Politechnika Wrocławska

ROZPRAWA DOKTORSKA

Lie-algebraiczne algorytmy suboptymalnego planowania ruchu układów nieholonomicznych w przestrzeni zadaniowej

mgr inż. Arkadiusz Mielczarek

Promotor: prof. dr hab. Ignacy Dulęba

Słowa kluczowe:
planowanie ruchu,
układy nieholonomiczne,
przestrzeń zadaniowa,
algebra Liego,
formuła gCBHD,
środowisko kolizyjne

WROCŁAW 2022

Dziękuję promotorowi, prof. Ignacemu Dulębie, za nieocenioną pomoc otrzymaną w trakcie przygotowywania niniejszej rozprawy.

Streszczenie

Planowanie ruchu jest jednym z podstawowych zadań robotyki. Wraz z rozwojem robotyki mobilnej (roboty: kołowe, kosmiczne, na- i podwodne) wzrasta zapotrzebowanie na algorytmy planowania ruchu dla układów z ograniczeniami nieholonomicznymi. Układy te generują trudne zadania planowania ze względu na mniejszą liczbę sterowań niż wymiarowość przestrzeni konfiguracyjnej. W dysertacji postawiono i rozwiązano zadanie adaptacji metody Lie-algebraicznej do uwzględnienia funkcji wyjścia i planowania w przestrzeni zadaniowej. Praktyczność uwzględnienia tej funkcji, oprócz poszerzenia klasy rozwiązywanych zadań, wynika z naturalnej konieczności sprawdzania kolizyjności w środowiskach z przeszkodami, gdzie nie wszystkie współrzędne wektora konfiguracji są istotne.

Bazowa, Lie-algebraiczna metoda planowania ruchu jest metodą ogólnego przeznaczenia i w literaturze przedmiotu była rozważana jedynie w przestrzeni konfiguracyjnej. Posiada zaletę łatwości kontrolowania objętości manewru i kształtu planowanej trajektorii, dlatego też jest predysponowana do planowań w środowiskach kolizyjnych. Metoda wykorzystuje uogólnioną formułą Campbella-Bakera-Hausdorffa-Dynkina (gCBHD), której dokładne zbadanie było celem dodatkowym pracy. Oprócz określenia wpływu reprezentacji sterowań oraz parametrów początkowych na generowaną trajektorię, zaproponowano algorytm kombinatoryczny wyliczający pre-sterowania formuły gCBHD o złożoności liniowej, znacznie poprawiający algorytmy literaturowe o złożoności eksponencjalnej.

W pracy zredefiniowano obiekty i pojęcia związane z planowaniem ruchu w przestrzeni konfiguracyjnej by dostosować je do przestrzeni zadaniowej. Zdefiniowano pojęcie osobliwości układu z wyjściem, nie występujące dla układów nieholonomicznych w przestrzeni konfiguracyjnej oraz przeanalizowano źródła osobliwości. W ramach transformacji pojęć, przedstawiono także wygląd sfer nieholonomicznych w przestrzeni zadaniowej, które są wizualnym obrazem zróżnicowania trudności ruchu w różnych kierunkach w przestrzeni.

Osiągnięciem rozprawy jest zaprojektowanie autorskiego algorytmu oceny trudności konfiguracji pośredniej w przypadku wieloetapowego planowania ruchu metodą Lie-algebraiczną w przestrzeni zadaniowej. Ocena konfiguracji polegała na uwzględnieniu zarówno parametrów geometrycznych pól wektorowych, jak i ich przynależności do warstw różnicujących energetyczną efektywność ruchu. Na bazie zaprojektowanego algorytmu zaproponowano algorytm planowania ruchu w przestrzeni zadaniowej umożliwiające optymalizację kształtu planowanej trajektorii, jak również jej parametrów takich jak długość czy energia ruchu (sterowań). Zwieńczeniem pracy jest autorski algorytm pozwalający na planowanie ruchu w przestrzeni zadaniowej w środowisku kolizyjnym w którym podcele są generowane planerem geometrycznym.

Wszystkie zaproponowane w dysertacji algorytmy zaimplementowano w wolframowskiej Mathematice i przetestowano na trzech modelach o dwóch wejściach, uwzględniających układy praktyczne i teoretyczne, nilpotentne i nie. Wyniki przestawiono w formie tabelarycznej i graficznej, a na ich bazie wyciągnięto wnioski szczegółowe dotyczące zagadnień warunkujących efektywność algorytmów.

Abstract

Motion planning is one of the fundamental tasks of robotics. With the development of mobile robotics (wheeled, space, on- and underwater) there is an increasing demand for motion planning algorithms for systems with nonholonomic constraints. These systems generate difficult planning tasks due to the smaller number of controls than the dimensionality of a configuration space. The dissertation poses and solves the task of adapting the Lie-algebraic method to take into account an output function and planning in a task space. The practicality of including this function, in addition to broadening the class of solved tasks, results from the natural necessity of checking collisions in environments with obstacles, where not all coordinates of the configuration vector are relevant.

In the literature, the underlying general-purpose, Lie-algebraic motion planning method has only been considered in the configuration space. It allows for controlling a volume of robot manoeuvres easily and also the shape of the planned trajectory and, therefore, it is predisposed to planning in collision environments. The method uses a generalised Campbell-Baker-Hausdorff-Dynkin (gCBHD) formula, the detailed study of which was an additional aim of this dissertation. Besides determining the influence of representation of controls and initial parameters on the generated trajectory, a combinatorial algorithm was proposed to compute pre-controls generated with the gCBHD formula with linear complexity, significantly improving the literature algorithms with exponential complexity.

In this work the objects and concepts related to motion planning in a configuration space have been redefined to adapt them to a task space. The notion of singularity of a system with an output, not occurring for nonholonomic systems in the configuration space, and sources of singularities were analysed. As a part of the transformation of concepts, nonholonomic spheres in the task space were displayed, which are a visual representation of the various difficulty of motion in different directions in a space.

The achievement of the thesis is the design of the author's algorithm for evaluating the difficulty of an intermediate configuration in the case of multistage motion planning using the Lie-algebraic method in a task space. The evaluation of the configuration consists in taking into account both geometrical parameters of vector fields, as well as their layers differentiating the energy efficiency of motion. On the basis of the designed algorithm an algorithm of motion planning in the task space was proposed allowing optimisation of the shape of a planned trajectory, as well as its parameters such as length or energy of motion. The culmination of the work is an original algorithm allowing for planning a motion in a task space in a collision environment in which subgoals are generated by a geometric planner.

All the algorithms proposed in the dissertation have been implemented in *Mathematica* and tested on three models with two inputs, including practical and theoretical, nilpotent and non-nilpotent systems. The results are presented in a tabular and graphical form, and on their basis detailed conclusions are drawn concerning some issues conditioning the efficiency of the algorithms.

Spis treści

Spis oznaczeń

a — skalar,
$\boldsymbol{a}$ — wektor,
$|a|$ — wartość bezwzględna z a,
a_i — i-ta współrzędna $\boldsymbol{a}$, bądź obiekt a w i-tej iteracji algorytmu iteracyjnego,
$\dot{\boldsymbol{a}}$ — pochodna po czasie z $\boldsymbol{a}$,
$\Delta\boldsymbol{a}$ — przyrost wektora $\boldsymbol{a}$,
$\partial\boldsymbol{a}$ — pochodna cząstkowa $\boldsymbol{a}$,
$\boldsymbol{a}_{\parallel}(\cdot)$ — rzut wektora $\boldsymbol{a}$ na podprzestrzeń, leżący w tej podprzestrzeni,
$\boldsymbol{a}_{\perp}(\cdot)$ — rzut wektora $\boldsymbol{a}$ na podprzestrzeń, leżący prostopadle do tej podprzestrzeni,
$\|\boldsymbol{a}\|$ — norma euklidesowa $\boldsymbol{a}$,
$\dim(\boldsymbol{a})$ — wymiarowość wektora $\boldsymbol{a}$,
$\boldsymbol{A}$ — macierz,
$\boldsymbol{I}_n$ — macierz identycznościowa rozmiaru $n \times n$,
$\boldsymbol{0}$ — wektor (macierz) o zerowych elementach,
$\boldsymbol{A}^T, \boldsymbol{q}^T$ — transpozycja macierzy, wektora,
$\boldsymbol{A}^{-1}$ — odwrotność macierzy $\boldsymbol{A}$,
$\boldsymbol{A}^{\#}$ — pseudoodwrotność macierzy $\boldsymbol{A}$ (Moore'a - Penrose'a),
$\det(\boldsymbol{A})$ — wyznacznik macierzy $\boldsymbol{A}$,
$\operatorname{rank}(\boldsymbol{A})$ — rząd macierzy $\boldsymbol{A}$,
$\operatorname{tr}(\boldsymbol{A})$ — ślad macierzy $\boldsymbol{A}$,
$\mathbb{A}$ — przestrzeń,
$\dim(\mathbb{A})$ — wymiar przestrzeni $\mathbb{A}$,
$\langle\cdot,\cdot\rangle$ — iloczyn skalarny,
$\mathbb{R}^n$ — n wymiarowa przestrzeń liczb rzeczywistych (euklidesowa),
$\mathbb{N}$ — zbiór liczb naturalnych,
$\mathbb{Z}$ — zbiór liczb całkowitych,
$\mathcal{C}^k$ — przestrzeń funkcji k-krotnie różniczkowalnych, ($k = 0$ — funkcje ciągłe, $k = \infty$ — funkcje gładkie),
$\mathbb{Q}$ — przestrzeń konfiguracyjna,
$\mathbb{X}$ — przestrzeń zadaniowa,
$\mathbb{U}$ — przestrzeń sterowań,
$\mathbb{P}$ — przestrzeń parametrów sterowań,
$\mathcal{U}$ — endogeniczna przestrzeń konfiguracyjna,

$\mathbb{L}_n^2[0,t]$ — przestrzeń funkcji całkowalnych z kwadratem,
n — wymiarowość przestrzeni konfiguracyjnej,
r — wymiarowość przestrzeni zadaniowej,
m — liczba sterowań (równa liczbie generatorów),
$\boldsymbol{q}$ — wektor konfiguracji w $\mathbb{Q}$,
$\boldsymbol{x}$ — wektor konfiguracji w $\mathbb{X}$,
$\boldsymbol{u}$ — wektor sterowań,
$\boldsymbol{p}$ — wektor parametrów sterowań,
$\boldsymbol{q}_0$ — konfiguracja początkowa w $\mathbb{Q}$,
$\boldsymbol{q}_f$ — konfiguracja docelowa w $\mathbb{Q}$,
$\boldsymbol{x}_0$ — konfiguracja początkowa w $\mathbb{X}$,
$\boldsymbol{x}_f$ — konfiguracja docelowa w $\mathbb{X}$,
$\boldsymbol{X}_i$ — jednomian Liego,
$\boldsymbol{f}(\boldsymbol{q})$ — pole wektorowe,
$\boldsymbol{g}_i(\boldsymbol{q})$ — generator układu nieholonomicznego, $\quad i = 1, \ldots, m$,
$\mathsf{H}(\boldsymbol{x}, \boldsymbol{p}, p_0, \boldsymbol{u})$ — Hamiltonian,
$\boldsymbol{G}(\boldsymbol{q})$ — macierz złożona (kolumnowo) z generatorów $\boldsymbol{g}_i$,
$[\boldsymbol{X}_i, \boldsymbol{X}_j]$ — nawias Liego jednomianów Liego,
$[\boldsymbol{f}_i, \boldsymbol{f}_j]$ — nawias Liego pól wektorowych,
$\operatorname{span}(\boldsymbol{A})$ — przestrzeń liniowa rozpinana przez kolumny macierzy $\boldsymbol{A}$,
$\#\boldsymbol{A}$ — liczba elementów (moc) zbioru $\boldsymbol{A}$,
c_θ, s_θ — cosinus, sinus kąta θ,
c_i, s_i — cosinus, sinus i-tej współrzędnej wektora konfiguracji $\boldsymbol{q}$,
$\boldsymbol{\sigma}$ — permutacja indeksów,
$\operatorname{err}(\boldsymbol{\sigma})$ — liczba błędów w permutacji indeksów,
$\binom{a}{b}$ — symbol Newtona,
$\boldsymbol{H}^i$ — i-ta warstwa bazy Ph. Halla,
$\boldsymbol{H}^i_j$ — j-ty element i-tej warstwy bazy Ph. Halla,
α^i_j — wyrażenie otrzymane z pre-sterowań odpowiadających $\boldsymbol{H}^i_j$,
$pre^{r,d}_p$ — p-te pre-sterowanie związane z d-tym elementem r-tej warstwy bazy algebry Liego,
$w^{r,d}_p$ — współczynnik liczbowy pre-sterowania, indeksy j.w.,
$\operatorname{vol}(\boldsymbol{B})$ — objętość bazy podprzestrzeni $\boldsymbol{B}$,
$LA(\boldsymbol{G}(\boldsymbol{q}))$ — algebra Liego rozpięta na generatorach $\boldsymbol{G}(\boldsymbol{q})$,
$\boldsymbol{D}_i$ — dystrybucja i-tego rzędu,
$\boldsymbol{F}^{\mathbb{Q}}_i / \boldsymbol{F}^{\mathbb{X}}_i$ — macierz złożona z warstw bazy algebry Liego do i-tej włącznie w $\mathbb{Q}/\mathbb{X}$,
$\boldsymbol{M}^{\mathbb{Q}}_i / \boldsymbol{M}^{\mathbb{X}}_i$ — macierz manipulowalności do i-tej warstwy w $\mathbb{Q}/\mathbb{X}$,
$\boldsymbol{J}(\cdot)$ — Jakobian przekształcenia,
$\rho^P_{\boldsymbol{A},\boldsymbol{B}} / \rho^S_{\boldsymbol{A},\boldsymbol{B}}$ — współczynnik korelacji liniowej Pearsona/rankingowej Spearmana,
$\mathbb{E}(\cdot)$ — wartość oczekiwana,
$\mu_{\boldsymbol{A}} / \sigma_{\boldsymbol{A}}$ — średnia/odchylenie standardowe w zbiorze $\boldsymbol{A}$.

Wstęp

Postępująca autonomizacja obiektów poruszających się w przestrzeniach wielowymiarowych wymusza planowanie ich ruchu jako jedno z podstawowych zadań współczesnej robotyki. Szczególnie trudnymi są zadania dla układów nieholonomicznych, rozważanych w niniejszej dysertacji, wśród których można wymienić kołowe roboty mobilne (również z przyczepami) [41, 51, 74, 81], samochody [14, 52, 75], łodzie [40, 98], okręty podwodne [1, 83], samoloty [82], sterowce [106, 107, 108], manipulatory nieholonomiczne [69, 88, 94], manipulatory mobilne [13, 95, 31], roboty kosmiczne [70, 78] i niektóre roboty kroczące [30].

W klasycznym trójpodziale [97], planowanie ruchu jest poprzedzone modelowaniem obiektu oraz jego otoczenia. Dla otrzymanego modelu, przed rozpoczęciem ruchu (w trybie *offline*), przeprowadzane jest planowanie ruchu, którego celem jest uzyskanie trajektorii (ścieżki) referencyjnej i/lub odpowiadające im sterowania. Ostatnim etapem jest sterowanie w czasie rzeczywistym przeprowadzające obiekt wzdłuż zaplanowanej trajektorii (ścieżki) referencyjnej, minimalizując jednocześnie błędy wynikające z niedokładności modelu i/lub niedotrzymania warunków początkowych oraz wykorzystujące informację sensoryczną. Często na etapie planowania uwzględniana jest jedynie kinematyka obiektu, a dynamika dopiero na etapie sterowania. Współcześnie podejmowanych jest coraz więcej udanych prób łączenia planowania i sterowania lub przynajmniej uwzględnienia dynamiki obiektu już podczas planowania ruchu [28, 49].

Układy nieholonomiczne w naturalny sposób są opisywane matematycznie w przestrzeni konfiguracyjnej, na którą nakłada się ograniczenia. Natomiast w przestrzeni zadaniowej opisywane są przeszkody i często nie wszystkie współrzędne konfiguracyjne są istotne podczas sprawdzania kolizyjności robota z przeszkodą. Ponadto w praktycznych zadaniach niekoniecznie wszystkie współrzędne opisujące robota są ważne w kontekście wykonywanego zadania. Typowo przestrzeń konfiguracyjna (gdzie zwykle odbywa się planowanie ruchu) jest związana statyczną funkcją wyjścia z zadaniową (w której łatwiej definiować praktyczne zadanie oraz opisać przeszkody).

Rozwiązywanie zadania planowania ruchu polega na zaprojektowaniu algorytmu pozwalającego na przeprowadzenie układu z dowolnego zadanego punktu początkowego do docelowego. Najczęściej punkty brzegowe planowania są opisywane w przestrzeni konfiguracyjnej. Zdarza się jednak, że punkt docelowy jest zdefiniowany w przestrzeni zadaniowej, a za punkt początkowy przyjmuje się odbicie konfiguracji początkowej do przestrzeni zadaniowej. Na zadanie planowania ruchu nierzadko są nakładane dodatkowe wymagania dotyczące bezkolizyjności obiektu z otoczeniem czy optymalizacji przyjętego kryterium jakości. Planowanie ruchu układów nieholonomicznych jest zadaniem trudnym, nawet w przypadku braku dodatkowych ograniczeń zarówno dla przestrzeni konfiguracyjnej jak i sterowań. Układy takie są opisane nieliniowymi równaniami różniczkowym, a liczba sterowań w nich występująca jest mniejsza od wymiarowości przestrzeni konfiguracji. Z tego powodu nie jest możliwe sterowania każdą współrzędną wektora konfiguracji niezależnie,

a ruch w kierunku celu często jest złożeniem ruchów elementarnych.

W rozprawie bazową metodą planowania ruchu jest metoda Lie-algebraiczna działająca ze swej istoty w przestrzeni konfiguracynej i rozwijana od lat w ośrodku wrocławskim [17, 18, 37, 45, 56, 72, 97]. Jest metodą lokalną, bazującą na uogólnionej formule Campbella-Bakera-Hausdorffa-Dynkina (gCBHD) i planującą ruch wokół bieżącej konfiguracji. Najczęściej konfiguracja docelowa planowania ruchu nie leży blisko początkowej, więc wymaganych jest kilka lokalnych planowań, a wynikowa trajektoria powstaje przez sklejenie kolejno wygenerowanych fragmentów.

Cele i teza pracy

Celem rozprawy doktorskiej jest taka adaptacja metody Lie-algebraicznej by uwzględniała także statyczne odwzorowanie wyjściowe, a jej tezą:

Metoda Lie-algebraiczna może być skutecznie adaptowana do rozwiązywania zadań planowania ruchu w przestrzeni zadaniowej dając dodatkowe, w stosunku do przestrzeni konfiguracyjnej, możliwości optymalizacyjne.

Wykorzystanie metody Lie-algebraicznej do planowania ruchu w przestrzeni zadaniowej jest szczególnie przydatne w środowiskach kolizyjnych, dzięki lokalności, łatwości modyfikowania kształtu wynikowej trajektorii oraz możliwości kontroli obszerności ruchu.

Celem naukowym rozprawy jest opracowanie podstaw teoretycznych dla uwzględnienia funkcji wyjścia w metodzie Lie-algebraicznej stosowanej dla układów nieholonomicznych, a celem praktycznym – zaproponowanie algorytmów planowania ruchu dla tej klasy układów. Realizując cel naukowy w rozprawie przeanalizowano pojęcia związane z metodą Lie-algebraiczną w przestrzeni konfiguracyjnej i zaproponowano ich analogony w przestrzeni zadaniowej. Dla celów praktycznych opracowano algorytmy pozwalające na optymalizację kryteriów jakości uzyskiwanej trajektorii referencyjnej zarówno w przestrzeni konfiguracyjnej jak i zadaniowej. Wśród optymalizowanych funkcji kryterialnych są energia ruchu, długość ścieżki czy kształt trajektorii pozwalający na unikanie przeszkód. Dodatkowymi celami dysertacji, tak poznawczymi jak i praktycznymi, są:

- określenie wpływu reprezentacji sterowań na jakość trajektorii wynikowej,
- zmniejszenie złożoności obliczeniowej metody Lie-algebraicznej,
- zbadanie wpływu punktu startowego dla algorytmu Newtona na wynikowe wartości parametrów sterowań,
- określenie jakości konfiguracji pod kątem realizacji ruchu do punktu w przestrzeni zadaniowej.

Zakres pracy

W rozprawie skupiono się na trzech obszarach związanych z planowaniem ruchu:

- formułą gCBHD będącą podstawowym narzędziem planowania ruchu w przestrzeni konfiguracyjnej,
- przeniesieniem obiektów i pojęć związanych z planowaniem w przestrzeni konfiguracyjnej do przestrzeni zadaniowej,
- planowaniem w przestrzeni zadaniowej.

Najważniejsze osiągnięcia dysertacji zebrano poniżej:

1. Planowanie w przestrzeni konfiguracyjnej i formuła gCBHD:
   - sformułowano i udowodniono twierdzenie o wartościach współczynników pre-sterowań w warstwie algebry Liego,
   - opracowano algorytm szybkiego wyliczania pre-sterowań i ich współczynników,
   - zbadano wpływ parametryzacji sterowań na planowaną trajektorię,
   - opracowano sposób oceny jakości parametryzacji sterowań,
   - zbadano wpływ parametrów sterowań na planowaną trajektorię.
2. Przeniesienie obiektów i pojęć związanych z planowaniem ruchu do przestrzeni zadaniowej:
   - Przeniesiono następujące obiekty i pojęcia:
     - sterowalność,
     - sterowalność w krótkim czasie (X-STLC),
     - stopień nieholonomiczności,
     - macierz manipulowalności,
   - zbadano osobliwości[1] układu nieholonomicznego w przestrzeni zadaniowej i określono ich źródła,
   - zbadano kształt sfer nieholonomicznych w przestrzeni zadaniowej.
3. Planowanie ruchu w przestrzeni zadaniowej:
   - zaprojektowano algorytm oceny trudności konfiguracji początkowej (pośredniej),
   - zaprojektowano algorytm zapobiegający przedwczesnej zbieżności,
   - zbadano możliwości optymalizacji planowanej trajektorii za pomocą wektora startowego algorytmu Newtona wyliczającego parametry sterowań,
   - zaprojektowano algorytm planowania ruchu w przestrzeni zadaniowej w obecności statycznych przeszkód.

Działanie zaproponowanych w dysertacji algorytmów zweryfikowano symulacyjnie dla trzech modeli trój- i czterowymiarowych: jednokołowca, samochodu kinematycznego oraz integratora Brocketta (opisanych w dodatku A). Algorytmy nie są dedykowane dla żadnego z tych modeli, więc mogą być stosowane do dowolnych układów nieholonomicznych.

Struktura pracy

Praca zorganizawana jest w jedenaście rozdziałów poprzedzonych spisem oznaczeń i uzupełnionych dodatkami, zawierającymi materiał pomocniczy, oraz bibliografią. Dla zminimalizowania konieczności odwoływania się do podstawowej literatury pomocniczej, w rozdziale 1 wprowadzono niezbędną terminologię. Zebrano podstawowe wiadomości o algebrze Liego, algebrze Liego pól wektorowych, algorytmie Newtona, parametryzacji funkcji (np. sterowań) za pomocą ortonormalnych baz funkcyjnych oraz korelacji pomiędzy zbiorami. Zamieszczenie opisu algorytmu Newtona i parametryzacji sterowań wynika z częstego rozwiązywania zadania odwrotnego pewnej funkcji z zadanym warunkiem końcowym, w którym znajdowanie sterowań jest sprowadzane do znalezienia ich parametrycznego przedstawienia.

[1] Przyjęto szerszą definicję osobliwości niż stosowaną dla manipulatorów.

W rozdziale 2 przedstawiono analizę literaturową metod planowania ruchu dla układów nieholonomicznych ogólnego przeznaczenia oraz wybranych metod planowania geometrycznego dla układów holonomicznych. Przypomniano pojęcie sterowalności w przestrzeni konfiguracyjnej wraz z jego mocniejszym wariantem: sterowalność w krótkim czasie (STLC) oraz pokazano sposób sprawdzania warunku sterowalności. STLC jest szczególnie przydatne w środowiskach kolizyjnych, gdzie kontrola obszerności ruchu ma istotne znaczenie.

Wśród metod planowania najwięcej uwagi poświęcono intensywnie wykorzystywanej w rozprawie metodzie Lie-algebraicznej planującej ruch w przestrzeni konfiguracyjnej oraz metodzie endogenicznej przestrzeni konfiguracyjnej. W metodzie Lie-algebraicznej opisano dwie formuły pozwalające na określenie sterowań potrzebnych do lokalnego przemieszczenia konfiguracji w dowolnym kierunku: formułę Campbella-Bakera-Hausdorffa-Dynkina (CBHD, sterowania kawałkami stałe) oraz uogólnioną formułę Campbella-Bakera-Hausdorffa-Dynkina (gCBHD, sterowania ciągłe). Metoda endogenicznej przestrzeni konfiguracyjnej również została opisana bardziej szczegółowo, gdyż pozwala na planowanie ruchu w przestrzeni zadaniowej i jest naturalnym punktem odniesienia do metody proponowanej w niniejszej dysertacji.

W kolejnych trzech rozdziałach wszechstronnie przebadano podstawowe narzędzie stosowane w metodzie Lie-algebraicznej - generację sterowań realizujących ruch w zadanym kierunku przy pomocy formuły gCBHD. W rozdziale 3 pokazano autorski algorytm kombinatoryczny poprawiający efektywność wyliczenia pre-sterowań wynikających z formuły gCBHD wraz z porównaniem z algorytmami literaturowych służącymi temu celowi. Pre-sterowanie jest wyrażeniem używanym w formule gCBHD służącym do określenia zależności pomiędzy przemieszczeniem konfiguracji a sterowaniami, bez konieczności ich konkretyzacji. Dopiero na etapie określania sterowań pozwalających na realizację przesunięcia w konkretnym kierunku pre-sterowania są zastępowane sterowaniami, najczęściej w formie parametrycznej, umożliwiającej zastosowanie algorytmu Newtona. Wyniki zaprezentowane w rozdziale zostały przedstawione na podstawie pracy [59].

Rozdział 4 zawiera opis badań symulacyjnych dotyczących wpływu parametrów sterowań na kształt trajektorii wynikającej z zastosowania formuły gCBHD. Zaproponowano szereg modyfikacji sterowań, m. in. liniowe skalowanie, obrót, zmiana proporcji amplitud oraz przesuwanie w fazie. Jakość uzyskanych trajektorii oceniano dwoma funkcjami jakości – błędem położenia końcowego oraz całką z błędu na całej trajektorii. Zawartość rozdziału bazuje na pracy [21].

Dobór parametryzacji sterowań zaprezentowano w rozdziale 5, które oceniano dwiema funkcjami jakości stosowanymi jako człon optymalizujący w przestrzeni zerowej algorytmu Newtona dla jakobianu generowanego formułą gCBHD. Pierwsza funkcja opisywała energię sterowań, druga natomiast bazowała na stopniu pól wektorowych. Dla uniezależnienia od wielu czynników wpływających na wyniki określenia jakości parametryzacji wykorzystano metody statystyczne. Podstawą rozdziału są wyniki uzyskane w pracy [16].

W rozdziale 6, bazującym na artykule [22], przeniesiono pojęcia i definicje używane podczas planowania ruchu układów bezdryfowych w przestrzeni konfiguracyjnej do przestrzeni zadaniowej. W szczególności pojęcie sterowalności w krótkim czasie. Praktycznym zyskiem z wprowadzenia przestrzeni zadaniowej (i przeniesienia pojęć) jest możliwe zredukowanie liczby parametrów przestrzeni sterowań zapewniających sterowalność w krótkim czasie. Przeanalizowano również konfiguracje osobliwe (w ujęciu innym niż znane dla manipulatorów, ale oznaczające brak możliwości infinitezymalnego ruchu w pewnym kierunku) układu nieholonomicznego z funkcją wyjścia. W końcowej części rozdziału porównano lokalną Lie-algebraiczną metodę planowania ruchu z funkcją wyjścia z globalną metodą

endogenicznej przestrzeni konfiguracyjnej.

Rozdział 7 zawiera wyniki symulacji sfer nieholonomicznych układów rozszerzonych o funkcję wyjścia, opisane pierwotnie w referacie [61]. Przeanalizowano wpływ funkcji wyjścia i parametryzacji sterowań na kształt sfer w przestrzeni zadaniowej na przykładach jednokołowca i samochodu kinematycznego. Określono również przydatność sfer nieholonomicznych w przestrzeni zadaniowej.

W rozdziale 8 opisano konsekwencje niejednoznaczności konfiguracji początkowej w przestrzeni konfiguracyjnej $\boldsymbol{q}_0$ odpowiadającej zadanemu punktowi początkowemu w przestrzeni zadaniowej $\boldsymbol{x}_0$. Niejednoznaczność taka istnieje dla układów nieholonomicznych z surjektywną funkcją wyjścia i została zasygnalizowany w rozdziale 6. Przeanalizowano możliwość potraktowania konfiguracji $\boldsymbol{q}_0$ jako parametru wybieranego w ramach podprzestrzeni odpowiadającej $\boldsymbol{x}_0$. W praktycznych zadaniach wielokrokowego planowania ruchu w przestrzeni zadaniowej zwykle nie mamy wpływu na konfigurację $\boldsymbol{q}_0$ podczas kolejnych planowań. Stąd też zbadano wpływ wyboru konfiguracji $\boldsymbol{q}_0$ na jakość (długość, obszerność, energię sterowań) uzyskanej trajektorii. Przedstawiono również autorski algorytm oceny trudności konfiguracji $\boldsymbol{q}_0$ w perspektywie ruchu w kierunku punktu docelowego $\boldsymbol{x}_f$.

W rozdziale 9 opisano problem przedwczesnej zbieżności algorytmów planowania ruchu bazujących na metodzie Lie-algebraicznej, polegający na tendencji do planowania trajektorii złożonych z trudnych do generacji i kosztownych energetycznie lokalnych trajektorii składowych, gdy odległość euklidesowa pomiędzy konfiguracjami początkową $\boldsymbol{q}_0$ a docelową $\boldsymbol{q}_f$ jest duża (większa niż zakres jednego planowania) a różnice pomiędzy "łatwymi" składowymi wektorów $\boldsymbol{q}_0$ i $\boldsymbol{q}_f$ małe. Zaproponowano algorytm wyboru punktów pośrednich pozwalający na uniknięcie przedwczesnej zbieżności. Część wyników zaprezentowanych w rozdziale została przedstawiona w pracy [60].

Rozdział 10, powstały na podstawie pracy [58], przedstawia wyniki badań symulacyjnych wpływu początkowego wektora parametrów sterowań, koniecznego w algorytmie Newtona, na kształt trajektorii planowanej za pomocą formuły gCBHD. Zadanie znalezienia wartości wektora parametrów sterowań ma wiele rozwiązań, jednak z powodu lokalności formuły gCBHD każde z otrzymanych sterowań przeprowadzi rzeczywisty układ wzdłuż innej trajektorii. Przeanalizowano możliwość wykorzystania tego faktu do optymalizacji jakości planowanej trajektorii.

W rozdziale 11 przedstawiono zadanie planowania ruchu układu nieholonomicznego w przestrzeni zadaniowej ze statycznymi przeszkodami. Zaprojektowano algorytm umożliwiający rozwiązanie zadania planowania oraz przeanalizowano wpływ parametrów algorytmu na jakość planowanej trajektorii.

Rozdział 1

Preliminaria matematyczne

W dysertacji będzie wykorzystywany dział algebry zwany algebrą Liego, której początki sięgają prac norweskiego matematyka Mariusa Sophusa Liego z drugiej połowy XIX wieku. Algebra ta z powodzeniem pozwala stosować koncepcje znane w teorii grup do opisywania pojęć i rozwiązywania problemów geometrii różniczkowej. Grupy i algebry Liego znajdują zastosowanie w fizyce teoretycznej i kwantowej, analizie numerycznej, mechanice oraz, przede wszystkim, w teorii sterowania.

W robotyce, między innymi, algebra Liego jest wykorzystywana w lokalnym planowaniu ruchu układów nieholonomicznych oraz stanowić będzie główne narzędzie matematyczne pracy. W rozdziale zostaną przedstawione definicje grupy i algebry Liego wraz z jej bazami oraz omówiona algebra Liego pól wektorowych wraz z dedykowanym nawiasem Liego. W dalszej części rozdziału przywołane zostaną pojęcia matematyczne nie związane bezpośrednio z algebrą Liego, jednak stanowiące uzupełnienie arsenału narzędzi matematycznych dysertacji. Wśród nich jest algorytm Newtona (również w wersji z optymalizacją w przestrzeni zerowej) służący do rozwiązania pewnego zadania odwrotnego, idea parametryzacji funkcji (konkretniej, funkcji sterowań) wraz z przedstawieniem wybranych baz funkcyjnych oraz współczynniki pozwalające na określanie korelacji pomiędzy zestawami danych wynikowych, wykorzystywane podczas ewaluacji rozwiązań. Definicje przywołane w rozdziale odnoszące się do Lie-algebry zaczerpnięte są z prac [7, 29, 47, 80, 84, 87, 91, 109].

1.1 Grupa i algebra Liego

Definicja 1 (grupa Liego) *grupa Liego* $(\mathcal{G},\mu(\cdot,\cdot))$ *jest gładką (klasy* $\mathcal{C}^\infty$*) rozmaitością skończonego wymiaru spełniającą warunki grupy.*

Warunkami grupy dla zbioru $\mathcal{G}$ z operacją dwuargumentową μ są: domknięcie zbioru $\mathcal{G}$ ze względu na μ ($x, y \in \mathcal{G} \Rightarrow \mu(x, y) \in \mathcal{G}$), łączność operatora μ, istnienie elementu neutralnego e ($\mu(x, e) = x$) oraz istnienie elementu odwrotnego x^{-1} dla każdego $x \in \mathcal{G}$. Jeżeli operacja μ jest przemienna, grupa jest abelową.

Przykładem nieabelowej grupy Liego jest grupa obrotów na płaszczyźnie $SO(2)$ oraz izomorficzna do niej grupa liczb zespolonych o module 1 (postaci $e^{i\phi}$).

Definicja 2 (algebra Liego) *algebra Liego* $\mathcal{L}(\boldsymbol{V})$ *nad ciałem* $\boldsymbol{K}$ *to struktura algebraiczna* $(\boldsymbol{V},[\cdot,\cdot])$ *składająca się z przestrzeni liniowej* $\boldsymbol{V}$ *nad ciałem* $\boldsymbol{K}$ *oraz z dwuargumentowego odwzorowania* $[\cdot,\cdot] : \boldsymbol{V}\times\boldsymbol{V} \to \boldsymbol{V}$ *nazywanego nawiasem Liego (komutatorem) spełniającym następujące warunki* $\forall \boldsymbol{X},\boldsymbol{Y},\boldsymbol{Z} \in \boldsymbol{V}, \alpha, \beta \in \mathbb{R}$*:*

- *dwuliniowość:*

$$[\alpha \boldsymbol{X} + \beta \boldsymbol{Y}, \boldsymbol{Z}] = \alpha[\boldsymbol{X}, \boldsymbol{Z}] + \beta[\boldsymbol{Y}, \boldsymbol{Z}], \tag{1.1}$$

$$[\boldsymbol{X}, \alpha \boldsymbol{Y} + \beta \boldsymbol{Z}] = \alpha[\boldsymbol{X}, \boldsymbol{Y}] + \beta[\boldsymbol{X}, \boldsymbol{Z}], \tag{1.2}$$

- *antysymetryczność:*

$$[\boldsymbol{X}, \boldsymbol{Y}] = -[\boldsymbol{Y}, \boldsymbol{X}], \tag{1.3}$$

- *tożsamość Jacobiego:*

$$[\boldsymbol{X}, [\boldsymbol{Y}, \boldsymbol{Z}]] + [\boldsymbol{Y}, [\boldsymbol{Z}, \boldsymbol{X}]] + [\boldsymbol{Z}, [\boldsymbol{X}, \boldsymbol{Y}]] = \boldsymbol{0}. \tag{1.4}$$

Przykładami algebr Liego są:
algebra macierzy kwadratowych $n \times n$ o elementach rzeczywistych z nawiasem Liego

$$[\boldsymbol{A}, \boldsymbol{B}] = \boldsymbol{AB} - \boldsymbol{BA}, \tag{1.5}$$

algebra wektorów w $\mathbb{R}^3$ z nawiasem Liego zadanym iloczynem wektorowym

$$[\boldsymbol{x}, \boldsymbol{y}] = \boldsymbol{x} \times \boldsymbol{y}, \tag{1.6}$$

czy algebra pól wektorowych omówiona w podrozdziale 1.3

1.2 Bazy algebry Liego

Algebra Liego jest rozpinana przez zbiór składający się z generatorów oraz elementów powstałych przez użycie nawiasu Liego na wszystkich kombinacjach elementów powstałych wcześniej. Zarówno generatory jak i elementy powstałe w wyniku rekurencyjnego zastosowania komutatora są nazywane jednomianami Liego. Dla każdego jednomianu Liego można określić rekurencyjnie jego stopień

$$\begin{cases} \deg(\boldsymbol{X}) = 1 & \text{gdy } \boldsymbol{X} \text{ jest generatorem,} \\ \deg([\boldsymbol{X}, \boldsymbol{Y}]) = \deg(\boldsymbol{X}) + \deg(\boldsymbol{Y}) & \text{dla jednomianów złożonych.} \end{cases} \tag{1.7}$$

Warstwą algebry Liego nazywamy zbiór jednomianów Liego o ustalonym stopniu

$$\mathcal{W}_i = \{\boldsymbol{X} \mid \deg(\boldsymbol{X}) = i\}. \tag{1.8}$$

Przykład 1. Elementy trzech pierwszych warstw algebry Liego o generatorach **X**,**Y**,**Z**.

$$\begin{aligned}
\mathcal{W}_1 :\ & \boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \\
\mathcal{W}_2 :\ & [\boldsymbol{X}, \boldsymbol{Y}], [\boldsymbol{X}, \boldsymbol{Z}], [\boldsymbol{Y}, \boldsymbol{Z}], [\boldsymbol{Y}, \boldsymbol{X}], [\boldsymbol{Z}, \boldsymbol{X}], [\boldsymbol{Z}, \boldsymbol{Y}] \\
\mathcal{W}_3 :\ & [\boldsymbol{X}, [\boldsymbol{X}, \boldsymbol{Y}]], [\boldsymbol{X}, [\boldsymbol{X}, \boldsymbol{Z}]], [\boldsymbol{X}, [\boldsymbol{Y}, \boldsymbol{Z}]], [\boldsymbol{X}, [\boldsymbol{Y}, \boldsymbol{X}]], [\boldsymbol{X}, [\boldsymbol{Z}, \boldsymbol{X}]], [\boldsymbol{X}, [\boldsymbol{Z}, \boldsymbol{Y}]], \\
& [\boldsymbol{Y}, [\boldsymbol{X}, \boldsymbol{Y}]], [\boldsymbol{Y}, [\boldsymbol{X}, \boldsymbol{Z}]], [\boldsymbol{Y}, [\boldsymbol{Y}, \boldsymbol{Z}]], [\boldsymbol{Y}, [\boldsymbol{Y}, \boldsymbol{X}]], [\boldsymbol{Y}, [\boldsymbol{Z}, \boldsymbol{X}]], [\boldsymbol{Y}, [\boldsymbol{Z}, \boldsymbol{Y}]], \\
& [\boldsymbol{Z}, [\boldsymbol{X}, \boldsymbol{Y}]], [\boldsymbol{Z}, [\boldsymbol{X}, \boldsymbol{Z}]], [\boldsymbol{Z}, [\boldsymbol{Y}, \boldsymbol{Z}]], [\boldsymbol{Z}, [\boldsymbol{Y}, \boldsymbol{X}]], [\boldsymbol{Z}, [\boldsymbol{Z}, \boldsymbol{X}]], [\boldsymbol{Z}, [\boldsymbol{Z}, \boldsymbol{Y}]], \\
& [[\boldsymbol{X}, \boldsymbol{Y}], \boldsymbol{X}], [[\boldsymbol{X}, \boldsymbol{Z}], \boldsymbol{X}], [[\boldsymbol{Y}, \boldsymbol{Z}], \boldsymbol{X}], [[\boldsymbol{Y}, \boldsymbol{X}], \boldsymbol{X}], [[\boldsymbol{Z}, \boldsymbol{X}], \boldsymbol{X}], [[\boldsymbol{Z}, \boldsymbol{Y}], \boldsymbol{X}], \\
& [[\boldsymbol{X}, \boldsymbol{Y}], \boldsymbol{Y}], [[\boldsymbol{X}, \boldsymbol{Z}], \boldsymbol{Y}], [[\boldsymbol{Y}, \boldsymbol{Z}], \boldsymbol{Y}], [[\boldsymbol{Y}, \boldsymbol{X}], \boldsymbol{Y}], [[\boldsymbol{Z}, \boldsymbol{X}], \boldsymbol{Y}], [[\boldsymbol{Z}, \boldsymbol{Y}], \boldsymbol{Y}], \\
& [[\boldsymbol{X}, \boldsymbol{Y}], \boldsymbol{Z}], [[\boldsymbol{X}, \boldsymbol{Z}], \boldsymbol{Z}], [[\boldsymbol{Y}, \boldsymbol{Z}], \boldsymbol{Z}], [[\boldsymbol{Y}, \boldsymbol{X}], \boldsymbol{Z}], [[\boldsymbol{Z}, \boldsymbol{X}], \boldsymbol{Z}], [[\boldsymbol{Z}, \boldsymbol{Y}], \boldsymbol{Z}].
\end{aligned}$$

Jak widać na przykładzie 1 liczba elementów algebry Liego rośnie bardzo szybko wraz z numerem warstwy. Jednak wiele elementów jest zależnych na podstawie antysymetryczności oraz tożsamości Jacobiego. Dla uniknięcia nadmiarowości reprezentacji algebry Liego można stosować jej bazę składającą się z niezależnych jednomianów Liego. Kolejne sekcje będą poświęcone przedstawieniu takich literaturowych baz.

