wraz z jednoczesnym utrzymywaniem wahadła w pionie i minimalizacją energii sterowania. Dodatkowe wyniki ilustrujące działanie algorytmu planowania ruchu robota balansującego złożonego z trzech zadań zawiera praca [38].

Tabela. C.2 Parametry robota balansującego

$m_k = 1$	$m_o = 10$	$l = 1.5$	$k_m = 2.6$
$m_w = 5$	$R = 0.12$	$b = 0.3$	$g = 9.81$

C.3 Poduszkowiec

Kolejnym przykładem układu robotycznego z deficytem napędów jest poduszkowiec (patrz rysunek C.3). Załóżmy, że model posiada dwa pędniki. Pierwszy będzie umożliwiał ruch do przodu (i do tyłu), a drugi będzie zapewniał ruch obrotowy wokół osi Z. Charakter



Rysunek C.3 Poduszkowiec

sygnałów sterujących jest podobny jak w dwukołowym robocie mobilnym. Główną cechą odróżniającą ten model od modelu kołowego robota jest fakt, że poduszkowiec może

swobodnie poruszać się poprzecznie do osi wzdłużnej robota, nawet w przypadku, gdy ten stopień swobody nie jest bezpośrednio napędzany. Podobnie jak w modelu statku [11] kinematykę i dynamikę platformy mobilnej z deficytem napędów opisują równania [10,31]

$$\begin{cases} \dot{\eta} = R(\theta)\nu, \\ M(\nu)\dot{\nu} + N(\nu)\nu = \tau, \end{cases}$$

gdzie $R(\theta)$ jest macierzą obrotu wokół osi Z o kąt θ, $M(\eta)$ stanowi macierz inercji, macierz $N(\nu)$ zawiera elementy związane z efektami odśrodkowymi, Coriolisa, tłumieniem hydrodynamicznym, aerodynamicznym, itp. a wektor $\tau = (\tau_u, 0, \tau_r)$ jest wektorem sterowań. Wektor $\eta = (x, y, \theta)$ określa współrzędne położenia i orientacji, a $\nu = (u, v, r)$ zawiera odpowiednio prędkość wzdłużną, poprzeczną i obrotową. Dla uproszczenia, przyjmijmy, że układ jest symetryczny względem każdej osi, a pędniki umieszczone są w środku masy. Ponadto, niech macierz inercji będzie diagonalna i równa macierzy jednostkowej. Ostatecznie, pominiemy także efekt tłumienia hydro-/aerodynamicznego. Powyższe założenia upraszczające nie wpływają na istotne cechy zachowania modelu nieliniowego [10,31]. Po powyższych przekształceniach dynamika poduszkowca przyjmuje postać

$$\begin{cases} \dot{u} = vr + \tau_u, \\ \dot{v} = -ur, \\ \dot{r} = \tau_r, \end{cases}$$

gdzie τ_u jest momentem sterującym prędkością wzdłużną, a τ_r momentem sterującym prędkością obrotową wokół osi Z. Wprowadzając nowy wektor stanu $x = (\eta, \nu) \in \mathbb{R}^6$ możemy zapisać kinematykę i dynamikę poduszkowca w postaci afinicznego układu sterowania

$$\dot{x} = f(x) + G(x)u,$$

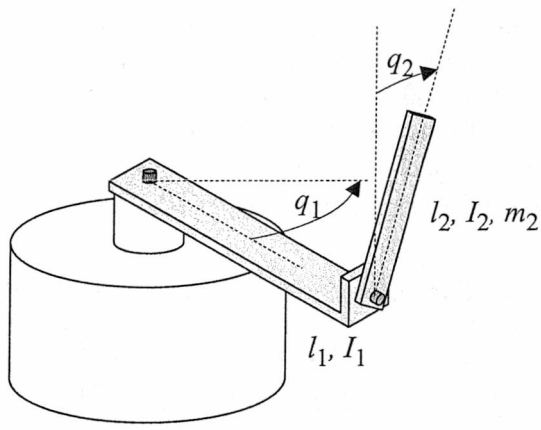
gdzie

$$f(x) = \begin{pmatrix} R(x_{13})x_2 \\ vr \\ -ur \\ 0 \end{pmatrix}, \qquad G(x) = \begin{bmatrix} 0_{3\times 2} \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}, \qquad u = (\tau_u, \tau_r),$$