1.2.1 Baza Ph. Halla

Baza Philipa Halla, $\mathbb{H}$, jest bazą wykorzystywaną w tej rozprawie. Powstaje przez konstruowanie kolejnych elementów z już wygenerowanych i wyborze tylko tych spełniających następujące reguły [97]:

1. $\boldsymbol{X}_i \in \mathbb{H},\ i = 1, \ldots, m$ (generatory należą do bazy i są arbitralnie uporządkowane),
2. jeżeli $\deg(\boldsymbol{B}_i) < \deg(\boldsymbol{B}_j)$, wtedy $(\boldsymbol{B}_i) \overset{\mathbb{H}}{<} (\boldsymbol{B}_j)$,
3. $[\boldsymbol{B}_i, \boldsymbol{B}_j] \in \mathbb{H}$ wtedy i tylko wtedy, gdy
   a) $\boldsymbol{B}_i, \boldsymbol{B}_j \in \mathbb{H}$ i $(\boldsymbol{B}_i) \overset{\mathbb{H}}{<} (\boldsymbol{B}_j)$, oraz
   b) albo $\boldsymbol{B}_j = \boldsymbol{X}_k$ dla $k \in 1, \ldots, m$, albo $\boldsymbol{B}_j = [\boldsymbol{B}_l, \boldsymbol{B}_r]$ dla $\boldsymbol{B}_l, \boldsymbol{B}_r \in \mathbb{H}$ i $\boldsymbol{B}_l \overset{\mathbb{H}}{\leqslant} \boldsymbol{B}_i$.

Warto zauważyć, że reguła druga nie jest warunkiem selekcjonującym, ale wprowadzającym uporządkowanie zależne od stopnia jednomianu Liego.

Dowody, że baza Halla stanowi bazę algebry Liego można znaleźć w [33, 87]. Wydajne algorytmy generujące bazę Halla zostały opisane w [17, 65]. W [65] znajduje się również implementacja takiego algorytmu w Wolfram Language (*Mathematica*). Początkowe elementy bazy Ph. Halla dla trzech generatorów podano w przykładzie 2, dla dwóch natomiast – w tab. 1.1.

Przykład 2. Elementy trzech warstw bazy Ph. Halla o generatorach **X**,**Y**,**Z**.

$$
\begin{aligned}
\boldsymbol{H}^1 :&\ \boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z} \\
\boldsymbol{H}^2 :&\ [\boldsymbol{X}, \boldsymbol{Y}], [\boldsymbol{X}, \boldsymbol{Z}], [\boldsymbol{Y}, \boldsymbol{Z}] \\
\boldsymbol{H}^3 :&\ [\boldsymbol{X}, [\boldsymbol{X}, \boldsymbol{Y}]], [\boldsymbol{X}, [\boldsymbol{X}, \boldsymbol{Z}]], [\boldsymbol{Y}, [\boldsymbol{X}, \boldsymbol{Y}]], [\boldsymbol{Y}, [\boldsymbol{X}, \boldsymbol{Z}]], \\
&\ [\boldsymbol{Y}, [\boldsymbol{Y}, \boldsymbol{Z}]], [\boldsymbol{Z}, [\boldsymbol{X}, \boldsymbol{Y}]], [\boldsymbol{Z}, [\boldsymbol{X}, \boldsymbol{Z}]], [\boldsymbol{Z}, [\boldsymbol{Y}, \boldsymbol{Z}]].
\end{aligned}
$$

1.2.2 Kombinatoryczne bazy algebry Liego

W odróżnieniu od bazy Ph. Halla bazy kombinatoryczne nie są konstruowanie z elementów wcześniej dodanych do bazy, lecz wywodzą się z kombinatoryki. W metodach tych najpierw definiuje się pewien alfabet, a następnie generuje słowa nad tym alfabetem za pomocą odpowiedniego algorytmu. Ostatnią częścią jest przekształcenie uzyskanych słów w jednomiany Liego. Znane są trzy algorytmy kombinatoryczne służące do uzyskania bazy algebry Liego, których autorami są Roger Lyndon [36, 85], Anatolij Shirshov [89] oraz Evgeniy Chibrikov [10]. Dwa pierwsze generują jednak identyczną bazę. Bazy kombinatoryczne zostały wyczerpująco opisane w rozprawie [37].

W tab. 1.1 zebrano kilka początkowych elementów baz Ph. Halla, Lyndona-Shirshova oraz Chibrikova dla układu dwuwejściowego. Bazy algebry Liego są oczywiście równoważne, jednak mogą się różnić efektywnością obliczeń w zależności od jednomianów na jakich są wykonywane operacje. Załóżmy bowiem identyczny koszt energetyczny generacji jednomianów $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ związanych tożsamości Jacobiego. Zatem jedna z baz zawiera jednomiany $\boldsymbol{A}, \boldsymbol{B}$, podczas gdy inna $\boldsymbol{A}, \boldsymbol{C}$. W przypadku gdy konieczne jest uzyskanie jednomianu $\boldsymbol{C}$, w jednej z baz należy wygenerować $\boldsymbol{A}$ oraz $\boldsymbol{B}$ i z nich otrzymać $\boldsymbol{C}$, podczas gdy w innej możliwe jest łatwiejsze, bo bezpośrednie, wygenerowanie $\boldsymbol{C}$.

Tabela 1.1 Elementy baz Halla, Lyndona-Shirshova i Chibrikova z pierwszych pięciu warstw bazy dla dwóch generatorów $\boldsymbol{X}, \boldsymbol{Y}$.

war.	baza		
	Halla	Lyndona-Shirshova	Chibrikova
1	$\boldsymbol{X}$ $\boldsymbol{Y}$	$\boldsymbol{X}$ $\boldsymbol{Y}$	$\boldsymbol{X}$ $\boldsymbol{Y}$
2	$[\boldsymbol{X},\boldsymbol{Y}]$	$[\boldsymbol{Y},\boldsymbol{X}]$	$[\boldsymbol{X},\boldsymbol{Y}]$
3	$[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]$ $[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]$	$[[\boldsymbol{Y},\boldsymbol{X}],\boldsymbol{X}]$ $[\boldsymbol{Y},[\boldsymbol{Y},\boldsymbol{X}]]$	$[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]$ $[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]$
4	$[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]$ $[\boldsymbol{Y},[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]$ $[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]$	$[[[\boldsymbol{Y},\boldsymbol{X}],\boldsymbol{X}],\boldsymbol{X}]$ $[[\boldsymbol{Y},[\boldsymbol{Y},\boldsymbol{X}]],\boldsymbol{X}]$ $[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{Y},\boldsymbol{X}]]]$	$[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]$ $[\boldsymbol{X},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]$ $[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]$
5	$[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]]$ $[\boldsymbol{Y},[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]]$ $[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]]$ $[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]]$ $[[\boldsymbol{X},\boldsymbol{Y}],[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]$ $[[\boldsymbol{X},\boldsymbol{Y}],[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]$	$[[[[\boldsymbol{Y},\boldsymbol{X}],\boldsymbol{X}],\boldsymbol{X}],\boldsymbol{X}]$ $[[[\boldsymbol{Y},[\boldsymbol{Y},\boldsymbol{X}]],\boldsymbol{X}],\boldsymbol{X}]$ $[[\boldsymbol{Y},\boldsymbol{X}],[[\boldsymbol{Y},\boldsymbol{X}],\boldsymbol{X}]]$ $[[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{Y},\boldsymbol{X}]]],\boldsymbol{X}]$ $[[\boldsymbol{Y},[\boldsymbol{Y},\boldsymbol{X}]],[\boldsymbol{Y},\boldsymbol{X}]]$ $[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]]$	$[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]]$ $[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]]$ $[\boldsymbol{X},[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]]$ $[\boldsymbol{Y},[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]]$ $[\boldsymbol{Y},[\boldsymbol{X},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]]$ $[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]]$

1.2.3 Formuła Witta

Z tab. 1.1 można wnioskować, że liczba elementów kolejnych warstw baz algebry Liego rośnie bardzo szybko. Analityczny wzór na liczność warstwy bazy dla r-generatorów nazywany jest, od nazwiska autora, formułą Witta [7, 64]

$$v(n,r) = \frac{1}{n}\sum_{d|n}\mu(d)s^{n/d}, \tag{1.9}$$

gdzie n oznacza numer warstwy, a r liczbę generatorów. Sumowanie następuje po całkowitych dzielnikach d numeru warstwy (np. dla $n=6, \quad d \in \{1,2,3,6\}$. Funkcja $\mu : \mathbb{N} \to \{-1,0,1\}$ dla danego rozkładu na czynniki pierwsze $d = p_1^{n_1}p_2^{n_2}\ldots p_q^{n_q}$, z $n_i > 0$ ma postać

$$\mu(d) = \begin{cases} 1 & \text{dla } d = 1 \\ (-1)^q & \text{gdy wszytkie } n_i = 1 \\ 0 & \text{w pozostałych przypadkach.} \end{cases} \tag{1.10}$$

W tab. 1.2 podano liczbę elementów bazy algebry Liego, do danej warstwy włącznie, wyliczoną z formuły Witta.

Tabela 1.2 Liczba elementów bazy algebry Liego do danej warstwy włącznie dla dwóch generatorów.

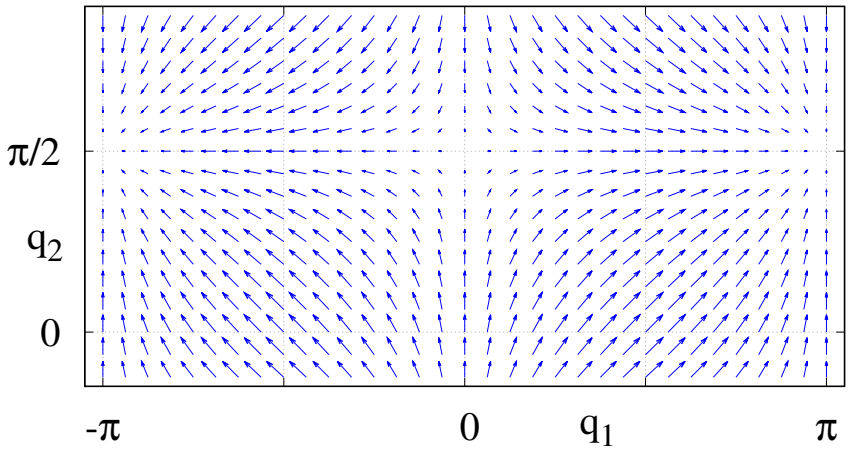
warstwa	1	2	3	4	5	6	7	8	9	10
# elem.	2	3	5	8	14	23	41	71	127	226
warstwa	11	12	13	14	15	16	17	18	19	20
# elem.	412	747	1377	2538	4720	8800	16510	31042	58636	111013

1.3 Algebra Liego pól wektorowych

Wektor, którego elementy są funkcjami nazywamy polem wektorowym $\boldsymbol{f}(\boldsymbol{q})$. Pole wektorowe przyporządkowuje każdemu punktowi dziedziny $\boldsymbol{q}$ (konfiguracji) odpowiedni wektor wskazujący infinitezymalną ewolucję układu zgodnie z owym polem. Przykładową graficzną interpretację pola wektorowego $\boldsymbol{f}(\boldsymbol{q}) = (\sin(q_1), \cos(q_2))^T$ przedstawiono na rys. 1.1. Przestrzeń liniowa gładkich pól wektorowych wraz z nawiasem Liego zdefiniowanym jako

$$[\boldsymbol{f}(\boldsymbol{q}), \boldsymbol{g}(\boldsymbol{q})] = \frac{\partial \boldsymbol{g}(\boldsymbol{q})}{\partial \boldsymbol{q}} \boldsymbol{f}(\boldsymbol{q}) - \frac{\partial \boldsymbol{f}(\boldsymbol{q})}{\partial \boldsymbol{q}} \boldsymbol{g}(\boldsymbol{q}), \tag{1.11}$$

tworzy algebrę Liego pól wektorowych. Zakładamy, że pola wektorowe są gładkie (o elementach klasy $\mathcal{C}^\infty$), aby wynik nawiasu Liego również dawał gładkie pole wektorowe.



Rysunek 1.1 Graficzna reprezentacja pola wektorowego $\boldsymbol{f}(\boldsymbol{q}) = (\sin(q_1), \cos(q_2))^T$ na płaszczyźnie q_1, q_2.

1.4 Algorytm Newtona

Klasyczny algorytm Newtona [68], dla zadanej funkcji $\boldsymbol{f}(\boldsymbol{x})$ i punktu docelowego $\boldsymbol{y}_d$ pozwala na wyznaczenie jednego z rozwiązań $\boldsymbol{x}^\star$ układu równań

$$\boldsymbol{y}_d = \boldsymbol{f}(\boldsymbol{x}^\star), \tag{1.12}$$

gdzie $\boldsymbol{y}_d \in \mathbb{R}^m$, $\boldsymbol{x}^\star \in \mathbb{R}^n$ i wymiar dziedziny przekształcenia $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m$ jest większy lub równy wymiarowi przeciwdziedziny ($n \geqslant m$). Będąc algorytmem lokalnym, wymaga podania punktu startowego $\boldsymbol{x}_0$, który jest przekształcany w kolejnych iteracjach. Dyskretna postać jednego kroku algorytmu jest następująca

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \zeta_i \cdot \boldsymbol{J}^{\#}(\boldsymbol{x}_i)(\boldsymbol{y}_d - \boldsymbol{f}(\boldsymbol{x}_i)), \tag{1.13}$$

gdzie i oznacza numer bieżącej iteracji, ζ_i jest dodatnim parametrem rzeczywistym służącym do skalowania zakresu ruchu pojedynczego kroku (iteracji), a $\boldsymbol{J}^{\#}(\boldsymbol{x}_i)$ – pseudoodwrotnością Moore'a-Penrose'a. Z definicji, pseudoodwrotność generuje rozwiązanie o najmniejszej długości euklidesowej i jest zadana wzorem

$$\boldsymbol{J}^{\#} = \boldsymbol{J}^T(\boldsymbol{J} \cdot \boldsymbol{J}^T)^{-1}, \tag{1.14}$$

gdzie $\boldsymbol{J}(\boldsymbol{x}) = \partial \boldsymbol{f}(\boldsymbol{x})/\partial \boldsymbol{x}$ jest jakobianem przekształcenia $\boldsymbol{f}(\boldsymbol{x})$.

Algorytm Newtona iteracyjnie wylicza kolejne wartości $\boldsymbol{x}_{i+1}$ do momentu spełnienia kryterium stopu $\|\boldsymbol{y}_d - \boldsymbol{f}(\boldsymbol{x}_{i+1})\| < \varepsilon$ dla dostatecznie małego ε.

1.4.1 Algorytm Newtona z optymalizacją w przestrzeni zerowej

Iteracja algorytmu w wersji pozwalającej na optymalizację w przestrzeni zerowej jakobianu jest zadana jako

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \zeta_i^1 \boldsymbol{J}^{\#}(\boldsymbol{x}_i)(\boldsymbol{y}_d - \boldsymbol{f}(\boldsymbol{x}_i)) + \zeta_i^2(\boldsymbol{J}^{\#}(\boldsymbol{x}_i)\boldsymbol{J}(\boldsymbol{x}_i) - \boldsymbol{I})\frac{\partial w(\boldsymbol{x})}{\partial \boldsymbol{x}}|_{\boldsymbol{x}=\boldsymbol{x}_i}, \tag{1.15}$$

gdzie $\boldsymbol{I}$ jest macierzą jednostkową, a $w(\boldsymbol{x})$ minimalizowaną funkcją kryterialną w przestrzeni zerowej. Parametry rzeczywiste ζ_i^1 i ζ_i^2 służą do skalowania zakresu ruchu w bieżącym kroku, a ich stosunek waży wpływ składowych równania (1.15).

1.5 Parametryzacja sterowań

Parametryzacja sterowań jest standardowym zabiegiem mającym na celu zastąpienie trudnego zadania znalezienia sterowania w (nieskończeniewymiarowej) przestrzeni funkcyjnej łatwiejszym zadaniem znalezienia wartości współczynników liczbowych dla pewnej skończonej bazy funkcji [17, 39]. Sterowania w parametrycznej formie są opisane równaniem

$$u_i(t) = \sum_{j=1}^{K_i} p_{i,j}\phi_j(t), \quad i = 1, \ldots, m, \tag{1.16}$$

gdzie u_i oznacza i-te sterowanie, $\phi_j(t)$ jest j-tym elementem ustalonej bazy funkcyjnej, $p_{i,j}$ są współczynnikami liczbowymi, a K_i to ich liczebność. W celu ułatwienia obliczeń często wymaga się, by baza była także ortonormalna. Najczęściej używane są bazy: harmoniczna (Fouriera) oraz wielomianowe. Dobrą praktyką jest dobieranie liczby elementów bazy K_i, tak by wymiarowość wektora parametrów $\dim(\boldsymbol{p})$ była niewiele większa od wymiarowości przestrzeni konfiguracyjnej $\dim(\boldsymbol{p}) > \dim(\mathbb{Q})$.

1.5.1 Baza harmoniczna (baza Fouriera)

Baza harmoniczna jest bazą przestrzeni $\boldsymbol{L}_2([0,T])$ wykorzystywaną między innymi przy aproksymacji funkcji za pomocą trygonometrycznego szeregu Fouriera [100]. Elementy bazy harmonicznej są ortogonalne, a przy odpowiednim przeskalowaniu również ortonormalne. Element j-ty ortonormalnej bazy harmonicznej

$$\phi_j(t) = \frac{\beta_j}{\sqrt{T}}\psi_j(t), \qquad \omega = \frac{2\pi}{T}, \tag{1.17}$$

gdzie funkcje trygonometryczne ψ_j oraz odpowiadające im współczynniki liczbowe β_j są następujące

$$\psi_j(t) = \begin{cases} 1 & \text{dla } j = 1 \\ \sin(k\omega t) & \text{dla } j = 2k \\ \cos(k\omega t) & \text{dla } j = 2k+1 \end{cases}, \quad k \in \mathbb{N}_+, \quad \beta_j = \begin{cases} 1 & \text{dla } j = 1 \\ \sqrt{2} & \text{dla } j > 1 \end{cases}. \tag{1.18}$$

Po podstawieniu do (1.16) zależności (1.17), (1.18) sterowanie u_i w ortonormalnej bazie harmonicznej jest opisane jako

$$u_i(t) = \frac{1}{\sqrt{T}}\Big(p_{i,1} + \sum_{j=1}^{N_i}\Big(p_{i,2j}\sqrt{2}\sin(j\omega t) + p_{i,2j+1}\sqrt{2}\cos(j\omega t)\Big)\Big), \quad K_i = 1 + 2N_i. \tag{1.19}$$

1.5.2 Bazy wielomianowe

Istnieje wiele baz wielomianowych przestrzeni funkcyjnej $\boldsymbol{L}_2([a,b])$. Najbardziej naturalną jest baza potęgowa $\mathcal{B} = \{1, t, t^2, \ldots\}$, jednak nie jest znormalizowana ani ortogonalna. Innymi bazami wielomianowymi są bazy wielomianów Bernsteina [55], Chebysheva (I-go i II-go stopnia) [93] czy Legendre'a [93]. W tab. 1.3 zebrano definicje i właściwości wyżej baz wielomianów, a w tab. 1.4 pokazano kilka ich pierwszych elementów.

Tabela 1.3 Definicje i właściwości baz wielomianów Bernsteina, Chebysheva i Legendre'a.

	wzór	dziedzina	ortogonalność
Bernstein	$B_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}$	$t \in [0,1]$	nie
Chebyshev I	$T_n(t) = \sum_{m=0}^{\lfloor n/2 \rfloor} \binom{n}{2m} t^{n-2m}(t^2-1)^m$	$t \in [-1,1]$	tak
Chebyshev II	$U_n(t) = \sum_{m=0}^{\lfloor n/2 \rfloor} \binom{n+1}{2m+1} t^{n-2m}(t^2-1)^m$	$t \in [-1,1]$	tak
Legendre	$P_n(t) = \sum_{m=0}^{n} \binom{n}{m}\binom{-n-1}{m}\left(\frac{1-t}{2}\right)^m$	$t \in [-1,1]$	tak

Tabela 1.4 Początkowe elementy baz wielomianowych Bernsteina, Chebysheva i Legendre'a.

Bernstein	Chebyshev I	Chebyshev II	Lengendre
$B_{0,0}(t) = 1$	$T_0(t) = 1$	$U_0(t) = 1$	$P_0(t) = 1$
$B_{0,1}(t) = 1 - t$	$T_1(t) = t$	$U_1(t) = 2t$	$P_1(t) = t$
$B_{1,1}(t) = t$	$T_2(t) = 2t^2 - 1$	$U_2(t) = 4t^2 - 1$	$P_2(t) = \frac{1}{2}(3t^2 - 1)$
$B_{0,2}(t) = (1-t)^2$	$T_3(t) = 4t^3 - 3t$	$U_3(t) = 8t^3 - 4t$	$P_3(t) = \frac{1}{2}(5t^3 - 3t)$
$B_{1,2}(t) = 2(1-t)t$	$T_4(t) = 8t^4+$	$U_4(t) = 16t^4$	$P_4(t) = \frac{1}{8}(35t^4+$
$B_{2,2}(t) = t^2$	$-8t^2 + 1$	$-12t^2 + 1$	$-30t^2 + 3)$
$B_{0,3}(t) = (1-t)^3$	$T_5(t) = 16t^5+$	$U_5(t) = 32t^5+$	$P_5(t) = \frac{1}{8}(63t^5+$
$B_{1,3}(t) = 3(1-t)^2t$	$-20t^3 + 5t$	$-32t^3 + 6t$	$-70t^3 + 15t)$
$B_{2,3}(t) = 3(1-t)t^2$	$T_6(t) = 32t^6+$	$U_6(t) = 64t^6+$	$P_6(t) = \frac{1}{16}(231t^6+$
$B_{3,3}(t) = t^3$	$-48t^4 + 18t^2 - 1$	$-80t^4 + 24t^2 - 1$	$-315t^4 + 105t^2 - 5)$

1.6 Korelacja między zestawami danych wynikowych

Aby ustalić liczbowo wpływ danych wejściowych na dane wynikowe chętnie stosuje się korelację, która określa jak silny jest związek pomiędzy dwoma zbiorami (lub zmiennymi losowymi). Do określania korelacji najczęściej stosuje się współczynnik korelacji liniowej Pearsona [15] oraz współczynnik korelacji rankingowej Spearmana [44]. Oba współczynniki przyjmują wartości z przedziału $[-1, 1]$, gdzie wartości bliskie ± 1 oznaczają silną korelację (dodatnią lub ujemną), a wartości bliskie zeru – jej brak.

Współczynnik korelacji liniowej Pearsona określa jak bardzo związek między zbiorami danych $\boldsymbol{A}$ i $\boldsymbol{B}$ jest bliski liniowemu i jest zdefiniowany jako [15]

$$\rho^P_{\boldsymbol{A},\boldsymbol{B}} = \frac{\mathbb{E}[(\boldsymbol{A} - \mu_{\boldsymbol{A}})(\boldsymbol{B} - \mu_{\boldsymbol{B}})]}{\sigma_{\boldsymbol{A}}\sigma_{\boldsymbol{B}}}, \tag{1.20}$$

gdzie $\mathbb{E}[\cdot]$ oznacza wartość oczekiwaną, $\mu_{\boldsymbol{A}}$ – średnią zbioru $\boldsymbol{A}$, a $\sigma_{\boldsymbol{A}}$ – odchylenie standardowe zbioru $\boldsymbol{A}$. Wzór (1.20) może być też określony w wersji dyskretnej dla dwóch zbiorów danych lub prób zmiennych losowych o liczebności n [15]

$$\rho^P_{\boldsymbol{A},\boldsymbol{B}} = \frac{\sum_{i=1}^n (a_i - \overline{a})(b_i - \overline{b})}{\sqrt{\sum_{i=1}^n (a_i - \overline{a})^2}\sqrt{\sum_{i=1}^n (b_i - \overline{b})^2}}, \tag{1.21}$$

gdzie $\overline{a}$ oznacza średnią zbioru $\boldsymbol{A}$.

Współczynnik korelacji rankingowej Spearmana używany jest do określenia jak dalece związek pomiędzy zbiorami danych $\boldsymbol{A}$ i $\boldsymbol{B}$ jest monotoniczny i dany jest wzorem [15, 44]

$$\rho^S_{\boldsymbol{A},\boldsymbol{B}} = \frac{\mathbb{E}[(\mathrm{rg}(\boldsymbol{A}) - \mu_{\mathrm{rg}(\boldsymbol{A})})(\mathrm{rg}(\boldsymbol{B}) - \mu_{\mathrm{rg}(\boldsymbol{B})})]}{\sigma_{\mathrm{rg}(\boldsymbol{A})}\sigma_{\mathrm{rg}(\boldsymbol{B})}}, \tag{1.22}$$

gdzie $\mathrm{rg}(\boldsymbol{A})$ oznacza funkcję określającą rangi na zbiorze $\boldsymbol{A}$.

Rozdział 2

Przegląd metod planowania ruchu układów bezdryfowych

W rozdziale przedstawiono obiekt dla którego rozwiązywane będą zadania planowania ruchu czyli nieholonomiczny, bezdryfowy układ sterowania oraz warunek sterowalności w krótkim czasie umożliwiający efektywne planowanie. W dalszej części rozdziału opisano pokrótce kilka literaturowych metod ogólnego przeznaczenia dla planowania ruchu w przestrzeni konfiguracyjnej. Ponadto przedstawiono popularne metody geometrycznego planowania ruchu dla układów holonomicznych wykorzystywane jako wspierające planowanie dla układów nieholonomicznych. Z przedstawionych metod bardziej szczegółowo opisano Lie-algebraiczną metodę planowania ruchu (której rozszerzenie o przestrzeń zadaniową jest jednym z głównych osiągnięć rozprawy) oraz metodę endogenicznej przestrzeni konfiguracyjnej (która jako jedna z nielicznych metod umożliwia planowanie w przestrzeni zadaniowej i z którą porównywano opracowane rozszerzenie metody Lie-algebraicznej o przestrzeń zadaniową). Pominięto metody specjalizowane działające efektywnie dla wąskiej klasy modeli (czasem jednoelementowej).

2.1 Nieholonomiczny, bezdryfowy układ sterowania

Do nieholonomicznych, bezdryfowych układów sterowania zaliczają się między innymi modele matematyczne kołowych robotów mobilnych, okrętów podwodnych, obiektów latających czy robotów kosmicznych. Układy te na poziomie kinematycznym są opisywane na przez równanie, powstałe z ograniczeń Pfaffa, łączące ze sobą charakterystyki położeniowe oraz prędkościowe [17, 97]. Nieholonomiczny, bezdryfowy układ sterowania jest opisany równaniem

$$\dot{\boldsymbol{q}}(t) = \boldsymbol{G}(\boldsymbol{q})\boldsymbol{u} = \sum_{i=1}^{m} \boldsymbol{g}_i(\boldsymbol{q})u_i, \tag{2.1}$$

$$\boldsymbol{q} \in \mathbb{Q}, \quad \boldsymbol{u} \in \mathbb{U}, \quad n = \dim\mathbb{Q} < m = \dim\mathbb{U},$$

gdzie:

$\boldsymbol{q}$ – wektor konfiguracji w przestrzeni konfiguracyjnej,

n – wymiar przestrzeni konfiguracyjnej,

m – liczba sterowań (wymiar przestrzeni sterowań),

u_i – i-te sterowanie,

$\boldsymbol{g}_i$ – i-ty generator układu, gładkie pole wektorowe stowarzyszone z układem (2.1).

2.1.1 Sterowalność bezdryfowego układu sterowania

Sterowalność układu sterowania oznacza istnienie sterowań dopuszczalnych, przeprowadzających układ z dowolnego stanu początkowego do dowolnego stanu końcowego w skończonym czasie.

Mocniejszą własnością jest lokalna sterowalność w krótkim czasie (STLC – *small-time local controllability*). Sterowalność w krótkim czasie można interpretować geometrycznie jako zdolność układu do ruchu w dowolnym kierunku z dowolnej konfiguracji. Rozumiemy przez to, że istnieje pewna trajektoria łącząca ze sobą dwa bliskie punkty zawarta w małym otoczeniu tych punktów. Warunek na sterowalność układu (2.1), znany jako twierdzenia Chow [11], stanowi, że jeśli w każdej konfiguracji $\boldsymbol{q}$ algebra Liego powstała z generatorów układu (2.1) rozpina całą przestrzeń konfiguracyjną, to układ jest STLC. Istnienie sterowalności w krótkim czasie implikuje również sterowalność układu. Zastosowanie twierdzenie Chow sprowadza się do sprawdzenia rzędu algebry Liego powstałej z generatorów

$$\forall \boldsymbol{q} \in \mathbb{Q} \quad \operatorname{rank}(LA(\boldsymbol{G}(\boldsymbol{q}))) = n. \tag{2.2}$$

Powyższy warunek jest często nazywany warunkiem rzędu (LARC – *Lie algebra rank condition*). O nieholonomiczności układu decyduje konstrukcja małej flagi dystrybucji $\boldsymbol{D}_i$, której pierwszym elementem jest dystrybucja $\boldsymbol{D}_0$ rozpięta przez generatory układu (2.1). Kolejne dystrybucje są tworzone iteracyjnie według reguły

$$\boldsymbol{D}_{i+1} = \boldsymbol{D}_i \oplus [\boldsymbol{D}_0, \boldsymbol{D}_i], \quad i = 0, 1, \ldots. \tag{2.3}$$

Nawias Liego dystrybucji rozumie się jako zbiór nawiasów Liego pomiędzy każdą parą pól rozpinających daną dystrybucję. Każdej dystrybucji przypisany jest rząd w konfiguracji $\boldsymbol{q}$ oznaczony jako

$$f_i^{\mathbb{Q}}(\boldsymbol{q}) = \dim \boldsymbol{D}_i(\boldsymbol{q}), \quad i = 0, 1, \ldots \tag{2.4}$$

Wektor składający się z kolejnych rzędów konfiguracji $f_i(\boldsymbol{q})$ nazywany jest wektorem wzrostu w danej konfiguracji. Jeśli dla każdej konfiguracji $\boldsymbol{q}$ jest on taki sam to nazywany jest wektorem wzrostu. Minimalna liczba $i^\star$ dla której $f_{i^\star}(\boldsymbol{q}) = n$ nazywamy stopniem nieholonomiczności w konfiguracji $\boldsymbol{q}$, a jeśli nie zależy od konfiguracji to (analogicznie jak z rzędem) stopniem nieholonomiczności. Układ jest nieholonomiczny, jeśli dla każdej konfiguracji rząd dystrybucji osiąga wartość wymiaru przestrzeni konfiguracyjnej n

$$\forall \boldsymbol{q} \in \mathbb{Q} \;\; \exists i^\star \quad \dim \boldsymbol{D}_{i^\star}(\boldsymbol{q}) = f_{i^\star}^{\mathbb{Q}}(\boldsymbol{q}) = n. \tag{2.5}$$

W przypadku gdy nie istnieje $i^\star$ wymagane w (2.5) układ jest holonomiczny i może być sterowany jedynie w odpowiedniej podprzestrzeni przestrzeni konfiguracyjnej. Praktyczna procedura ustalania rzędów dystrybucji $\boldsymbol{D}_i$ polega na badaniu rzędu macierzy skonstruowanej z kolejnych (niezależnych od już dodanych) nawiasów Liego wynikających ze wzoru (2.3).

Do badania warunku LARC można podejść analogicznie jak do badania rzędu dystrybucji, czyli poprzez zbudowanie macierzy $\boldsymbol{F}_i(\boldsymbol{q})$ składającej się z kolejnych elementów bazy algebry Liego i zastąpieniem warunku (2.2) warunkiem sprawdzającym rząd $f_i(\boldsymbol{q})$ tej macierzy. Macierz $\boldsymbol{F}_i(\boldsymbol{q})$ składa się z kolejnych warstw bazy algebry Liego

$$\boldsymbol{F}_i(\boldsymbol{q}) = [\boldsymbol{H}^1, \boldsymbol{H}^2, ..., \boldsymbol{H}^i]. \tag{2.6}$$

Warunek (2.2) jest wtedy następujący

$$\forall \boldsymbol{q} \in \mathbb{Q} \quad \exists i^\star \quad \operatorname{rank}(\boldsymbol{F}_{i^\star}(\boldsymbol{q})) = n, \tag{2.7}$$

gdzie $i^\star$ oznacza numer warstwy dla której warunek (2.7) jest spełniony. Jeśli takie $i^\star$ istnieje, to znaczy że układ jest sterowalny w krótkim czasie. Macierz $\boldsymbol{F}_i$ tworzy się iteracyjnie, zaczynając od $i = 0$, aż do momentu spełnienia warunku (2.7). W przypadku kiedy wszystkie elementy warstwy $\boldsymbol{H}^{i+1}$ są zależne od już istniejących w $\boldsymbol{F}_i$ układ nie jest sterowalny w krótkim czasie.

W dalszej części rozprawy będziemy rozróżniać STLC w przestrzeni konfiguracyjnej od STLC w przestrzeni zadaniowej. STLC opisane w tym rozdziale jest definiowane w przestrzeni konfiguracyjnej i oznaczane będzie jako Q-STLC.

Przykład 3. Sprawdzenie sterowalności jednokołowego robota mobilnego (jednokołowca), opis modelu w dodatku A.1.

Generatory: $\boldsymbol{g}_1(\boldsymbol{q}) = (\cos(q_3), \sin(q_3), 0)^T$, $\boldsymbol{g}_2(\boldsymbol{q}) = (0, 0, 1)^T$

$$[\boldsymbol{g}_1, \boldsymbol{g}_2] = \frac{\partial \boldsymbol{g}_2(\boldsymbol{q})}{\partial \boldsymbol{q}} \boldsymbol{g}_1(\boldsymbol{q}) - \frac{\partial \boldsymbol{g}_1(\boldsymbol{q})}{\partial \boldsymbol{q}} \boldsymbol{g}_2(\boldsymbol{q}) = (\sin(q_3), -\cos(q_3), 0)^T$$

$$\boldsymbol{F}_2(\boldsymbol{q}) = [\boldsymbol{H}^1, \boldsymbol{H}^2] = \begin{bmatrix} \cos(q_3) & 0 & \sin(q_3) \\ \sin(q_3) & 0 & -\cos(q_3) \\ 0 & 1 & 0 \end{bmatrix},$$

$\det \boldsymbol{F}_2(\boldsymbol{q}) = 1$, więc $\operatorname{rank}(\boldsymbol{F}_2(\boldsymbol{q})) = 3$, i rząd równy jest wymiarowi przestrzeni konfiguracyjnej n. Z tego wynika, że układ jest sterowalny w krótkim czasie.

Generowanie baz algebry Liego jest operacją, która może być wykonana w trybie pre-planowania (*offline*), zatem sprawdzenie warunku LARC również może być przeprowadzone przed przystąpieniem do planowania. Zatem dla układów sterowalnych na etapie planowania jest wiadomo, które pola wektorowe z bazy rozpinają przestrzeń konfiguracyjną w jakim punkcie i jest możliwy wybór minimalnego zbioru pól na potrzeby planowania.

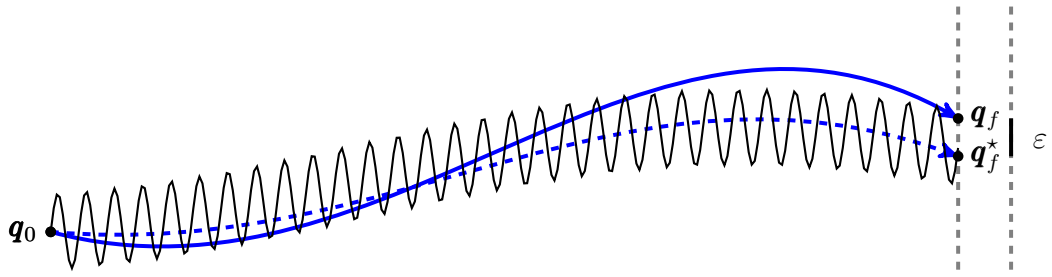
2.2 Metody ogólne

Poniżej opisano niektóre z metod planowania ruchu układów nieholonomicznych w przestrzeni konfiguracyjnej Spośród innych, niewymienionych w podrozdziale można wymienić metody typu lapunowskiego [9, 70, 101], metodę gradientów [6, 79, 105] oraz metody dedykowane dla niektórych typów układów nieholonomicznych jak metoda sterowań sinusoidalnych [67, 67, 99], metoda bazująca na twierdzeniu Stokesa [63] czy metoda osiągania podcelów [17].

2.2.1 Metoda uśredniania (Sussmann-Liu)

Metoda uśredniania wykorzystuje założenie, że zmiana stanu układu w czasie może być zdekomponowana na składowe wolno- i szybkozmienne [53]. Przy takim założeniu składowe szybkozmienne wprowadzają tylko oscylacje wokół trajektorii odpowiadającej składowej wolnozmiennej. Przebiegi wolno- i szybkozmienne opisuje się za pomocą kombinacji

liniowej elementów pewnej bazy. Następnie szuka współczynników liczbowych tej kombinacji, przy założonej maksymalnej wartości błędu osiągniecia stanu końcowego ε. Rys. 2.1 ilustruje ideę metody uśredniania. Kolejne kroki metody uśredniania dla przypadku gdy



Rysunek 2.1 Graficzna ilustracja idei metody uśredniania dla układów nieholonomicznych.

generatory wraz z nawiasami Liego drugiego stopnia rozpinają przestrzeń konfiguracyjną (macierz $\boldsymbol{F}_2$ pełnego rzędu) zebrano w Algorytmie 2.1.

Algorytm 2.1 Metoda uśredniania

Dane wejściowe: Układ nieholonomiczny (2.1), macierz $\boldsymbol{F}_2$, konfiguracja początkowa $\boldsymbol{q}_0$ oraz docelowa $\boldsymbol{q}_f$, margines błędu ε.

Krok 1. Wyznaczyć trajektorię gładką $\boldsymbol{q}(t)$, łączącą konfiguracje $\boldsymbol{q}_0$ i $\boldsymbol{q}_f$. Poprzez różniczkowanie otrzymać $\dot{\boldsymbol{q}} = \boldsymbol{f}(\boldsymbol{q})$ z warunkiem początkowym $\boldsymbol{q}(0) = \boldsymbol{q}_0$.

Krok 2. Wyliczyć sterowanie rozszerzone $\boldsymbol{v} = (v_1, \ldots, v_l)^T, \quad l = \frac{m(m-1)}{2} \geqslant n$ z równania

$$\boldsymbol{F}_2(\boldsymbol{q})\boldsymbol{v} = \boldsymbol{f}(\boldsymbol{q}) \quad \Longrightarrow \quad \boldsymbol{v} = \boldsymbol{F}^{\#}(\boldsymbol{q})\boldsymbol{f}(\boldsymbol{q}). \tag{2.8}$$

Krok 3. Znaleźć sterowania układu (2.1) postaci

$$u_i^\varepsilon(\boldsymbol{q}, t) = \alpha_i(\boldsymbol{q}) + \sqrt{\frac{2}{\varepsilon}} \sum_{j=1}^{l} \left(\beta_i^j(\boldsymbol{q}) \sin(\frac{jt}{\varepsilon}) + \gamma_i^j(\boldsymbol{q}) \cos(\frac{jt}{\varepsilon}) \right), \quad i \in \{0, \ldots, m\}, \tag{2.9}$$

aproksymujące efekt sterowań rozszerzonych.

2.2.2 Metoda Newtona dla układów nieholonomicznych

Podstawowa wersja algorytmu Newtona dla rozwiązywania równania macierzowego (jak dla kinematyki manipulatorów) została przedstawiona w podrozdziale 1.4. Dla układów nieholonomicznych tę ideę należy dostosować do specyfiki układów przez odpowiednią redefinicję pojęcia kinematyki. Na gruncie robotyki nieholonomicznej prekursorem był Wen i współpracownicy [76], a wersję z optymalizacją energii przedstawiono w pracy [23].

Bezdryfowy układ sterowania (2.1) przy ustalonym horyzoncie czasowym $T > 0$, stanie początkowym $\boldsymbol{q}_0 \in \mathbb{R}^n$ oraz docelowym $\boldsymbol{q}_f \in \mathbb{R}^n$ definiuje odwzorowanie

$$\boldsymbol{k}_{\boldsymbol{q}_0,t}(\boldsymbol{u}(\cdot)) : \mathbb{L}_m^2[0,t] \to \mathbb{R}^n, \tag{2.10}$$

nazywane w kontekście robotów mobilnych kinematyką robota mobilnego w chwili t. Zadanie znalezienia sterowań spełniających warunek

$$\boldsymbol{q}_d = \boldsymbol{k}_{\boldsymbol{q}_0,T}(\boldsymbol{u}(\cdot)), \tag{2.11}$$

nosi nazwę zadania planowania ruchu lub odwrotnego zadania kinematyki.

Dla danych sterowań $\boldsymbol{u}(\cdot)$ układ (2.1) można zastąpić jego zlinearyzowaną wersją, powstałą z rozwinięcia w szereg Taylora z pominieciem składowych wyższych niż liniowa,

$$\boldsymbol{\delta\dot{q}} = \boldsymbol{A}(t)\boldsymbol{\delta q} + \boldsymbol{B}(t)\boldsymbol{\delta u}, \quad \text{gdzie} \quad \boldsymbol{A}(t) = \frac{\partial(\boldsymbol{G}(\boldsymbol{q}(t))\boldsymbol{u}(t))}{\partial \boldsymbol{q}}, \quad \boldsymbol{B} = \boldsymbol{G}(\boldsymbol{q}(t)), \tag{2.12}$$

gdzie δx oznacza małe zaburzenie (czyli wariację) x. Zmiana stanu końcowego $\boldsymbol{\delta q}(T)$ spowodowana małą modyfikacją sterowań $\boldsymbol{\delta u}(\cdot)$ jest opisana równaniem

$$\boldsymbol{\delta q}(T) = \int_0^T \boldsymbol{\Phi}(t,s)\boldsymbol{B}(s)\boldsymbol{\delta u}(s)ds, \quad \boldsymbol{\delta q}(0) = \boldsymbol{0}, \tag{2.13}$$

gdzie T jest ustalonym horyzontem czasu, a macierz fundamentalna $\boldsymbol{\Phi}(t,s)$ spełnia warunek [34]

$$\frac{d}{dt}\boldsymbol{\Phi}(t,s) = \boldsymbol{A}(t)\boldsymbol{\Phi}(t,s), \quad \boldsymbol{\Phi}(s,s) = \boldsymbol{I}_n. \tag{2.14}$$

Aby przenieść zadanie (2.11) z nieskończonej przestrzeni funkcyjnej $\mathbb{L}^2_m[0,t]$ w skończoną przestrzeń parametrów zakłada się sterowania kawałkami stałe, lub parametryzuje przestrzeń sterowań za pomocą pewnej bazy, jak opisano w podrozdziale 1.5.