a x_{13} jest trzecią współrzędną wektora η. Układ poduszkowca jest obiektem testowym dla algorytmu planowania ruchu z unikaniem przeszkód przedstawionym w podrozdziale 4.1.3. Model stanowi także obiekt badań porównawczych algorytmu egalitarnego oraz algorytmu z priorytetowaniem opisanych w rozdziale 4.2. Podobnie, jak w przypadku robota balansującego, nie definiujemy funkcji wyjścia, ponieważ będziemy planowali ruch całego układu, więc $y = k(x) = x$.

C.4 Nieplanarne dwuwahadło $R\overline{R}$

Jednym z modeli stacjonarnych robotów z deficytem napędów jest nieplanarny manipulator $R\overline{R}$. Pracę nad tym modelem zapoczątkował Furuta [12]; na jego cześć model nazywa się także *wahadłem Furuty*. Manipulator (rysunek C.4) składa się z napędzanego przegubu obrotowego, który porusza ramieniem w płaszczyźnie poziomej. Do końca ramienia, przez kolejny przegub obrotowy dołączone jest wahadło, w taki sposób, aby jego ruch odbywał się w płaszczyźnie pionowej. Pierwszy przegub jest przegubem aktywnym, natomiast



Rysunek C.4 Nieplanarne dwuwahadło $R\overline{R}$ (Wahadło Furuty)

drugi, przegubem pasywnym. Dynamikę manipulatora $R\overline{R}$ opisuje formuła

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + D(q) = \begin{pmatrix} u \\ 0 \end{pmatrix}, \tag{C.9}$$

gdzie $q \in \mathbb{R}^2$ są współrzędnymi przegubowymi, macierz $M(q)$ jest symetryczną i dodatnio określoną macierzą inercji, $C(q,\dot{q})$ jest macierzą sił odśrodkowych i Coriolisa, $D(q)$ jest wektorem sił potencjalnych, a u jest momentem sterującym w pierwszym przegubie. Załóżmy, że opory tarcia są pomijalnie małe. Oznaczmy przez l_1 całkowitą długość ramienia, l_2 odległość do środka masy wahadła. Niech m_2 jest masą wahadła, I_1 oznacza bezwładność ramienia, a I_2 oznacza inercję wahadła wokół środka masy. Wówczas elementy pojawiające się w dynamice (C.9) definiujemy następująco [10]

$$\begin{aligned}
M(q) &= \begin{bmatrix} I_1 + m_2(l_1^2 + l_2^2 \sin^2 q_2) & m_2 l_1 l_2 \cos q_2 \\ m_2 l_1 l_2 \cos q_2 & I_2 + m_2 l_2^2 \end{bmatrix}, \\
C(q,\dot{q}) &= \begin{bmatrix} \frac{1}{2} m_2 l_2^2 \sin(2q_2)\dot{q}_2 & -m_2 l_1 l_2 \sin q_2\, \dot{q}_2 + \frac{1}{2} m_2 l_2^2 \sin(2q_2)\dot{q}_1 \\ -\frac{1}{2} m_2 l_2^2 \sin(2q_2)\dot{q}_1 & 0 \end{bmatrix}, \\
D(q) &= \begin{pmatrix} 0 \\ -m_2 g l_2 \sin q_2 \end{pmatrix}.
\end{aligned}$$

Następnie, wprowadzając nowy wektor stanu $x = (q, \dot{q})$, otrzymujemy afiniczny układ sterowania z dryfem

$$\dot{x} = f(x) + G(x)u,$$

gdzie

$$f(x) = \begin{pmatrix} x_3 \\ x_4 \\ -M^{-1}(x)\left(C(x)\begin{pmatrix} x_3 \\ x_4 \end{pmatrix} + D(x)\right) \end{pmatrix}, \qquad G(x) = \begin{pmatrix} 0 \\ 0 \\ M^{-1}(x)\begin{pmatrix} 1 \\ 0 \end{pmatrix} \end{pmatrix}.$$

Kinematykę manipulatora możemy zapisać jako

$$K(x) = \begin{pmatrix} l_1 \cos x_1 - l_2 \sin x_1 \sin x_2 \\ l_1 \sin x_1 + l_2 \cos x_1 \sin x_2 \\ l_2 \cos x_2 \end{pmatrix}.$$

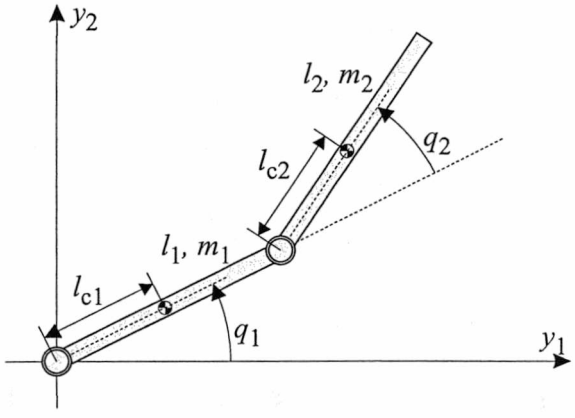
Planowanie ruchu będzie odbywało się w przestrzeni stanu, więc $y = k(x) = x$. Wprowadzony model jest obiektem badań algorytmu planowania ruchu przy niepełnej znajomości modelu przedstawionych w rozdziale 4.3. Do obliczeń numerycznych przyjęliśmy wartości parametrów modelu zebranych w tabeli C.3 [10]. Dodatkowe wyniki działania algorytmu planowania ruchu opartego na metodzie endogenicznej przestrzeni konfiguracyjnej wahadła Furuty przy pełnej znajomości modelu zostały opisane w [32].

Tabela. C.3 Parametry nieplanarnego dwuwahadła $R\overline{R}$

l_1 =0.215	I_1 =1.7510^{-}2	m_1 =5.3810^{-2}
l_2 =0.113	I_2 =1.9810^{-}4	g =9.81

C.5 Planarne dwuwahadło $R\overline{R}$

Ostatnim modelem, z którego korzystamy w rozprawie jest manipulator planarny typu $R\overline{R}$. Jest to układ planarnego dwuwahadła (rysunek C.5), w którym pierwszy przegub obrotowy jest napędzany, a drugi, również obrotowy, jest przegubem pasywnym. W li-



Rysunek C.5 Planarne dwuwahadło $R\overline{R}$

teraturze taki układ jest często nazywany *Pendubotem* [49]. Niech $q = (q_1, q_2) \in \mathbb{R}^2$ będą współrzędnymi przegubowymi, m_1, m_2 masami odpowiednio pierwszego i drugiego ramienia, l_1, l_2 długościami ramion, l_{c1}, l_{c2} odległościami do środka masy ramion, a g przyspieszeniem ziemskim. Wówczas dynamikę manipulatora $R\overline{R}$ możemy wyrazić przez

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + D(q) = \begin{pmatrix} \tau \\ 0 \end{pmatrix}, \tag{C.10}$$

gdzie $M(q)$ jest macierzą inercji, $C(q,\dot{q})$ macierzą Coriolisa, a $D(q)$ wektorem grawitacji i są zdefiniowane następująco

$$M(q) = \begin{bmatrix} k_1 + k_2 + 2k_3 \cos q_2 & k_2 + k_3 \cos q_2 \\ k_2 + k_3 \cos q_2 & k_2 \end{bmatrix},$$

$$C(q,\dot{q}) = \begin{bmatrix} -k_3\dot{q}_2 \sin q_2 & -k_3\dot{q}_2 \sin q_2 - k_3\dot{q}_1 \sin q_2 \\ k_3\dot{q}_1 \sin q_2 & 0 \end{bmatrix},$$

$$D(q) = \begin{pmatrix} k_4 g \cos q_1 + k_5 g \cos(q_1 + q_2) \\ k_5 g \cos(q_1 + q_2) \end{pmatrix}.$$