W przypadku parametryzacji sterowań wektorem $\boldsymbol{p} \in \mathbb{R}^m$ układ (2.12) jest redefiniowany do postaci

$$\boldsymbol{\delta\dot{q}} = \boldsymbol{A}_{\boldsymbol{p}}(t)\boldsymbol{\delta q} + \boldsymbol{B}_{\boldsymbol{p}}(t)\boldsymbol{\delta u}, \quad \text{gdzie}\boldsymbol{A}_{\boldsymbol{p}}(t) = \boldsymbol{A}(t), \quad \boldsymbol{B}_{\boldsymbol{p}} = \boldsymbol{B}(t)\frac{\partial \boldsymbol{u}(t,\boldsymbol{p})}{\partial \boldsymbol{p}}, \tag{2.15}$$

a równanie określające końcowe przemieszczenie (2.13) jako

$$\boldsymbol{\delta q}(T) = \int_0^T \boldsymbol{\Phi}_{\boldsymbol{p}}(T,s)\boldsymbol{B}_{\boldsymbol{p}}(s)\boldsymbol{\delta u}(s)ds\boldsymbol{\delta p} = \boldsymbol{J}_{\boldsymbol{q}_0,T}(\boldsymbol{u}(\cdot,\boldsymbol{p}))\boldsymbol{\delta p}, \tag{2.16}$$

gdzie $\boldsymbol{\Phi}_{\boldsymbol{p}}(T,s) = \boldsymbol{\Phi}(T,s)$, a jakobian $\boldsymbol{J}_{\boldsymbol{q}_o,T}(\boldsymbol{u}(\cdot,\boldsymbol{p})) = \boldsymbol{J}_{\boldsymbol{q}_0,T}(\boldsymbol{p})$ jest jakobianem układu (2.1).

W i-tym kroku algorytmu Newtona wektor parametrów jest modyfikowany według wzoru

$$\boldsymbol{p}_{i+1}(\cdot) = \boldsymbol{p}_i(\cdot) + \xi_i \boldsymbol{J}^{\#}_{\boldsymbol{q}_0,T}(\boldsymbol{q}_d - \boldsymbol{k}_{\boldsymbol{q}_0,T}(\boldsymbol{u}_i(\cdot,\boldsymbol{p}_i))), \tag{2.17}$$

a dane do $\boldsymbol{J}_{\boldsymbol{q}_0,T}(\boldsymbol{p})$ są wyliczane według formuł

$$\boldsymbol{B}_{\boldsymbol{p}}(t) = [\boldsymbol{g}_1\boldsymbol{\phi}^1, \boldsymbol{g}_2\boldsymbol{\phi}^2, \ldots, \boldsymbol{g}_m\boldsymbol{\phi}^m], \quad \text{z} \quad \boldsymbol{g}_k\boldsymbol{\phi}^k = [\boldsymbol{g}_k\phi_1, \boldsymbol{g}_k\phi^2, \ldots, \boldsymbol{g}_k\phi^{K_i}], \tag{2.18}$$

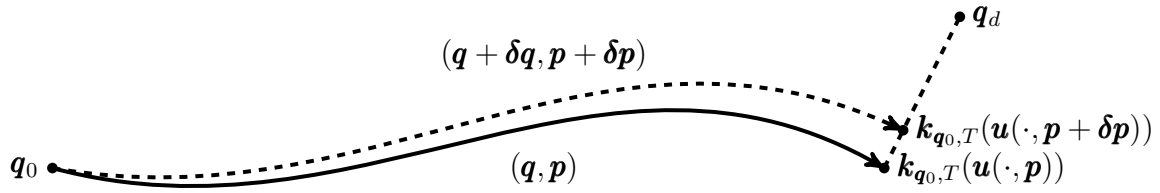
gdzie $\boldsymbol{g}_i$ odnosi się do układu (2.1), a $\phi_i, i \in \{0, \ldots, K_i\}$ są funkcjami bazowymi parametryzacji sterowań.

$$\boldsymbol{\Phi}_{\boldsymbol{p}}(T,s) = \lim_{\Delta s \to 0}(\boldsymbol{I}_n + \boldsymbol{A}_{\boldsymbol{p}}(\tau_r)\Delta s)\ldots(\boldsymbol{I}_n + \boldsymbol{A}_{\boldsymbol{p}}(\tau_1)\Delta s) \tag{2.19}$$

gdzie $\Delta s = (T-s)/r$, $\tau_k = (k+0.5)\Delta s$, a r jest liczbą podprzedziałów przedziału $[s,T]$.

Na rys. 2.2 przedstawiono graficznie interpretację działania algorytmu Newtona dla

układów nieholonomicznych.



Rysunek 2.2 Graficzna ilustracja idei metody Newtona dla układów nieholonomicznych.

2.2.3 Metoda endogenicznej przestrzeni konfiguracyjnej

Metoda przestrzeni endogenicznej jest poszerzeniem metody Newtona i umożliwia planowanie ruchu afinicznych układów sterowania z funkcją wyjścia [77, 96]

$$\begin{aligned} \dot{\boldsymbol{q}} &= \boldsymbol{f}(\boldsymbol{q}) + \boldsymbol{G}(\boldsymbol{q})\boldsymbol{u}, \\ \boldsymbol{x} &= \boldsymbol{k}(\boldsymbol{q}), \end{aligned} \tag{2.20}$$

$$\boldsymbol{q} \in \mathbb{Q}, \quad \boldsymbol{x} \in \mathbb{X}, \quad \boldsymbol{u} \in \mathbb{U}, \quad n = \dim \mathbb{Q}, \quad r = \dim \mathbb{X}, \quad m = \dim \mathbb{U}, \quad n \geqslant r, \quad n > m.$$

W przypadku braku dryfu ($\boldsymbol{f}(\boldsymbol{q}) = 0$) oraz identycznościowej funkcji wyjścia $\boldsymbol{k}$ układ (2.20) sprowadza się do bezdryfowego układu sterowania (2.1). Jak uprzednio, zadanie planowanie ruchu układu polega na znalezieniu sterowania $\boldsymbol{u}^\star(\cdot)$ które przeprowadzi układ z konfiguracji początkowej $\boldsymbol{q}_0 = \boldsymbol{q}(0)$ do położenia docelowego $\boldsymbol{x}_f$ w określonym czasie T

$$\boldsymbol{k}(\boldsymbol{q}_0) = \boldsymbol{k}(\boldsymbol{q}(0)) \xrightarrow{\boldsymbol{u}^\star([0,T])} \boldsymbol{k}(\boldsymbol{q}(T)) = \boldsymbol{x}_f. \tag{2.21}$$

Endogeniczna przestrzeń konfiguracyjna $\mathcal{U}$ zawiera dopuszczalne funkcje sterujące dla układu (2.20), $\boldsymbol{u}(\cdot) \in \mathcal{U} = \mathbb{L}^2_m[0,T]$ i jest przestrzenią Hilberta z iloczynem skalarnym

$$\langle \boldsymbol{u}_1(\cdot), \boldsymbol{u}_2(\cdot) \rangle = \int_0^T \boldsymbol{u}_1^T(t)\boldsymbol{u}_2(t)dt \tag{2.22}$$

oraz normą $\|\boldsymbol{u}(\cdot)\|^2_{\mathcal{U}} = \langle \boldsymbol{u}(\cdot), \boldsymbol{u}(\cdot) \rangle$. Niech $\boldsymbol{\varphi}_{\boldsymbol{q}_0,t}(\boldsymbol{u}(\cdot))$ będzie strumieniem układu (2.20) poddanemu sterowaniu $\boldsymbol{u}(\cdot)$, wyznaczonym w chwili t i zapoczątkowanym w punkcie $\boldsymbol{q}_0$. Wówczas

$$\boldsymbol{K}_{\boldsymbol{q}_0,T}(\boldsymbol{u}(\cdot)) = \boldsymbol{k}(\boldsymbol{\varphi}_{\boldsymbol{q}_0,T}(\boldsymbol{u}(\cdot))) = \boldsymbol{k}(\boldsymbol{q}(T)) = \boldsymbol{x}(T) \tag{2.23}$$

będzie odwzorowaniem końcowym, określającym wartość funkcji wyjścia układu (2.20) poddanemu sterowaniu $\boldsymbol{u}(\cdot)$ z konfiguracji początkowej $\boldsymbol{q}_0$ w chwili T. Różniczkując odwzorowanie końcowe (2.23) otrzymujemy jakobian

$$\boldsymbol{J}_{\boldsymbol{q}_0,T}(\boldsymbol{u}(\cdot))\boldsymbol{v}(\cdot) = \frac{d}{d\alpha}|_{\alpha=0}\boldsymbol{K}_{\boldsymbol{q}_0,T}(\boldsymbol{u}(\cdot) + \alpha\boldsymbol{v}(\cdot)) = \boldsymbol{C}(T)\int_0^T \boldsymbol{\Phi}(T,s)\boldsymbol{B}(s)\boldsymbol{v}(s)ds. \tag{2.24}$$

Macierz $\boldsymbol{\Phi}(t,s)$ w równaniu (2.24) jest macierzą tranzycji przybliżenia liniowego

$$\begin{aligned}\dot{\boldsymbol{\xi}}(t) &= \boldsymbol{A}(t)\boldsymbol{\xi}(t) + \boldsymbol{B}(t)\boldsymbol{v}(t),\\ \boldsymbol{\eta} &= \boldsymbol{C}(t)\boldsymbol{\xi},\end{aligned} \tag{2.25}$$

układu (2.20) wzdłuż trajektorii $(\boldsymbol{q}(t),\boldsymbol{u}(t))$. Macierz tranzycji przybliżenia liniowego można wyznaczyć z równania różniczkowego

$$\frac{\partial}{\partial t}\boldsymbol{\Phi}(t,s) = \boldsymbol{A}(t)\boldsymbol{\Phi}(t,s), \quad \text{z} \quad \boldsymbol{\Phi}(s,s) = \boldsymbol{I}_n, \tag{2.26}$$

a macierze przybliżenia definiowane są jako

$$\boldsymbol{A}(t) = \frac{\partial\Big(\boldsymbol{f}(\boldsymbol{q}(t)) + \boldsymbol{G}(\boldsymbol{q}(t))\boldsymbol{u}(t)\Big)}{\partial \boldsymbol{q}}, \quad \boldsymbol{B}(t) = \boldsymbol{G}(\boldsymbol{q}(t)), \quad \boldsymbol{C}(t) = \frac{\partial \boldsymbol{k}(\boldsymbol{q}(t))}{\partial \boldsymbol{q}}.$$

Jakobian pseudoodwrotny jakobianu (2.24) przyjmuje następującą postać

$$\Big(\boldsymbol{J}^{\#}_{\boldsymbol{q}_0,T}(\boldsymbol{u}(\cdot))\boldsymbol{\eta}\Big)(t) = \boldsymbol{B}^T(t)\boldsymbol{\Phi}^T(T,t)\boldsymbol{C}^T(T)\boldsymbol{\mathcal{D}}^{-1}_{\boldsymbol{q}_0,T}(\boldsymbol{u}(\cdot))\boldsymbol{\eta}, \tag{2.27}$$

gdzie

$$\boldsymbol{\mathcal{D}}_{\boldsymbol{q}_0,T}(\boldsymbol{u}(\cdot)) = \boldsymbol{C}(T)\int_0^T \boldsymbol{\Phi}(T,s)\boldsymbol{B}(s)\boldsymbol{B}^T(s)\boldsymbol{\Phi}^T(T,s)\,ds\,\boldsymbol{C}^T(T) \tag{2.28}$$

jest macierzą Grama układu (2.25). Taka pseudoodwrotność minimalizuje kwadrat normy przyrostów sterowania $\|\boldsymbol{v}(\cdot)\|^2_{\mathcal{U}}$. Powyższy jakobian pseudoodwrotny jest jednym z kilku możliwych, a właściwości innych zostały opisane w pracy [96].

Algorytm planowania ruchu bazuje na metodzie kontynuacji. Należy wybrać gładką krzywą $\boldsymbol{u}_\vartheta(\cdot), \vartheta \in \mathbb{R}$ w endogenicznej przestrzeni konfiguracyjnej $\mathcal{U}$, przechodzącą przez pewną konfigurację początkową $\boldsymbol{u}_0(\cdot)$. Zakłada się, że błąd planowania ruchu wzdłuż krzywej $\boldsymbol{u}_\vartheta(\cdot) \in \mathcal{U}$

$$e(\vartheta) = \boldsymbol{K}_{\boldsymbol{q}_0,T}(\boldsymbol{u}_\vartheta(\cdot)) - \boldsymbol{x}_f, \tag{2.29}$$

będzie zbiegał eksponencjalne do zera z parametrem $\gamma > 0$

$$\frac{de(\vartheta)}{d\vartheta} = -\gamma e(\vartheta). \tag{2.30}$$

Po podstawieniu równania (2.29) do (2.30), zróżniczkowaniu zgodnie z (2.24) otrzymuje się równanie Ważewskiego-Dawidenki

$$\boldsymbol{J}_{\boldsymbol{q}_0,T}(\boldsymbol{u}_\vartheta(\cdot)) = \frac{d\boldsymbol{u}_\vartheta(\cdot)}{d\vartheta} = -\gamma e(\vartheta). \tag{2.31}$$

Z równania (2.31), dzięki skorzystaniu z pseudoodwrotności (2.27), uzyskuje się równanie definiujące algorytm ciągły planowania ruchu

$$\frac{d\boldsymbol{u}_\vartheta(\cdot)}{d\vartheta} = -\gamma\Big(\boldsymbol{J}^{\#}_{\boldsymbol{q}_0,T}(\boldsymbol{u}(\cdot))e(\vartheta)\Big)(t). \tag{2.32}$$

W wersji dyskretnej równanie (2.32) przyjmuje postać

$$\boldsymbol{u}_{\vartheta+1}(t) = \boldsymbol{u}_\vartheta(t) - \gamma\Big(\boldsymbol{J}^{\#}_{\boldsymbol{q}_0,T}(\boldsymbol{u}(\cdot))e(\vartheta)\Big)(t), \quad \vartheta = 1,2,\ldots \tag{2.33}$$

Równanie (2.33) można rozwiązać za pomocą metody Eulera rozwiązywania równań różniczkowych. Zazwyczaj jednak korzysta się z parametryzacji sterowań opisanej w podrozdziale 1.5, a parametryczna wersja (2.33) jest następująca

$$\boldsymbol{p}_{\vartheta+1} = \boldsymbol{p}_{\vartheta} - \gamma\Big(\boldsymbol{J}^{\#}_{\boldsymbol{q}_0,T}(\boldsymbol{p})e(\vartheta)\Big)(t), \quad \vartheta = 1, 2, \ldots, \tag{2.34}$$

gdzie $\boldsymbol{J}^{\#}_{\boldsymbol{q}_0,T}(\boldsymbol{p}) = \boldsymbol{J}^{T}_{\boldsymbol{q}_0,T}(\boldsymbol{p})\Big(\boldsymbol{J}_{\boldsymbol{q}_0,T}(\boldsymbol{p})\boldsymbol{J}^{T}_{\boldsymbol{q}_0,T}(\boldsymbol{p})\Big)^{-1}$. Jakobian $\boldsymbol{J}_{\boldsymbol{q}_0,T}(\boldsymbol{p})$ można wyznaczyć korzystając z równania różniczkowego

$$\frac{d\boldsymbol{J}_{\boldsymbol{q}_0,t}(\boldsymbol{p})}{dt} = \boldsymbol{A}(t)\boldsymbol{J}_{\boldsymbol{q}_0,t}(\boldsymbol{p}) + \boldsymbol{B}(t)\boldsymbol{P}(t), \tag{2.35}$$

gdzie $\boldsymbol{P}(t)$ jest blokowo-diagonalną macierzą zawierającą funkcje bazowe parametryzacji sterowań ($\boldsymbol{u}(t) = \boldsymbol{P}(t)\boldsymbol{p}$).

2.2.4 Metoda Lie-algebraiczna

Metoda Lie-algebraiczna jest lokalną metodą planowania ruchu układów nieholonomicznych bazującą na algebrze Liego układu [23, 97]. Bazuje na iteracyjnym generowaniu możliwych kierunków ruchu (pól wektorowych) w aktualnej konfiguracji przestrzeni stanu, wyliczeniu sterowań dla konkretnego kierunku, a następnie realizacji ruchu. Lokalność metody jest warunkowana tym, że ruch wzdłuż wygenerowanego kierunku zachodzi jedynie dla infinitezymalnych przemieszczeń – im większy ruch zakładamy, tym różnica pomiędzy przewidywanym kierunkiem ruchu a rzeczywistym może być większa. Zaletami metody są konstruktywność i ogólność, jak również duże możliwości optymalizacji pojedynczego kroku.

Zadanie planowania ruchu polega na znalezieniu takich sterowań $\boldsymbol{u}(\cdot)$, które pozwolą na przeprowadzenie układu (2.1) z konfiguracji początkowej $\boldsymbol{q}_0$ do docelowej $\boldsymbol{q}_f$. Ponieważ w układzie tym liczba sterowań jest mniejsza od wymiarowości przestrzeni konfiguracyjnej należy wygenerować odpowiednią sekwencję ruchów pozwalającą na ruch także w kierunkach innych niż bezpośrednio stowarzyszonych ze sterowaniami. Istnieją dwie drogi do osiągnięcia tego celu wykorzystujące algebrę Liego układu (2.1). W jednej wykorzystuje się formułę Campbella-Bakera-Hausdorffa-Dynkina (CBHD) i otrzymuje się sterowania kawałkami stałe [17, 92]. Drugą drogą jest zastosowanie uogólnionej wersji formuły (gCBHD) w wyniku której otrzymuje się sterowania kawałkami ciągłe.

2.2.4.1 Sterowania kawałkami stałe – formuła CBHD

Formuła Campbella-Bakera-Hausdorffa-Dynkina (CBHD) dla analitycznych pól wektorowych $\boldsymbol{X}, \boldsymbol{Y}$ ma postać

$$\begin{aligned}\exp(t\boldsymbol{X})\exp(t\boldsymbol{Y}) = \exp\Big(&t\boldsymbol{X} + t\boldsymbol{Y} + \frac{t^2}{2}[\boldsymbol{X},\boldsymbol{Y}] + \frac{t^3}{12}[[\boldsymbol{X},\boldsymbol{Y}],\boldsymbol{Y}] + \\ &- \frac{t^3}{12}[[\boldsymbol{X},\boldsymbol{Y}],\boldsymbol{X}] - \frac{t^4}{24}[\boldsymbol{X},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]] + \ldots\Big)\end{aligned}, \tag{2.36}$$

Przyjęta konwencja zapisu (2.36) warunkuje kolejność wykonywania ruchów wzdłuż strumieni pól wektorowych. I tak w przypadku (2.36) najpierw przez czas t układ porusza się wzdłuż pola wektorowego $\boldsymbol{X}$, a następnie przez ten sam czas wzdłuż $\boldsymbol{Y}$. Rezultatem

(wypadkowym) takiego działania jest nie tylko składnik pola $t\boldsymbol{X} + t\boldsymbol{Y}$, ale także reszta szeregu, w którym dla małych t najbardziej znaczącym składnikiem jest pole $[\boldsymbol{X}, \boldsymbol{Y}]$.

W ogólnej postaci formuła CBHD jest następująca [24]

$$\exp(t\boldsymbol{X})\exp(t\boldsymbol{Y}) = \exp\Big(\sum_{m=1}^{\infty}\sum \frac{(-1)^{m-1}(\mathrm{ad}_{\boldsymbol{Y}})^{q_m}(\mathrm{ad}_{\boldsymbol{X}})^{p_m}\ldots\mathrm{ad}_{\boldsymbol{Y}})^{q_1}(\mathrm{ad}_{\boldsymbol{X}})^{p_1}}{m\sum_{i=1}^{m}(p_1+q_1)\prod_{i=1}^{m}(p_i!q_i!)}\Big), \tag{2.37}$$

gdzie wewnętrzna suma jest wyliczana po wszystkich m-krotkach par nieujemnych liczb całkowitych (p_i, q_i) takich, że $p_i + q_i > 0$. Operator $\mathrm{ad}_{\boldsymbol{X}} : \boldsymbol{Y} \to [\boldsymbol{X}, \boldsymbol{Y}]$, z zastrzeżeniem, że dla skrajnie prawego elementu $\mathrm{ad}^0_{\boldsymbol{X}} = 1$ i $\mathrm{ad}^1_{\boldsymbol{X}} = \boldsymbol{X}$. Formułę CBHD często przestawia się również w postaci [17]

$$\exp(t\boldsymbol{X})\exp(t\boldsymbol{Y})\exp(-t\boldsymbol{X}) = \exp\Big(\sum_{i=0}^{\infty}(\frac{t^{i+1}}{i!}\mathrm{ad}^i_{\boldsymbol{X}}\boldsymbol{Y})\Big), \tag{2.38}$$

gdzie $\mathrm{ad}^0_{\boldsymbol{X}}\boldsymbol{Y} = \boldsymbol{Y}$ i $\mathrm{ad}^i_{\boldsymbol{X}}\boldsymbol{Y} = [\boldsymbol{X}, \mathrm{ad}^{i-1}_{\boldsymbol{X}}\boldsymbol{Y}]$. Warto zwrócić uwagę że formuła ta w każdym wariancie jest szeregiem nieskończonym. Specjalną klasą układów, dla których formuła CBHD staje się szeregiem skończonym są układu nilpotentne, tj. takie dla których wszystkie elementy algebry Liego powyżej pewnej warstwy są równe $\mathbf{0}$ (baza algebry Liego jest skończeniewymiarowa).

W przypadku układów nienilpotentnych formułę CBHD należy na odpowiedniej warstwie "przyciąć" – analogicznie jak w przypadku rozwinięcia funkcji w szereg Taylora. Ideę wykorzystania formuły CBHD w generowaniu sterowań pokazuje przykład 4

Przykład 4. Generowanie ruchu w kierunku $[\boldsymbol{X}, \boldsymbol{Y}]$ przy pomocy pól $\boldsymbol{X}$ i $\boldsymbol{Y}$.

$$\begin{aligned}
&\log\Big(\exp(t\boldsymbol{X})\exp(t\boldsymbol{Y})\exp(-t\boldsymbol{X})\exp(-t\boldsymbol{Y})\Big) \overset{(2.38)}{=} \\
&\overset{(2.38)}{=} \log\Big(\exp\Big(t\boldsymbol{Y} + t^2[\boldsymbol{X},\boldsymbol{Y}] + \frac{t^3}{2}[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]] + o(t^3)\Big)\exp(-t\boldsymbol{Y})\Big) \overset{(2.36)}{=} \\
&\overset{(2.36)}{=} t\boldsymbol{Y} + t^2[\boldsymbol{X},\boldsymbol{Y}] + \frac{t^3}{2}[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]] - t\boldsymbol{Y} + \frac{t^2}{2}[\boldsymbol{Y} + t[\boldsymbol{X},\boldsymbol{Y}] + \\
&+ \frac{t^2}{2}[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]], -\boldsymbol{Y}] + o(t^3) = t^2[\boldsymbol{X},\boldsymbol{Y}] + \frac{t^3}{2}[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]] + \frac{t^3}{2}[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]] + o(t^3).
\end{aligned}$$

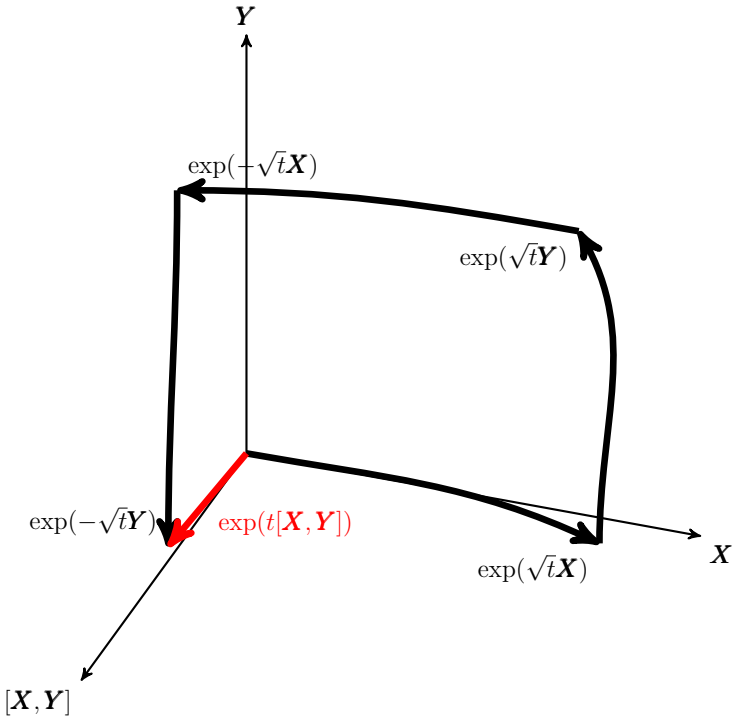
Dla małych wartości t najbardziej znaczącą częścią równania jest $t^2[\boldsymbol{X}, \boldsymbol{Y}]$, reszta jest pomijana. Aby uzależnić kierunek $[\boldsymbol{X}, \boldsymbol{Y}]$ od t zamiast t^2 stosuje się przeskalowanie

$$\begin{aligned}
&\log\Big(\exp(\sqrt{t}\boldsymbol{X})\exp(\sqrt{t}\boldsymbol{Y})\exp(-\sqrt{t}\boldsymbol{X})\exp(-\sqrt{t}\boldsymbol{Y})\Big) \overset{(2.38)}{=} \\
&= t[\boldsymbol{X},\boldsymbol{Y}] + \frac{t^{3/2}}{2}[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]] + \frac{t^{3/2}}{2}[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]] + o(t^{3/2})
\end{aligned}. \tag{2.39}$$

Sterowania generujące ruch w kierunku $[\boldsymbol{X}, \boldsymbol{Y}]$ są łatwe do określenia z lewej część równania (2.39) i przyjmują wartości

$$u_1 = \begin{cases} 1 & \text{dla } t \in [0,1] \\ 0 & \text{dla } t \in (1,2] \\ -1 & \text{dla } t \in (2,3] \\ 0 & \text{dla } t \in (3,4] \end{cases}, \quad u_2 = \begin{cases} 0 & \text{dla } t \in [0,1] \\ 1 & \text{dla } t \in (1,2] \\ 0 & \text{dla } t \in (2,3] \\ -1 & \text{dla } t \in (3,4] \end{cases},$$

W tym przypadku rzeczywisty czas ruchu wynosi 4, który oczywiście można przeskalować, np. zmniejszając go kosztem zwiększenia amplitudy sterowań. Na rys. 2.3 zilustrowano graficznie trajektorię odpowiadającą powyższemu scenariuszowi przełączeń sterowań.



Rysunek 2.3 Wypadkowy ruch w kierunku pola $[\boldsymbol{X},\boldsymbol{Y}]$ jako złożenie ruchów wzdłuż generatorów układu z przykładu 4.

2.2.4.2 Sterowania ciągłe – formuła gCBHD

Uogólniona formuła Campbella-Bakera-Hausdorffa-Dynkina (gCBHD) opisuje (lokalne) rozwiązanie nieautonomicznego układu równań różniczkowych z danym warunkiem początkowym [92]

$$\dot{\boldsymbol{q}}(t) = \boldsymbol{A}(t)(\boldsymbol{q}(t)), \quad \boldsymbol{q}(0) = \boldsymbol{q}_0, \tag{2.40}$$

gdzie $\boldsymbol{A}(t)(\cdot)$ to rodzina analitycznych pól wektorowych parametryzowanych przez ciągłe t. Układ (2.40) może być utożsamiany z bezdryfowym układem sterowania (2.1)

$$\dot{\boldsymbol{q}}(t) = \boldsymbol{A}(t)(\boldsymbol{q}(t)) = \boldsymbol{G}(\boldsymbol{q})\boldsymbol{u} = \sum_{i=1}^{m} \boldsymbol{g}_i(\boldsymbol{q})u_i = \sum_{i=1}^{m} \boldsymbol{X}_i(\boldsymbol{q})u_i, \quad \boldsymbol{q} \in \mathbb{Q}, \quad n = \dim \mathbb{Q} > m. \tag{2.41}$$

Rozwiązanie takiego układu równań różniczkowych przybiera formę

$$\boldsymbol{q}(t) = \exp \boldsymbol{z}(t, \boldsymbol{q}(0)) \simeq \boldsymbol{z}(t)(\boldsymbol{q}(0)) + \boldsymbol{q}(0), \tag{2.42}$$

gdzie $\boldsymbol{z}(t)(\boldsymbol{q}_0)$ pełni rolę operatora przesunięcia, a $\exp \boldsymbol{z}(t)(\boldsymbol{q}(0))$ jest rozwiązaniem równania

$$\frac{d}{ds}\nu(s,t) = \boldsymbol{z}(t)\nu(s), \quad \nu(0,0) = \boldsymbol{q}(0), \tag{2.43}$$

$$\nu(s,t) = \exp(s\ \boldsymbol{z}(t))(\boldsymbol{q}(0)) \Rightarrow \boldsymbol{q}(t) = \nu(1,t) = \exp \boldsymbol{z}(t)(\boldsymbol{q}(0)). \tag{2.44}$$

Dla $t \to 0$, $\exp \boldsymbol{z}(t)(\boldsymbol{q}(0))$ przybiera formę nieskończonego szeregu

$$\boldsymbol{z}(t)(\boldsymbol{q}_0) \simeq \sum_{r=1}^{\infty} \sum_{\boldsymbol{\sigma}\in\mathbb{P}_r} \frac{(-1)^{\mathrm{err}(\boldsymbol{\sigma})}}{r^2 \binom{r-1}{\mathrm{err}(\boldsymbol{\sigma})}} \int_{\mathbb{T}_r(t)} [\ldots[\boldsymbol{A}(s_{\boldsymbol{\sigma}(1)}), \boldsymbol{A}(s_{\boldsymbol{\sigma}(2)})], \ldots], \boldsymbol{A}(s_{\boldsymbol{\sigma}(r)})] d\boldsymbol{s}^r, \tag{2.45}$$

gdzie:

$\mathbb{P}_r$ jest zbiorem wszystkich permutacji zbioru $\{1, \ldots, r\}$.
Na przykład $\mathbb{P}_3 = \{(1,2,3), (1,3,2), (2,1,3), (2,3,1), (3,1,2), (3,2,1)\}$

$\mathrm{err}(\boldsymbol{\sigma})$ to liczba błędów w permutacji $\boldsymbol{\sigma} = \{\sigma(1), \sigma(1), \ldots, \sigma(r)\}$. Błąd permutacji jest liczony w przypadku kiedy kolejny element permutacji jest mniejszy od obecnego.
Na przykład $\mathrm{err}((1,2,3)) = 0$, $\mathrm{err}((4,1,2,3)) = 1$, $\mathrm{err}((4,2,3,1)) = 2$

$\mathbb{T}_r(t)$ to r-wymiarowy sympleks, $\mathbb{T}_r = \{s \in \mathbb{R}^r : 0 \leqslant s_1 \leqslant s_2 \leqslant \ldots \leqslant s_r \leqslant t\}$. Całka po sympleksie $\int_{\mathbb{T}_r(t)}$ jest iteracyjną całką

$$\int_{T_r(t)} = \int_{s_r=0}^{t} \int_{s_{r-1}=0}^{s_r} \ldots \int_{s_1=0}^{s_2}. \tag{2.46}$$

$d\boldsymbol{s}^r$ jest uproszczonym zapisem $ds_1 ds_2 \ldots ds_r$.

Formuła (2.45) jest również czasem zapisywana w skróconej formie

$$\boldsymbol{z}(t)(\boldsymbol{q}_0) \simeq \sum_{r=1}^{\infty} \int_{T_r(t)} \left(\sum_{\boldsymbol{\sigma}\in P_r} c(\boldsymbol{\sigma}) \boldsymbol{E}_{\boldsymbol{\sigma}} \right) d\boldsymbol{s}^r, \tag{2.47}$$

jako suma po warstwach całek po sympleksie z sumy wszystkich elementów zaliczających się do tej warstwy. W formule (2.47) $\boldsymbol{E}_{\boldsymbol{\sigma}}$ dostajemy z odpowiedniej permutacji argumentów rekurencyjnego nawiasu Liego

$$\boldsymbol{E}_{\boldsymbol{\sigma}} = [\ldots[\boldsymbol{A}(s_{\boldsymbol{\sigma}(1)}), \boldsymbol{A}(s_{\boldsymbol{\sigma}(2)})], \ldots], \boldsymbol{A}(s_{\boldsymbol{\sigma}(r)})], \tag{2.48}$$

a $c(\boldsymbol{\sigma})$ jest współczynnikiem liczbowym zależnym od permutacji

$$c(\boldsymbol{\sigma}) = (-1)^{\mathrm{err}(\boldsymbol{\sigma})} \Big/ \left\{ r^2 \binom{r-1}{\mathrm{err}(\boldsymbol{\sigma})} \right\}. \tag{2.49}$$

Przemieszczenie konfiguracji $\boldsymbol{z}(t)(\boldsymbol{q}_0)$ wyliczone za pomocą formuły gCBHD przyjmuje formę sumy iloczynów, gdzie każdy iloczyn jest złożony z pola wektorowego wyliczonego w punkcie $\boldsymbol{q}_0$ oraz wyrażenia α zależnego od sterowań. Przykładowo, dla dwuwejściowego układu (2.41) ($m = 2$) przemieszczenie $\boldsymbol{z}(t)(\boldsymbol{q}_0)$ przyjmuje postać

$$\begin{aligned} \boldsymbol{z}(t)(\boldsymbol{q}_0) =& \alpha_1^1(\boldsymbol{u}(\cdot))\boldsymbol{X}_1(\boldsymbol{q}_0) + \alpha_2^1(\boldsymbol{u}(\cdot))\boldsymbol{X}_2(\boldsymbol{q}_0) + \alpha_1^2(\boldsymbol{u}(\cdot))[\boldsymbol{X}_1, \boldsymbol{X}_2](\boldsymbol{q}_0) + \\ &+ \alpha_1^3(\boldsymbol{u}(\cdot))[\boldsymbol{X}_1, [\boldsymbol{X}_1, \boldsymbol{X}_2]](\boldsymbol{q}_0) + \alpha_2^3(\boldsymbol{u}(\cdot))[\boldsymbol{X}_2, [\boldsymbol{X}_1, \boldsymbol{X}_2]](\boldsymbol{q}_0) + \\ &+ \alpha_1^4(\boldsymbol{u}(\cdot))[\boldsymbol{X}_1, [\boldsymbol{X}_1, [\boldsymbol{X}_1, \boldsymbol{X}_2]]](\boldsymbol{q}_0) + \alpha_2^4(\boldsymbol{u}(\cdot))[\boldsymbol{X}_2, [\boldsymbol{X}_1, [\boldsymbol{X}_1, \boldsymbol{X}_2]]](\boldsymbol{q}_0) + \\ &+ \alpha_3^4(\boldsymbol{u}(\cdot))[\boldsymbol{X}_2, [\boldsymbol{X}_2, [\boldsymbol{X}_1, \boldsymbol{X}_2]]](\boldsymbol{q}_0) + \ldots. \end{aligned} \tag{2.50}$$

Jakkolwiek formalnie zarówno $\boldsymbol{z}(t, \boldsymbol{q}_0, \boldsymbol{u}(\cdot))$ jak i funkcje $\boldsymbol{\alpha}(\boldsymbol{u}(\cdot))$ są zależne od sterowań $\boldsymbol{u}(\cdot)$, to dla skrócenia zapisu ten argument będzie często pomijany. Podobnie jak zależność pól wektorowych od $\boldsymbol{q}_0$ w formule gCBHD. Warto zwrócić uwagę, że w równaniu (2.50) wszystkie pola wektorowe wygenerowane z formuły gCBHD są już przedstawione jako kombinacje liniowe elementów bazy Ph. Halla. Sprowadzenie do bazy pozwala na zmniejszenie liczby składników w sumie i ułatwienie obliczeń bez straty ogólności. W kolejnych rozdziałach, tam gdzie stosowano bazę Ph. Halla można użyć dowolną inną bazę, chyba że wprost napisano inaczej. Algorytm reprezentujący jednomian Liego jako kombinację liniową elementów bazy Halla jest opisany w pracy [17]. W przykładzie (2.50) wykorzystano dwuwejściowy układ (2.41), gdyż dla takich układów konieczne elementy algebry Liego wyższych stopni pojawiają się najszybciej, czyniąc układy najciekawszymi z punktu widzenia trudności planowania ruchu.

Zależność wyrażeń α_d^r, w równaniu (2.50), od sterowań jest następująca [20]

$$\begin{aligned}
&\alpha_1^1 = \textstyle\int_0^t u_1 ds_1, \quad \alpha_2^1 = \int_0^t u_2 ds_1,\\
&\alpha_1^2 = \tfrac{1}{2}\textstyle\int_{T_2(t)}(u_{12} - u_{21})d\boldsymbol{s}^2,\\
&\alpha_1^3 = \tfrac{1}{6}\textstyle\int_{T_3(t)}(u_{112} - 2u_{121} + u_{211})d\boldsymbol{s}^3,\\
&\alpha_2^3 = \tfrac{1}{6}\textstyle\int_{T_3(t)}(-u_{122} + 2u_{212} - u_{221})d\boldsymbol{s}^3,\\
&\alpha_1^4 = \tfrac{1}{24}\textstyle\int_{T_4(t)}(-4u_{1121} + 4u_{1211})d\boldsymbol{s}^4,\\
&\alpha_2^4 = \tfrac{1}{24}\textstyle\int_{T_4(t)}(-4u_{1122} + 4u_{1212} - 4u_{2121} + 4u_{2211})d\boldsymbol{s}^4,\\
&\alpha_3^4 = \tfrac{1}{24}\textstyle\int_{T_4(t)}(-4u_{2122} + 4u_{2212})d\boldsymbol{s}^4,
\end{aligned} \tag{2.51}$$

a indeksy r (górny) i d (dolny) odpowiadają numerowi warstwy i elementu bazy Halla, odpowiednio. Wyrażenia $u_{i_1 i_2 \ldots i_r}$, będące częścią wyrażenia α_d^r, nazywamy pre-sterowaniami. Wprowadzenie pre-sterowań pozwala opisać zależność pomiędzy przesunięciem $\boldsymbol{z}(t)(\boldsymbol{q}_0)$ a sterowaniami bez konieczności określania ich natury. Pre-sterowania zapisywane są często w skróconej formie

$$u_{i_1 i_2 \ldots i_r} \stackrel{\text{def}}{=} u_{i_1}(s_1) u_{i_2}(s_2) \ldots u_{i_r}(s_r). \tag{2.52}$$

Indeks dolny w pre-sterowaniu $u_{i_1 i_2 \ldots i_r}$ opisuje sekwencję numerów kolejnych sterowań mnożonych w kolejności wynikającej z rosnących indeksów występujących w argumentach sterowań.

Ogólna postać wzoru (2.50) na przesunięcie konfiguracji $\boldsymbol{z}(t, \boldsymbol{q}_0)$ w otoczeniu konfiguracji $\boldsymbol{q}_0$ jest następująca

$$\boldsymbol{z}(t, \boldsymbol{q}_0, \boldsymbol{u}(\cdot)) = \sum_{r=1}^{\infty} \sum_{d=1}^{\#\boldsymbol{H}^r} \alpha_d^r \, \boldsymbol{H}_d^r(\boldsymbol{q}_0), \tag{2.53}$$

gdzie $\boldsymbol{H}_d^r$ jest d-tym elementem r-tej warstwy bazy Ph. Halla, a $\#\boldsymbol{H}^r$ liczbą elementów w r-tej warstwie. Dla układów nienilpotentnych, szereg (2.53) jest nieskończony. Ze względu obliczeniowego należy więc szereg aproksymować szeregiem skończonym z zachowaniem warunku LARC. Otrzymujemy dzięki temu równanie macierzowo-wektorowe prawdziwe w otoczeniu $\boldsymbol{q}_0$

$$\boldsymbol{z}(t, \boldsymbol{q}_0, \boldsymbol{u}(\cdot)) \simeq \boldsymbol{F}_{i^\star}(\boldsymbol{q}_0) \, \boldsymbol{\alpha}(\boldsymbol{u}(\cdot)), \tag{2.54}$$

gdzie macierz $\boldsymbol{F}_{i^\star}$ jest złożona z elementów bazy algebry Liego do i-tej warstwy włącznie (2.6), a $\boldsymbol{\alpha}(\boldsymbol{u}(\cdot))$ jest wektorem wyrażeń zależnych od sterowań. Należy zwrócić uwagę, że macierz $\boldsymbol{F}_{i^\star}$ niekoniecznie jest minimalną, w sensie liczby kolumn, macierzą spełniającą

warunek rzędu. Podobnie jak dla CBHD wszystkie elementy ustalonej warstwy porównywalnie wpływają na przesunięcie $\boldsymbol{z}(t, \boldsymbol{q}_0, \boldsymbol{u}(\cdot))$. Jeśli więc jednomian Liego j-tego stopnia jest niezbędny do spełnienia twierdzenia Chow, to macierz $\boldsymbol{F}_{i^\star}$ musi zawierać wszystkie elementy bazowe z warstw do j-tej włącznie. Warto podkreślić, że macierz $\boldsymbol{F}_{i^\star}$ jest wartościowana w punkcie $\boldsymbol{q}_0$, więc ucięty szereg $\boldsymbol{z}(t, \boldsymbol{q}_0, \boldsymbol{u}(\cdot))$ dobrze opisuje ruch jedynie w otoczeniu punktu $\boldsymbol{q}_0$, w którym pola wektorowe nie będą się drastycznie zmieniać. Na przekształcenie równania (2.53) w (2.54) można spojrzeć analogicznie jak na ucięcie nieskończonego szeregu Taylora przy określaniu przybliżonej wartości funkcji w danym punkcie.

Przy założeniu stałego czasu ruchu T, równanie (2.54) jest wykorzystywane do opisu jednego lokalnego planowania odpowiadającego przemieszczeniu układu w kierunku punktu docelowego $\boldsymbol{q}_f$

$$\boldsymbol{z}(T, \boldsymbol{q}_0, \boldsymbol{u}(\cdot)) = \xi \cdot (\boldsymbol{q}_f - \boldsymbol{q}_0), \tag{2.55}$$

gdzie dodatni parametr ξ powinien być dobrany tak, by zapewnić przesunięcie dla którego przybliżenie (2.54) jest wiarygodne. Algorytmiczne znalezienie sterowań $\boldsymbol{u}(\cdot)$ rozwiązujących równanie (2.54) nie jest zadaniem łatwym, gdyż sterowania $\boldsymbol{u}(\cdot)$ mogą być dowolnymi funkcjami ciągłymi. Z tego powodu stosuje się często parametryzacje sterowań za pomocą skończonej bazy funkcji oraz ich współczynników liczbowych dla poszczególnych sterowań zgromadzonych w wektorze $\boldsymbol{p}$, zamieniając zadanie znalezienia sterowań $\boldsymbol{u}(\cdot)$ na zadanie znalezienia parametrów $\boldsymbol{p}$. Algorytmy rozwiązujące równanie macierzowe

$$\boldsymbol{z}(t, \boldsymbol{q}_0, \boldsymbol{u}(\boldsymbol{p})) \simeq \boldsymbol{F}_{i^\star}(\boldsymbol{q}_0)\, \boldsymbol{\alpha}(\boldsymbol{u}(\boldsymbol{p})), \tag{2.56}$$

dla przyjętej parametryzacji $\boldsymbol{u}(\boldsymbol{p})$, są opisane w podrozdziale 1.4. Wymiarowość wektora parametrów $\boldsymbol{p}$ musi wynosić przynajmniej n. W przeciwnym przypadku nie ma możliwości ruchu w każdym kierunku zapewnionym przez STLC. Teoretycznie wektor $\boldsymbol{p}$ o wymiarze n może umożliwić ruch w każdym kierunku, jednak w takim przypadku funkcje bazowe parametryzacji sterowań muszą być właściwie dobrane. W praktyce najczęściej stosuje się wektory parametryzacji posiadające niewiele więcej elementów niż n. Problem doboru parametrów sterowań został bardziej szczegółowo opisany w rozdziale 5. W dalszej części pracy używany będzie skrócony zapis $\boldsymbol{\alpha}(\boldsymbol{p})$ oznaczający faktycznie $\boldsymbol{\alpha}(\boldsymbol{u}(\boldsymbol{p}))$.

Warto zwrócić uwagę, że $\boldsymbol{F}_{i^\star}(\boldsymbol{q}_0)$ we wzorze (2.56) jest stałą dla ustalonego $\boldsymbol{q}_0$, dlatego jakobian przekształcenia przybiera łatwiejszą obliczeniowo postać

$$\boldsymbol{J} = \boldsymbol{J}_{q_0}(\boldsymbol{p}) = \frac{\partial(\boldsymbol{F}_{i^\star}(\boldsymbol{q}_0)\boldsymbol{\alpha}(\boldsymbol{p}))}{\partial \boldsymbol{p}} = \boldsymbol{F}_{i^\star}(\boldsymbol{q}_0)\frac{\partial \boldsymbol{\alpha}(\boldsymbol{p})}{\partial \boldsymbol{p}}. \tag{2.57}$$

Po przyjęciu parametryzacji, analityczna postać pochodnej $\partial\boldsymbol{\alpha}(\boldsymbol{p})/\partial\boldsymbol{p}$ może być wyliczona raz (w trybie *offline*), przed uruchomieniem algorytmu Newtona.

2.2.4.3 Algorytm metody Lie-algebraicznej

W ramach podsumowania wszystkie kroki Lie-algebraicznej metody planowania ruchu zostały zebrane w Algorytmie 2.2.

Ze względu na aproksymacyjny charakter przesunięcia $\boldsymbol{z}(t, \boldsymbol{q}_c)$ (2.56) zwykle $\boldsymbol{q}_c^\star$ niekoniecznie będzie równe $\boldsymbol{q}_c + \xi\boldsymbol{v}$, jednak dla odpowiednio małych parametrów ξ oraz T różnica pomiędzy tymi konfiguracjami może być dowolnie mała.

Algorytm 2.2 Lie-algebraiczna metoda planowania ruchu.

Dane wejściowe: Bezdryfowy układ nieholonomiczny (2.1), konfiguracje początkowa $\boldsymbol{q}_0 \in \mathbb{R}^n$ oraz docelowa $\boldsymbol{q}_f \in \mathbb{R}^n$, dokładność osiągnięcia celu $\varepsilon > 0$, maksymalny zakres pojedynczego planowania ξ oraz pojedynczego ruchu T.

Krok 1. Wygenerować pierwsze $i^\star$ warstw bazy algebry Liego (np. bazy Ph. Halla) tak, aby macierz $\boldsymbol{F}_{i^\star}$ złożona z tych warstw (patrz (2.6)) spełniała warunek LARC (2.7) w każdym punkcie $\boldsymbol{q} \in \mathbb{Q}$.

Krok 2. Za konfigurację bieżącą $\boldsymbol{q}_c$ podstawić konfigurację początkową $\boldsymbol{q}_0$ i zainicjować pustą trajektorię wynikową, z czasem początkowym $t^\star \leftarrow 0$.

Krok 3. Sprawdzić warunek osiągnięcia konfiguracji docelowej $\|\boldsymbol{q}_c - \boldsymbol{q}_f\| < \varepsilon$. Jeśli warunek jest spełniony – zakończyć algorytm zwracając trajektorię wynikową, jeśli nie – przejść do następnego kroku.

Krok 4. Dla bieżącej iteracji określić pożądany kierunek ruchu $\boldsymbol{v}$. Może to być kierunek stowarzyszony z jednym z pól wektorowych wchodzących w skład $\boldsymbol{F}_{i^\star}$ wyliczonym w punkcie $\boldsymbol{q}_c$, lub określony inaczej (np. $\boldsymbol{v} \leftarrow \boldsymbol{q}_f - \boldsymbol{q}_0$). Kierunek ten powinien zapewnić wypadkowy ruch ku punktowi $\boldsymbol{q}_f$, czyli spełniać warunek $\langle \boldsymbol{v}, \boldsymbol{q}_f - \boldsymbol{q}_0 \rangle > 0$.

Krok 5. Wykorzystując formułę CBHD lub gCBHD wyznaczyć sterowania realizujące ruch w kierunku $\xi \boldsymbol{v}$ w czasie T, gdzie ξ jest rzeczywistym, dodatnim parametrem określającym zakres pojedynczego kroku.

Krok 6. Wyznaczyć trajektorię częściową z konfiguracji $\boldsymbol{q}_c$ do nowej $\boldsymbol{q}_c^\star$ za pomocą wyliczonych sterowań na przedziale $[0, T]$. Nową konfigurację przyjąć za aktualną $\boldsymbol{q}_c \leftarrow \boldsymbol{q}_c^\star$ i dokleić otrzymaną trajektorię do wynikowej na przedziale $[t^\star, t^\star + T]$. Przyjąć czas początkowy następnej części trajektorii wynikowej $t^\star \leftarrow t^\star + T$.

Krok 7. Powrócić do kroku 3.

2.2.4.4 Inne podejścia wykorzystujące Lie-algebrę

Metody Lie-algebraiczne posiadają także inne oblicza. W pierszym podejściu zamiast planować trajektorię jako konkatenację lokalnych trajektorii częściowych stosuje się sterowania okresowe o wysokich częstotliwościach w celu realizacji zadania śledzenia trajektorii referencyjnej z zadaną dokładnością. Ten sposób rozwiązania zadania śledzenia wykorzystujący rozwinięcie wyjścia układu w szereg Volterry zawiera referat [38] czerpiąc ideę aproksymacyjną z wcześniejszej pracy [53]. Druga grupa metod Lie-algebraicznych, zaproponowana przez Lafferriera i Sussmanna [48], odnosi się do planowania ruchu układów nilpotentnych. Tu także zadawana jest trajektoria śledzona, która dzięki nilpotentności układu może być odtwarzana globalnie. Metoda została szczegółowo zbadana w doktoracie [37].

2.2.5 Metoda wykorzystująca Zasadę Maksimum Pontriagina

Zasada Maksimum Pontriagina dostarcza ogólnej metody rozwiązań zadań sterowania optymalnego układów w postaci

$$\dot{\boldsymbol{q}} = \boldsymbol{f}(\boldsymbol{q}, \boldsymbol{u}), \tag{2.58}$$

gdzie $\boldsymbol{q} \in \mathbb{R}^n$ jest wektorem stanu, a $\boldsymbol{u} \in \mathbb{R}^m$ wektorem sterowań [25, 103]. Układ (2.1) jest jedną ze szczególnych postaci (2.58). Zadanie sterowania optymalnego polega na znalezieniu takiego wektora sterowań $\boldsymbol{u}$ który dla układu, stanów początkowego $\boldsymbol{q}_0$ i końcowego $\boldsymbol{q}_1$ zminimalizuje wskaźnik jakości $\mathcal{J}(\boldsymbol{u}(\cdot))$. W przypadku nieholonomicznego układu bez

dryfu (2.1) zadanie sterowania optymalnego polega na zmianie stanu układu z punktu początkowego $\boldsymbol{q}_0$ do punktu docelowego $\boldsymbol{q}_f$ w zadanym przedziale czasu $t \in [0, T]$. Rozwiązanie może być optymalne ze względu na dowolną funkcję jakości, jednak najczęściej przyjmuje się funkcję energetyczną

$$\boldsymbol{E}(\boldsymbol{u}(\cdot)) = \frac{1}{2}\int_0^T \langle \boldsymbol{u}(t), \boldsymbol{C}\boldsymbol{u}(t)\rangle dt, \tag{2.59}$$

gdzie $\boldsymbol{C}$ jest symetryczną i dodatnio określoną macierzą wagową. Dla powyższej funkcji jakości Hamiltonian układu (2.1) ma postać

$$\mathsf{H}(\boldsymbol{q}, \boldsymbol{p}, \boldsymbol{u}) = -\frac{1}{2}\langle \boldsymbol{u}, \boldsymbol{C}\boldsymbol{u}\rangle + \langle \boldsymbol{p}, \boldsymbol{G}(\boldsymbol{q})\boldsymbol{u}\rangle, \tag{2.60}$$

gdzie $\boldsymbol{p} \in \mathbb{R}^n$ oznacza zmienną dołączoną. Natomiast równania kanoniczne Hamiltona są następujące

$$\begin{cases} \dot{\boldsymbol{q}} &= \frac{\partial \mathsf{H}}{\partial \boldsymbol{\psi}}(\boldsymbol{q}(t), \boldsymbol{p}(t), p_0, \boldsymbol{u}(t)), = \boldsymbol{G}(\boldsymbol{q})\boldsymbol{u} \\ \dot{\boldsymbol{p}} &= -\frac{\partial \mathsf{H}}{\partial \boldsymbol{x}}(\boldsymbol{q}(t), \boldsymbol{p}(t), p_0, \boldsymbol{u}(t)) = -\left(\frac{\partial (\boldsymbol{G}(\boldsymbol{q})\boldsymbol{u})}{\partial \boldsymbol{q}}\right)^T \boldsymbol{p}. \end{cases} \tag{2.61}$$

Dla nieograniczonych sterowań ekstremum (2.59) względem sterowań uzyskuje się z warunku

$$\frac{\partial \mathsf{H}(\boldsymbol{q}, \boldsymbol{p}, \boldsymbol{u})}{\partial \boldsymbol{u}} = \boldsymbol{0}. \tag{2.62}$$

Z (2.62) można wyznaczyć sterowania

$$\boldsymbol{u} = \boldsymbol{C}^{-1}\boldsymbol{G}^T(\boldsymbol{q})\boldsymbol{p}, \tag{2.63}$$

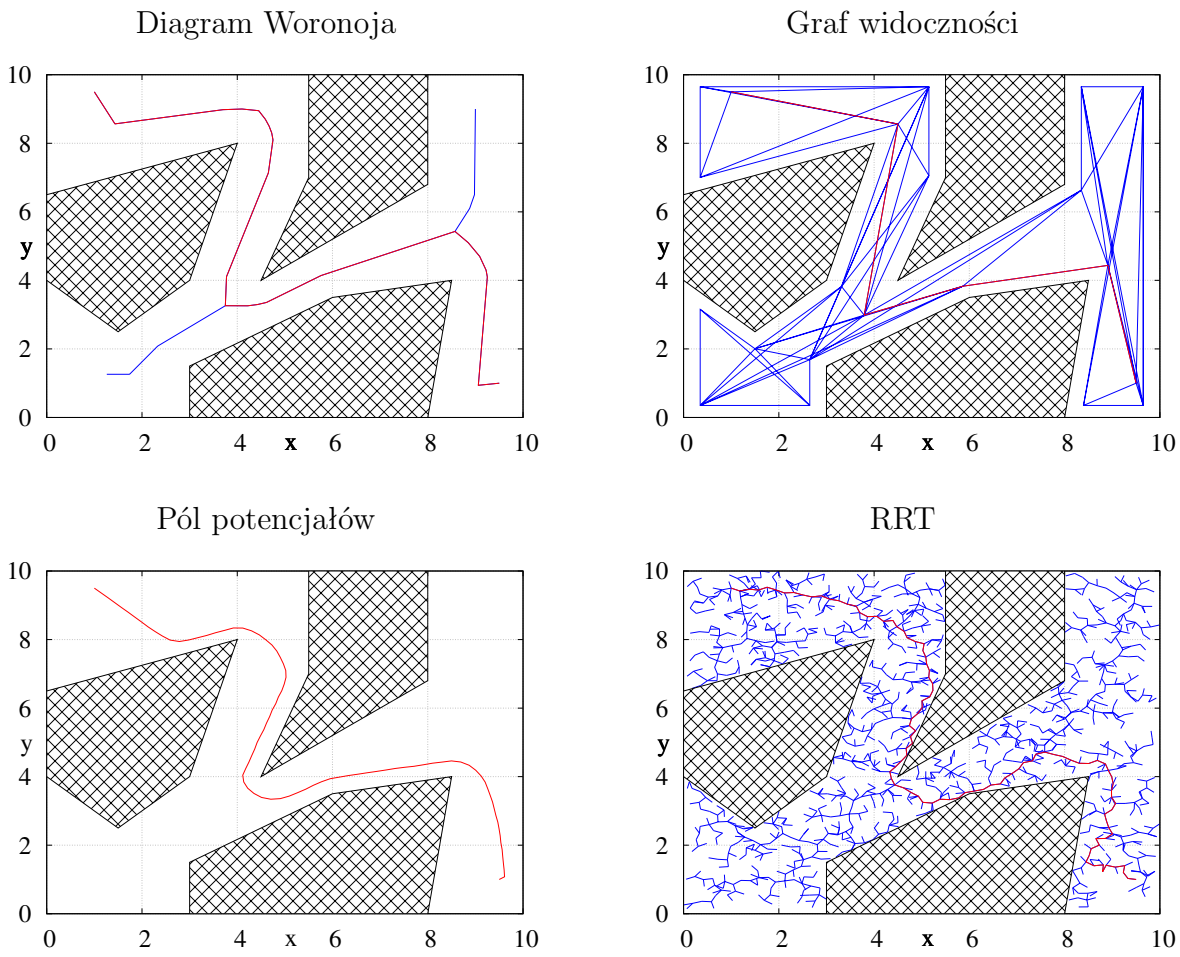
oraz podstawić je do równań kanonicznych Hamiltona

$$\begin{cases} \dot{\boldsymbol{q}} &= \boldsymbol{G}(\boldsymbol{q})\boldsymbol{C}^{-1}\boldsymbol{G}^T(\boldsymbol{q})\boldsymbol{p} \\ \dot{\boldsymbol{p}} &= -\left(\frac{\partial}{\partial \boldsymbol{q}}\boldsymbol{G}(\boldsymbol{q})\boldsymbol{C}^{-1}\boldsymbol{G}^T(\boldsymbol{q})\boldsymbol{p}\right)^T \boldsymbol{p}, \end{cases} \tag{2.64}$$

otrzymując (wraz z zadanymi T, $\boldsymbol{q}(0) = \boldsymbol{q}_0$ i $\boldsymbol{q}(T) = \boldsymbol{q}_f$) zadanie dwubrzegowe. Niestety, zadanie dwubrzegowe jest zadaniem trudnym, a metody umożliwiające rozwiązanie takiego zadania (np. metoda strzałów) są kosztowne obliczeniowo. Ponadto metoda bazująca na ZMP jest czuła na ograniczenia w przestrzeni konfiguracyjnej, zatem nie może być stosowana w środowiskach kolizyjnych.

2.3 Metody geometryczne

Geometryczne metody planowania ruchu znajdują szerokie zastosowanie w przypadku środowisk kolizyjnych, znajdując bezkolizyjną ścieżkę z punktu początkowego $\boldsymbol{x}_0$ do docelowego $\boldsymbol{x}_f$. W metodach tych jednak nie są brane pod uwagę ograniczenia nieholonomiczne (lub brane w uproszczonej formie). Teoretycznie każda z wymienionych poniżej metod może być zastosowana w dowolnie wymiarowej przestrzeni $\mathbb{X} = \mathbb{R}^n$. Najczęściej jednak, z powodu gwałtownie rosnącej złożoności obliczeniowej, wykorzystuje się je do planowania ruchu na płaszczyźnie $\mathbb{X} = \mathbb{R}^2$. Niektóre z wymienionych metod zostały użyte w roli pomocniczego planera geometrycznego w rozdziale 11. Przegląd algorytmów planowania geometrycznego można znaleźć w [27, 50]. Przykładowe wyniki działania wymienionych poniżej metod przedstawione są na rys. 2.4.



Rysunek 2.4 Ścieżki uzyskane z algorytmów planowania geometrycznego (czerwone) pomiędzy punktami $\boldsymbol{x}_0 = (1, 9.5)^T, \boldsymbol{x}_f = (9.5, 1)^T$. Kolorem niebieskim oznaczono grafy powstałe w wyniku działania planerów geometrycznych.

2.3.1 Metoda diagramu Woronoja (Voronoi diagram)

Metoda diagramu Woronoja opiera się na podziale przestrzeni $\mathbb{X}$ zawierającej zbiór przeszkód $\mathbb{O} = \{\boldsymbol{O}_1, \ldots, \boldsymbol{O}_k\}$ na k części nazywane obszarami Woronoja $\boldsymbol{Vor}(\boldsymbol{O}_i)$. Każdy z obszarów $\boldsymbol{Vor}(\boldsymbol{O}_i)$ zawiera punkty przestrzeni $\mathbb{X}$ będące najbliżej (według przyjętej metryki) przeszkody $\boldsymbol{O}_i$. Punkty znajdujące się na granicy obszarów Woronoja tworzą mapę drogową (*roadmap*), której każdy punkt znajduje się w maksymalnej odległości od przeszkód. Po dołączeniu do otrzymanej mapy drogowej punktów początkowego $\boldsymbol{x}_0$ i docelowego $\boldsymbol{x}_f$ wraz z niekolizyjną ścieżką do wymienionych punktów otrzymuje się mapę drogową umożliwiającą przemieszczenie pomiędzy punktami $\boldsymbol{x}_0$ a $\boldsymbol{x}_f$ bez kolizji z przeszkodami. Taką mapę drogową można potraktować jak graf, którego wierzchołkami są punkty $\boldsymbol{x}_0, \boldsymbol{x}_f$ oraz wszystkie punkty rozgałęzień mapy drogowej. Najkrótsza ścieżka w grafie może być znaleziona np. algorytmami Dijkstry lub $A^\star$. Metoda została szczegółowo opisana w pracach [4, 26, 57].

Praktyczne implementacje za przeszkody $\boldsymbol{O}_i$ przyjmują również dolne i górne ograniczenia współrzędnych przestrzeni $\mathbb{X}$. Ponadto, w celu zmniejszenia złożoności obliczeniowej, przeszkody często zastępowane są zbiorem punktów na krawędziach przeszkody,

a $\boldsymbol{Vor}(\boldsymbol{O}_i)$ jest sumą obszarów Woronoja $\boldsymbol{Vor}(\boldsymbol{o}_{i,j})$ dla każdego z tych punktów

$$\boldsymbol{Vor}(\boldsymbol{O}_i) = \bigcup_{\boldsymbol{o}_{i,j} \in \boldsymbol{O}_i} \boldsymbol{Vor}(\boldsymbol{o}_{i,j}) \tag{2.65}$$

$$\boldsymbol{Vor}(\boldsymbol{o}_{i,j}) = \{\boldsymbol{x} : \|\boldsymbol{x} - \boldsymbol{o}_{i,j}\| < \|\boldsymbol{x} - \boldsymbol{o}_{k,l}\|, \quad k \neq i, \quad l \neq j\}. \tag{2.66}$$

2.3.2 Metoda grafu widoczności

Metoda grafu widoczności składa się z dwóch kroków. Pierwszym krokiem jest stworzenie grafu widoczności, którego wierzchołkami są wierzchołki przeszkód (modelowanych jako wielokąty w $\mathbb{X} = \mathbb{R}^2$) oraz punkty początkowy $\boldsymbol{x}_0$ i docelowy $\boldsymbol{x}_f$. Krawędziami grafu widoczności są odcinki łączące wierzchołki grafu nie mające żadnej części wspólnej z którąkolwiek z przeszkód. Porównanie różnych algorytmów tworzenia grafu widzialności znajduje się w pracy [46]. Drugim krokiem algorytmu jest znalezienie najkrótszej ścieżki (np. algorytmem Dijkstry lub $A^\star$) w grafie łączącej wierzchołki odpowiadające punktom $\boldsymbol{x}_0$ i $\boldsymbol{x}_f$.

Metoda grafu widoczności pozwala na otrzymanie najkrótszej ścieżki między punktami $\boldsymbol{x}_0$ i $\boldsymbol{x}_f$. Ścieżka przebiega jednak często po krawędziach przeszkód, co nie jest dobrym rozwiązaniem w praktycznych zastosowaniach, gdy robot nie jest punktem materialnym. Z tego powodu na potrzeby tworzenia grafu często powiększa się przeszkody o rozmiar robota (promień okręgu opisanego na nim) i pewien dodatkowy margines bezpieczeństwa.

2.3.3 Metoda pól potencjałów

Metoda pól potencjałów polega na przypisaniu każdemu punktowi przestrzeni $\mathbb{X}$ wartości odpowiednio dobranej funkcji potencjału. Funkcja powinna mieć minimum globalne w punkcie docelowym $\boldsymbol{x}_f$ i najlepiej nie posiadać minimów lokalnych. Dodatkowo, w środowiskach kolizyjnych powinna być tak skonstruowana, aby jej wartości gwałtownie rosły w okolicach przeszkód. Po skonstruowaniu właściwej funkcji planowanie ruchu odbywa się lokalnie, poprzez poruszanie się w kierunku najszybszego spadku jej wartości w punkcie (antygradientu). Funkcja potencjału zwykle składa się z dwóch składników

$$U(\boldsymbol{x}) = U_{att}(\boldsymbol{x}, \boldsymbol{x}_f) + U_{rep}(\boldsymbol{x}, \mathbb{O}), \tag{2.67}$$

gdzie składnik $U_{att}(\boldsymbol{x}, \boldsymbol{x}_f)$ jest odpowiedzialny za zbliżanie się do punktu docelowego, a $U_{rep}(\boldsymbol{x}, \mathbb{O})$ odpowiada za unikanie przeszkód. Jedną z prostszych propozycji konstrukcji składników funkcji jest

$$U_{att}(\boldsymbol{x}, \boldsymbol{x}_f) = \|\boldsymbol{x} - \boldsymbol{x}_f\|^2, \quad U_{rep}(\boldsymbol{x}, \mathbb{O}) = \lambda \frac{1}{\|\boldsymbol{x} - \mathbb{O}\|^2}, \tag{2.68}$$

gdzie $\|\boldsymbol{x} - \mathbb{O}\|$ oznacza odległość od najbliższej przeszkody, a λ jest współczynnikiem skalującym wpływ obu składników na wynik funkcji $U(\boldsymbol{x})$.

W przypadku gdy układ znajdzie się w minimum lokalnym funkcji, należy go z tego minimum wyprowadzić, np. za pomocą błądzenia losowego. Metoda została szczegółowo opisana w pracach [12, 57, 62].

2.3.4 Algorytm RRT (Rapidly Exploring Random Tree)

Algorytm RRT jest algorytmem probabilistycznym, ktorego działanie polega na stopniowym rozroście drzewa o korzeniu w punkcie początkowym $\boldsymbol{x}_0$ [8, 104]. Rozrost drzewa odbywa się poprzez dodawanie do niego nowych, losowych punktów-wierzchołków, łącząc je krawędzią z najbliższym wierzchołkiem drzewa. Dodanie nowego punktu-wierzchołka, jest możliwe jedynie wtedy, gdy odcinek między tym wierzchołkiem a najbliższym wierzchołkiem drzewa jest bezkolizyjny. W większości wersji algorytm kończy działanie, gdy dołączono do drzewa punkt dostatecznie bliski docelowego $\boldsymbol{x}_f$. Kroki procedury rozrostu drzewa zebrano w Algorytmie 2.3

Algorytm 2.3 RRT

Dane wejściowe: Punkt początkowy $\boldsymbol{x}_0$, punkt docelowy $\boldsymbol{x}_f$, obszar wyłączony z planowania (przeszkody) $\mathbb{X}_{obs}$, obszar z jakiego można losować punkty $\mathbb{X}_{rand}$, maksymalne oddalenie nowego punktu od drzewa ΔL, maksymalne liczba iteracji it_{max}, akceptowalny błąd osiągnięcia punktu docelowego ε.

Krok 1. Utwórz drzewo $\mathcal{G}$ złożone z wierzchołka v_1 odpowiadającego punktowi $\boldsymbol{x}_0$. Ustaw licznik iteracji $it \leftarrow 0$

Krok 2. Sprawdź czy jakikolwiek wierzchołek drzewa v_i, odpowiadający punktowi $\boldsymbol{x}_i$ jest dostatecznie blisko punktu docelowego $\|\boldsymbol{x}_f - \boldsymbol{x}_i\| < \varepsilon$.
Jeśli tak, zakończ algorytm i zwróć ścieżkę łączącą $\boldsymbol{x}_0$ z $\boldsymbol{x}_i$.
Jeśli nie, kontynuuj krokiem 3.

Krok 3. Zwiększ licznik iteracji $it \leftarrow it + 1$. Jeśli $it = it_{max}$ zakończ algorytm i zwróć niepowodzenie.

Krok 4. Wylosuj nowy punkt $\boldsymbol{x}_{rand}$ z obszaru $\mathbb{X}_{rand}$.

Krok 5. Znajdź najbliższego sąsiada $\boldsymbol{x}_{near}$ punktu $\boldsymbol{x}_{rand}$ spośród punktów $\boldsymbol{x}_i$ odpowiadających wierzchołkom v_i drzewa.

Krok 6. Oblicz położenie punktu $\boldsymbol{x}_{new}$ jako

$$\boldsymbol{x}_{new} \leftarrow \boldsymbol{x}_{near} + \Delta L \frac{\boldsymbol{x}_{rand} - \boldsymbol{x}_{near}}{\|\boldsymbol{x}_{rand} - \boldsymbol{x}_{near}\|} \tag{2.69}$$

Krok 7. Jeśli odcinek $\overline{\boldsymbol{x}_{near}\boldsymbol{x}_{new}}$ nie przechodzi przez przeszkodę dodaj $\boldsymbol{x}_{new}$ jako kolejny wierzchołek, a parę $(\boldsymbol{x}_{near}, \boldsymbol{x}_{new})$ jako kolejną krawędź drzewa $\mathcal{G}$.
Następnie przejdź do kroku 2.

Wynikiem algorytmu RRT jest zwykle ścieżka daleka od optymalnej pod względem długości i składająca się z dużej liczby, często gwałtownych zakrętów. Z tego powodu ścieżka wynikowa podlega wygładzaniu polegającym na eliminowaniu kolejnych punktów odpowiadających wierzchołkom, o ile można poprowadzić odcinek nie przechodzący przez przeszkody pomiędzy punktami odpowiadającymi poprzedniemu i następnemu wierzchołkowi względem usuwanego.

Algorytm RRT występuje w bardzo wielu wariantach lekko się różniących. Spośród nich warto wymienić RRT* [42] w każdym kroku przebudowujący drzewo tak, aby odległość do wszystkich wierzchołków z korzenia była najkrótsza, RRT*-Smart [35, 71] wpływający dodatkowo na rozkład prawdopodobieństwa czy RRT^X [73] używany w środowiskach zmieniających się dynamicznie.

Rozdział 3

Generacja pre-sterowań i ich współczynników z formuły gCBHD

W rozdziale przedstawiono generację pre-sterowań według formuły gCBHD, która jest jednym z elementów Lie-algebraicznego zadania planowania ruchu. W podrozdziale 3.1 opisano zaobserwowaną zależność pomiędzy współczynnikami liczbowymi pre-sterowań, oraz ją udowodniono. Podrozdział 3.2 zawiera kombinatoryczny algorytm wyliczania współczynników pre-sterowań inspirowany wcześniej wspomnianą zależnością.

W paragrafie 2.2.4.2 opisano otrzymywanie sterowań ciągłych za pomocą formuły gCBHD dla jednego kroku algorytmu Lie-algebraicznej metody planowania ruchu. Wprowadzono również definicję wyrażenia α_d^r zależnego od sterowań oraz pre-sterowania $u_{i_1 i_2 \ldots i_r}$ będącego elementem α_d^r. Przypomnijmy wzór (2.53) opisujący przyrost $\boldsymbol{z}(T, \boldsymbol{q}_0, \boldsymbol{u}(\cdot)) = \xi(\boldsymbol{q}_f - \boldsymbol{q}_0)$ w zależności od sterowań $\boldsymbol{u}$ i konfiguracji początkowej $\boldsymbol{q}_0$ (strona 28). Wyrażenie $\alpha_d^r(\boldsymbol{u}(\cdot))$ ze wzoru (2.53) jest zależne od stowarzyszonego z nią elementu bazy Ph. Halla i może być opisane ogólnym wzorem jako

$$\alpha_d^r = \int_{T_r(t)} \sum_{p=1}^{\#PS_d^r} (w_p^{r,d} \cdot pre_p^{r,d}) d\boldsymbol{s}^r, \tag{3.1}$$

gdzie $pre_p^{r,d}$ oznacza pre-sterowanie w formie $u_{i_1 i_2 \ldots i_r}$, a $w_p^{r,d}$ jest współczynnikiem liczbowym związany z tym pre-sterowaniem. Górny indeks $\#PS_d^r$ sumatora we wzorze (3.1) oznacza liczbę pre-sterowań związanych z danym wyrażeniem α_d^r.

W tab. 3.1 zebrano kombinacje liniowe pre-sterowań dla elementów baz Ph. Halla i Chibrikova do warstwy piątej włącznie. Puste pola w tabeli wynikają z różnicy elementów między bazami.

3.1 Zależność między współczynnikami pre-sterowań

Analizując tab. 3.1 można zauważyć pewne zależności występujące między współczynnikami liczbowymi pre-sterowań związanych z tym samym elementem bazy, a nawet tą samą warstwą. Obserwacja prowadzi do ogólniejszego twierdzenia, prawdziwego dla dowolnej liczby sterowań r.

Twierdzenie 1 *Dla m-wejściowych układów bezdryfowych* (2.41)*, współczynniki $w_p^{r,d}$ w definicji współczynnika α_d^r (por.* (3.1)*) występujące we wzorze na przyrost konfiguracji* (2.53)

Tabela 3.1 Suma pre-sterowań stowarzyszonych z elementami baz Ph. Halla i Chibrikova do warstwy piątej włącznie, dla dwóch generatorów $\boldsymbol{X},\boldsymbol{Y}$. [37]

$\sum_p^{\#PS_d^r}(w_p^{r,d}\cdot pre_p^{r,d})$	baza	
	Ph. Halla	Chibrikova
u_1	$\boldsymbol{X}$	$\boldsymbol{X}$
u_2	$\boldsymbol{Y}$	$\boldsymbol{Y}$
$\frac{1}{2}(u_{12}-u_{21})$	$[\boldsymbol{X},\boldsymbol{Y}]$	$[\boldsymbol{X},\boldsymbol{Y}]$
$\frac{1}{3!}(u_{112}-2u_{121}+u_{211})$	$[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]$	$[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]$
$\frac{1}{3!}(-u_{122}+2u_{212}-u_{221})$	$[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]$	$[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]$
$\frac{1}{4!}(-4u_{1121}+4u_{1211})$	$[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]$	$[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]$
$\frac{1}{4!}(-4u_{1122}+4u_{1212}-4u_{2121}+4u_{2211})$	$[\boldsymbol{Y},[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]$	$[\boldsymbol{X},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]$
$\frac{1}{4!}(-4u_{2122}+4u_{2212})$	$[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]$	$[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]$
$\frac{1}{5!}(-4u_{11112}-4u_{11121}+16u_{11211}-4u_{12111}+$ $-4u_{21111})$	$[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]]$	$[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]]$
$\frac{1}{5!}(-8u_{11122}+12u_{11212}+12u_{11221}-8u_{12112}+$ $-8u_{12121}+12u_{12211}-8u_{21112}-8u_{21121}+$ $+12u_{21211}-8u_{22111})$	$[\boldsymbol{Y},[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]]$	
$\frac{1}{5!}(8u_{11222}-12u_{12122}+8u_{12212}+8u_{12221}+$ $-12u_{21122}+8u_{21212}+8u_{21221}-12u_{22112}+$ $-12u_{22121}+8u_{22211})$	$[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]]$	
$\frac{1}{5!}(4u_{12222}+4u_{21222}-16u_{22122}+4u_{22212}+$ $+4u_{22221})$	$[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]]$	$[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]]$
$\frac{1}{5!}(-12u_{11122}+8u_{11212}+8u_{11221}+8u_{12112}+$ $-12u_{12121}+8u_{12211}-12u_{21112}+8u_{21121}+$ $+8u_{21211}-12u_{22111})$	$[[\boldsymbol{X},\boldsymbol{Y}],[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]$	$[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]]$
$\frac{1}{5!}(-4u_{11222}-4u_{12122}+16u_{12212}-4u_{12221}+$ $-4u_{21122}-4u_{21212}+16u_{21221}-4u_{22112}+$ $-4u_{22121}-4u_{22211})$	$[[\boldsymbol{X},\boldsymbol{Y}],[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]$	$[\boldsymbol{X},[\boldsymbol{Y},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]]$
$\frac{1}{5!}(4u_{11122}+4u_{11212}+4u_{11221}-16u_{12112}+$ $+4u_{12121}+4u_{12211}+4u_{21112}-16u_{21121}+$ $+4u_{21211}+4u_{22111})$		$[\boldsymbol{Y},[\boldsymbol{X},[\boldsymbol{X},[\boldsymbol{X},\boldsymbol{Y}]]]]$
$\frac{1}{5!}(12u_{11222}-8u_{12122}-8u_{12212}+12u_{12221}+$ $-8u_{21122}+12u_{21212}-8u_{21221}-8u_{22112}+$ $-8u_{22121}+12u_{22211})$		$[\boldsymbol{Y},[\boldsymbol{X},[\boldsymbol{Y},[\boldsymbol{X},\boldsymbol{Y}]]]]$

spełniają warunek

$$\forall_{r\geqslant 2}\,\forall_{d\in\{1,\ldots,\#\boldsymbol{H}^r\}} \quad \sum_{p=1}^{\#PS_d^r} w_p^{r,d} = 0. \tag{3.2}$$

Innymi słowy, suma wszystkich współczynników $w_p^{r,d}$ występujących w α_d^r wynosi 0 dla każdej warstwy bazy Ph. Halla poza warstwą generatorów, dla których suma wynosi 1, a $\alpha_d^1 = \int_0^t u_d ds$.

Dowód 1 *Porównując zapisy formuły gCBHD w postaciach* (2.47) *i* (2.53) *oraz uwzględniając* (3.1)*, w każdej warstwie r, przetwarzane są elementy postaci*

$$\sum_{\boldsymbol{\sigma}\in P_r} c(\boldsymbol{\sigma})\boldsymbol{E}_{\boldsymbol{\sigma}} = \sum_{d=1}^{\#\boldsymbol{H}^r} \alpha_d^r \cdot \boldsymbol{H}_d^r = \sum_{d=1}^{\#\boldsymbol{H}^r} \Big(\sum_{p=1}^{\#PS_d^r} w_p^{r,d} \cdot pre_p^{r,d} \Big) \cdot \boldsymbol{H}_d^r. \tag{3.3}$$

Podstawiając do definicji elementu $\boldsymbol{E}_{\boldsymbol{\sigma}}$ (2.48) *wzór* (2.41) *opisujący układ bezdryfowy jako kombinację liniową iloczynów par generator - sterowanie otrzymujemy zależność $\boldsymbol{E}_{\boldsymbol{\sigma}}$ od generatorów $\boldsymbol{X}_i$ i sterowań u_i*

$$\boldsymbol{E}_{\boldsymbol{\sigma}} = [[\ldots[\sum_{i_1=1}^{m} \boldsymbol{X}_{i_1} u_{i_1}(s_{\sigma(1)}), \sum_{i_2=1}^{m} \boldsymbol{X}_{i_2} u_{i_2}(s_{\sigma(2)})]\ldots, \sum_{i_{r-1}=1}^{m} \boldsymbol{X}_{i_{r-1}} u_{i_{r-1}}(s_{\sigma(r-1)})], \sum_{i_r=1}^{m} \boldsymbol{X}_{i_r} u_{i_r}(s_{\sigma(r)})]. \tag{3.4}$$

Wykorzystując własności antysymetrii (1.3) *oraz tożsamość Jacobiego* (1.4) *równanie* (3.4) *przyjmuje postać*

$$\boldsymbol{E}_{\boldsymbol{\sigma}} = \sum_{i_1=1}^{m}\sum_{i_2=1}^{m}\cdots\sum_{i_r=1}^{m} [[\ldots[\boldsymbol{X}_{i_1}, \boldsymbol{X}_{i_2}]\ldots, \boldsymbol{X}_{i_{r-1}}], \boldsymbol{X}_{i_r}] \cdot u_{i_1}(s_{\sigma(1)}) \cdot \ldots \cdot u_{i_r}(s_{\sigma(r)}) \tag{3.5}$$

Dalsza część dowodu zostanie przeprowadzona przez indukcję.

Dla $r = 2$ teza twierdzenia 1 jest prawdziwa co ilustruje następujące przekształcenie wykorzystujące operację antysymetrii

$$\sum_{i_1=1}^{m}\sum_{i_2=1}^{m} [\boldsymbol{X}_{i_1}, \boldsymbol{X}_{i_2}] u_{i_1}(s_{\sigma(1)}) u_{i_2}(s_{\sigma(2)}) = \overbrace{\sum_{i_1=1}^{m}\sum_{i_2=i_1+1}^{m} [\boldsymbol{X}_{i_1}, \boldsymbol{X}_{i_2}] u_{i_1}(s_{\sigma(1)}) u_{i_2}(s_{\sigma(2)})}^{i_1<i_2} +$$
$$+ \overbrace{\sum_{i_1=1}^{m}\sum_{i_2=i_1+1}^{m} [\boldsymbol{X}_{i_2}, \boldsymbol{X}_{i_1}] u_{i_2}(s_{\sigma(1)}) u_{i_1}(s_{\sigma(2)})}^{i_1>i_2,\ zamiana\ indeksów} + \overbrace{\sum_{i_1=1}^{m} [\boldsymbol{X}_{i_1}, \boldsymbol{X}_{i_1}] u_{i_1}(s_{\sigma(1)}) u_{i_1}(s_{\sigma(2)})}^{i_1=i_2} =$$
$$= \overbrace{\sum_{i_1=1}^{m}\sum_{i_2=i_1+1}^{m} [\boldsymbol{X}_{i_1}, \boldsymbol{X}_{i_2}] u_{i_1}(s_{\sigma(1)}) u_{i_2}(s_{\sigma(2)})}^{i_1<i_2} + \overbrace{\sum_{i_1=1}^{m}\sum_{i_2=i_1+1}^{m} -[\boldsymbol{X}_{i_1}, \boldsymbol{X}_{i_2}] u_{i_2}(s_{\sigma(1)}) u_{i_1}(s_{\sigma(2)})}^{i_1>i_2,\ antysymatria} +$$
$$+ \overbrace{\boldsymbol{0}}^{i_1=i_2} = \sum_{i_1=1}^{m}\sum_{i_2=i_1+1}^{m} [\boldsymbol{X}_{i_1}, \boldsymbol{X}_{i_2}] \cdot (u_{i_1}(s_{\sigma(1)}) u_{i_2}(s_{\sigma(2)}) - u_{i_1}(s_{\sigma(2)}) u_{i_2}(s_{\sigma(1)})).$$

$\forall(i_1 < i_2)$ $[\boldsymbol{X}_{i_1}, \boldsymbol{X}_{i_2}]$ należy do bazy Ph. Halla, a suma współczynników pre-sterowań pod wewnętrzną sumą jest równa zero, $1 - 1 = 0$.