Parametry $k_1, \ldots, k_5$ są stałymi modelu i są równe

$$\begin{aligned} k_1 &= m_1 l_{c_1}^2 + m_2 l_1^2, & k_2 &= m_2 l_{c_2}^2, \\ k_3 &= m_2 l_1 l_{c2}, & k_4 &= m_1 l_{c1} + m_2 l_1, \\ k_5 &= m_2 l_{c2}. \end{aligned}$$

C.5.1 Częściowo linearyzujące sprzężenie zwrotne dwuwahadła $R\overline{R}$

Stosując do dynamiki (C.10) częściowo linearyzujące sprzężenie zwrotne

$$\begin{aligned} \tau = {} & (M_{11}(q) - M_{12}(q) M_{22}^{-1}(q) M_{12}^{\mathrm{T}}(q)) u \\ & - M_{12}(q) M_{22}^{-1}(q) (\begin{bmatrix} C_{21}(q,\dot{q}) & C_{22}(q,\dot{q}) \end{bmatrix} \dot{q} + D_2(q)) + \begin{bmatrix} C_{11}(q,\dot{q}) & C_{12}(q,\dot{q}) \end{bmatrix} \dot{q} + D_1(q) \end{aligned}$$

i wprowadzając nowe zmienne stanu $x = (q, \dot{q})$ otrzymujemy częściowo zlinearyzowany afiniczny układ sterowania z dryfem

$$\dot{x} = f(x) + G(x)u,$$

gdzie

$$f(x) = \begin{pmatrix} x_3 \\ x_4 \\ 0 \\ -\begin{bmatrix} C_{2,1} & C_{22} \end{bmatrix} M_{22}^{-1} \begin{pmatrix} x_3 \\ x_4 \end{pmatrix} - D_{21} M_{22}^{-1} \end{pmatrix}, \qquad G(x) = \begin{pmatrix} 0 \\ 0 \\ 1 \\ -M_{21} M_{22}^{-1} \end{pmatrix}.$$

Kinematykę manipulatora opisuje zależność

$$K(x) = \begin{pmatrix} l_1 \cos x_1 + l_2 \cos(x_1 + x_2) \\ l_1 \sin x_1 + l_2 \sin(x_1 + x_2) \end{pmatrix}.$$

Funkcją wyjścia jest identycznością $y = k(x) = x$, zatem będziemy planować ruch w przestrzeni stanu układu. Nowe sterowanie u uzyskane w wyniku częściowo linearyzującego sprzężenia zwrotnego ma znaczenie przyspieszenia w napędzanym przegubie. Przedstawiony model posłużył do uzyskania wyników symulacyjnych algorytmu planowania ruchu, w którym korzystaliśmy z nieparametrycznej reprezentacji funkcji sterującej (rozdział 4.4) oraz do rozwiązania problemu odtwarzania trajektorii (rozdział 4.5), zarówno we współrzędnych stanu jak i we współrzędnych zadaniowych. Do uzyskania wyników obliczeń numerycznych niezbędne są wartości parametrów manipulatora, które zawarliśmy w tabeli C.4. Dodatkowe wyniki symulacyjne kilku algorytmów planowania, zastosowanych do powyższego modelu można znaleźć w pracach [32–34, 37].

Tabela. C.4 Wartości parametrów manipulatora planarnego $R\overline{R}$

$l_1 = 0.5$	$l_{c1} = l_1$	$m_1 = 1$	$g = 0$
$l_2 = 0.5$	$l_{c2} = l_2$	$m_2 = 0.5$	

Bibliografia

[1] P. Baerlocher, R. Boulic. Task-priority formulations for the kinematic control of highly redundant articulated structures. *IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS 1998)*, wolumen 1, strony 323–329, Victoria, Kanada, Październik 1998.

[2] A. Ben-Israel, T. N. Greville. *Generalized Inverses: Theory and Applications.* CMS Books in Mathematics. Springer, Nowy Jork, USA, 2003.

[3] R. M. Bianchini, G. Stefani. Controllability along a trajectory: A variational approach. *SIAM Journal on Control and Optimization*, 31(4):900–927, 1993.