Dla $r \geqslant 3$, zdefiniujmy pre-sterowanie należące do r-tej warstwy i zależne od wektora indeksów sterowań $\boldsymbol{i}$ oraz permutacji $\boldsymbol{\sigma}$ zawierającej indeksy argumentów całkowania po

sympleksie $\boldsymbol{s_i}$ *jako*

$$pre_r(\boldsymbol{\sigma},\boldsymbol{i}) = u_{i_1}(s_{\boldsymbol{\sigma}(1)})u_{i_2}(s_{\boldsymbol{\sigma}(2)}) \cdot \ldots \cdot u_{i_r}(s_{\boldsymbol{\sigma}(r)}) \tag{3.6}$$

i przeanalizujmy pojedynczy element $\boldsymbol{E_\sigma}$ *ze wzoru* (3.5) *określony jako*

$$\boldsymbol{N}(\boldsymbol{\sigma},\boldsymbol{i},r) \stackrel{def}{=} pre_r(\boldsymbol{\sigma},\boldsymbol{i}) \cdot [[\ldots[\boldsymbol{X}_{i_1},\boldsymbol{X}_{i_2}]\ldots,\boldsymbol{X}_{i_{r-1}}],\boldsymbol{X}_{i_r}]. \tag{3.7}$$

Element $\boldsymbol{N}(\boldsymbol{\sigma},\boldsymbol{i},r)$ *jest zależny od ustalonej permutacji* $\boldsymbol{\sigma} = (\sigma_1,\sigma_2,\ldots,\sigma_r)$ *oraz ustalonego wektora* $\boldsymbol{i} = (i_1,i_2,\ldots,i_r)$ *zbudowanego z elementów odpowiadających indeksom generatorów* $i_j \in \{1,\ldots,m\}$. *Obydwa wektory* $\boldsymbol{\sigma}$, $\boldsymbol{i}$ *mają wymiarowość numeru warstwy równą* r. *Po przekształceniu jednomianów Liego występujących w zależności* (3.7) *do bazy Ph. Halla, za pomocą antysymetrii oraz tożsamości Jacobiego, można jej nadać postać*

$$\boldsymbol{N}(\boldsymbol{\sigma},\boldsymbol{i},r) = pre_r(\boldsymbol{\sigma},\boldsymbol{i}) \cdot [[\ldots[\boldsymbol{X}_{i_1},\boldsymbol{X}_{i_2}]\ldots,\boldsymbol{X}_{i_{r-1}}],\boldsymbol{X}_{i_r}] = pre_r(\boldsymbol{\sigma},\boldsymbol{i}) \sum_{d=1}^{\#\boldsymbol{H}^r} \beta_d \boldsymbol{H}_d^r. \tag{3.8}$$

Jeśli wektor $\boldsymbol{i}$ *ma właściwość* $i_1 = i_2$, *to* $[\boldsymbol{X}_{i_1},\boldsymbol{X}_{i_2}] = 0$ *i całe wyrażenie* (3.8) *jest równe* 0. *W przeciwnym przypadku* ($i_1 \neq i_2$) *istnieje wektor indeksów* $\bar{\boldsymbol{i}} = (i_2,i_1,i_3,\ldots,i_r)$ *komplementarny do* $\boldsymbol{i}$. *Dla permutacji* $\boldsymbol{\sigma}$ *i wektora* $\bar{\boldsymbol{i}}$ *istnieje element* $\boldsymbol{N}(\boldsymbol{\sigma},\bar{\boldsymbol{i}},r)$ *komplementarny do* (3.8) *określony następująco*

$$\begin{aligned}\boldsymbol{N}(\boldsymbol{\sigma},\bar{\boldsymbol{i}},r) &= [[\ldots[\boldsymbol{X}_{i_2},\boldsymbol{X}_{i_1}]\ldots,\boldsymbol{X}_{i_{r-1}}],\boldsymbol{X}_{i_r}] \cdot u_{i_1}(s_{\sigma(2)})u_{i_2}(s_{\sigma(1)}) \cdot \ldots \cdot u_{i_r}(s_{\sigma(r)}) = \\ &= -[[\ldots[\boldsymbol{X}_{i_1},\boldsymbol{X}_{i_2}]\ldots,\boldsymbol{X}_{i_{r-1}}],\boldsymbol{X}_{i_r}] \cdot pre_r(\boldsymbol{\sigma},\bar{\boldsymbol{i}}) = -pre_r(\boldsymbol{\sigma},\bar{\boldsymbol{i}}) \cdot \sum_{d=1}^{\#\boldsymbol{H}^r} \beta_d \boldsymbol{H}_d^r.\end{aligned} \tag{3.9}$$

Minus pojawiający się w (3.9) *wynika z zastosowania antysymetrii na nawiasie Liego* $[\boldsymbol{X}_{i_2},\boldsymbol{X}_{i_1}]$. *Po podstawieniu równań* (3.8), (3.9) *do równania* (3.5) *otrzymujemy*

$$\begin{aligned}&\sum_{i_3,\ldots,i_r=1}^{m}\sum_{i_1=1}^{m}\sum_{i_2=1}^{m}\boldsymbol{N}(\boldsymbol{\sigma},\boldsymbol{i},r) = \sum_{i_3,\ldots,i_r=1}^{m}\left(\sum_{i_1=1}^{m}\sum_{i_2>i_1}^{m}\boldsymbol{N}(\boldsymbol{\sigma},\boldsymbol{i},r) + \sum_{i_1=i_2=1}^{m}\boldsymbol{N}(\boldsymbol{\sigma},\boldsymbol{i},r) + \right. \\ &\left. + \sum_{i_1=1}^{m}\sum_{i_2<i_1}^{m}\boldsymbol{N}(\boldsymbol{\sigma},\boldsymbol{i},r)\right) = \sum_{i_1,i_3,\ldots,i_r=1}^{m}\sum_{i_2>i_1}^{m}\Big(\boldsymbol{N}(\boldsymbol{\sigma},\boldsymbol{i},r) + \boldsymbol{N}(\boldsymbol{\sigma},\bar{\boldsymbol{i}},r)\Big) = \\ &= \sum_{\boldsymbol{i}}\Big(pre_r(\boldsymbol{\sigma},\boldsymbol{i}) - pre_r(\boldsymbol{\sigma},\bar{\boldsymbol{i}})\Big) \cdot \sum_{d=1}^{\#\boldsymbol{H}^r}\beta_d\boldsymbol{H}_d^r = \sum_{\boldsymbol{i}}\sum_{d=1}^{\#\boldsymbol{H}^r}\Big(\beta_d \cdot pre_r(\boldsymbol{\sigma},\boldsymbol{i}) - \beta_d \cdot pre_r(\boldsymbol{\sigma},\bar{\boldsymbol{i}})\Big)\boldsymbol{H}_d^r,\end{aligned} \tag{3.10}$$

gdzie $\boldsymbol{i} = (i_1, i_2 > i_1, i_3, \ldots, i_r)$ *oraz* $i_k \in \{1,\ldots,m\}$. *Z równania* (3.10) *wynika, że współczynniki związane z pre-sterowaniami w pojedynczym elemencie* $\boldsymbol{E_\sigma}$ *sumują się do zera* ($\beta_d - \beta_d = 0$). *Ta właściwość przenosi się na całą warstwę* $\sum_{\boldsymbol{\sigma}\in P_r} c(\boldsymbol{\sigma})\boldsymbol{E_\sigma}$ *przez liniowość, udowadniając tezę twierdzenia 1.* □

Twierdzenie 1, jakkolwiek niekonstruktywne, jest użyteczne podczas zgrubnego sprawdzenia poprawności skomplikowanych wyprowadzeń, gdzie o pomyłkę (tę ręczną jak i w programie komputerowym) nietrudno.

3.2 Kombinatoryczny algorytm wyliczania współczynników pre-sterowań

Podstawowy, naiwny algorytm wyliczający pre-sterowania z formuły gCBHD przetwarza każdy element gCBHD (dany równaniem (2.48)) z osobna. Przykładowo: rozwija wszystkie elementy $\boldsymbol{A}(s_{\boldsymbol{\sigma}(i)})$ równania (2.45), sprowadza je do bazy Ph. Halla a następnie sumuje wyliczone pre-sterowania. Algorytm ma dużą złożoność obliczeniową, gdyż często wyznacza uprzednio już obliczone elementy podczas przeliczania kolejnej permutacji. Aby zredukować złożoność zadania przeliczania formuły gCBHD do formy (2.53) w pracy [20] zaproponowany został inny algorytm, którego ideę przedstawiono poniżej.

- Algorytm przetwarza formułę gCBHD warstwa po warstwie.
- W określonej r-tej warstwie elementy $\boldsymbol{A}(s_j) = \boldsymbol{Z}_j$ są tymczasowo traktowane jako niezależny generator $\boldsymbol{Z}_j$ na potrzeby obliczenia bazy Ph. Halla rozpiętej przez $\boldsymbol{Z}_j$.
- Następnie każdy nawias jest rozwijany do zredukowanej bazy Ph. Halla (jest to podzbiór bazy Ph. Halla w skład którego wchodzą elementy w których każdy generator występuje tylko raz). Dzięki temu liczba elementów przetwarzanych zmniejsza się z $r!$ (dla naiwnego algorytmu) do liczności zredukowanej bazy Ph. Halla.
- Ostatnim krokiem jest podstawienie za tymczasowy generator $\boldsymbol{Z}_j$ sumy $\sum_{i=1}^{m} \boldsymbol{X}_i u_i(s_j)$ w otrzymanej zredukowanej bazie Ph. Halla i raz jeszcze sprowadzenie wyniku do bazy, tym razem przyjmującej za generatory oryginalne $\boldsymbol{X}_i$.

Odautorski algorytm zaproponowany w tym podrozdziale wykorzystuje nieco inne podejście od opisanych wyżej. Punktem wyjścia dla opracowania algorytmu była następująca obserwacja: elementy $\boldsymbol{E}_{\boldsymbol{\sigma}}$ w danej warstwie (patrz wzór (2.48)) różnią się między sobą jedynie indeksami generatorów określonymi przez permutacje $\boldsymbol{\sigma}$, natomiast postać elementu po sprowadzeniu do bazy algebry Liego pozostaje taka sama. Znając więc postać elementu $\boldsymbol{E}_{\boldsymbol{\sigma}}$ w danej bazie dla ustalonej permutacji $\boldsymbol{\sigma}$ (np. identycznościowej $\boldsymbol{\sigma} = \boldsymbol{id} = (1, \ldots, r)$), można uzyskać je dla innych elementów $\boldsymbol{E}_{\boldsymbol{\sigma}}$ danej warstwy manipulując jedynie indeksami. Kroki pozwalające na wyliczenie pre-sterowań r-tej warstwy formuły gCBHD dla układów bezdryfowych zebrano w Algorytmie 3.1.

Komentarze do Algorytmu 3.1:

- Przykład odwrotnej permutacji: odwrotną permutacją do $(2, 3, 1, 4)$ jest $(3, 1, 2, 4)$. Oczywiście, złożenie permutacji z jej odwrotnością daje permutację identycznościową

$$(2, 3, 1, 4) \circ (3, 1, 2, 4) = (1, 2, 3, 4).$$

- Przykład użycia permutacji $\boldsymbol{\sigma}$ (i odwrotnej permutacji $\mathrm{inv}(\boldsymbol{\sigma})$) na indeksach $pre^{r,d}_{p(\boldsymbol{id})}$ dla uzyskania $pre^{r,d}_{p(\boldsymbol{\sigma})}$:
  a) permutacja $\boldsymbol{\sigma} = (3, 1, 2)$ przekształca sekwencję (s_1, s_2, s_3) następująco

$$u_{i_1}(s_1)u_{i_2}(s_2)u_{i_3}(s_3) \xrightarrow{(3,1,2)} u_{i_1}(s_3)u_{i_2}(s_1)u_{i_3}(s_2) = u_{i_2}(s_1)u_{i_3}(s_2)u_{i_1}(s_3).$$

  b) odwrotna permutacja $\mathrm{inv}(\boldsymbol{\sigma}) = (2, 3, 1)$ przekształca indeksy (i_1, i_2, i_3) (krótka forma $u_{i_1 i_2 \ldots i_r}$) w następujący sposób

$$u_{i_1 i_2 i_3} \xrightarrow{(2,3,1)} u_{i_2 i_3 i_1} = u_{i_2}(s_1)u_{i_3}(s_2)u_{i_1}(s_3).$$

Algorytm 3.1 Wyznaczenie pre-sterowań r-tej warstwy bazy Ph. Halla.

Dane wejściowe: Liczba sterowań m, numer warstwy r.

Krok 1. Jako $\boldsymbol{\sigma}$ wybrać permutację identycznościową $\boldsymbol{\sigma} = \boldsymbol{id} = (1, 2, \ldots, r)$.

Krok 2. Korzystajac z liniowości nawiasu Liego przedstawić $\boldsymbol{E_{id}}$ jako

$$\begin{aligned}\boldsymbol{E_{id}} &= [[\ldots[\sum_{i_1=1}^{m} \boldsymbol{X}_{i_1} u_{i_1}(s_1), \sum_{i_2=1}^{m} \boldsymbol{X}_{i_2} u_{i_2}(s_2)] \ldots \sum_{i_{r-1}=1}^{m} \boldsymbol{X}_{i_{r-1}} u_{i_{r-1}}(s_{r-1})], \sum_{i_r=1}^{m} \boldsymbol{X}_{i_r} u_{i_r}(s_r)] \\ &= \sum_{i_1=1}^{m}\sum_{i_2=1}^{m}\ldots\sum_{i_r=1}^{m}[[\ldots[\boldsymbol{X}_{i_1}, \boldsymbol{X}_{i_2}]\ldots, \boldsymbol{X}_{i_{r-1}}], \boldsymbol{X}_{i_r}] \cdot \cdot u_{i_1}(s_1) \cdot \ldots \cdot u_{i_r}(s_r).\end{aligned} \tag{3.11}$$

Krok 3. Przedstawić $\boldsymbol{E_{id}}$ jako kombinację liniową elementów bazy Ph. Halla, korzystając z właściwości antysymetrii i tożsamości Jacobiego

$$\boldsymbol{E_{id}} = \sum_{d=1}^{\#\boldsymbol{H}^r} (\sum_{p(\boldsymbol{id})=1}^{\#PS_d^r} w_{p(\boldsymbol{id})}^{r,d} pre_{p(\boldsymbol{id})}^{r,d}) \boldsymbol{H}_d^r. \tag{3.12}$$

Warto zwrócić uwagę, że przekształcenie każdego jednomianu Liego pociąga również odpowiednie przekształcenie pre-sterowań związanych z tym jednomianem.

Krok 4. Dla każdej permutacji $\boldsymbol{\sigma} \in P_r$, wyliczyć $\boldsymbol{E_\sigma}$

$$\boldsymbol{E_\sigma} = \sum_{d=1}^{\#\boldsymbol{H}^r} (\sum_{p(\boldsymbol{\sigma})=1}^{\#PS_d^r} w_{p(\boldsymbol{\sigma})}^{r,d} pre_{p(\boldsymbol{\sigma})}^{r,d}) \boldsymbol{H}_d^r, \tag{3.13}$$

gdzie indeksy argumentu s w pre-sterowaniu $pre_{p(\boldsymbol{\sigma})}^{r,d}$ powstają poprzez nałożenie permutacji $\boldsymbol{\sigma}$ na indeksy $pre_{p(\boldsymbol{id})}^{r,d}$ (porównaj wzór (3.6)). Warto wspomnieć, że przekształcenie indeksów s permtuacją $\boldsymbol{\sigma}$ oznacza przekształcenie indeksów w krótkiej formie zapisu $u_{i_1 i_2 \ldots i_r}$ (wzór (2.52)) permutacją odwrotną do $\boldsymbol{\sigma}$.

Krok 5. Zebrać wszystkie pre-sterowania $\cup_{\boldsymbol{\sigma}\in P_r}\ pre_{p(\boldsymbol{\sigma})}^{r,d}$ i uporządkować je w sekwencję (indeksowaną za pomocą p)

Krok 6. Dla wszystkich permutacji $\boldsymbol{\sigma}$ wyliczyć współczynnik $c(\boldsymbol{\sigma})$ według wzoru (2.49) i włączyć je do wyrażenia (3.3), w którym

$$w_p^{r,d} = \sum_{\boldsymbol{\sigma}\in P_r} w_{p(\boldsymbol{\sigma})}^{r,d} \quad \text{oraz} \quad pre_p^{r,d} = pre_{p(\boldsymbol{\sigma})}^{r,d}. \tag{3.14}$$

- Współczynniki w równaniach (3.12) i (3.13) są takie same $w_{p(\boldsymbol{id})}^{r,d} = w_{p(\boldsymbol{\sigma})}^{r,d}$.
- Algorytm przekształcający jednomiany Liego do postaci kombinacji liniowej elementów bazy Ph. Halla zaczerpnięto z pracy [20].
- Algorytm został przedstawiony dla bazy Ph. Halla, jednak może być stosowany także dla dowolnej bazy Lie-algebry.

Poniżej zilustrowano działanie algorytmu na prostym, ale nietrywialnym, przykładzie.

Przykład 5. Wyliczanie pre-sterowań oraz ich współczynników dla trzeciej warstwy $r = 3$ dwuwejściowego układu $m = 2$ za pomocą algorytmu kombinatorycznego.

Dla identycznościowej permutacji $\boldsymbol{id} = (1, 2, 3)$ element $\boldsymbol{E_{id}}$ jest postaci

$$E_{\boldsymbol{id}} = [[\boldsymbol{A}(s_1), \boldsymbol{A}(s_2)], \boldsymbol{A}(s_3)], \quad \text{gdzie} \quad \boldsymbol{A}(s_j) = \sum_{i=1}^{2} \boldsymbol{X}_i u_i(s_j).$$

Następnie wszystkie jednomiany Liego są przedstawiane jako kombinacje liniowe elementów bazy Ph. Halla za pomocą antysymetrii i tożsamości Jacobiego

$$\begin{aligned} E_{\boldsymbol{id}} &= [[\boldsymbol{X}_1 u_1(s_1) + \boldsymbol{X}_2 u_2(s_1), \boldsymbol{X}_1 u_1(s_2) + \boldsymbol{X}_2 u_2(s_2)], \boldsymbol{X}_1 u_1(s_3) + \boldsymbol{X}_2 u_2(s_3)] = \\ &= [[\boldsymbol{X}_1, \boldsymbol{X}_2](u_1(s_1)u_2(s_2) - u_2(s_1)u_1(s_2)), \boldsymbol{X}_1 u_1(s_3) + \boldsymbol{X}_2 u_2(s_3)] = \\ &= [[\boldsymbol{X}_1, \boldsymbol{X}_2], \boldsymbol{X}_1](u_{121} - u_{211}) + [[\boldsymbol{X}_1, \boldsymbol{X}_2], \boldsymbol{X}_2](u_{122} - u_{212}) \\ &= [\boldsymbol{X}_1, [\boldsymbol{X}_1, \boldsymbol{X}_2]](u_{211} - u_{121}) + [\boldsymbol{X}_2, [\boldsymbol{X}_1, \boldsymbol{X}_2]](u_{212} - u_{122}). \end{aligned} \tag{3.15}$$

Pre-sterowania oraz związane z nimi współczynniki są następujące

$$\begin{aligned} w^{31}_{1(\boldsymbol{id})} = 1,\ pre^{31}_{1(\boldsymbol{id})} = u_{211},\ w^{31}_{2(\boldsymbol{id})} = -1,\ pre^{31}_{2(\boldsymbol{id})} = u_{121}, \\ w^{32}_{1(\boldsymbol{id})} = 1,\ pre^{32}_{1(\boldsymbol{id})} = u_{212},\ w^{32}_{2(\boldsymbol{id})} = -1,\ pre^{32}_{2(\boldsymbol{id})} = u_{122}. \end{aligned}$$

Dla pozostałych permutacji pre-sterowania z równania (3.15) są permutowane. Dla przykładowej permutacji $\boldsymbol{\sigma} = (1, 3, 2)$ ($\mathrm{inv}(\boldsymbol{\sigma}) = (1, 3, 2)$) element $\boldsymbol{E_\sigma}$ określany jest wzorem

$$\boldsymbol{E_\sigma} = [[\boldsymbol{X}_1 u_1(s_1) + \boldsymbol{X}_2 u_2(s_1), \boldsymbol{X}_1 u_1(s_3) + \boldsymbol{X}_2 u_2(s_3)], \boldsymbol{X}_1 u_1(s_2) + \boldsymbol{X}_2 u_2(s_2)]. \tag{3.16}$$

Równanie (3.16) nie jest rozwijane i sprowadzane do bazy Ph. Halla, lecz wykorzystywana jest permutacja indeksów pre-sterowań

$$\begin{aligned} &[\boldsymbol{X}_1, [\boldsymbol{X}_1, \boldsymbol{X}_2]](u_{211} - u_{121}) + [\boldsymbol{X}_2, [\boldsymbol{X}_1, \boldsymbol{X}_2]](u_{212} - u_{122}) \xrightarrow{(1,3,2)} \\ &\xrightarrow{(1,3,2)} [\boldsymbol{X}_1, [\boldsymbol{X}_1, \boldsymbol{X}_2]](u_{211} - u_{112}) + [\boldsymbol{X}_2, [\boldsymbol{X}_1, \boldsymbol{X}_2]](u_{221} - u_{122}). \end{aligned}$$

Dla każdej permutacji $\boldsymbol{\sigma}$ wyliczany jest współczynnik $c(\boldsymbol{\sigma})$ (wzór (2.49)), a następnie wyniki częściowe są sumowane (wzór (3.3)). Poniżej przedstawiono wyniki tych operacji dla pre-sterowań stowarzyszonych z jednomianem $[\boldsymbol{X}_1, [\boldsymbol{X}_1, \boldsymbol{X}_2]]$

$$\begin{aligned} &\frac{1}{18}\Big(2 \cdot \overbrace{(u_{211} - u_{121})}^{(1,2,3)} - \overbrace{(u_{121} - u_{211})}^{(2,1,3)} - \overbrace{(u_{211} - u_{112})}^{(1,3,2)} - \overbrace{(u_{121} - u_{112})}^{(2,3,1)} + \\ &+ 2 \cdot \overbrace{(u_{112} - u_{121})}^{(3,2,1)} - \overbrace{(u_{112} - u_{211})}^{(3,1,2)}\Big) = \frac{1}{6}(u_{112} - 2u_{121} + u_{211}). \end{aligned}$$

Pre-sterowania są ułożone w kolejności leksykograficznej (można użyć dowolnej, ustalonej)

$$(u_{112}, u_{121}, u_{211}),$$

zatem

$$w_1^{31} = \frac{1}{6},\ pre_1^{31} = u_{112},\ w_2^{31} = -\frac{1}{3},\ pre_2^{31} = u_{121}, w_3^{31} = \frac{1}{6},\ pre_3^{31} = u_{211}.$$

Współczynnik α_1^3 stowarzyszony z jednomianem $[\boldsymbol{X}_1, [\boldsymbol{X}_1, \boldsymbol{X}_2]]$ jest równy

$$\alpha_1^3 = \frac{1}{6}\int_{T_{r(t)}} (u_{112} - 2u_{121} + u_{211}) d\boldsymbol{s}^3$$

Pozostałe współczynniki α_d^3 są obliczane analogicznie.

Tabela 3.2 Liczba elementów bazy Ph. Halla dla m generatorów do r-tej warstwy włącznie.

	r							
m	2	3	4	5	6	7	8	9
2	**3**	5	8	14	23	41	71	127
3	6	**14**	32	80	196	508	1318	3502
4	10	30	**90**	294	964	3304	11464	40584
5	15	55	205	**829**	3409	14569	63319	
6	21	91	406	1960	**9695**	49685		
7	28	140	728	4088	23632			

Główną zaletą prezentowanego algorytmu kombinatorycznego jest istotnie mniejsza złożoność obliczeniowa w porównaniu do istniejących algorytmów [17, 20], co zostało osiągnięte przez minimalizację użycia kosztownych operacji sprowadzenia do bazy Ph. Halla oraz przetwarzania jedynie niezbędnych jednomianów. W tab. 3.2 przedstawiono liczbę elementów w bazie Ph. Halla w zależności od liczby sterowań m oraz warstwy do której baza jest generowana r. W tabeli elementy na diagonali są pogrubione, pokazując liczbę elementów generowanych podczas działania algorytmu opisanego w [20]. Algorytm opisany w pracy [20] wymaga generowania baz Ph. Halla kilkukrotnie, ponieważ dla każdej warstwy elementy $\boldsymbol{A}(s_j) = \boldsymbol{Z}_j$ są traktowane jako generatory tymczasowego układu (gdzie $m = r$). Przykładowo, dla układu z $n = 9$ wymiarową przestrzenią stanu i $m = 2$ sterowaniami minimalna warstwa mogąca zapewnić STLC jest równa $r = 5$, ponieważ $8 < 9 < 14$ (porównaj, tab 3.2). Używając algorytmu z [20] w momencie przetwarzania elementów piątej warstwy należy wygenerować (i sprowadzić wynik do formy wykorzystującej) 829 elementy, gdyż algorytm ten podczas przetwarzania piątej warstwy korzysta z pięciu abstrakcyjnych generatorów.

W tab. 3.3 zebrano główne różnice między trzema algorytmami obliczającymi presterowania: algorytmem naiwnym, opisanym w [20] oraz kombinatorycznym prezentowanym w tym podrozdziale. Wiersz *l_baz.* określa liczbę baz Ph. Halla generowanych w trakcie działania algorytmu, natomiast wiersz *perm.* opisuje liczbę permutacji liczonych bezpośrednio z podstawienia układu (2.1) do formuły gCBHD. W tab. 3.4 zestawiono czasy działania algorytmów, gdzie m oznacza liczbę sterowań (generatorów), natomiast r warstwę do której włącznie pre-sterowania zostały wyliczone. Dla wszystkich przypadków opisanych w tym rozdziale algorytm kombinatoryczny był znacząco szybszy od zaproponowanego w pracy [20]. Algorytmy zaimplementowano w pakiecie *Mathematica*, w wersji

Tabela 3.3 Porównanie głównych właściwości obliczeniowych algorytmów liczących pre-sterowania formuły gCBHD.

	Algorytm kombinatoryczny	Algorithm z pracy [20]	Algorytm naiwny
$l\_baz.$	1 (baza do r-tej warstwy dla m sterowań)	$r+1$ (bazy zredukowane, liczone do r-tej warstwy dla r sterowań)	$r+1$ (bazy do r-tej warstwy dla r sterowań)
$perm.$	r	$\sum_{k=1}^{r}(k-1)!$	$\sum_{k=1}^{r} k!$

Tabela 3.4 Czas obliczeń (w sekundach) 1000-krotnego uruchomienia algorytmu liczącego pre-sterowania.

	Algorytm kombinatoryczny				
	r				
m	2	3	4	5	6
2	0,756	4,43	17,5	157	4551
3	1,15	16,1	168	5568	551752
4	1,84	43,2	829	55820	
5	3,12	102	3215	320656	
6	4,79	237	10620		
7	6,12	427	30892		
	Algorytm z pracy [20]				
	r				
m	2	3	4	5	6
2	2,01	13	104	3185	zbyt długo
3	3,82	39,6	415	12896	zbyt długo
4	6,21	98,8	1902	144812	
5	8,71	208	7852	zbyt długo	
6	13,2	470	34396		
7	18,5	1018	122684		

11.3. Czasy zamieszczone w tab. 3.4 zostały osiągnięte na komputerze osobistym, z procesorem *Intel Core i5-8400, 2.80GHz* $\times$ *6* i pamięcią 16 GB RAM.

Rozdział 4

Wpływ parametrów sterowań na kształt trajektorii generowanej formułą gCBHD

W rozdziale opisano wpływ parametrów sterowań na kształt trajektorii otrzymanej z algorytmu planowania ruchu wykorzystującego formułę gCBHD opisaną w paragrafie 2.2.4.2 (strona 26). Wpływ ten pokazano na prostych, ale nietrywialnych, przykładach układów nieholonomicznych (2.1) jakimi są jednokołowiec A.1, samochód kinematyczny A.2 oraz integrator Brocketta A.3 (jako przykład układu nilpotentnego). Wszystkie wymienione modele są układami dwuwejściowymi. Punkt docelowy $\boldsymbol{q}_f$ dla każdego przypadku znajduje się w najtrudniejszym do osiągnięcia kierunku $[\boldsymbol{g}_1, \boldsymbol{g}_2]$, będącym w punkcie $\boldsymbol{q}_0 = \boldsymbol{0}$ prostopadłym (dla podanych modeli) do kierunków bezpośrednio sterowanych (uzyskiwanych z generatorów). W przypadku jednokołowca i samochodu kinematycznego osiągnięcie punktu docelowego definiuje manewr parkowania równoległego.

Najprostszym zestawem sterowań pozwalającym na rozwiązanie zadania planowania ruchu w kierunku $[\boldsymbol{g}_1, \boldsymbol{g}_2]$ są sterowania

$$u_1(t) = p_1 \sin(\omega t), \quad u_2(t) = p_2 \cos(\omega t), \quad \omega = 2\pi/T, \tag{4.1}$$

gdzie T oznacza czas ruchu. Dla powyższej parametryzacji ($\boldsymbol{p} = (p_1, p_2)^T$) współczynniki $\boldsymbol{\alpha}(\boldsymbol{p})$, wyliczone zgodnie z równaniami (2.50), przyjmują postać

$$\begin{aligned} &\alpha_1^1 = p_1 \frac{1-\cos(\omega t)}{\omega}, \quad \alpha_2^1 = p_2 \frac{\sin(\omega t)}{\omega}, \quad \alpha_1^2 = p_1 p_2 \frac{\sin(\omega t) - \omega t}{2\omega^2} \\ &\alpha_1^3 = \eta p_1^2 p_2 \cos(\frac{\omega t}{2}), \quad \alpha_2^3 = -\eta p_1 p_2^2 \sin(\frac{\omega t}{2}), \quad \ldots \\ &\eta = \frac{-6\omega t \cos(\omega t/2) + 9\sin(\omega t/2) + \sin(3\omega t/2)}{12\omega^3}. \end{aligned} \tag{4.2}$$

Łatwo pokazać, że sterowania (4.1) pozwalają na poruszanie się w kierunku $[\boldsymbol{g}_1, \boldsymbol{g}_2]$ przy braku przemieszczenia (punktu końcowego) w kierunkach $\boldsymbol{g}_1$ i $\boldsymbol{g}_2$, po podstawieniu $t = T$, $\alpha_1^1 = \alpha_2^1 = 0$ oraz $\alpha_1^2 \neq 0$.

Aby sterowania (4.1) mogły reprezentować szerszą rodzinę sterowań wprowadzono operatory skalowania liniowego oraz (bardziej istotny) rotacji sterowań. Oba operatory w prosty sposób wpływają na energię sterowania oraz nie zależą od stanu układu. Operator liniowego skalowania sterowań

$$\tilde{\boldsymbol{u}} = a \ \cdot \boldsymbol{u} \tag{4.3}$$

wpływa na współczynniki $\boldsymbol{\alpha}$ w następujący sposób

$$\begin{pmatrix}\tilde{\alpha}_1^1\\ \tilde{\alpha}_2^1\end{pmatrix} = a\begin{pmatrix}\alpha_1^1\\ \alpha_2^1\end{pmatrix}, \quad \tilde{\alpha}_1^2 = a^2\alpha_1^2, \quad \begin{pmatrix}\tilde{\alpha}_1^3\\ \tilde{\alpha}_2^3\end{pmatrix} = a^3\begin{pmatrix}\alpha_1^3\\ \alpha_2^3\end{pmatrix}, \ldots \tag{4.4}$$

Operator skalowania liniowego sterowań przekształca więc wyrażenia α_d^r przemnażając je przez potęgę amplitudy a stopnia równego numerowi warstwy r. Drugim operatorem jest operator rotujący sterowania

$$\tilde{\boldsymbol{u}} = Rot(\beta)\boldsymbol{u} = \begin{pmatrix}\cos(\beta) & -\sin(\beta)\\ \sin(\beta) & \cos(\beta)\end{pmatrix}\boldsymbol{u}. \tag{4.5}$$

Operator rotacji nałożony na sterowania wpływa na pierwszych kilka współczynników $\boldsymbol{\alpha}$ w następujący sposób

$$\begin{pmatrix}\tilde{\alpha}_1^1\\ \tilde{\alpha}_2^1\end{pmatrix} = Rot(\beta)\begin{pmatrix}\alpha_1^1\\ \alpha_2^1\end{pmatrix}, \quad \tilde{\alpha}_1^2 = \alpha_1^2, \quad \begin{pmatrix}\tilde{\alpha}_1^3\\ \tilde{\alpha}_2^3\end{pmatrix} = Rot(\beta)\begin{pmatrix}\alpha_1^3\\ \alpha_2^3\end{pmatrix}. \tag{4.6}$$

W szczególnym przypadku, gdy współczynniki $p_1 = p_2 = p$, rotacja sterowań o kąt β sprowadza się do przesuwania fazy funkcji trygonometrycznych o kąt $\phi(\beta)$

$$\tilde{u}_1(t) = p\sin(\omega t - \phi(\beta)), \quad \tilde{u}_2(t) = -p\cos(\omega t - \phi(\beta)). \tag{4.7}$$

W celu porównania trajektorii uzyskanych za pomocą formuły gCBHD zaproponowano dwa wskaźniki jakości. Pierwszym jest odległość między punktami końcowymi trajektorii – uzyskanej za pomocą gCBHD (wzór (2.56), strona 29, oznaczonej jako *test*) oraz uzyskanej za pomocą całkowania numerycznego układu (2.1) ze sterowaniami uzyskanymi ze wzoru (2.56) (*real*)

$$J_1(\boldsymbol{q}_0, \boldsymbol{u}(\boldsymbol{p})) = \|\boldsymbol{q}_{real}(T) - \boldsymbol{q}_{test}(T)\|. \tag{4.8}$$

Drugim jest rozszerzenie pierwszego wskaźnika na interwał $t \in [0, T]$ uzyskując skumulowany błąd trajektorii

$$J_2(\boldsymbol{q}_0, \boldsymbol{u}(\boldsymbol{p})) = \int_0^T \|\boldsymbol{q}_{real}(t) - \boldsymbol{q}_{test}(t)\|dt. \tag{4.9}$$

W równaniach (4.8), (4.9) przyjęto odległość rozumianą euklidesowo. Taki wybór wiąże się jednak z pewnymi niedogodnościami. W przypadku robotów mobilnych (w tym jednokołowca i samochodu kinematycznego) oraz robotów kosmicznych wektor $\boldsymbol{q}$ składa się ze współrzędnych opisujących pozycję i/lub orientację. Dlatego warto rozważyć stosowanie zamiast odległości euklidesowej jej zmodyfikowaną wersję, gdzie współrzędne mają różne wagi. Przykładowo, jeśli orientacja jednokołowca jest nieistotna można jej wpływ pominąć przyjmując odpowiednią wagę równą 0.

W przypadku jednokołowca i integratora Brocketta w formule gCBHD wykorzystano pola wektorowe i współczynniki $\boldsymbol{\alpha}$ do drugiej warstwy włącznie, a dla samochodu kinematycznego – do trzeciej. W każdym przypadku jest to minimalna warstwa zapewniająca STLC (patrz sekcja 2.1.1). Dla wszystkich modeli przeanalizowano wpływ czasu trwania ruchu T, amplitudy sterowań a (równanie (4.3)), początkowej fazy sterowań $\phi(\beta)$ (równanie (4.7)), stosunku p_1 do p_2 przy stałej energii ruchu

$$p_1^2 + p_2^2 = E = \text{const}, \tag{4.10}$$

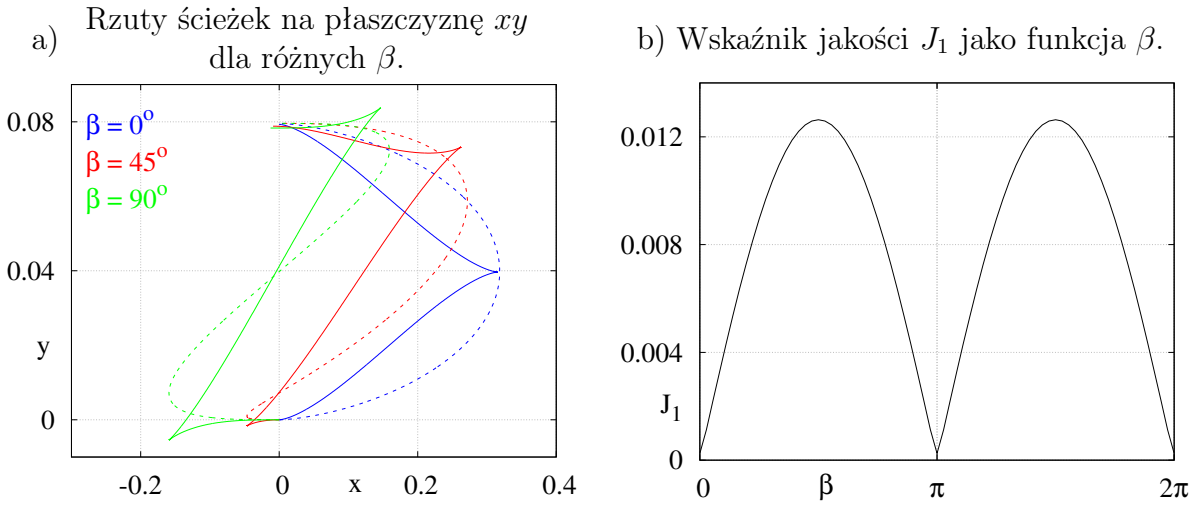
przesunięcia fazowego ϕ między sterowaniami

$$u_1(t) = p_1 \sin(\omega t), \quad u_2 = p_2 \sin(\omega t + \phi) \tag{4.11}$$

oraz zwiększenia częstotliwości składowych sterowań

$$u_1(t) = p_1 \sin(k\omega t), \quad u_2(t) = p_2 \cos(k\omega t), \quad k \in \mathbb{N}_+ \tag{4.12}$$

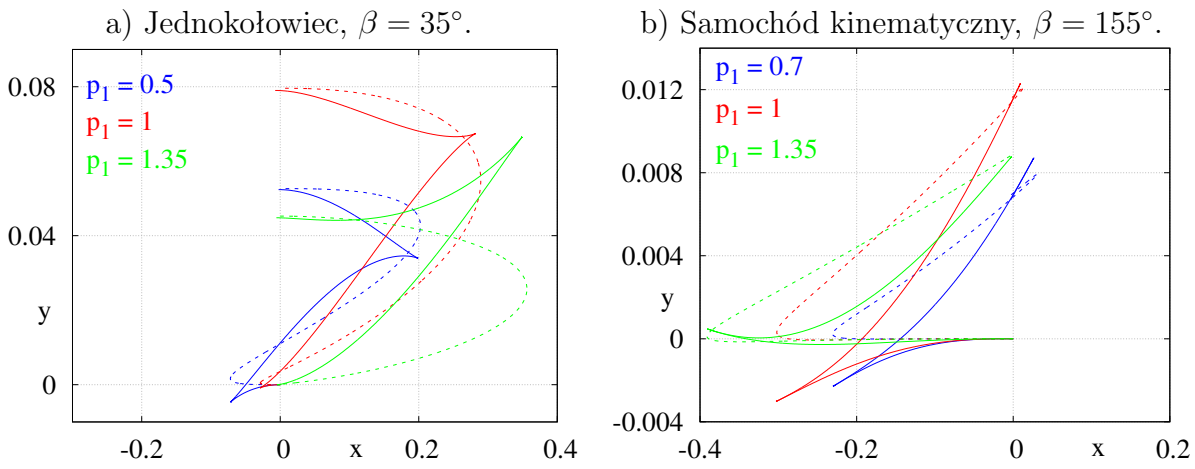
na kształt oraz jakość trajektorii (ścieżki) ruchu ocenianą zgodnie z (4.8), (4.9). Otrzymane ścieżki ruchu zostały przedstawione na rysunkach, gdzie linią przerywaną oznaczono ścieżkę otrzymaną z formuły gCBHD (uciętej w odpowiedniej warstwie, planowaną), a ciągłą – za pomocą całkowania numerycznego (rzeczywistą) modeli ze sterowaniami otrzymanymi z gCBHD. w każdym przypadku przyjęto punkt początkowy $\boldsymbol{q}_0 = \mathbf{0}$, a pożądany kierunek ruchu wyznacza oś $0Y$.



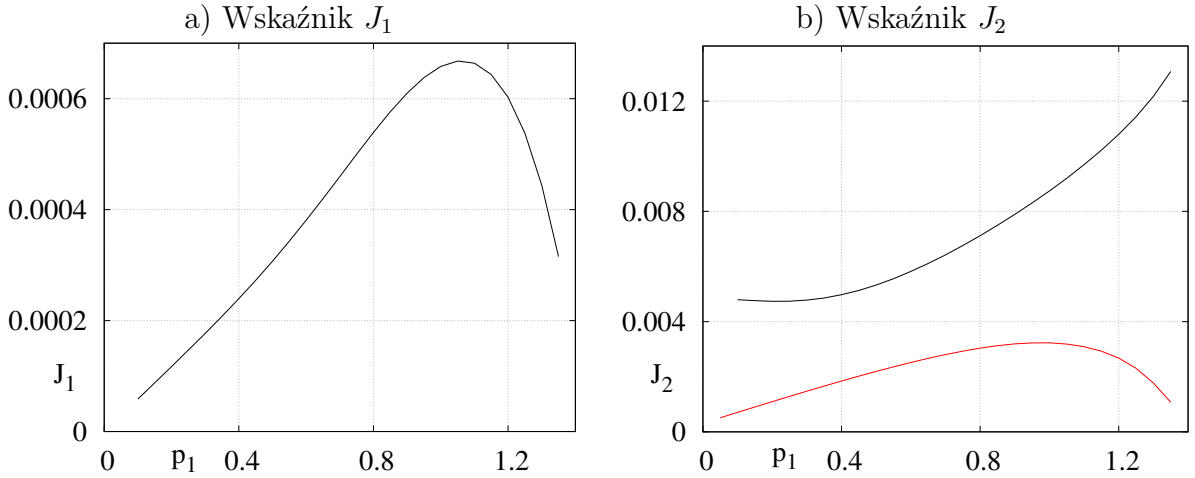
Rysunek 4.1 Zależność trajektorii jednokołowca od kąta obrotu sterowań β.

Rys. 4.1a przedstawia rzuty planowanej i rzeczywistej trajektorii na płaszczyznę xy dla jednokołowca otrzymane dla sterowań (4.1) obróconych o różne kąty β, gdzie pozostałe parametry to $p_1 = p_2 = T = 1$. Jak można zauważyć, sterowania wygenerowały (dla czasu $t = T$) ruch w kierunku $0Y$ (kierunek $[\boldsymbol{g}_1, \boldsymbol{g}_2](t = 0)$), niedostępny bezpośrednio dla przestrzeni rozpiętej przez generatory. Planowana trajektoria nie jest dokładnym odzwierciedleniem trajektorii rzeczywistej. Przykładowo: rzeczywista trajektoria ma szpice pojawiające się gdy prędkość postępowa układu zmienia znak. Jednak obie trajektorie są jakościowo podobne, co potwierdza użyteczność formuły gCBHD w planowaniu ruchu. Kształt i lokalizacja wynikowych trajektorii mocno zależy od kąta obrotu β. Ten fakt może być efektywnie wykorzystany w środowisku kolizyjnym, gdzie odpowiednia lokalizacja ścieżki ruchu jest kluczowa podczas omijania przeszkód. Na rys. 4.1b wskaźnik jakości J_1, czyli odległość pomiędzy końcowymi punktami (dla $t = T$) obu trajektorii jednokołowca, został przedstawiony jako funkcja kąta obrotu β. Wskaźnik J_1 silnie zależy od obrotu. Najlepsze (minimalne) wartości przyjmuje dla $\beta = k\pi, k \in \mathbb{Z}$, jednak trajektoria dla takich wartości β jest najobszerniejsza. Najgorsze, i relatywnie duże w stosunku do przebytego dystansu $\|\boldsymbol{q}(0) - \boldsymbol{q}(T)\| \approx 0.08$, wartości J_1 przyjmuje dla kąta β zapewniającego najmniej obszerny manewr.