[4] L. Birglen, T. Laliberté, C. Gosselin. *Underactuated Robotic Hands*, wolumen 40 serii *Springer Tracts in Advanced Robotics.* Springer-Verlag, Berlin, Niemcy, 2008.

[5] F. Bullo, A. D. Lewis. *Geometric Control of Mechanical Systems: Modeling, Analysis, and Design for Simple Mechanical Control Systems.* Texts in Applied Mathematics. Springer, Nowy Jork, USA, 2005.

[6] S. Chiaverini. Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators. *IEEE Transactions on Robotics and Automation*, 13(3):398–410, Czerwiec 1997.

[7] Y. Chitour. A continuation method for motion-planning problems. *ESAIM: Control, Optimisation and Calculus of Variations*, 12:139–168, 2006.

[8] A. De Luca, S. Iannitti. A simple STLC test for mechanical systems underactuated by one control. *IEEE International Conference on Robotics and Automation (ICRA 2002)*, strony 1735–1740, Waszyngton, USA, Maj 2002.

[9] A. De Luca, S. Iannitti, R. Mattone, G. Oriolo. Control problems in underactuated manipulators. *IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2001)*, strony 855–861, Como, Włochy, Lipiec 2001.

[10] I. Fantoni, R. Lozano. *Non-linear Control for Underactuated Mechanical Systems.* Springer-Verlag, Londyn, Wielka Brytania, 2002.

[11] T. I. Fossen. *Guidance and Control of Ocean Vehicles.* J. Wiley, Nowy Jork, USA, 1994.

[12] K. Furuta, M. Iwase. Swing-up time analysis of pendulum. *Bulletin of the Polish Academy of Sciences: Technical Sciences*, 52(3):153–163, Wrzesień 2004.

[13] H. Goldstein. *Classical Mechanics.* Addison-Wesley, Reading, USA, wydanie 2, 1980.

[14] J. W. Grizzle, C. H. Moog, C. Chevallereau. Nonlinear control of mechanical systems with an unactuated cyclic variable. *IEEE Transactions on Automatic Control*, 50(5):559–576, Maj 2005.

[15] H. Hertz. *Die Prinzipien der Mechanik in Neuem Zusammenhange Dargestellt.* Gesammelte Werke von Heinrich Hertz. Lipsk, Niemcy, 1894.

[16] IFToMM. Terminology for the theory of machines and mechanisms. *Mechanism and Machine Theory*, 26:435–539, 1991.

[17] J. Jakubiak, K. Tchoń. Fourier vs. non-Fourier band-limited jacobian inverse kinematics algorithms for mobile manipulators. *10th IEEE International Conference on Methods and Models in Automation and Robotics. (MMAR 2004)*, strony 1005–1010, Szczecin, Polska, Sierpień 2004.

[18] M. Janiak. *Jakobianowe algorytmy kinematyki odwrotnej manipulatorów mobilnych z ograniczeniami na sterowanie, stan i zachowanie.* Praca doktorska, Politechnika Wrocławska, Wrocław, Polska, 2009.

[19] S. Jung, J. T. Wen. Nonlinear model predictive control for the swing-up of a rotary inverted pendulum. *Journal of Dynamic Systems, Measurement, and Control*, 126(3):666–673, 2004.

[20] J. Kędzierski, K. Tchoń. Feedback control of a balancing robot. *14th IEEE International Conference on Methods and Models in Automation and Robotics. (MMAR 2009)*, Międzyzdroje, Polska, Sier. 2009.

[21] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *IEEE International Conference on Robotics and Automation (ICRA 1985)*, strony 500–505, St. Louis, USA, Marzec 1985.

[22] O. Khatib, J.-F. Le Maitre. Dynamic control of manipulators operating in a complex environment. *3rd International CISM–IFToMM Symposium*, strony 267–282, Seul, Korea, Wrzesień 1978.

[23] U. Kiencke, L. Nielsen. *Automotive Control Systems, For Engine, Driveline, and Vehicle.* Springer Verlag, Berlin, Niemcy, wydanie 2nd, 2005.

[24] A. Liégeois. Automatic supervisory control of the configuration and behavior of multibody mechanisms. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-7(12):868–871, Grudzień 1977.

[25] A. D. Luca, G. Oriolo. Trajectory planning and control for planar robots with passive last joint. *International Journal of Robotic Research*, 21(5-6):575–590, Maj 2002.

[26] A. A. Maciejewski, C. A. Klein. Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. *International Journal of Robotics Research*, 4(3):109–117, Wrzesień 1985.

[27] Y. Nakamura. *Advanced Theoretical Robotics: Redundancy and Optimization.* Prentice Hall, New Jersey, USA, 1991.

[28] C. Olech. On the Ważewski equation. *Conference of Topological Methods in Differential Equations and Dynamical Systems*, strony 55–64, Kraków-Przegorzały, Polska, 1996.

[29] G. Oriolo, Y. Nakamura. Control of mechanical systems with second-order nonholonomic constraints: Underactuated manipulators. *30th IEEE Conference on Decision and Control. (CDC 1991)*, strony 2398–2403, Brighton, Wielka Brytania, Grudzień 1991.

[30] G. Oriolo, Y. Nakamura. Free-joint manipulators: motion control under second-order nonholonomic constraints. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 1991)*, wolumen 3, strony 1248–1253, Osaka , Japonia, Listopad 1991.

[31] K. Pettersen, O. Egeland. Exponential stabilization of an underactuated surface vessel. *35th IEEE Conference on Decision and Control. (CDC 1996)*, strony 967 – 972, vol.1, Kobe, Japonia, Grudzień 1996.

[32] A. Ratajczak. Układy robotyczne z pasywnym stopniem swobody. Praca magisterska, Politechnika Wrocławska, Wrocław, Polska, 2007.

[33] A. Ratajczak, M. Janiak. Planowanie ruchu manipulatorów z pasywnym przegubem przy ograniczonych zmiennych stanu. *Prace Naukowe - Politechnika Warszawska. Elektronika*, wolumen 175, strony 495–504, t.2, XI KKR, Karpacz, Polska, 2010.

[34] A. Ratajczak, M. Janiak. Motion planning of the underactuated manipulators with friction in constrained state space. *Journal of Automation, Mobile Robotics & Intelligent Systems, (JAMRIS)*, 5(3):33–40, 2011.

[35] A. Ratajczak, J. Karpińska, K. Tchoń. Algorytm planowania ruchu statku z priorytetowaniem zadań: metoda endogenicznej przestrzeni konfiguracyjnej. *Prace Naukowe - Politechnika Warszawska. Elektronika*, wolumen 166, strony 535–544, X KKR, Piechowice, Polska, 2008.

[36] A. Ratajczak, J. Karpińska, K. Tchoń. Task-priority motion planning of underactuated systems: an endogenous configuration space approach. *Robotica*, 28(6):885–892, 2010.

[37] A. Ratajczak, K. Tchoń. Control of underactuated robotic manipulators: an endogenous configuration space approach. *13th International Conference on Methods and Models in Automation and Robotics. (MMAR 2007)*, strony 985 – 990, Szczecin, Polska, Sierpień 2007.

[38] A. Ratajczak, K. Tchoń. Motion planning of a balancing robot with threefold subtasks: An endogenous configuration space approach. *IEEE International Conference on Robotics and Automation. (ICRA 2011)*, strony 6096–6101, Szanghaj, Chiny, Maj 2011.

[39] M. Reyhanoglu, A. van der Schaft, N. Mcclamroch, I. Kolmanovsky. Dynamics and control of a class of underactuated mechanical systems. *IEEE Transactions on Automatic Control*, 44(9):1663–1671, Wrzesień 1999.

[40] S. Ricardo. *Geometry of mechanical control systems with applications to systems subject to second-order nonholonomic constraints.* Praca doktorska, Institut National des Sciences Appliquées, Rouen, Francja, 2008.

[41] S. Seereeram, J. Wen. A global approach to path planning for redundant manipulators. *IEEE Transactions on Robotics and Automation*, 11(1):152–160, Luty 1995.