Na rys. 4.2 przedstawiono przykładowe rzuty ścieżek na płaszczyznę xy dla jednokołowca i samochodu kinematycznego przy stałej energii ruchu $E = 2$ rozdysponowanej



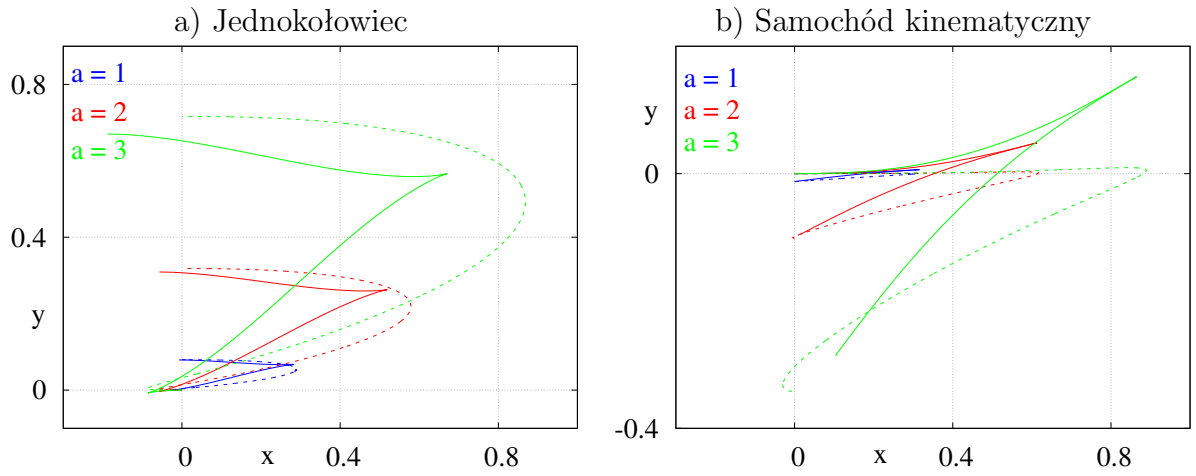
Rysunek 4.2 Rzuty ścieżek na płaszczyznę xy dla jednokołowca i samochodu kinematycznego dla różnych wartości p_1 i p_2 oraz stałej energii sterowania $E = 2$.



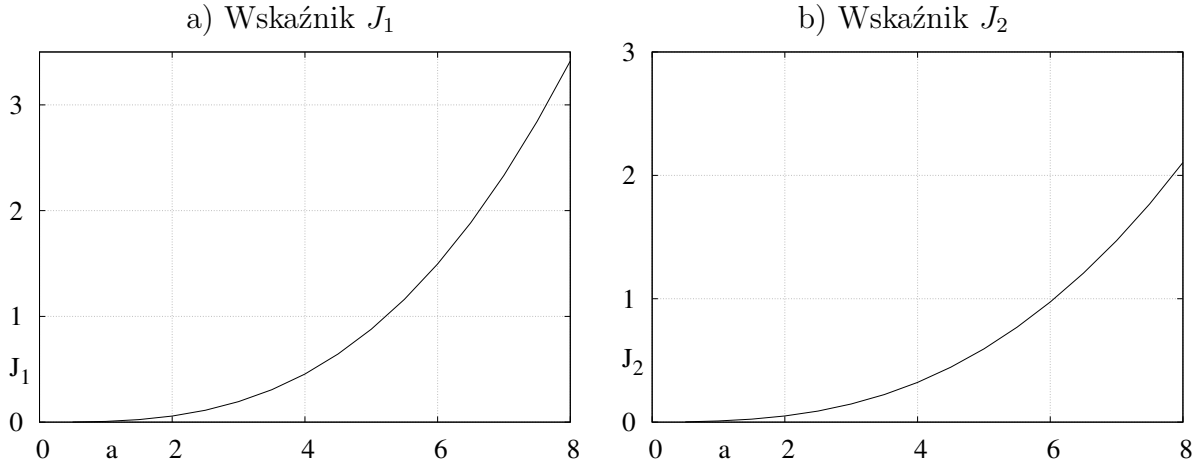
Rysunek 4.3 Wskaźniki jakości J_1 i J_2 jako funkcja p_1 dla stałej energii sterowań $E = 2$ samochodu kinematycznego i $\beta = 155°$.

w różnej proporcji między sterowania $u_1(p_1)$ i $u_2(p_2)$. Można zauważyć, że wielkość przesunięcia w pożądanym kierunku $0Y$ zależy od proporcji p_1/p_2. Najdłuższe przesunięcie otrzymano dla proporcji $p_1/p_2 = 1$. Na rys. 4.3 zobrazowano wpływ tej proporcji (a konkretnie, wartości p_1 przy stałej energii $E = 2$) na wartości wskaźników jakości J_1 i J_2. Na rys 4.3b czerwoną linią oznaczono dodatkowo wskaźnik jakości J_2 liczony jedynie dla współrzędnych pozycyjnych, aby podkreślić duży wpływ współrzędnych kątowych na wartość J_2 dla dużych p_1. Z rysunków wnioskuje się, że największe różnice między ścieżkami (trajektoriami) występują dla najbardziej efektywnego ruchu, tj. $p_1 = p_2 = 1$, jednak wartości te są małe w porównaniu z całkowitą wartością przesunięcia. Ponadto, łatwo zauważyć, że rozdysponowanie energii między sterowania wpływa na kształt otrzymanych trajektorii. Dla niektórych wartości p_1 trajektoria rzeczywista ma dość znaczący fragment mający wartości $y < 0$. Zatem zmianę stosunku amplitud sterowań można rozważać jako dodatkowy sposób na znajdywanie trajektorii o oczekiwanym kształcie.

Rys. 4.4 ilustruje rzuty ścieżek na płaszczyznę xy dla jednokołowca i samochodu kinematycznego przy parametrach sterowań $p_1 = p_2 = T = 1$, $\beta = 35°$ (na 4.4a) lub $\beta = 0$ (na 4.4b) przy różnych wartościach liniowego skalowania amplitudy sterowań parameterem a. Identyczne wyniki można uzyskać w przypadku stałego współczynnika $a = 1$ i zmiennej



Rysunek 4.4 Rzuty ścieżek na płaszczyznę xy dla jednokołowca oraz samochodu kinematycznego przy sterowaniach o różnych wartościach amplitudy sterowań a.

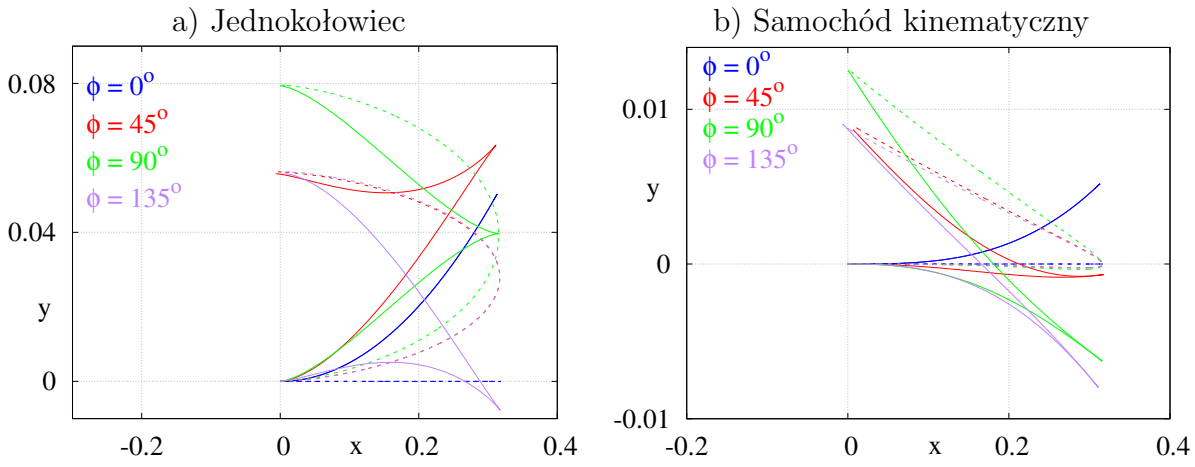


Rysunek 4.5 Wskaźniki jakości J_1 i J_2 jako funkcja amplitudy sterowań a przy stałej wartości T dla jednokołowca.

wartości czasu ruchu T. O wymienności parametrów a i T można także wywnioskować analizując równania (4.2) oraz (4.4). Zależność ta jest niezależna od modelu i przydatna w planowaniu ruchu. Przykładowo, w przypadku ograniczenia amplitud sterowań można uzyskać pożądaną trajektorię poprzez wydłużenie czasu ruchu. Na rysunkach można zauważyć spadek dokładności planowanej ścieżki (trajektorii) wobec rzeczywistej wraz ze wzrostem wartości iloczynu $a \cdot T$. Wspomnianą zależność można też zaobserwować na rys. 4.5 przedstawiającym wzrost kryteriów jakości (interpretowanych jako różnica pomiędzy trajektoriami) wraz ze wzrostem wartości a. Spadek dokładności planowania trajektorii, zwłaszcza dla $a \cdot T > 1$, wynika z większego wpływu iloczynu $a \cdot T$ na współczynniki $\boldsymbol{\alpha}$ odpowiadające wyższym warstwom. Metoda wykorzystująca formułę gCBHD opiera się na nieskończonym[1] szeregu (2.50). Z przyczyn praktycznych szereg ten jest ograniczany do warstw niezbędnych do uzyskania STLC. Przy $a \cdot T < 1$ wpływ warstw nie branych pod uwagę jest pomijalny, natomiast przy $a \cdot T > 1$ staje się dominujący.

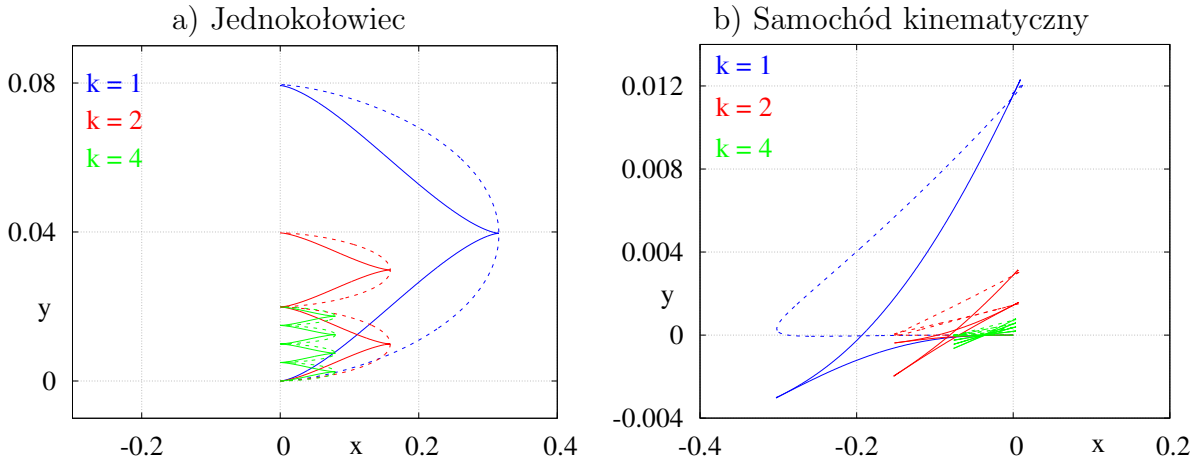
Na rys. 4.6 pokazano wpływ kąta przesunięcia fazowego ϕ dla sterowań (4.11) (parametry $p_1 = p_2 = T = 1$) na kształt otrzymanych ścieżek (trajektorii) dla jednokołowca

[1]Dla układów nienilpotentnych.



Rysunek 4.6 Rzuty ścieżek na płaszczyznę xy dla jednokołowca i samochodu kinematycznego przy sterowaniach (4.11) i różnej wartości przesunięcia fazowego ϕ.

i samochodu kinematycznego. Z powodu symetrii trajektorii względem osi x dla dodatnich i ujemnych kątów ϕ, rysunki ilustrują jedynie przypadki z dodatnią wartością ϕ. Można zaobserwować, że istnieje optymalna wartość $\phi^\star = 90°$ dla której przemieszczenie w kierunku y jest największe. Od wartości optymalnego przesunięcia istnieją dwie gałęzie $\phi = \phi^\star \pm \delta\phi$ dla których planowane trajektorie znajdują się bardzo blisko siebie a ich punkty końcowe niemal się pokrywają. Jednak trajektorie rzeczywiste dla obu gałęzi różnią się znacząco. Dla najniekorzystniejszego przypadku $\phi = 0°, \phi = 180°$ sterowania dla obu generatorów są zależne liniowo i punkt końcowy trajektorii pokrywa się z początkowym.



Rysunek 4.7 Rzuty ścieżek na płaszczyznę xy dla jednokołowca i samochodu kinematycznego przy sterowaniach (4.12) i różnej wartości k.

Rys. 4.7 przedstawia wpływ zwiększania częstotliwości sterowań (4.12) na kształt ścieżki (trajektorii) dla jednokołowca i samochodu kinematycznego z parametrami $p_1 = p_2 = T = 1$. Dla wyższych częstotliwości identyczny szablon ruchu zostaje powielony k-krotnie. Obszerność ruchu jest mniejsza, kosztem jednak wartości przesunięcia wzdłuż osi y. Co ciekawe, skalowanie przesunięcia i obszerności ruchu nie musi być liniowe co pokazuje rys. 4.7b. Uzyskane wyniki potwierdzają poprzednią obserwację zależności pomiędzy wielkością przesunięcia (długością ścieżki) a czasem trwania ruchu T, gdyż zwiększenie częstotliwości można interpretować jako k krotne powtórzenie ruchu ze zmienionym czasem

jego trwania $\tilde{T} = T/k$.

Wyniki symulacji dla integratora Brocketta są mniej interesujące, aczkolwiek wciąż pouczające. Ponieważ układ jest nilpotentny i wszystkie warstwy bazy Ph. Halla powyżej drugiej zerują się tożsamościowo, planowana i rzeczywista ścieżka w każdym przypadku pokrywają się , a wskaźniki $J_1 = J_2 = 0$. Dla nilpotentnych układów formula gCBHD daje idealną prognozę ścieżki rzeczywistej, więc układy takie są bardzo pożądane w planowaniu ruchu. Istnieje opisana w literaturze przedmiotu procedura aproksymacji nieliniowych układów układami nilpotentnymi [102]. Niestety działa ona jedynie lokalnie i przekształca sterowania do formy nieliniowej i zależnej od aktualnej konfiguracji $\boldsymbol{q}$. W takiej formie optymalizacja i ograniczanie sterowań staje się zadaniem trudnym. Nieliniowa zależność pomiędzy przesunięciem w pożądanym kierunku a iloczynem $a \cdot T$ istnieje również dla układów nilpotentnych.

Rozdział 5

Ocena jakości parametryzacji sterowań

W wielu metodach planowania ruchu, zarówno globalnych (np. endogenicznej przestrzeni konfiguracyjnej 2.2.3) jak i lokalnych (np. Lie-algebraiczna 2.2.4), często występuje konieczność reprezentowania sterowań naturalnie występujących w nieskończenie wymiarowej przestrzeni funkcyjnej. W celu znaczącego ułatwienia obliczeń numerycznych reprezentuje się sterowania jako kombinacje liniowe funkcji bazowych pewnej skończonej bazy funkcyjnej opisanej w podrozdziale 1.5. Jednak wybór bazy funkcyjnej, oraz przede wszystkim konkretnych funkcji bazowych wykorzystanych do opisania sterowań, może mieć kluczowe znaczenie dla jakości (lub nawet istnienia) rozwiązania zadania planowania ruchu.

W niniejszym rozdziale skupiono się na doborze parametryzacji sterowań w metodzie Lie-algebraicznej planowania ruchu wykorzystującej formułę gCBHD. Za bazę funkcyjną przyjęto ortonormalną bazę harmoniczną, zob. sekcja 1.5.1, jako najczęściej stosowaną w planowaniu ruchu. Badania oparto na analizie statystycznej danych liczbowych dla sterowań i trajektorii uzyskanych dla różnych zestawów funkcji bazowych. Modelem dla przeprowadzonych badań był jednokołowiec A.1, wybrany z powodu niskiej wymiarowości, jednakże dobrze ilustrujący wszystkie aspekty planowania ruchu układów nieholonomicznych za pomocą formuły gCBHD. Do rozwiązania równania (2.56) użyto algorytmu Newtona, zob. podrozdział 1.4 w wersji podstawowej (tj. bez optymalizacji) jako metody odniesienia, oraz z dwoma funkcjami optymalizującymi w przestrzeni zerowej jakobianu.

Dla testowanych układów, przez zmianę współrzędnych każdą parę punktów brzegowych planowania $(\boldsymbol{q}_0, \boldsymbol{q}_f)$ można sprowadzić do pary $(\mathbf{0}, \boldsymbol{q}_f^\star)$, więc bez straty ogólności można założyć, że konfiguracją początkową układu jest punkt $\boldsymbol{q}_0 = \mathbf{0}$.

Jednokołowiec jest układem trójwymiarowym, spełniającym warunek STLC przez dwie pierwsze warstwy bazy Ph. Halla (zobacz dodatek A.1). Dla jednokołowca, w punkcie $\boldsymbol{q}_0 = \mathbf{0}$, parametryczny wzór (2.56) określający przesunięcie konfiguracji przyjmuje postać

$$\boldsymbol{z}(T, \boldsymbol{q}_0, \boldsymbol{u}(\boldsymbol{p})) = \xi \cdot (\boldsymbol{q}_f - \mathbf{0}) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \boldsymbol{\alpha}(\boldsymbol{p}), \tag{5.1}$$

gdzie $\boldsymbol{\alpha}(\boldsymbol{p})$ zależy od wybranej parametryzacji sterowań. Dla uzyskania ruchu w każdym kierunku, wektor $\boldsymbol{p}$ określający (wraz z wybraną bazą) parametryzację powinien mieć

wymiarowość przynajmniej równą wymiarowości wektora konfiguracji

$$\dim(\boldsymbol{p}) \geqslant n. \tag{5.2}$$

Tabela 5.1 Pięć pierwszych elementów harmonicznej bazy ortonormalnej i ich kody (i).

kod i	β_i	$\psi_i(t)$
1	1	1
2	$\sqrt{2}$	$\sin(\omega t)$
3	$\sqrt{2}$	$\cos(\omega t)$
4	$\sqrt{2}$	$\sin(2\,\omega t)$
5	$\sqrt{2}$	$\cos(2\,\omega t)$
$\omega = 2\pi/T$		

Tabela 5.2 Kody sterowań odpowiadających wybranym parametryzacjom.

nr	u_1	u_2
#1	(1,2)	(1,3)
#2	(1,3)	(1,2)
#3	(1,4)	(1,5)
#4	(1,5)	(1,4)
#5	(1,2,3)	(1,2,3)
#6	(1,4,5)	(1,4,5)
#7	(1,2,5)	(1,3,4)
#8	(1,3,4)	(1,2,5)
#9	(1,2,3,4,5)	(1,2,3,4,5)

Do testów wybrano dziewięć parametryzacji bazujących na ortonormalnej bazie harmonicznej (patrz wzór (1.17)). Pięć pierwszych elementów bazy oraz wybrane parametryzacje wraz z ich kodami zawierają tab. 5.1, 5.2, odpowiednio. Przykładowo: dla ruchu w przedziale $t \in [0, T = 1]$ parametryzacja #7 jesto opisana następującym zestawem danych

$$\begin{aligned} \boldsymbol{p} &= (p_{1,1}, p_{1,2}, p_{1,5}, p_{2,1}, p_{2,3}, p_{2,4})^T, \\ u_1 &= p_{1,1} + p_{1,2}\sqrt{2}\sin(\omega t) + p_{1,5}\sqrt{2}\cos(2\omega t), \\ u_2 &= p_{2,1} + p_{2,3}\sqrt{2}\cos(\omega t) + p_{2,4}\sqrt{2}\sin(2\omega t). \end{aligned} \tag{5.3}$$

Do rozwiązania równania (5.1) wykorzystano algorytm Newtona 1.4. Wzory (1.13) oraz (1.15) dla wersji podstawowej i z optymalizacją, dla przyjętych notacji, przyjmują postać

$$\boldsymbol{p}_{i+1} = \boldsymbol{p}_i + \zeta \boldsymbol{J}^{\#}(\boldsymbol{p}_i)(\xi(\boldsymbol{q}_f - \boldsymbol{q}_0) - \boldsymbol{z}(T, \boldsymbol{q}_0, \boldsymbol{u}(\boldsymbol{p}_i))), \tag{5.4}$$

$$\boldsymbol{p}_{i+1} = \boldsymbol{p}_i + \zeta_1 \boldsymbol{J}^{\#}(\boldsymbol{p}_i)(\xi(\boldsymbol{q}_f - \boldsymbol{q}_0) - \boldsymbol{z}(T, \boldsymbol{q}_0, \boldsymbol{u}(\boldsymbol{p}_i))) + \zeta_2(\boldsymbol{J}^{\#}(\boldsymbol{p}_i)\boldsymbol{J}(\boldsymbol{p}_i) - \boldsymbol{I})\frac{\partial w(\boldsymbol{p})}{\partial \boldsymbol{p}}|_{\boldsymbol{p}=\boldsymbol{p}_i}. \tag{5.5}$$

Wyrażenie $\boldsymbol{z}(T, \boldsymbol{q}_0, \boldsymbol{u}(\boldsymbol{p}_i))$ użyte w równaniach (5.4) i (5.5) należy rozumieć jako prawą stronę wzoru (2.56). Wektor wartości początkowych $\boldsymbol{p}_0$ składa się z zer, z wyjątkiem stałych składowych każdego sterowania (na przykładzie z równania (5.3) $\boldsymbol{p}_0 = (1, 0, 0, 1, 0, 0)^T$). Wybór takich danych początkowych wynika z konieczności zapewnienia niezerowego wektora $\boldsymbol{p}$ dla poprawnego działania algorytmu Newtona, a składowa stała jest jedyną występującą w każdej z parametryzacji. W celu zmniejszenia wpływu parametrów ζ, ζ_1, ζ_2 algorytmu Newtona na wynik jego działania w każdym kroku przeprowadzono dodatkowo ich optymalizację. W przypadku parametru ζ optymalizacja polegała na wyborze wartości dającej najdłuższy krok w kierunku celu (najmniejszą różnicę $\xi(\boldsymbol{q}_f - \boldsymbol{q}_0) - \boldsymbol{z}(T, \boldsymbol{q}_0, \boldsymbol{u}(\boldsymbol{p}_i))$). W przypadku pary (ζ_1, ζ_2) wybierano wartość realizującą najmniejszą wartość funkcji jakości $w(\boldsymbol{p}_i)$ zapewniając jednocześnie zbieżność w każdym kroku. W obu przypadkach korzystano z dyskretnej optymalizacji, analizując zbiór wartości parametru $\{0.01, 0.02, \ldots, 1\}$

lub siatkę $\{0.01, 0.02, \ldots, 1\} \times \{0.01, 0.02, \ldots, 1\}$. $\boldsymbol{q}_f$ we wzorach (5.4), (5.5) oznacza wybrany punkt docelowy.

Jako funkcję jakości $w(\boldsymbol{u}(\cdot))$ zaproponowano klasyczną funkcję opisującą energię sterowań, która dla unormowanej bazy jest zadana zależnością

$$w(\boldsymbol{u}(\cdot)) = \int_0^T \boldsymbol{u}^T(t)\boldsymbol{u}(t)dt = \sum_{i=1}^{m}\sum_{j=1}^{N_i} p_{i,j}^2, \tag{5.6}$$

oraz funkcję bazującą na wyrażeniach $\boldsymbol{\alpha}$, która bierze pod uwagę rosnący wraz z warstwą algebry Liego koszt ruchu wzdłuż konkretnego pola wektorowego

$$w(\boldsymbol{u}(\cdot)) = \sum_{i=1}^{i^\star}\left(c_i \sum_{d=1}^{\#\boldsymbol{H}^i}(\alpha_d^i(\boldsymbol{p}))^2\right). \tag{5.7}$$

W definicji funkcji (5.7) $i^\star$ oznacza najwyższą rozważaną warstwę bazy Ph. Halla, a $\#\boldsymbol{H}^i$ – liczność danej warstwy. Współczynnik c_i, zależny od numeru warstwy, oznacza trudność ruchu wzdłuż pól z tej warstwy. Wartości współczynnika c_i dla poszczególnych warstw określono za pomocą formuły CBHD (strona 24). Poniżej uzasadniono ich (energetyczne) wartości dla pierwszych trzech warstw, na przedziale $[0, T]$:

$$\begin{aligned}
c_1 = T: &\quad e(T\boldsymbol{g}_1),\\
c_2 = 4\sqrt{T}: &\quad e(T[\boldsymbol{g}_1,\boldsymbol{g}_2]) = e(\sqrt{T}\boldsymbol{g}_1)e(\sqrt{T}\boldsymbol{g}_2)e(-\sqrt{T}\boldsymbol{g}_1)e(-\sqrt{T}\boldsymbol{g}_2),\\
c_3 = 8\sqrt[3]{T}: &\quad e(T[\boldsymbol{g}_1,[\boldsymbol{g}_1,\boldsymbol{g}_2]]) = e(\sqrt[3]{T}\boldsymbol{g}_1)e(\sqrt[3]{T}\boldsymbol{g}_2)e(-\sqrt[3]{T}\boldsymbol{g}_1)e(-\sqrt[3]{T}\boldsymbol{g}_2)\\
&\quad e(-\sqrt[3]{T}\boldsymbol{g}_1)e(\sqrt[3]{T}\boldsymbol{g}_2)e(\sqrt[3]{T}\boldsymbol{g}_1)e(-\sqrt[3]{T}\boldsymbol{g}_2),
\end{aligned} \tag{5.8}$$

gdzie $e(\cdot)$ oznacza $\exp(\cdot)$, a T jest odpowiednio małe. Współczynnik c_i odpowiada energii sekwencji ruchów elementarnych, ze sterowaniem aktywnym o wartości ± 1, pozwalającej na wygenerowanie ruchu wynikowego o głównej składowej wzdłuż wybranego pola z i-tej warstwy.

Poniżej przedstawiono przykłady wyrażenia $\boldsymbol{\alpha}(\boldsymbol{p}) = (\alpha_1^1(\boldsymbol{p}), \alpha_2^1(\boldsymbol{p}), \alpha_1^2(\boldsymbol{p}))^T$ do trzeciej warstwy włącznie dla parametryzacji o numerach #2 i #7:

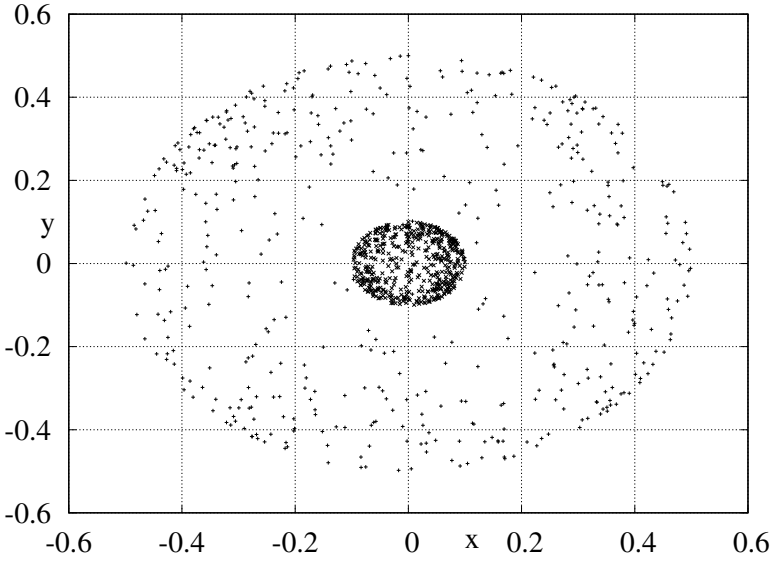
$$\begin{aligned}
\#2: &\quad \boldsymbol{\alpha}(\boldsymbol{p}) = (p_{1,1}, p_{2,1}, \frac{1}{2\pi}(-2p_{1,1}p_{2,2} + p_{1,3}p_{2,2}))^T,\\
\#7: &\quad \boldsymbol{\alpha}(\boldsymbol{p}) = (p_{1,1}, p_{2,1}, \frac{1}{8\pi}(4p_{1,2}p_{2,1} - 2p_{1,2}p_{2,3} - 2p_{1,1}p_{2,4} + p_{1,5}p_{2,4}))^T.
\end{aligned}$$

Metoda Lie-algebraiczna jest lokalna, więc planowanie globalne składa się zazwyczaj z wielu lokalnych planowań, a wynik poprzedniego planowania wpływa czasem znacząco na kolejne. Zatem by ten wpływ ograniczyć, w badaniach skupiono się na analizie tylko jednego kroku planowania ruchu metodą Lie-algebraiczną. Dodatkowo, aby uniezależnić wyniki od wybranego kierunku ruchu wygenerowano 1000 zadań planowania ruchu losując punkty końcowe (kierunki), bliskie punktowi $\boldsymbol{q}_0 = \boldsymbol{0}$, według rozkładu

$$\Delta\boldsymbol{q}_i = \delta \cdot \frac{\boldsymbol{q}_{rand}}{\|\boldsymbol{q}_{rand}\|}, \quad \boldsymbol{q}_{rand} = \begin{pmatrix}\mathrm{rand}(-1,1)\\ \mathrm{rand}(-1,1)\\ \mathrm{rand}(-1,1)\end{pmatrix}, \tag{5.9}$$

gdzie funkcja $\mathrm{rand}(a, b)$ generuje liczbę z przedziału $[a, b]$ według rozkładu jednostajnego

ciągłego. Współczynnik δ ze wzoru (5.9) przyjmował wartości $\delta = 0.1$ lub $\delta = 0.5$, generując po 500 konfiguracji docelowych dla każdej wartości współczynnika. Na rys. 5.1 przedstawiono rzut wylosowanych punktów na płaszczyznę xy, czyli pierwszych dwóch zmiennych wektora konfiguracji jednokołowca.



Rysunek 5.1 Rzut na płaszczyznę xy końcowych punktów $\Delta\boldsymbol{q}$ wygenerowanych w przestrzeni konfiguracyjnej.

Operator przesunięcia $\boldsymbol{z}(t, \boldsymbol{q}_0, \boldsymbol{u}(\boldsymbol{p}))$ z formuły gCBHD jest szeregiem nieskończonym, ograniczonym w przypadku równania (5.1) do $\boldsymbol{i}^\star = 2$ warstw. Z tego powodu występują różnice w położeniach zadanego punktu końcowego $\boldsymbol{q}_f$ oraz faktycznie zrealizowanego $\boldsymbol{q}_{freal}$. Punkt $\boldsymbol{q}_{freal}$ jest określony jako koniec trajektorii układu (2.1), zainicjalizowanej w $\boldsymbol{q}_0$, odpowiadającej sterowaniom w formie parametrycznej otrzymanych z formuły gCBHD i algorytmu Newtona. W celu odrzucenia trajektorii dla których różnica jest zbyt duża wprowadzono warunek

$$\|\boldsymbol{q}_f - \boldsymbol{q}_{freal}\| = \|\boldsymbol{q}_0 + \Delta\boldsymbol{q} - \boldsymbol{q}_{freal}\| < 0.1 \cdot \delta \tag{5.10}$$

sprawdzający czy różnica odległości między punktami $\boldsymbol{q}_f$ i $\boldsymbol{q}_{freal}$ nie przekracza 10% maksymalnego zakresu ruchu (odległość $\boldsymbol{q}_0$ od $\boldsymbol{q}_f$). Należy zwrócić uwagę, że otrzymana trajektoria może nie spełniać warunku (5.10) z dwóch powodów:

1. Wektor parametrów $\boldsymbol{p}$ jest źle skonstruowany i uniemożliwia osiągnięcie pewnych kierunków. Przykładem jest zestaw parametrów z rozdziału 4, równanie (4.11), rys. (4.6), dla $\phi = 0°$. Przypadek ten jest niezwykle rzadki.
2. Zakres pojedynczego planowania δ (zobacz równanie (5.9)) algorytmu metody Lie-algebraicznej jest zbyt duży. Skrócenie zakresu spowoduje, że ogon (warstwy większe od $i^\star$-ej) szeregu $\boldsymbol{z}(t, \boldsymbol{q}_0, \boldsymbol{u}(\boldsymbol{p}))$ ma mniejszy wpływ na położenie punktu końcowego $\boldsymbol{q}_{freal}$.

Dla porównania jakości parametryzacji, dla każdego wygenerowanego kierunku i każdej parametryzacji przydzielono rangę (1-najlepsza, 9-najgorsza) według jednego z trzech kryteriów: długości ścieżki, energetycznej funkcji jakości (5.6) lub funkcji jakości bazującej

na wyrażeniach $\boldsymbol{\alpha}$ (5.7). Dla kryterium długości ścieżki do obliczeń używano podstawowego algorytmu Newtona (5.4), natomiast dla obu funkcji jakości – algorytmu Newtona z optymalizacją w przestrzeni zerowej (5.5).

Metodologię badań podsumowano w Algorytmie 5.1, a wyniki przeprowadzonych testów zebrano w tab. 5.3, 5.4, 5.5.

Algorytm 5.1 Metodologia oceny jakości parametryzacji sterowań.

Dane wejściowe: Układ nieholonomiczny (2.1), konfiguracja początkowa $\boldsymbol{q}_0$, zakres planowania δ.

Krok 1. Określić odpowiednią liczbę warstw $i^\star$ zapewniającą STLC, zgodnie z (2.7).

Krok 2. Określić zbiór K_1 składający się z parametryzacji spełniających warunek (5.2). Wypełnić zerami tabelę zawierającą sumy rang dla każdej parametryzacji oraz tabelę pomocniczą.

Krok 3. Wygenerować losowo (według rozkładu jednorodnego) K_2 przemieszczenie $\Delta\boldsymbol{q}_i,\ i = 1, \ldots, K_2$, o długości δ.

Krok 4. Dla każdego przemieszczenia $i = 1, \ldots, K_2$ powtarzać kroki 5-6.

Krok 5. Dla każdej parametryzacji uruchomić odpowiedni algorytm Newtona inicjalizowany w identycznym $\boldsymbol{p}_0$ otrzymując wynikowy wektor $\boldsymbol{p}$. Określić koszty ruchu w każdym z wylosowanych kierunków $\Delta\boldsymbol{q}_i$ i wpisać je do tymczasowej tabeli.

Krok 6. Posortować tymczasową tabelę w porządku rosnącym. Indeksy tymczasowej tabeli potraktować jako rangi i dodać do tablicy rang dla odpowiedniej parametryzacji. Tymczasową tabelę wyzerować.

Krok 7. Podzielić dane w tabeli rang przez K_1 uzyskując średnie rangi. Zaprezentować wszystkie parametryzacje wraz ich uśrednionymi rangami.

W tab. 5.3, 5.4 przedstawiono średnią rangę parametryzacji w kilku kategoriach. Tab. 5.3/5.4 opisują wyniki testów 500 kierunków o zakresie planowania $\delta = 0.1/0.5$, odpowiednio. W obu tabelach kolumny długość i jakość określają średnią rangę długości ścieżki i funkcji jakości. Numery $1, 2, 3$ dodane do nazwy kolumny należy interpretować następująco:

1. przy przypisywaniu rangi brano pod uwagę wszystkie trajektorie wynikowe, nawet nie spełniające warunku (5.10),
2. przy przypisywaniu rangi parametryzacjom dla których trajektorie nie spełniały warunku (5.10) przypisywano najgorszą rangę (9),
3. jeśli którakolwiek z trajektorii odpowiadająca testowanym parametryzacjom nie spełniała warunku (5.10), ten przypadek nie był brany pod uwagę w określaniu średniej rangi dla żadnej z parametryzacji.

Dodatkowo kolumna o nazwie *nc* zawiera informację jaki procent wygenerowanych trajektorii nie spełniło warunku (5.10). W tab. 5.5 przedstawiono średnie długości ścieżek, wartości funkcji jakości i odległości $\|\boldsymbol{q}_0 + \Delta\boldsymbol{q} - \boldsymbol{q}_{freal}\|$. W tym przypadku numery $1, 2$ dodane do nazwy kolumny należy rozumieć jako:

1. średnia ze wszystkich trajektorii,
2. średnia z trajektorii spełniających warunek (5.10).

Na rys. 5.2 przedstawiono rzut na płaszczyznę xy ścieżek wygenerowanych dla różnych parametryzacji w przykładowym kierunku $\Delta\boldsymbol{q} = (0.026, 0.097, 0)^T$. Kolumny tablicy rysunków są uszeregowane w następujący sposób:

Tabela 5.3 Uśrednione rangi z 500 losowych przemieszczeń na odległości $\delta = 0.1$ dla testowanych parametryzacji i funkcji jakości.

$w(\boldsymbol{u}(\cdot))$	nr par.	długość1	długość2	długość3	jakość1	jakość2	jakość3	nc [%]
0	#1	6.7	5.0	6.3	—	—	—	0.0
	#2	1.4	5.3	1.8	—	—	—	28.0
	#3	8.9	7.1	8.8	—	—	—	23.0
	#4	5.1	5.8	5.2	—	—	—	0.0
	#5	3.8	4.9	3.7	—	—	—	47.6
	#6	7.8	6.7	7.7	—	—	—	40.4
	#7	5.4	3.7	4.8	—	—	—	0.0
	#8	2.5	5.3	3.0	—	—	—	9.4
	#9	3.4	4.3	3.7	—	—	—	27.0
wzór (5.6)	#1	4.5	4.0	4.7	4.3	3.8	4.5	0.0
	#2	4.1	5.5	4.1	4.3	5.8	4.5	13.6
	#3	8.2	7.2	8.2	8.1	7.2	8.1	13.4
	#4	8.0	7.6	8.0	8.1	7.7	8.1	0.0
	#5	3.3	3.9	3.1	3.3	3.9	3.0	28.0
	#6	7.8	7.0	7.7	7.7	7.0	7.7	28.0
	#7	3.7	3.1	3.8	3.6	3.1	3.7	0.0
	#8	2.9	4.6	2.9	3.5	5.1	3.6	1.8
	#9	2.8	3.5	2.6	2.0	2.8	1.7	11.6
wzór (5.7)	#1	6.8	5.3	6.7	5.0	4.0	4.7	0.0
	#2	1.2	4.5	1.3	5.1	6.7	5.0	24.4
	#3	9.0	7.4	9.0	7.4	6.1	7.4	24.2
	#4	5.4	6.0	5.6	7.5	7.6	7.5	0.0
	#5	3.5	4.6	3.6	2.9	4.4	3.1	41.4
	#6	7.9	6.8	7.8	6.3	5.6	6.4	38.0
	#7	5.7	4.2	5.5	4.3	3.5	4.3	0.0
	#8	2.2	4.8	2.3	4.3	6.0	4.3	6.8
	#9	3.4	4.4	3.2	2.2	3.9	2.4	23.6

- pierwsza kolumna obrazuje ścieżkę parametryzacji zawierającej wszystkie harmoniczne w obu sterowaniach,
- druga i trzecia kolumna ilustrują ścieżki dla parametryzacji będących podzbiorami właściwymi parametryzacji z pierwszej kolumny.

Pierwszy/drugi/trzeci wiersz rys. 5.2 zawierają wyniki dla parametryzacji zawierającej odpowiednio {0,1}/{0,2}/{0,1,2} harmoniczne.

Na podstawie tab. 5.3,5.4,5.5 oraz rys. 5.2 można sformułować następujące wnioski:

- pod względem długości ścieżki i dokładności osiągnięcia punktu końcowego, najlepszą jest energetyczna funkcja jakości (5.6),
- wartość funkcji jakości bazującej na wyrażeniach $\boldsymbol{\alpha}$ (5.7) nie zależy od parametryzacji sterowań,

Tabela 5.4 Uśrednione rangi z 500 losowych przemieszczeń na odległości $\delta = 0.5$ dla testowanych parametryzacji i funkcji jakości.

$w(\boldsymbol{u}(\cdot))$	nr par.	długość1	długość2	długość3	jakość1	jakość2	jakość3	nc [%]
0	#1	5.8	5.8	6.2	—	—	—	51.0
	#2	1.6	8.0	2.7	—	—	—	75.8
	#3	8.8	6.6	8.3	—	—	—	75.8
	#4	6.0	7.9	5.8	—	—	—	53.8
	#5	3.7	7.6	3.4	—	—	—	85.6
	#6	7.8	7.6	7.0	—	—	—	83.8
	#7	5.5	6.2	5.1	—	—	—	50.8
	#8	2.6	8.0	3.6	—	—	—	63.6
	#9	3.3	7.4	3.0	—	—	—	75.2
wzór (5.6)	#1	5.1	6.1	6.2	5.7	6.3	6.3	51.0
	#2	4.7	7.9	4.3	5.7	8.0	4.8	71.8
	#3	8.2	6.8	8.3	6.7	6.5	8.6	69.8
	#4	8.1	7.7	7.4	6.6	7.8	7.5	52.8
	#5	2.3	7.0	2.2	2.9	7.1	2.3	80.0
	#6	6.8	7.5	6.2	6.3	7.4	5.8	78.2
	#7	3.9	5.8	4.3	4.7	6.0	5.0	50.6
	#8	3.6	7.6	3.6	4.6	7.7	3.8	66.0
	#9	2.3	6.8	2.4	1.8	6.6	1.0	68.8
wzór (5.7)	#1	5.6	5.8	6.2	5.6	5.9	5.4	50.8
	#2	1.9	8.0	2.7	5.5	8.3	4.7	75.8
	#3	8.7	6.6	8.2	8.0	6.5	7.3	75.2
	#4	6.2	7.8	5.6	8.0	8.1	7.1	51.2
	#5	3.7	7.6	3.4	2.5	7.5	3.6	84.8
	#6	7.8	7.6	7.2	6.3	7.2	6.0	83.0
	#7	5.4	6.0	5.2	3.8	5.7	4.2	50.6
	#8	2.4	7.9	3.3	3.8	8.1	4.0	63.0
	#9	3.4	7.4	3.0	1.5	7.2	2.6	74.4

- dla energetycznej funkcji jakości (5.6) najlepsze rezultaty pod względem długości ścieżki oraz efektywności energetycznej otrzymano dla parametryzacji zawierającej dwie pełne harmoniczne $(1, 2, 3, 4, 5)(1, 2, 3, 4, 5)$, czyli najbardziej złożonej obliczeniowo. Parametryzacje zawierające przynajmniej jedną harmoniczną pierwszego rzędu dają jednak tylko minimalnie pogorszenie rezultatów. Najgorzej wypadają parametryzacje zawierające jedynie harmoniczne drugiego rzędu.