[42] A. Shiriaev, A. Robertsson, P. Pacull, T. I. Fossen. Motion planning and its feedback stabilization for underactuated ships: Virtual constraints approach. *16th Triennial World Congress*, strony 67–72, Praga, Czechy, Lipiec 2005.

[43] B. Siciliano, O. Khatib, redaktorzy. *Springer Handbook of Robotics.* Springer, 2008.

[44] S. K. Soltakhanov, M. P. Yushkov, S. A. Zegzhda. *Mechanics of Non-holonomic Systems: A New Class of Control Systems.* Springer, wydanie 1st, 2009.

[45] E. D. Sontag. *Mathematical control theory: Deterministic finite dimensional systems.* Springer-Verlag, Nowy Jork, USA, wydanie 2, 1998.

[46] M. W. Spong. Partial feedback linearization of underactuated mechanical systems. *The international conference on intelligent robots & systems. (IROS 1994)*, strony 314–321, Monachium, Niemcy, Wrzesień 1994.

[47] M. W. Spong. The swing up control problem for the Acrobot. *IEEE Control Systems Magazine*, 15(1):49–55, 1995.

[48] M. W. Spong. Underactuated mechanical systems. K. P. Valavanis, redaktor, *Control Problems in Robotics and Automation.* Springer-Verlag, 1998.

[49] M. W. Spong, D. J. Block. The Pendubot: A mechatronic system for control research and education. *IEEE Conference on Decision and Control. (CDC 1995)*, strony 555–557, Nowy Orlean, USA, Grudzień 1995.

[50] A. Stephenson. On a new type of dynamical stability. *Manchester Memoirs*, 8:1–10, 1908.

[51] H. J. Sussmann. Lie brackets, real analyticity and geometric control. *Differential Geometric Control Theory.* Birkhäuser, Boston, USA, 1983.

[52] H. J. Sussmann. A general theorem on local controllability. *SIAM Journal on Control and Optimization*, 25(1):158–194, 1987.

[53] H. J. Sussmann. A continuation method for nonholonomic path-finding problems. *32th IEEE Conference on Decision and Control. (CDC 1993)*, strony 2718–2723, t.3, San Antonio, USA, Grudzień 1993.

[54] K. Tchoń. Continuation method in robotics. *7th Conference on Computer Methods and Systems*, strony 17–24, Kraków, Polska, 2007.

[55] K. Tchoń, J. Jakubiak. Endogenous configuration space approach to mobile manipulators: a derivation and performance assessment of jacobian inverse kinematics algorithms. *International Journal of Control*, 76(14):1387–1419, Wrzesień 2003.

[56] K. Tchoń, J. Jakubiak. Extendend Jacobian motion planning algorithm for mobile manipulators. *Robot Control 2003 (SYROCO '03): 7th IFAC Symposium*, strony 147–152, Wrocław, Polska, Wrzesień 2003. Elsevier.

[57] K. Tchoń, J. Jakubiak, K. Zadarnowska. Mobile manipulators with affine control system representation. *Cybernetics of Robotic Systems*, strony 75–116. WKŁ, Wrocław, Polska, 2004.

[58] K. Tchoń, A. Mazur, I. Dulęba, R. Hossa, R. Muszyński. *Manipulatory i roboty mobilne*. Problemy współczesnej nauki teoria i zastosowania: Robotyka. Akademicka Oficyna Wydawnicza PLJ, Warszawa, Polska, 2000.

[59] K. Y. Wichlund, O. J. Sordalen, O. Egeland. Control of vehicles with second-order nonholonomic constraints: Underactuated vehicles. *3rd European Control Conference. (ECC 1995)*, strony 3086–3091, Rzym, Włochy, Wrzesień 1995.

[60] T. Yoshikawa. Manipulability of robotic mechanisms. *International Journal of Robotics Research*, 4(2):3–9, 1985.

[61] K. Zadarnowska, A. Ratajczak. Task–priority motion planning of wheeled mobile robots subject to slipping. *Robot Motion and Control (RoMoCo 2011); Lecture Notes in Control and Information Sciences*. Springer-Verlag, 2011. w druku.