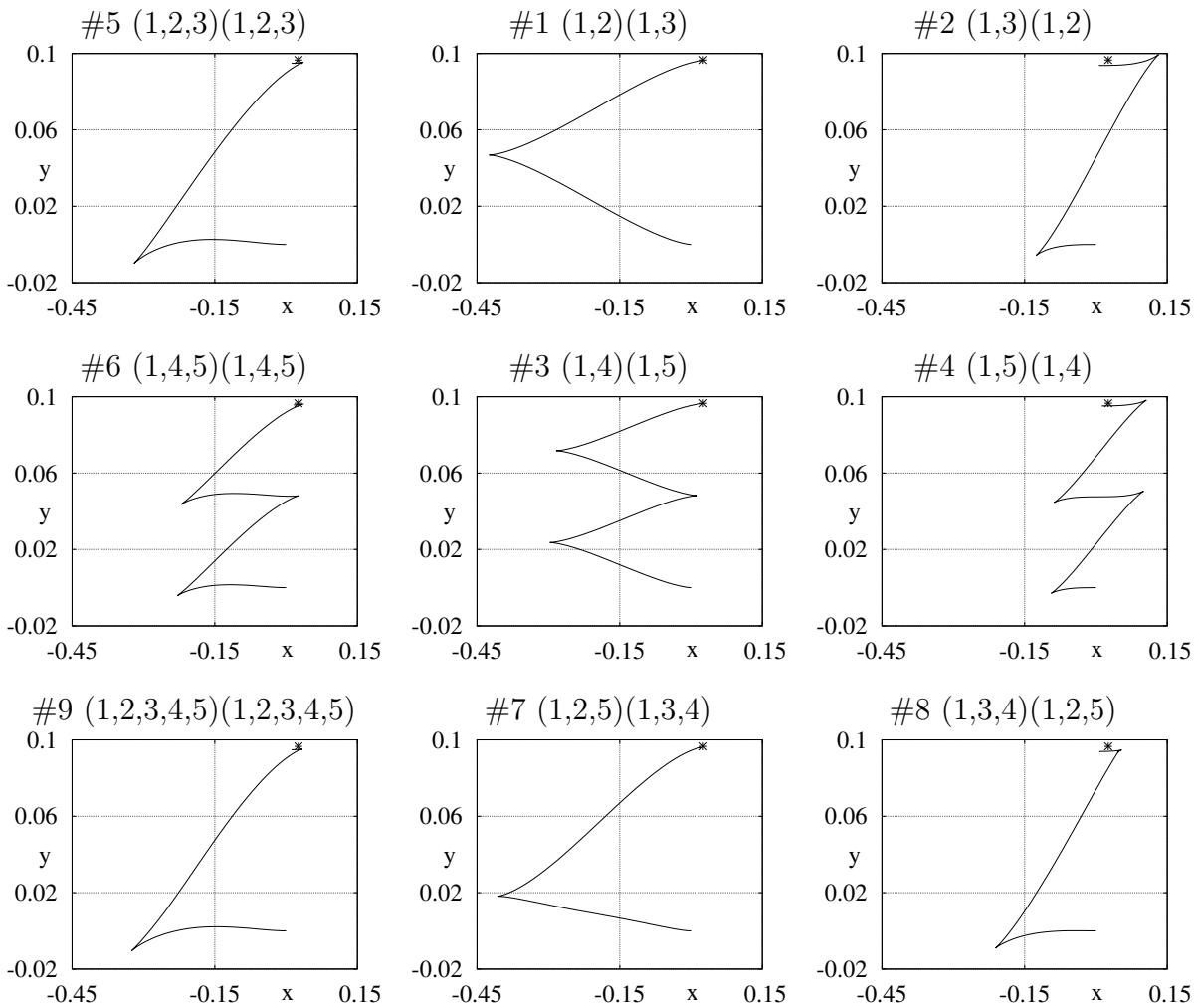
- Dla przypadków wykorzystujących funkcję jakości (5.7) jak również z pominięciem funkcji jakości, wynikowe długości ścieżek były zbliżone, minimalnie lepsze dla funkcji jakości (5.7). W obu przypadkach najgorszą pod tym względem okazała się parametryzacja typu $(1, 4)(1, 5)$, a najlepszą $(1, 3)(1, 2)$, co zostało jednak okupione sporym procentem trajektorii nie spełniających warunku na bliskość punktów końcowych (5.10) oraz istotnym średnim błędem osiągnięcia punktu końcowego. Zaskakująco, parame-

Tabela 5.5 Uśrednione długości ścieżki, wartości funkcji jakości i odległości $\|\boldsymbol{q}_0 + \Delta\boldsymbol{q} - \boldsymbol{q}_{freal}\|$ dla testowanych parametryzacji i funkcji jakości.

$w(\boldsymbol{u}(\cdot))$	nr par.	śr. długość1	śr. długość2	śr. jakość1	śr. jakość2	śr. błąd punktu końcowego
0	#1	0.6493	0.4843	—	—	0.0019
	#2	0.3668	0.2934	—	—	0.0101
	#3	0.8896	0.7019	—	—	0.0019
	#4	0.5343	0.4016	—	—	0.0070
	#5	0.5020	0.3709	—	—	0.0067
	#6	0.7202	0.5459	—	—	0.0051
	#7	0.5914	0.4248	—	—	0.0040
	#8	0.4204	0.3414	—	—	0.0092
	#9	0.4965	0.3716	—	—	0.0062
wzór (5.6)	#1	0.4871	0.4227	0.6500	0.4933	0.0021
	#2	0.4828	0.4181	0.6433	0.4873	0.0071
	#3	0.6890	0.5980	1.2918	0.9773	0.0021
	#4	0.6844	0.5931	1.2824	0.9688	0.0052
	#5	0.4634	0.3934	0.6006	0.4362	0.0051
	#6	0.6684	0.5723	1.2334	0.9089	0.0039
	#7	0.4824	0.4166	0.6479	0.4906	0.0022
	#8	0.4797	0.4141	0.6412	0.4847	0.0070
	#9	0.4608	0.3903	0.5954	0.4303	0.0051
wzór (5.7)	#1	0.6145	0.4897	0.0199	0.0140	0.0020
	#2	0.3823	0.3057	0.0199	0.0140	0.0091
	#3	0.8601	0.6941	0.0199	0.0141	0.0020
	#4	0.5484	0.4338	0.0199	0.0141	0.0065
	#5	0.5004	0.3908	0.0199	0.0140	0.0065
	#6	0.7153	0.5668	0.0199	0.0141	0.0048
	#7	0.5836	0.4579	0.0199	0.0140	0.0035
	#8	0.4050	0.3298	0.0199	0.0140	0.0086
	#9	0.4982	0.3859	0.0199	0.0140	0.0065

tryzacje generujące najdłuższą średnią ścieżkę ruchu okazały się tymi, których punkty końcowe trajektorii $\boldsymbol{q}_{freal}$ znajdują się najbliżej oczekiwanych.

- W przypadku braku harmonicznych niskiego rzędu, należy oczekiwać większej liczby zwrotów w ścieżkach wynikowych. Zwroty pojawiają się w miejscach, gdzie sterowania zmieniają znak, a w przypadku wyższych harmonicznych jest to przypadek częstszy. Również obszerność ścieżki w przestrzeni konfiguracyjnej prawdopodobnie będzie mniejsza niż w przypadku parametryzacji zawierającej harmoniczne niższego rzędu. Ta pożądana cecha może mieć praktyczne zastosowanie w zadaniach planowania ruchu w środowiskach kolizyjnych.
- Dla modelu jednokołowca można zaproponować uzasadnienie kształtu uzyskanych ścieżek. Bazując na kształcie ścieżek uzyskanych przy parametryzacjach #1, #2 oraz #7, #8 różnice w długości ścieżek oraz dokładności punktu końcowego są łatwe do wytłumacze-



Rysunek 5.2 Rzut na płaszczyznę xy ścieżek w kierunku $\Delta\boldsymbol{q} = (0.026, 0.097, 0)^T$ z punktu $\boldsymbol{q}_0 = \mathbf{0}$ dla funkcji jakości (5.7) i wszystkich parametryzacji. Gwiazdką zaznaczono planowany punkt końcowy ruchu.

nia. W trajektoriach uzyskanych za pomocą parametryzacji #2 i #8 robot najbardziej zmienia orientację (i kierunek ruchu) na początku trajektorii (w punkcie bliskim $\boldsymbol{q}_0$, więc przybliżenie kształtu pól wektorowych przez szereg $\boldsymbol{z}(t, \boldsymbol{q}_0, \boldsymbol{u}(\boldsymbol{p}))$ jest dobre), oraz na końcu, w okolicach $\boldsymbol{q}_f$ (gdzie wspomniane przybliżenie jest obarczone zauważalnym błędem). Dla parametryzacji #1 i #7 robot mocno zmienia swoją orientację i kierunek ruchu raz najpóźniej w połowie ścieżki (więc przy wciąż niezłym przybliżeniu kształtu pól wektorowych w tym punkcie przez szereg $\boldsymbol{z}(t, \boldsymbol{q}_0, \boldsymbol{u}(\boldsymbol{p}))$). Wynika z tego, że robot przy parametryzacjach #2 i #8 szybciej przeorientuje się w kierunku punktu końcowego, co implikuje krótszą ścieżkę, jednak kolejne przeorientowanie się w pobliżu punktu końcowego będzie obarczone błędem istotnie wpływającym na niedokładność ruchu.

- Najbezpieczniejszą strategią wyboru parametryzacji sterowań wykorzystując bazę harmoniczną jest wybór parametryzacji redundantnej zakładającej pełne harmoniczne do właściwego rzędu. Wymiarowość wektora $\boldsymbol{p}$ musi być wszak większa lub równa wymiarowości wektora wyrażeń $\boldsymbol{\alpha}$. Niestety taka strategia może okazać się kosztowna obliczeniowo. W celu uzyskania trajektorii o lepszych właściwościach polecana jest optymalizacja energii sterowań, przez zastosowanie algorytmu Newtona z optymalizacją w przestrzeni

zerowej energetycznej funkcji jakości.

Rozdział 6

Sterowalność w przestrzeni zadaniowej

Sterowalność jest pożądaną cechą każdego układu sterowania określającą, czy istnieją sterowania dopuszczalne przeprowadzające układ między dowolnymi punktami w przestrzeni konfiguracyjnej. W sekcji 2.1.1 omówiono sterowalność bezdryfowego układu sterowania (2.1) oraz jej mocniejszą wersję nazywaną sterowalnością w krótkim czasie (STLC).

W wielu zastosowaniach praktycznych wektor konfiguracji zawiera elementy wpływające, przez model, na planowanie ruchu, jednocześnie nie będąc istotnymi z punktu widzenia zadania dla robota. Przykładowo: współrzędne położenia kół samochodu kinematycznego niekoniecznie są istotne dla zadania planowania ruchu obiektu, jednakże występują w równaniach generujących jego trasę. W niniejszym rozdziale zdefiniowano odpowiednik sterowalności w krótkim czasie dla układów będących rozszerzeniem klasycznych bezdryfowych układów sterowania o funkcję wyjścia. Funkcja wyjścia jest odwzorowaniem pomiędzy przestrzenią konfiguracyjną $\mathbb{Q}$ a zadaniową $\mathbb{X}$, najczęściej mniej wymiarową. Przykładem może być projekcja wektora konfiguracji na przestrzeń zadaniową pomijająca współrzędne położenia kół. Przeniesienie sterowalności w krótkim czasie (Q-STLC) do przestrzeni zadaniowej (X-STLC) bazuje na pewnych algebraicznych konstruktach oraz uogólnionej formule Campbella-Bakera-Hausdorffa-Dynkina (gCBHD, paragraf 2.2.4.2). W dalszych podrozdziałach wyjaśniono praktyczne zastosowanie X-STLC w doborze sterowań, sekcja 6.4, oraz pokazano przykłady obliczeniowe. Przeanalizowano również osobliwości odwzorowania Q-STLC na X-STLC, podrozdział 6.3, oraz porównano otrzymaną metodę z metodą endogenicznej przestrzeni konfiguracyjnej, sekcja 6.5. Na rys. 6.1 przedstawiono transformacje pomiędzy przestrzeniami parametryzacji $\mathbb{P}$, sterowań $\mathbb{U}$, konfiguracyjną $\mathbb{Q}$ oraz zadaniową $\mathbb{X}$.

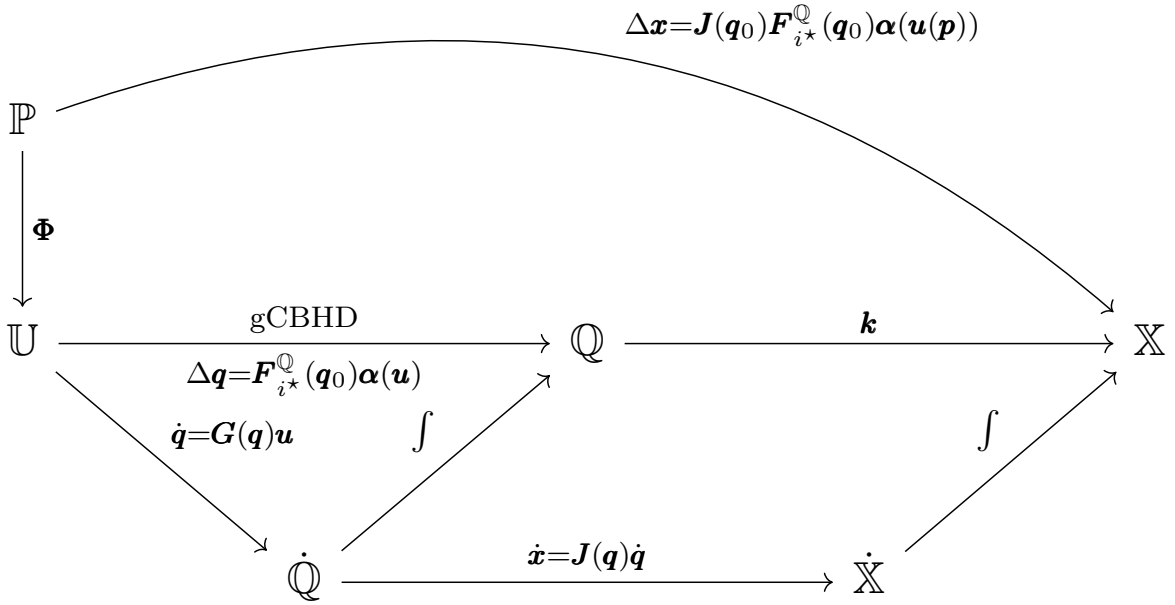
6.1 Nieholonomiczny, bezdryfowy układ sterowania z funkcją wyjścia

Nieholonomiczny, bezdryfowy układ sterowania (2.1) poszerzony o funkcję wyjścia $\boldsymbol{x} = \boldsymbol{k}(\boldsymbol{q})$ opisuje układ równań

$$\begin{cases} \dot{\boldsymbol{q}}(t) = \boldsymbol{G}(\boldsymbol{q})\boldsymbol{u} = \sum\limits_{i=1}^{m} \boldsymbol{g}_i(\boldsymbol{q})u_i = \sum\limits_{i=1}^{m} \boldsymbol{X}_i(\boldsymbol{q})u_i, \\ \quad \boldsymbol{x} = \boldsymbol{k}(\boldsymbol{q}), \end{cases} \tag{6.1}$$

gdzie

$$\boldsymbol{q} \in \mathbb{Q}, \quad \boldsymbol{x} \in \mathbb{X}, \quad \boldsymbol{u} \in \mathbb{U}, \quad n = \dim \mathbb{Q}, \quad r = \dim \mathbb{X}, \quad m = \dim \mathbb{U}, \quad n \geqslant r, \quad n > m.$$



Rysunek 6.1 Przestrzenie i transformacje między nimi

6.2 Sterowalność w krótkim czasie w przestrzeni zadaniowej (X-STLC)

Pola wektorowe $\boldsymbol{g}_i(\boldsymbol{q})$ składające się na macierz $\boldsymbol{G}(\boldsymbol{q})$ mają fizyczną interpretację prędkości skalowanych sterowaniami. Można je więc przetransferować z przestrzeni stycznej do przestrzeni konfiguracyjnej $\mathbb{Q}$ do przestrzeni stycznej do przestrzeni zadaniowej $\mathbb{X}$ za pomocą jakobianu

$$\boldsymbol{J}(\boldsymbol{q}) = \frac{\partial \boldsymbol{k}(\boldsymbol{q})}{\partial \boldsymbol{q}}, \tag{6.2}$$

uzyskując zależność zmiany położenia w $\mathbb{X}$ od stanu $\boldsymbol{q}$ i sterowań $\boldsymbol{u}$ w postaci

$$\dot{\boldsymbol{x}} = \frac{\partial \boldsymbol{k}(\boldsymbol{q})}{\partial \boldsymbol{q}}\dot{\boldsymbol{q}} = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}} = \boldsymbol{J}(\boldsymbol{q})\boldsymbol{G}(\boldsymbol{q})\boldsymbol{u}. \tag{6.3}$$

Analogicznie do zaprezentowanych w 2.1.1 koncepcji[1], można zdefiniować macierze $\boldsymbol{F}^{\mathbb{X}}_i(\boldsymbol{q})$ i ich rzędy $f^{\mathbb{X}}_i$ w konfiguracji $\boldsymbol{q}$ jako

$$\boldsymbol{F}^{\mathbb{X}}_i(\boldsymbol{q}) = \boldsymbol{J}(\boldsymbol{q}) \cdot \boldsymbol{F}^{\mathbb{Q}}_i(\boldsymbol{q}), \quad i = 1, \ldots \tag{6.4}$$

[1]Wszystkie oznaczenia odnoszące się do przestrzeni konfiguracyjnej $\mathbb{Q}$ zaprezentowane w sekcji 2.1.1, w niniejszym rozdziale będą posiadały górny indeks $\mathbb{Q}$.

oraz

$$f_i^{\mathbb{X}}(\boldsymbol{q}) = \operatorname{rank}(\boldsymbol{F}_i^{\mathbb{X}}(\boldsymbol{q})), \quad i = 1, \ldots. \tag{6.5}$$

Wykorzystując $\boldsymbol{F}_i^{\mathbb{X}}$ i stosując zasadę analogii, można przenieść warunek sterowalności w krótkim czasie Q-STLC (2.7) do przestrzeni zadaniowej otrzymując warunek X-STLC w konfiguracji $\boldsymbol{q}$

$$\exists\, \hat{p}(\boldsymbol{q}) \in \mathbb{N}: \quad f_{\hat{p}(\boldsymbol{q})}^{\mathbb{X}}(\boldsymbol{q}) = r, \tag{6.6}$$

oraz zdefiniować warunek sterowalności w krótkim czasie dla przestrzeni zadaniowej:

Definicja 3 *Układ* (6.1) *jest X-STLC jeśli jest X-STLC w każdej konfiguracji* $\boldsymbol{q} \in \mathbb{Q}$.

Liczba $\hat{p}(\boldsymbol{q})$, o ile istnieje, musi być minimalna. Minimalną (maksymalną) wartość $\hat{p}$ dla punktów z przestrzeni konfiguracyjnej nazywamy minimalną (maksymalną) X-warstwą i definiujemy następująco

$$\hat{p}_{\min} = \min_{\boldsymbol{q}\in\mathbb{Q}} \hat{p}(\boldsymbol{q}), \quad \hat{p}_{\max} = \max_{\boldsymbol{q}\in\mathbb{Q}} \hat{p}(\boldsymbol{q}), \quad \hat{p}_{\min} \leqslant \hat{p}_{\max}. \tag{6.7}$$

Jeśli $\hat{p}_{\min} = \hat{p}_{\max} = \hat{p}$, liczbę $\hat{p}$ nazywamy stopniem nieholonomiczności przestrzeni zadaniowej. W przypadku regularnym (brak degeneracji zarówno macierzy Jacobiego $\boldsymbol{J}(\boldsymbol{q})$ jak i pól utworzonych z generatorów układu zachodzi $\hat{p}_{\min} = \hat{p}_{\max}$. W typowym przypadku zdegenerowanym jest natomiast $\hat{p}_{\max} - \hat{p}_{\min} = 1$. Różnica $\hat{p}_{\max} - \hat{p}_{\min}$ może być jednak dowolnie duża, zgodnie z twierdzeniem 2.

Twierdzenie 2 *Dla dowolnej liczby naturalnej* $j \in \mathbb{N}$ *istnieje układ nieholonomiczny z funkcją wyjścia* (6.1) *dla którego różnica pomiędzy maksymalną i minimalną X-warstwą wynosi* j.

$$j = \hat{p}_{max} - \hat{p}_{min} \tag{6.8}$$

Dowód 2 *Weźmy dwuwejściowy układ nieholonomiczny*

$$\dot{\boldsymbol{q}} = \boldsymbol{g}_1 u_1 + \boldsymbol{g}_2(\boldsymbol{q}) u_2 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} u_1 + \begin{pmatrix} 0 \\ q_1^j \end{pmatrix} u_2 \tag{6.9}$$

z funkcją wyjścia $\boldsymbol{k}(\boldsymbol{q}) = q_2$ *i jej jakobianem* $\boldsymbol{J}(q) = (0, 1)$ *stałym dla każdego* $\boldsymbol{q}$. *Macierz* $\boldsymbol{F}_i^{\mathbb{X}}$ *powstała ze wzoru 6.4 zawierać będzie więc jedynie drugi wiersz macierzy* $\boldsymbol{F}_i^{\mathbb{Q}}$.

Dla $i = 1$ *kolumnami* $\boldsymbol{F}_i^{\mathbb{Q}}$ *są jedynie generatory*

$$\boldsymbol{F}_1^{\mathbb{Q}} = \begin{pmatrix} 1 & 0 \\ 0 & q_1^j \end{pmatrix}, \quad \boldsymbol{F}_1^{\mathbb{X}} = \begin{pmatrix} 0 & q_1^j \end{pmatrix}. \tag{6.10}$$

Dla warstw $i > 1$ *w dowodzie wykorzystamy bazę Chibrikova wspomnianą w sekcji 1.2.2. Jest to unormowana baza prawostronna [10], tj. jej elementy są postaci:*

$$[\boldsymbol{g}_{k_1}, [\boldsymbol{g}_{k_2}, [\ldots, [\boldsymbol{g}_{k_{i-1}}, \boldsymbol{g}_{k_i}] \ldots]]]. \tag{6.11}$$

Oznacza to, że dla układu (6.9) *elementy kolejnych warstw i przyjmują jedną z form*

$$[\boldsymbol{g}_1, \boldsymbol{C}_d^{i-1}], \tag{6.12}$$

$$[\boldsymbol{g}_2, \boldsymbol{C}_d^{i-1}], \tag{6.13}$$

gdzie $\boldsymbol{C}_d^i$ *oznacza d-ty element i-tej warstwy bazy Chibrikova.*

W przypadku formy (6.12) *wynik nawiasu jest niezerowy jedynie jeśli* $\boldsymbol{C}_d^{i-1}$ *zależy od* q_1. *W przypadku formy* (6.13) *wynik nawiasu jest niezerowy jedynie jeśli pierwsza współrzędna* $\boldsymbol{C}_d^{i-1}$ *jest niezerowa.*
Dla $i = 2$ *jedynym elementem bazy Chibrikova jest*

$$[\boldsymbol{g}_1, \boldsymbol{g}_2] = (0, jq_1^{j-1})^T \tag{6.14}$$

Wynika z tego, że jedynym niezerowym elementem warstwy trzeciej jest

$$[\boldsymbol{g}_1, [\boldsymbol{g}_1, \boldsymbol{g}_2]] = (0, (j-1)jq_1^{j-2})^T \tag{6.15}$$

Wykorzystując indukcje można pokazać, że jedyny możliwy niezerowy element i-tej warstwy (dla $i > 1$*) ma postać*

$$\begin{array}{ll} ad_{\boldsymbol{g}_1}^{i-1}\boldsymbol{g}_2 = \left(0, \frac{j!}{(j+1-i)!}q_1^{j+1-i}\right)^T & dla\ i \leqslant j+1, \\ ad_{\boldsymbol{g}_1}^{i-1}\boldsymbol{g}_2 = \boldsymbol{0} & dla\ i > j+1. \end{array} \tag{6.16}$$

Notacja $ad_{\boldsymbol{X}}^i\boldsymbol{Y}$ *użyta we wzorze* (6.16) *została wprowadzona na stronie 25. Macierz* $\boldsymbol{F}_{j+1}^{\mathbb{X}}$ *po odrzuceniu zerowych kolumn przybiera formę*

$$\boldsymbol{F}_{j+1}^{\mathbb{X}} = (q_1^j, jq_1^{j-1}, (j-1)jq_1^{j-2}, \ldots, \frac{j!}{(j-k)!}q_1^{j-k}, \ldots, j!q_1^0), \quad k \in \{0, 1, ...j\}. \tag{6.17}$$

Dla $q_1 \neq 0$ *rząd macierzy* $\boldsymbol{F}_1^{\mathbb{X}}$ *jest niezerowy, więc* $\hat{p}_{min} = 1$, *natomiast dla* $q_1 = 0$ *niezerowy jest dopiero rząd macierzy* $\boldsymbol{F}_{j+1}^{\mathbb{X}}$, *więc* $\hat{p}_{max} = j + 1$. *Różnica* $\hat{p}_{max} - \hat{p}_{min} = j$, *co należało udowodnić.* □

Warto zauważyć, że mimo określania sterowalności w przestrzeni zadaniowej wartościowanie pól wektorowych odbywa się w przestrzeni konfiguracyjnej. Przy próbie wykluczenia konfiguracji $\boldsymbol{q}$ z macierzowego warunku na sterowalność otrzymujemy

$$\forall \boldsymbol{x} \in \mathbb{X} \quad \exists i^\star \quad \operatorname{rank}(\boldsymbol{J}(\boldsymbol{k}^{-1}(\boldsymbol{x}))\boldsymbol{F}_{i^\star}^{\mathbb{Q}}(\boldsymbol{k}^{-1}(\boldsymbol{x})) = r \tag{6.18}$$

i pojawia się problem niejednoznaczności przekształcenia $\boldsymbol{k}^{-1}(\boldsymbol{x})$ odwrotnego do $\boldsymbol{k}(\boldsymbol{q})$.

Istnieją teoretycznie przynajmniej dwa sposoby uniezależnienia definicji własności określanych dla przestrzeni zadaniowej od konfiguracji. Załóżmy, że badamy własność w punkcie $\boldsymbol{x}^\star \in \mathbb{X}$. Najpierw definiujemy przeciwobraz $\boldsymbol{x}^\star$

$$\mathbb{Q}_{\boldsymbol{x}^\star} = \{\boldsymbol{q} : \boldsymbol{k}(\boldsymbol{q}) = \boldsymbol{x}^\star\} \subset \mathbb{Q}. \tag{6.19}$$

Pierwszy sposób (słaba wersja warunku X-STLC dla $\boldsymbol{x}^\star$) zakłada istnienie pewnej konfiguracji $\boldsymbol{q}^\star \in \mathbb{Q}_{\boldsymbol{x}^\star}$, dla której definiowana własność zachodzi $\hat{p}(\boldsymbol{q}^\star)$

$$\exists \hat{p}(\boldsymbol{q}^\star): \quad f_{\hat{p}}^{\mathbb{X}}(\boldsymbol{q}^\star) = r. \tag{6.20}$$

Drugi sposób (mocna wersja X-STLC dla $\boldsymbol{x}^\star$) zakłada spełnienie własność X-STLC (6.20) w każdej konfiguracji z $\forall \boldsymbol{q}^\star \in \mathbb{Q}_{\boldsymbol{x}^\star}$. Żaden ze sposobów nie jest uniwersalny, czy też łatwo stosowalny.

6.3 Konfiguracje osobliwe układu z funkcją wyjścia

Przywołajmy na początek kilka pojęć i definicji odnoszących się do manipulatorów, bowiem te intuicje i pojęcia chcemy przenieść do układów nieholonomicznych z wyjściem. Manipulator stacjonarny jest opisany za pomocą kinematyki na bazie, której tworzona jest macierz Jacobiego

$$\boldsymbol{x} = \boldsymbol{k}(\boldsymbol{q}), \quad \boldsymbol{J}(\boldsymbol{q}) = \frac{\partial \boldsymbol{k}}{\partial \boldsymbol{q}}, \quad \dim \mathbb{X} = r. \tag{6.21}$$

W konfiguracji osobliwej manipulatora przemieszczenie efektora w niektórych kierunkach nie jest możliwe [97]. Algebraicznie konfiguracja osobliwa jest wykrywana, gdy rząd macierzy Jacobiego jest mniejszy niż maksymalnie możliwy (równy wymiarowości przestrzeni zadaniowej). Alternatywnie i równoważnie, rząd macierzy manipulowalności $\boldsymbol{M}(\boldsymbol{q})$ jest mniejszy niż r

$$\operatorname{rank}(\boldsymbol{M}(\boldsymbol{q})) = \operatorname{rank}(\boldsymbol{J}(\boldsymbol{q})\boldsymbol{J}^T(\boldsymbol{q})) < r. \tag{6.22}$$

Dla układów nieholonomicznych taka definicja konfiguracji osobliwej jest niewystarczająca, ponieważ układ posiada jedynie m niezależnych sterowań (generatorów, bezpośrednich kierunków ruchu). Przestrzeń kierunków sterowanych bezpośrednio w konfiguracji $\boldsymbol{q}_0$ $\operatorname{span}_{\mathbb{R}}\{\boldsymbol{g}_1(\boldsymbol{q}_0), \ldots, \boldsymbol{g}_m(\boldsymbol{q}_0)\}$ jest podprzestrzenią n-wymiarowej przestrzeni konfiguracyjnej, więc w myśl powyższej definicji konfiguracji osobliwej wszystkie konfiguracje układu nieholonomicznego dla którego $m < r$ są osobliwe. Ruch w kierunkach nie odpowiadających bezpośredniemu sterowaniu jest możliwy dla układów nieholonomicznych, ale wymaga złożenia kilku ruchów elementarnych w manewr (czego przykładem jest parkowanie samochodu). Z tego powodu definiując konfiguracje osobliwe dla układów nieholonomicznych z funkcją wyjścia należy zastosować szerszą definicję uwzględniającą wszystkie możliwe do uzyskania kierunki ruchu. Za osobliwą konfigurację układu (6.3) uznamy konfigurację $\boldsymbol{q}$, dla której rząd (zdefiniowany w (6.20)) nie osiąga wartości maksymalnej r

$$f_i^{\mathbb{X}} < r, \quad i = 1, 2, \ldots. \tag{6.23}$$

Na podstawie wyników zaczerpniętych z analizy macierzowej [34] można przytoczyć nierówność

$$\operatorname{rank}(\boldsymbol{J}(\boldsymbol{q}) \cdot \boldsymbol{F}_i^{\mathbb{Q}}(\boldsymbol{q})) \leqslant \min(\operatorname{rank}(\boldsymbol{J}(\boldsymbol{q})), \operatorname{rank}(\boldsymbol{F}_i^{\mathbb{Q}}(\boldsymbol{q}))). \tag{6.24}$$

Przyjmując założenie o nieholonomiczności układu (2.1) można zauważyć, że dla pewnego $p^\star$ istnieje macierz $\boldsymbol{F}_{p^\star}^{\mathbb{Q}}(\boldsymbol{q})$ pełnego rzędu. Dla tego $p^\star$ można więc zastąpić równanie (6.24) następującym

$$\operatorname{rank}(\boldsymbol{J}(\boldsymbol{q}) \cdot \boldsymbol{F}_{p^\star}^{\mathbb{Q}}(\boldsymbol{q})) = \operatorname{rank}(\boldsymbol{J}(\boldsymbol{q})). \tag{6.25}$$

Konsekwencje zależności (6.25) są następujące:

(1) jedynym elementem układu (6.1) mogącym wprowadzić konfiguracje osobliwe jest funkcja wyjścia $\boldsymbol{k}(\boldsymbol{q})$.

(2) spełnienie Q-STLC dla układu (2.1) nie implikuje spełnienia X-STLC dla układu (6.1). Gwarantuje jednak spełnienie X-STLC we wszystkich nieosobliwych konfiguracjach odwzorowania $\boldsymbol{k}(\boldsymbol{q})$.

Szczególną i mającą praktyczne zastosowanie podklasą funkcji wyjścia są projekcje, które wybierają pewien właściwy podzbiór ($r < n$) zmiennych z wektora konfiguracji $\boldsymbol{q}$. Dla tej podklasy macierz Jacobiego $\boldsymbol{J}(\boldsymbol{q})$ jest stałą o elementach równych 0 lub 1 i ma pełen

rząd równy r. W tym przypadku istnienie Q-STLC układu (2.1) pociąga za sobą istnienie X-STLC układu (6.1), a macierz $\boldsymbol{F}_i^{\mathbb{X}}(\boldsymbol{q})$ jest złożona z wybranych wierszy macierzy $\boldsymbol{F}_i^{\mathbb{Q}}(\boldsymbol{q})$.

Podobnie jak w przypadku manipulatorów, dla układu (6.1) konfiguracje osobliwe można uzyskać z macierzy manipulowalności $\boldsymbol{M}_i^{\mathbb{X}}(\boldsymbol{q})$

$$\boldsymbol{M}_i^{\mathbb{X}}(\boldsymbol{q}) = \boldsymbol{J}(\boldsymbol{q})\boldsymbol{M}_i^{\mathbb{Q}}(\boldsymbol{q})\boldsymbol{J}^T(\boldsymbol{q}), \tag{6.26}$$

gdzie

$$\boldsymbol{M}_i^{\mathbb{Q}}(\boldsymbol{q}) = \boldsymbol{F}_i^{\mathbb{Q}}(\boldsymbol{q})(\boldsymbol{F}_i^{\mathbb{Q}}(\boldsymbol{q}))^T. \tag{6.27}$$

6.4 Przykłady obliczeniowe

Przykłady opracowano z wykorzystaniem metody Lie-algebraicznej opisanej w sekcji 2.2.4 i bazującej na formule gCBHD (paragraf 2.2.4.2). Wykorzystano również harmoniczną parametryzację sterowań (paragraf 2.2.4.2).

Rozważmy bezdryfowy układ sterowania (2.1) o dwóch wejściach $m = 2$, ze sterowaniami harmonicznymi postaci

$$u_1 = a_1 + a_2\sin(\omega t) + a_3\cos(\omega t), \qquad u_2 = b_1 + b_2\sin(\omega t) + b_3\cos(\omega t), \tag{6.28}$$

gdzie $\omega = 2\pi/T$, T jest małym czasem ruchu, a wektor parametrów sterowań jest równy $\boldsymbol{p} = (a_1, a_2, a_3, b_1, b_2, b_3)^T$. Po zastosowaniu formuły gCBHD otrzymujemy

$$\Delta\boldsymbol{q} \simeq \boldsymbol{F}_\infty(\boldsymbol{q})\boldsymbol{\alpha}(\boldsymbol{p}) = \Big(\boldsymbol{g}_1, \boldsymbol{g}_2, [\boldsymbol{g}_1, \boldsymbol{g}_2], [\boldsymbol{g}_1, [\boldsymbol{g}_1, \boldsymbol{g}_2]], [\boldsymbol{g}_2, [\boldsymbol{g}_1, \boldsymbol{g}_2], \ldots]\Big)\Big(\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \ldots\Big)^T. \tag{6.29}$$

Dla małych T, dominujące początkowe współczynniki wektora $\boldsymbol{\alpha}(\boldsymbol{p})$ są następujące

$$\begin{aligned}
&\alpha_1 = a_1,\\
&\alpha_2 = b_1,\\
&\alpha_3 = (2a_2b_1 - 2a_1b_2 + a_3b_2 - a_2b_3)/(4\pi),\\
&\alpha_4 = \{a_2b_2(-3a_1 + 2a_3) + a_2^2(3b_1 - 2b_3) + (4a_1 - a_3)(a_1b_3 - a_3b_1)\}/(16\pi^2),\\
&\alpha_5 = \{-a_2b_2(-3b_1 + 2b_3) - b_2^2(3a_1 - 2a_3) + (4b_1 - b_3)(a_1b_3 - a_3b_1)\}/(16\pi^2).
\end{aligned} \tag{6.30}$$

Kwestia optymalnego doboru wektora parametryzacji $\boldsymbol{p}$ przestrzeni sterowań została poruszona w rozdziale 5. W kontekście układów z funkcją wyjścia interesującym zagadnieniem jest znalezienie minimalnego zestawu parametrów przestrzeni sterowań zapewniającego sterowalność X-STLC.

Przykład 6. X-STLC oraz minimalny zestaw parametrów $\boldsymbol{p}$ zapewniający X-STLC dla jednokołowca.

Jak pokazano w dodatku A.1, pola wektorowe $\boldsymbol{g}_1, \boldsymbol{g}_2, [\boldsymbol{g}_1, \boldsymbol{g}_2]$ zapewniają sterowalność Q-STLC i nieholonomiczność układu dla $i^\star = 2$ (przykład 3).

Dla trywialnej funkcji wyjścia $\boldsymbol{k}(\boldsymbol{q}) = q_3$, pierwsza warstwa jest wystarczająca do spełnienia warunku X-STLC, $\hat{p} = 1$. Minimalnym zestawem parametrów sterowania realizujący ruch w przestrzeni zadaniowej jest $\boldsymbol{p} = (b_1)^T$.

Dla funkcji wyjścia opisującej położenie (bez orientacji) $\boldsymbol{k}(\boldsymbol{q}) = (q_1, q_2)^T$ do uzyskania X-STLC konieczne są już pola $\boldsymbol{g}_1, \boldsymbol{g}_2, [\boldsymbol{g}_1, \boldsymbol{g}_2]$, czyli $\hat{p} = 2$. Minimalne wektory

parametrów sterowań dla dowolnego ruchu są następujące

$$\begin{aligned} &\boldsymbol{p} = (a_1, b_1, b_2)^T && \text{gdy} \quad a_1 \neq 0, \\ &\boldsymbol{p} = (a_1, b_1, a_2)^T && \text{gdy} \quad b_1 \neq 0, \\ &\boldsymbol{p} = (a_2, b_3)^T \quad \text{lub} \quad \boldsymbol{p} = (a_3, b_2)^T && \text{gdy} \quad b_1 = a_1 = 0. \end{aligned} \tag{6.31}$$

Z (6.31) wynika, że istnieją trzy minimalne zestawy parametrów sterowań realizujące lokalnie ruch w dowolnym kierunku. W obszarach przestrzeni konfiguracyjnej wymagających by $a_1 \simeq b_1 \simeq 0$ można zastosować dwie pierwsze parametryzacje, jednak wartości pozostałych parametrów będą najpewniej bardzo duże, przez co praktycznie fizycznie nierealizowalne. Z tego właśnie powodu w zadaniach praktycznych zaleca się małą redundancję w wektorze parametrów nie powodującą nadmiernego wzrostu złożoności obliczeniowej, a ułatwiającej rozwiązanie zadania planowania.

Dla funkcji wyjścia $\boldsymbol{k}(\boldsymbol{q}) = (q_1, q_3)^T$ pola wektorowe $\boldsymbol{g}_1, \boldsymbol{g}_2$ spełniają warunek X-STLC dla $q_3 \neq \pi/2 + k\pi$. Poza tym obszarem do spełnienia X-STLC konieczne jest również pole $[\boldsymbol{g}_1, \boldsymbol{g}_2]$, więc $\hat{p}_{min} = 1$, $\hat{p}_{max} = 2$.

W przypadku funkcji wyjścia $\boldsymbol{k}(\boldsymbol{q}) = (q_1 \cos(q_3) + q_2 \sin(q_3))^T$ posiadającej nieosobliwy jakobian, otrzymujemy $\hat{p}_{min} = \hat{p}_{max} = \hat{p} = 1$ i warstwa złożona jedynie z generatorów jest wystarczająca do spełnienia X-STLC.

Powyższe przykłady pokazują, że nie istnieje jeden minimalny zestaw sterowań pozwalający na spełnienie X-STLC w każdej konfiguracji $\boldsymbol{q}$, co utrudnia implementację algorytmów planowania ruchu przez konieczność uwzględnienia wariantowości i detekcji konieczności zmiany parametryzacji sterowań.

Przykład 7. X-STLC oraz minimalny zestaw parametrów $\boldsymbol{p}$ zapewniający X-STLC dla samochodu kinematycznego.

Jak pokazano w dodatku A.2, pola wektorowe $\boldsymbol{g}_1, \boldsymbol{g}_2, [\boldsymbol{g}_1, \boldsymbol{g}_2], [\boldsymbol{g}_1, [\boldsymbol{g}_1, \boldsymbol{g}_2]]$ zapewniają sterowalność Q-STLC i nieholonomiczność układu dla $i^\star = 3$.

Dla każdej funkcji wyjścia polegającej na wyborze trójelementowego podzbioru współrzędnych $\boldsymbol{q}$, zbiór pól wektorowych $\boldsymbol{g}_1, \boldsymbol{g}_2, [\boldsymbol{g}_1, \boldsymbol{g}_2]$ (czyli $\boldsymbol{F}_2^{\mathbb{Q}}$) nie spełnia warunku X-STLC. Zatem dla takich funkcji wyjścia należy również użyć pola $[\boldsymbol{g}_1, [\boldsymbol{g}_1, \boldsymbol{g}_2]]$, a liczba parametrów sterowań potrzebna do uzyskania X-STLC jest równa tej dla Q-STLC.

Dla rodziny funkcji wyjścia $\boldsymbol{k}(\boldsymbol{q}) = (\xi(q_3, q_4)\{q_1 \cos(q_3) + q_2(\sin(q_3)\}, q_3, q_4)^T$ z funkcją $\xi(q_3, q_4)$ spełniającą warunki $\xi \in \mathcal{C}^1$ oraz $\forall q_3, q_4 \quad \xi \neq 0$, do spełnienia warunku X-STLC wystarczą pola wektorowe z dwóch pierwszych warstw i $\hat{p} = 2$.

6.5 Porównanie rozszerzenia metody Lie-algebraicznej o funkcję wyjścia z metodą endogenicznej przestrzeni konfiguracyjnej

Metoda endogenicznej przestrzeni konfiguracyjnej została opisana szczegółowo w sekcji 2.2.3 (strona 22). W odróżnieniu od innych metod planowania ruchu opisanych w rozdziale 2 umożliwia planowanie z funkcją wyjścia $\boldsymbol{x} = \boldsymbol{k}(\boldsymbol{q})$ oraz znajduje również zasto-

sowanie dla ogólniejszych układów niż (6.1), dopuszczających również dryf. Porównując obie metody można zauważyć, że używają one przestrzeni sterowań $\mathbb{U}$ i konfiguracji $\mathbb{Q}$ w innych kontekstach. W metodzie bazującej na algebrze Liego przestrzeń $\mathbb{U}$ jest wejściową, a $\mathbb{Q}$ konfiguracyjną. Metoda przestrzeni endogenicznej traktuje przestrzeń $\mathbb{U}$ jako konfiguracyjną, a przestrzeń $\mathbb{Q}$ pełni rolę przestrzeni w której obserwowane są wyniki zastosowania sterowań w układzie. W konsekwencji wymiar przestrzeni konfiguracyjnej dla metody Lie-algebraicznej jest skończony i równy n, a dla metody endogenicznej – nieskończony.

Tabela 6.1 Porównanie metody Lie-algebraicznej z funkcją wyjścia z metodą endogenicznej przestrzeni konfiguracyjnej.

	Lie-algebraiczna	endogeniczna
przestrzeń wejściowa	$\mathbb{U}$	—
przestrzeń konfiguracyjna	$\mathbb{Q}$	$\mathbb{U}$
przestrzeń przekształceń	—	$\mathbb{Q}$
przestrzeń zadaniowa	$\mathbb{X}$	$\mathbb{X}$
typ metody	lokalna	globalna
czas ruchu $T > 0$	mały	dowolny
sterowania	parametryzowane	zwykle parametryzowane
ciągłość sterowań	kawałkami ciągłe	ciągłe
znajdowanie osobliwości	analitycznie	numerycznie
ruch planowany w otoczeniu	$\boldsymbol{q}_0$	$\boldsymbol{q}(T)$
złożoność obliczeniowa	mała	duża
kontrolowanie kształtu ścieżki	łatwe	trudne

Sterowania $\boldsymbol{u}(\cdot)$ realizowane w przedziale $[0, T]$ zastosowane do układu (2.1), inicjowanego w $\boldsymbol{q}_0$ pozwalają na określenie punktu końcowego $\boldsymbol{q}(T)$ wynikowej trajektorii. Przemieszczenie w otoczeniu punktu $\boldsymbol{q}(T)$ realizowane za pomocą wariacji (małej zmiany) sterowania $\boldsymbol{v}(\cdot)$ określone jest następująco [96]

$$\boldsymbol{\eta}(T, \boldsymbol{u}(\cdot), \boldsymbol{v}(\cdot)) = \boldsymbol{C}(T) \int_0^T \boldsymbol{\Phi}(T, s)\boldsymbol{B}(s)\boldsymbol{v}(s)ds. \tag{6.32}$$

Natomiast macierze manipulowalności określające osobliwości przekształcenia są następujące

$$\boldsymbol{M}^{\mathbb{U}}(T, \boldsymbol{u}(\cdot)) = \int_0^T \boldsymbol{\Phi}(T, s)\boldsymbol{B}(s)\boldsymbol{B}^T(s)\boldsymbol{\Phi}^T(T, s)ds, \tag{6.33}$$

$$\boldsymbol{M}^{\mathbb{X}}(T, \boldsymbol{u}(\cdot)) = \boldsymbol{C}(T)\boldsymbol{M}^{\mathbb{U}}\boldsymbol{C}^T(T). \tag{6.34}$$

Konfiguracje osobliwe w przestrzeni sterowań dla zadania z funkcją wyjścia pojawiają się gdy rząd macierzy $\boldsymbol{M}^{\mathbb{X}}$ jest mniejszy od maksymalnego możliwego (r). Z powyższych wzorów wynikają kolejne różnice pomiędzy podejściem Lie-algebraicznym a metodą endogeniczną. Dla metody Lie-algebraicznej osobliwości są określone w otoczeniu punktu $\boldsymbol{q}_0$, natomiast dla endogenicznej w otoczeniu punktu $\boldsymbol{q}(T)$ (i analogicznie w przestrzeni $\mathbb{X}$ w otoczeniach punktów $\boldsymbol{k}(\boldsymbol{q}_0)$ i $\boldsymbol{k}(\boldsymbol{q}(T))$ odpowiednio). Ponadto w metodzie endogenicznej jedyną konfiguracją, zawsze osobliwą, możliwą do uzyskania analitycznie jest tożsamościowo zerowe sterowanie $\boldsymbol{u}(\cdot) = \boldsymbol{0}$. Inne konfiguracje osobliwe mogą być uzyskane jedynie numerycznie, bowiem macierz fundamentalna $\boldsymbol{\Phi}(T, s)$ nie może być wyznaczona analitycz-

nie dla prawie wszystkich praktycznych układów. W metodzie Lie-algebraicznej konfiguracje osobliwe mogą być określone analitycznie. Pozostałe różnice wraz z wymienionymi powyżej zostały zebrane w tab. 6.1. Z praktycznego punktu widzenia obydwie porównywane metody są użyteczne w planowaniu ruchu układów nieholonomicznych posiadając komplementarne zalety i wady.

Rozdział 7

Sfery nieholonomiczne

W poprzednich rozdziałach wzmiankowano o zróżnicowanej trudności ruchu w kierunkach odpowiadających elementom bazy Ph. Halla o różnym stopniu. W praktycznych zastosowaniach bardziej interesującą jest określenie jakości ruchu w dowolnym kierunku w przestrzeni konfiguracyjnej czy zadaniowej, która formalnie jest opisana sferą nieholonomiczną układu (2.1) lub (6.1). Znajomość kształtu sfery nieholonomicznej może być pomocna w konstruowaniu algorytmów planowania ruchu z minimalizacją energii (szczególnie w środowisku zawierającym przeszkody) lub być wykorzystana w grafowych algorytmach planowania ruchu do właściwego doboru podcelów (wierzchołków tworzonego grafu). Pojęcie sfery nieholonomicznej dla układu nieholonomicznego może być rozumiane analogicznie jak pojęcie elipsoidy manipulowalności dla manipulatorów [97].

W niniejszym rozdziale opisano algorytm pozwalający na numeryczne przybliżenie sfery nieholonomicznej w przestrzeniach konfiguracyjnej i zadaniowej. Przybliżenie wynika z lokalności formuły gCBHD, użytej do określenia sfery. Formuła gCBHD jest nieskończonym szeregiem, który dla małego otoczenia punktu $\boldsymbol{q}_0$, w którym jest centrowana sfera w przestrzeni konfiguracyjnej, może być aproksymowany kilkoma pierwszymi elementami (spełniającymi warunek LARC). W celu spełnienia warunku małego otoczenia konfiguracji $\boldsymbol{q}_0$ założono stały, krótki czas ruchu T oraz ustaloną, niską energię sterowań $E(\boldsymbol{u}(\cdot))$ dla ograniczenia obszerności manewru. Sfera nieholonomiczna w przestrzeni zadaniowej jest obrazem sfery w przestrzeni konfiguracyjnej poprzez funkcję wyjścia $\boldsymbol{k}(\boldsymbol{q})$. W konsekwencji jest to sfera centrowana w $\boldsymbol{x}_0 = \boldsymbol{k}(\boldsymbol{q}_0)$, a elementy formuły gCBHD są liczone dla konfiguracji $\boldsymbol{q}_0$.

Dla określenia kształtu sfery nieholonomicznej zastosowano technikę kierunkowej optymalizacji, w której pierwszym krokiem jest wybranie kierunku w r-wymiarowej przestrzeni zadaniowej $\mathbb{X} \cong \mathbb{R}^r$. Następnie należy wyznaczyć najdalszy osiagalny punkt w tym kierunku dla ustalonej energii ruchu. Zbiór powstałych punktów przy uzmiennieniu wybieranych kierunków tworzy sferą nieholonomiczną układu w przestrzeni zadaniowej. Przy założeniu identycznościowej funkcji wyjścia $\boldsymbol{k}(\boldsymbol{q}) = \boldsymbol{q}$ powstaje sfera nieholonomiczna w przestrzeni konfiguracyjnej. W celu łatwego określenia dowolnego kierunku zakresu ruchu w $\mathbb{R}^r$ zastosowano współrzędne hipersferyczne określone jako

$$\begin{cases} w_1 = R\ \cos(\beta_1), \\ w_i = R\ \prod_{j=1}^{i-1} \sin(\beta_j) \cos(\beta_i), \quad i = 2, \ldots, r-1, \\ w_r = R\ \prod_{j=1}^{r-1} \sin(\beta_j), \end{cases} \tag{7.1}$$

gdzie R to odległość punktu od środka układu współrzędnych (promień hipersfery), a β_i to kąt pomiędzy rzutem wektora w_i na podprzestrzeń rozpinaną przez $\{R, \beta_1, \ldots, \beta_{i-1}\}$

a wersorem i-tej osi kartezjańskiego układu współrzędnych $e_i \in \mathbb{R}^r$. Szczegółowe kroki pozwalające na wyznaczenia sfery nieholonomicznej zebrano w Algorytmie 7.1.

Algorytm 7.1 Wyznaczanie sfery nieholonomicznej w przestrzeni zadaniowej

Dane wejściowe: Układ nieholonomiczny z funkcją wyjścia $\boldsymbol{k}(\boldsymbol{q})$ (6.1), konfiguracja $\boldsymbol{q}_0, \in \mathbb{Q}$ wokół obrazu której $\boldsymbol{k}(\boldsymbol{q}_0)$ powstaje sfera, czas planowania T, maksymalna energia ruchu (sterowań) E.

Krok 1. Wyliczyć jakobian $\boldsymbol{J}(\boldsymbol{q}_0)$ funkcji wyjścia $\boldsymbol{k}(\boldsymbol{q})$. Wyliczyć niezbędne pola wektorowe od określenia macierzy $\boldsymbol{F}^{\mathbb{Q}}_{i^\star}(\boldsymbol{q})$ (wzory (2.6),(2.7))oraz $\boldsymbol{F}^{\mathbb{X}}_{i^\star}(\boldsymbol{q})$ (wzór (6.4)).

Krok 2. Wybrać bazę parametryzacji funkcji sterowań (strona 14) oraz wektor parametrów $\boldsymbol{p}$ odpowiedniej wymiarowości. Określić zależność energii od wektora parametrów $\boldsymbol{p}$

$$\boldsymbol{E}(\boldsymbol{u}(\cdot)) = \int_0^T \boldsymbol{u}^T(t)\boldsymbol{u}(t)dt = \sum_{i=1}^{m} \int_{t=0}^{T} \left(\sum_{j=1}^{K_i} p_{i,j}\phi_j(t) \right)^2 dt = E = const. \tag{7.2}$$

Krok 3. Wyliczyć wyrażenia $\boldsymbol{\alpha}(\boldsymbol{p})$ z formuły gCBHD (wzory (2.51), (3.1)).

Krok 4. Wygenerować siatkę w przestrzeni $r-1$ wymiarowych wektorów kątów $(\beta_1, \ldots, \beta_{r-1})^T \in \mathbb{S}^{r-1}$.

Krok 5. Dla każdego wektora kątów $(\beta_1, \ldots, \beta_{r-1})$ z siatki powtórzyć kroki 6-8.

Krok 6. Określić ograniczenia w formie

$$\boldsymbol{w}(R) = \boldsymbol{F}^{\mathbb{X}}_{i^\star}(\boldsymbol{q}_0)\boldsymbol{\alpha}(\boldsymbol{p}) = \boldsymbol{J}(\boldsymbol{q}_0)\boldsymbol{F}^{\mathbb{Q}}_{i^\star}(\boldsymbol{q}_0)\boldsymbol{\alpha}(\boldsymbol{p}), \tag{7.3}$$

powstałe ze złożenia wzorów (7.1), (6.4) i (2.56).

Krok 7. Znaleźć maksimum funkcji $f(\boldsymbol{p})$ określającej odległość punktu końcowego ruchu od środka układu współrzędnych

$$f(\boldsymbol{p}) = R, \tag{7.4}$$

z ograniczeniami (7.2) i (7.3) ustalającymi odpowiednio energię i kierunek ruchu.

Krok 8. Dla wektora $(\beta_1, \ldots, \beta_{r-1}, R)^T$, wyliczyć z zależności (7.1) punkt należący do sfery nieholonomicznej w $\mathbb{X} \cong \mathbb{R}^r$ wokół punktu $\mathbf{0} \in \mathbb{X}$.

Krok 9. Przesunąć sferę nieholonomiczną powstałą w Kroku 8 o wektor $\boldsymbol{k}(\boldsymbol{q}_0)$.

Krok 10. Zwizualizować sferę nieholonomiczną lub jej niżej wymiarowe przekroje.

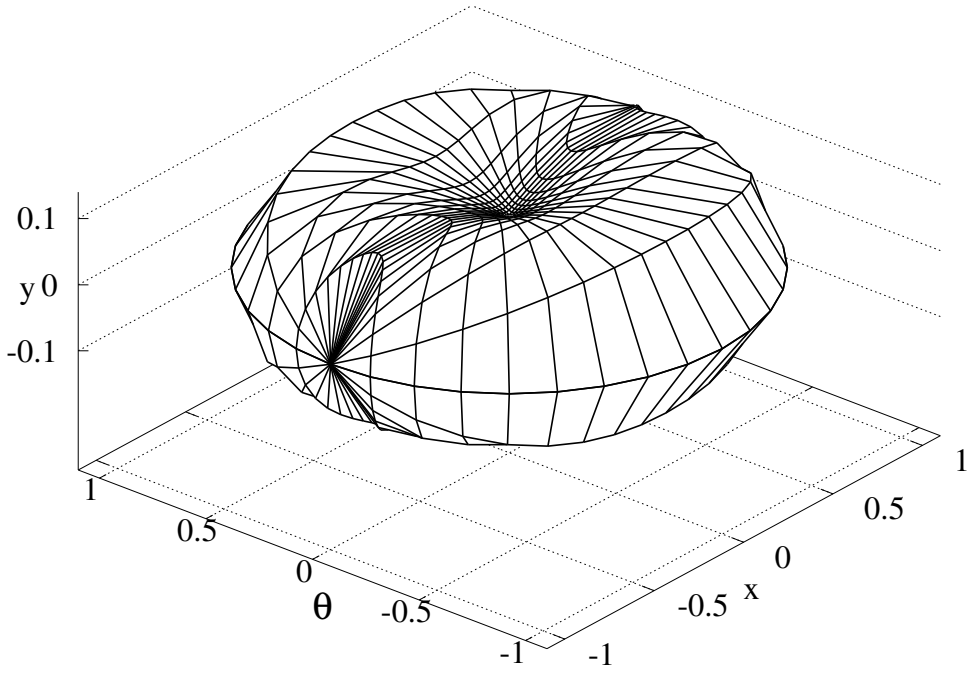
Uwagi do algorytmu:

- Algorytm opisuje procedurę wyznaczania sfery nieholonomicznej w przestrzeni zadaniowej. Sferę w przestrzeni konfiguracyjnej można otrzymać przyjmując identycznościową funkcję wyjścia $\boldsymbol{k}(\boldsymbol{q}) = \mathbf{id}(\boldsymbol{q})$.
- W kroku 1. macierz $\boldsymbol{F}^{\mathbb{X}}_{i}(\boldsymbol{q})$ musi spełniać warunek LARC dla przestrzeni zadaniowej (twierdzenie Chow dla układów z funkcją wyjścia, strona 65). Jednak, jeśli pewne pole wektorowe (np. $\boldsymbol{Z}$ o stopniu $z = \deg(\boldsymbol{Z})$) zostało włączone do macierzy $\boldsymbol{F}^{\mathbb{Q}}_{i^\star}(\boldsymbol{q})$, to także wszystkie pola wektorowe o tym samym stopniu (z tej samej warstwy) muszą być również włączone do macierzy $\boldsymbol{F}^{\mathbb{Q}}_{i^\star}(\boldsymbol{q})$, bowiem mają porównywalny wpływ na maksymalną odległość $f(\boldsymbol{p})$ w danym kierunku $(\beta_1, \ldots, \beta_{r-1})^T$, i przez to na kształt sfery.
- W kroku 2. wymiarowość wektora $\boldsymbol{p}$, spełniająca warunek $\dim(\boldsymbol{p}) \geqslant r$, powinna być rozsądnym kompromisem pomiędzy dokładnością otrzymanej sfery a złożonością obliczeniową. Zwiększenie wymiarowości $\boldsymbol{p}$ poprawia dokładność przybliżenia sfery nie-

holonomicznej, jednak drastycznie zwiększa złożoność obliczeniową zadania optymalizacyjnego z kroku 7. Szczegółowy opis wpływu parametryzacji sterowań na położenie punktu końcowego ruchu znajduje się w rozdziale 5.

- W kroku 7. rozwiązywane jest klasyczne zadanie optymalizacyjne z ograniczeniami. Jednym ze sposobów rozwiązywania takiego zadania jest metoda mnożników Lagrange'a [3].

Symulacje Algorytmu 7.1 zostały przeprowadzone na dwóch dwuwejściowych modelach: jednokołowcu A.1 i samochodzie kinematycznym A.2. Do parametryzacji sterowań wykorzystano ortonormalną bazę harmoniczną 1.5.1. We wszystkich symulacjach ruch był określany na przedziale $[0, T]$, z ustalonym czasem końcowym $T = 1$ i stałą energią sterowań wynoszącą $E = 1$. W każdym zadaniu za konfigurację początkową wybrano $\boldsymbol{q}_0 = \mathbf{0}$, gdyż równania symulowanych układów są niezależne od położenia $(q_1, q_2) = (x, y)$, a kąt $q_3 = \theta$ można sprowadzić do 0 poprzez odpowiedni obrót płaszczyzny xy. Za funkcje wyjścia $\boldsymbol{k}(\boldsymbol{q})$ przyjęto właściwy podzbiór współrzędnych wektora konfiguracji $\boldsymbol{q}$. Wybór takich funkcji ma uzasadnienie praktyczne, gdyż zwykle nie wszystkie współrzędne wektora konfiguracji są istotne. Przykładowo, podczas manewrowania samochodem kinematycznym (np. parkując w środowisku kolizyjnym) nie jest ważna końcowa orientacja osi sterującej więc funkcja wyjścia zawiera tylko położenie i orientacje $\boldsymbol{k}(\boldsymbol{q}) = (x, y, \theta)$.



Rysunek 7.1 Sfera nieholonomiczna jednokołowca w przestrzeni konfiguracyjnej ($\mathbf{k}(\mathbf{q}) = \mathbf{q}$) uzyskana za pomocą algorytmu 7.1; sterowania (7.5).

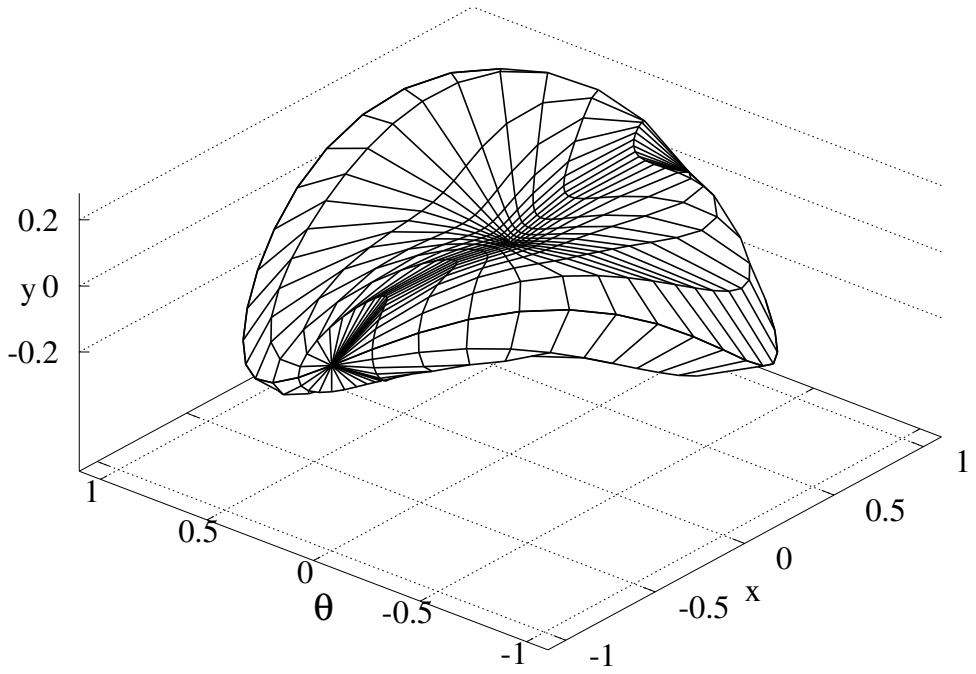
Na rys. 7.1 przedstawiono przybliżoną sferę nieholonomiczną dla jednokołowca z identycznościową funkcją wyjścia $\boldsymbol{k}(\boldsymbol{q}) = \boldsymbol{q}$, wyliczoną za pomocą Algorytmu 7.1. Jako parametryzację sterowań przyjęto pełną pierwszą harmoniczną dla każdego sterowania

$$\begin{cases} u_1 = \frac{1}{\sqrt{T}}(p_{1,1} + p_{1,2}\sqrt{2}\sin(2\pi t) + p_{1,3}\sqrt{2}\cos(2\pi t)), \\ u_2 = \frac{1}{\sqrt{T}}(p_{2,1} + p_{2,2}\sqrt{2}\sin(2\pi t) + p_{2,3}\sqrt{2}\cos(2\pi t)). \end{cases} \tag{7.5}$$

Do określenia wybranych kierunków ruchu wykorzystano regularną siatkę złożoną z 684 punktów opisaną wzorem

$$(\beta_1, \beta_2) = \frac{\pi}{18}(i, j), \quad i = 0, 1, \ldots, 35, \quad j = 0, 1, \ldots, 18. \tag{7.6}$$

Wynikowa sfera jest symetryczna względem konfiguracji $\boldsymbol{q}_0$ oraz każdej osi i płaszczyzny układu współrzędnych. Kierunkiem najmniej efektywnie energetycznym jest ruch wzdłuż osi y, czyli wzdłuż pola wektorowego $[\boldsymbol{g}_1, \boldsymbol{g}_2](\boldsymbol{q}_0)$.

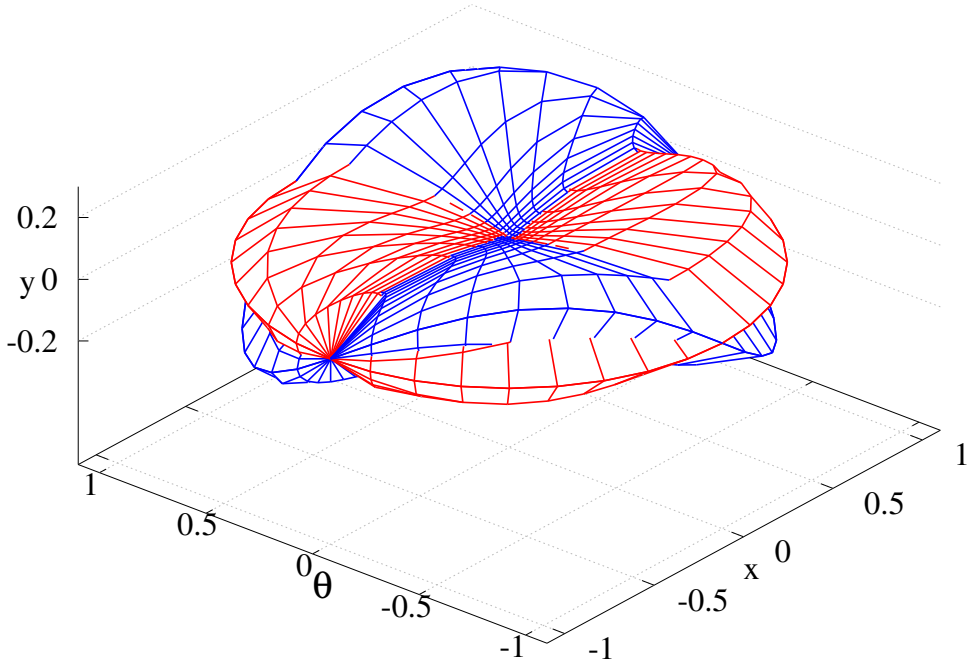


Rysunek 7.2 Sfera nieholonomiczna jednokołowca w przestrzeni konfiguracyjnej ($\mathbf{k}(\mathbf{q}) = \mathbf{q}$) uzyskana za pomocą całkowania numerycznego dla sterowań (7.5)

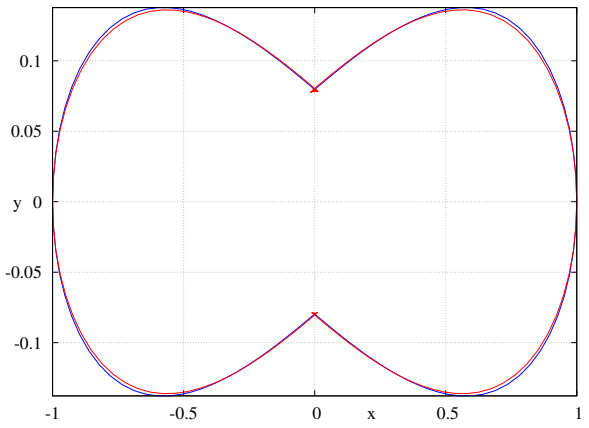
.

Dla ustalonego kierunku, zadanego parą kątów (β_1, β_2), wynikiem zadania optymalizacyjnego jest nie tylko maksymalna wartość R, ale również wektor parametrów $\boldsymbol{p}$ określających sterowania realizująca ruch do punktu (β_1, β_2, R). Z powodu lokalności formuły gCBHD zastosowanie tych sterowań do układu (6.1) skutkuje osiągnięciem punktu $(\beta_1^\star, \beta_2^\star, R^\star)$, niekoniecznie identycznego z punktem (β_1, β_2, R). Na rys. 7.2 zobrazowano sferę nieholonomiczną uzyskaną przez numeryczne scałkowanie równań układu (6.1) dla otrzymanych sterowań. Otrzymana sfera jest dokładniejszą aproksymacją rzeczywistego kształtu sfery nieholonomicznej. Obie, różniące się nieznacznie sfery, przedstawiono na rys. 7.3 z widocznym wpływem pól wektorowych wyższych warstw, nieuwzględnionych w macierzy $\boldsymbol{F}_2^{\mathbb{X}}$, a branych niejawnie pod uwagę przez procedurę całkowania numerycznego. Różnice są zauważalnie widoczne w kierunkach, gdzie nie ma dominującego pola wektorowego będącego elementem $\boldsymbol{F}_2^{\mathbb{X}}$. Niemniej jednak różnice pomiędzy obiema sferami są wystarczająco małe by sfera powstała w wyniku zaproponowanego algorytmu była użyteczna w planowaniu ruchu.

Na rys. 7.4 zaprezentowano przekrój sfer z rys. 7.3 dla kąta $\theta = 0$, czyli możliwe punkty docelowe z końcową orientacją robota równą początkowej. Jak można zauważyć, najtrudniejszym jest ruch wzdłuż osi y, odpowiadający polu drugiego stopnia $[\boldsymbol{g}_1, \boldsymbol{g}_2]$.



Rysunek 7.3 Sfery nieholonomiczna jednokołowca w przestrzeni konfiguracyjnej ($\mathbf{k}(\mathbf{q}) = \mathbf{q}$) dla sterowań (7.5) uzyskana za pomocą: sfera czerwona – algorytmu 7.1, sfera niebieska – całkowania numerycznego modelu.



Rysunek 7.4 Przekrój sfer nieholonomicznych jednokołowca z rysunku 7.3 dla $\theta = 0$; uzyskana za pomocą: linia czerwona – algorytmu, linia niebieska – całkowania numerycznego.

W porównaniu z ruchem w kierunku określonym przez jeden z generatorów, ruch w kierunku pola z drugiej warstwy jest około trzynastu razy bardziej kosztowny energetycznie.

Na rys. 7.5 i 7.6 zilustrowano wpływ funkcji wyjścia $\boldsymbol{k}(\boldsymbol{q})$ oraz parametryzacji sterowań na kształt sfery nieholonomicznej jednokołowca. Rys. 7.5 przedstawia przekrój sfery dla identycznościowej funkcji wyjścia, natomiast na rys. 7.6 – funkcji $\boldsymbol{k}(\boldsymbol{q}) = (x, y)$. Linią czerwoną oznaczono przekrój dla sterowań określonych równaniem (7.5). Linia zielona

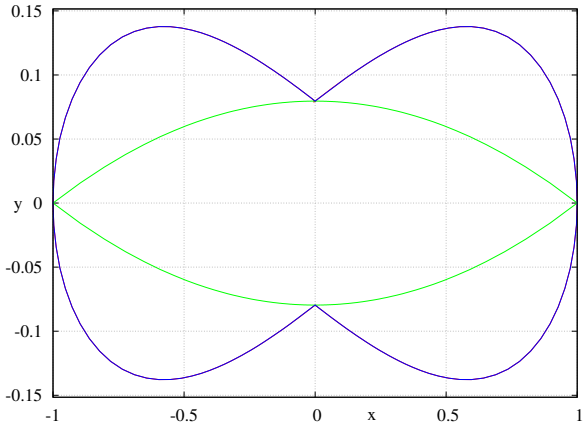
odpowiada parze sterowań

$$\begin{cases} u_1 = \frac{1}{\sqrt{T}}(p_{1,1} + p_{1,2}\sqrt{2}\sin(2\pi t)), \\ u_2 = \frac{1}{\sqrt{T}}(p_{2,1} + p_{2,3}\sqrt{2}\cos(2\pi t)), \end{cases} \tag{7.7}$$

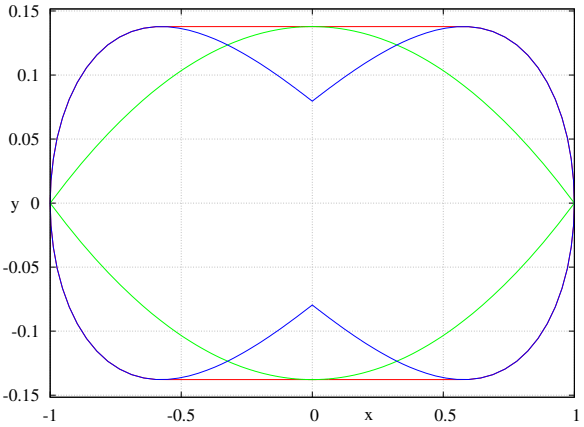
natomiast niebieska parze

$$\begin{cases} u_1 = \frac{1}{\sqrt{T}}(p_{1,1} + p_{1,3}\sqrt{2}\cos(2\pi t)), \\ u_2 = \frac{1}{\sqrt{T}}(p_{2,1} + p_{2,2}\sqrt{2}\sin(2\pi t)). \end{cases} \tag{7.8}$$

Na rys. 7.5 linie niebieska i czerwona się pokrywają. Z rys. 7.5 i 7.6 można wywnioskować, że wybór parametryzacji ma istotny wpływ na kształt sfery. Nawet gdy wybrane parametryzacje są identycznej wymiarowości, różnice mogą być także znaczące. Porównując sfery otrzymane dla najdokładniejszej parametryzacji, można zauważyć wpływ funkcji wyjścia na kształt sfery. Dla funkcji $\boldsymbol{k}(\boldsymbol{q}) = (x, y)$ brak wymogu zachowania startowej orientacji w punkcie końcowym powoduje, że maksymalna odległość jaką można osiągnąć w osi y utrzymuje się przez większość dostępnych wartości współrzędnej x, a kształt przekroju zamiast ósemki przypomina prostokąt z zaokrąglonymi rogami.



Rysunek 7.5 Przekrój ($\theta = 0$) sfer nieholonomicznych jednokołowca w przestrzeni konfiguracyjnej ($\mathbf{k}(\mathbf{q}) = \mathbf{q}$) dla różnych sterowań.

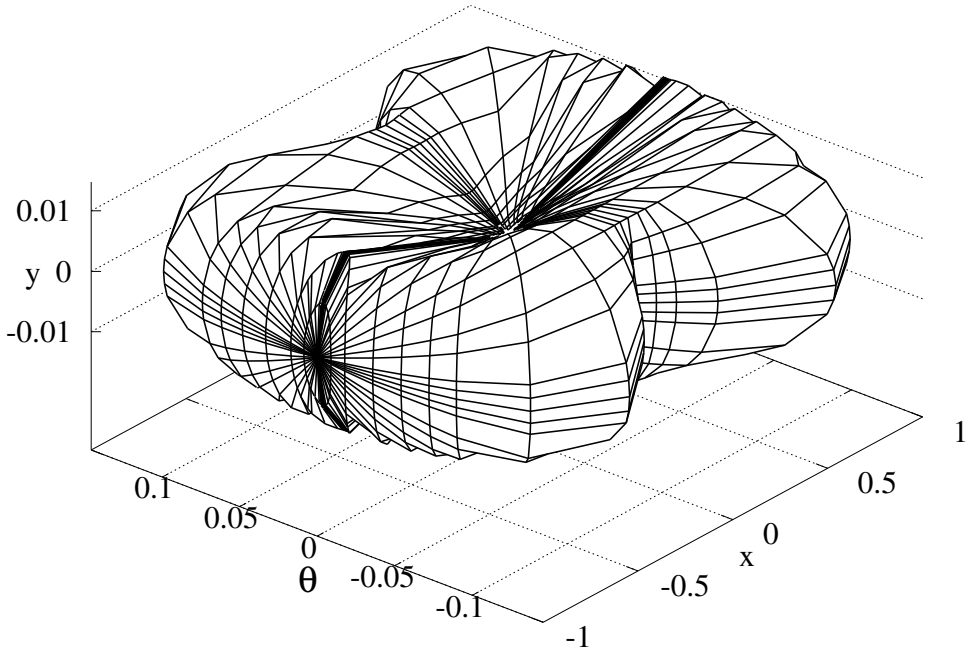


Rysunek 7.6 Przekrój ($\theta = 0$) sfer nieholonomicznych jednokołowca w przestrzeni zadaniowej ($\mathbf{k}(\mathbf{q}) = (\mathrm{x,y})$) dla różnych sterowań.

Rys. 7.7 przedstawia trójwymiarowy przekrój ($\psi = 0$) czterowymiarowej sfery nieholonomicznej samochodu kinematycznego (A.2) z identycznościową funkcją wyjścia i sterowaniami zawierającymi pełne harmoniczne drugiego stopnia

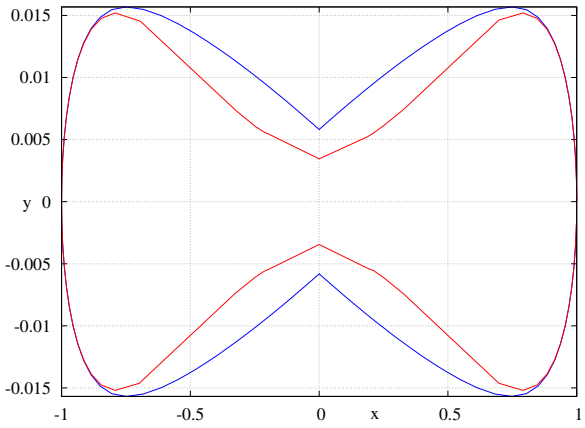
$$\begin{cases} u_1 = \frac{1}{\sqrt{T}}(p_{1,1} + p_{1,2}\sqrt{2}\sin(2\pi t) + p_{1,3}\sqrt{2}\cos(2\pi t) + p_{1,4}\sqrt{2}\sin(4\pi t) + p_{1,5}\sqrt{2}\cos(4\pi t)), \\ u_2 = \frac{1}{\sqrt{T}}(p_{2,1} + p_{2,2}\sqrt{2}\sin(2\pi t) + p_{2,3}\sqrt{2}\cos(2\pi t) + p_{2,4}\sqrt{2}\sin(4\pi t) + p_{2,5}\sqrt{2}\cos(4\pi t)). \end{cases} \tag{7.9}$$

Siatka określająca wybrane kierunki ruchu była dwuwymiarowa (ponieważ liczono jedynie przekrój, a nie całą sferę) i nieregularna. Zawierała 47/43 punktów dyskretyzujących współrzędną β_1/β_2. Użycie nieregularnej siatki było podyktowane koniecznością kompromisu pomiędzy złożonością obliczeniową, a jakością kształtu sfery. Dla regularnej siatki zdecydowana większość uzyskanych punktów była w bliskiej odległości od płaszczyzn



Rysunek 7.7 Przekrój ($\psi = 0$) sfery nieholonomicznej samochodu kinematycznego w przestrzeni konfiguracyjnej ($\mathbf{k}(\mathbf{q}) = \mathbf{q}$) uzyskana za pomocą algorytmu 7.1; sterowania (7.9).

$x = 0$ lub $\theta = 0$. Warto zwrócić uwagę, że kierunki wzdłuż osi x, θ i y w punkcie $\boldsymbol{q}_0 = \mathbf{0}$ odpowiadają polom wektorowym z warstw pierwszej, drugiej i trzeciej, odpowiednio. Występujące na sferze "zagłębienia" można powiązać z ruchem w kierunkach pól wektorowych z wyższych warstw. Im wyższa warstwa, tym trudniejszy ruch, więc poruszanie się wzdłuż kierunku z tej warstwy jest bardziej kosztowne energetycznie.



Rysunek 7.8 Przekroje ($\theta = \psi = 0$) sfer nieholonomicznych samochodu kinematycznego w przestrzeni konfiguracyjnej ($\mathbf{k}(\mathbf{q}) = \mathbf{q}$); sterowania (7.9); uzyskane za pomocą: linia czerwona – algorytmu, linia niebiska – całkowania numerycznego.

Przekroje dwuwymiarowe ($\theta = \psi = 0$) czterowymiarowej sfery nieholonomicznej samochodu kinematycznego przedstawiono na rys. 7.8. Funkcja wyjścia jak i sterowania były identyczne jak w przypadku poprzednim. W przypadku samochodu kinematycznego różnice pomiędzy sferami są większe, jednak wciąż stosunkowo niewielkie.

Wszystkie symulacje zostały autorsko zaimplementowane w pakiecie *Mathematica*, w wersji 11.3, a obliczenia wykonano na komputerze z procesorem *Intel Core i5-8400, 2.80GHz* $\times$ *6* i pamięcią 8 GB RAM. Czas wykonania wybranych zadań był następujący:

- sfera nieholonomiczna dla jednokołowca, sterowania (7.5), $210[s]$, 36×19 podstawowych zadań optymalizacyjnych, $0,3[s]$/zadanie,
- przekrój sfery nieholonomicznej samochodu kinematycznego ($\psi = 0$), sterowania (7.9), 390 min, 47×43 podstawowych zadań optymalizacyjnych, $11,5$ s/zadanie.

Warto nadmienić, że do rozwiązywania zadań optymalizacyjnych użyto funkcji `NMinimize` z pakietu *Mathematica*, która usiłuje znaleźć rozwiązanie globalnie optymalne i jest czasochłonna. W rzeczywistych zastosowaniach sugerowane jest używanie szybszych metod optymalizacji lokalnych (np. algorytmu Newtona, podrozdział 1.4) zamiast globalnych.

Rozdział 8

Wybór konfiguracji początkowej

Klasycznie przez pojęcie planowania ruchu rozumie się znalezienie sterowań pozwalających na przeprowadzenie układu z punktu początkowego do docelowego. Równania układu oraz punkty brzegowe są niezbędne do otrzymania wynikowej trajektorii oraz sterowań. Standardowo zakłada się, że punkty początkowy oraz docelowy należą do tej samej przestrzeni. Przypadek w którym rozważany jest układ nieholonomiczny z funkcją wyjścia (6.1) przenoszącą planowanie do przestrzeni zadaniowej jest bardziej skomplikowany. W rozdziale 6 adaptowano pojęcia i metody planowania ruchu do przestrzeni zadaniowej. Pokazano, że mimo planowania pomiędzy punktami początkowym $\boldsymbol{x}_0$ oraz docelowym $\boldsymbol{x}_f$ w przestrzeni zadaniowej, metoda Lie-algebraiczna wartościuje pola wektorowe w punkcie $\boldsymbol{q}_0$, gdzie $\boldsymbol{x}_0 = \boldsymbol{k}(\boldsymbol{q}_0)$. Dla interesujących praktycznie przypadków ($r < n$) funkcja odwrotna do funkcji wyjścia $\boldsymbol{k}(\boldsymbol{q})$ jest niejednoznaczna. Niejednoznaczność nie jest problematyczna w przypadku pojedynczego planowania, bowiem realistycznie zakłada się, że układ jest w znanej konfiguracji początkowej $\boldsymbol{q}_0$, wynikającej z fizycznych uwarunkowań, będąc daną wejściową (najczęściej zastępującą $\boldsymbol{x}_0$). Sytuacja zmienia się w przypadku ciągu pojedynczych planowań, gdy podcele kolejnych planowań są definiowane jedynie w przestrzeni zadaniowej. Oczywiście, po realizacji ruchu do poprzedniego podcelu można odczytać konfigurację początkową dla bieżącego podcelu układu wynikającą z realizacji uprzedniego planowania. Jednak potencjalnie różne punkty końcowe $\boldsymbol{q}_f$ odpowiadające identycznemu $\boldsymbol{x}_f$ w i-tym planowaniu warunkują różną trudność ruchu w $(i+1)$-szym planowaniu. Jest więc tu miejsce na optymalizację polegającą na wyborze najlepszej konfiguracji $\boldsymbol{q}^\star$, takiej że $\boldsymbol{x}_f = \boldsymbol{k}(\boldsymbol{q}^\star)$, z punktu widzenia następnego kroku metody Lie-algebraicznej. Optymalizacja wyboru podcelu w przestrzeni konfiguracyjnej $\boldsymbol{q}^\star$ jest zadaniem trudnym oraz (co bardziej istotne) złożonym obliczeniowo. W celu zmniejszenia złożoności obliczeniowej zaproponowano algorytm oceny jakości konfiguracji $\boldsymbol{q}^\star$ w perspektywie przemieszczenia układu do punktu docelowego $\boldsymbol{x}_f$. Podstawowym narzędziem będzie algebra liniowa, z której przytoczono kilka użytecznych faktów:

- Pole wektorowe zwartościowane w pewnej konfiguracji jest wektorem.
- Zbiór (baza) liniowo niezależnych wektorów $\boldsymbol{B} = \{\boldsymbol{b}_i \in \mathbb{R}^n\}, \quad i = 1, \ldots, r$ jest ortonormalizowalny algorytmem Grama-Schmidta [5], a wynikowy zbiór $\tilde{\boldsymbol{B}} = \{\tilde{\boldsymbol{b}}_1, \ldots, \tilde{\boldsymbol{b}}_r\}$ będzie określany jako $\tilde{\boldsymbol{B}} = \text{orthonorm}(\boldsymbol{B})$.
- Rzut wektora $\boldsymbol{v} \in \mathbb{R}^n$ na podprzestrzeń rozpiętą przez $\tilde{\boldsymbol{B}}$ zadany jest wzorem

$$\boldsymbol{v}_{\|}(\tilde{\boldsymbol{B}}) = \text{proj}_{\tilde{\boldsymbol{B}}}(\boldsymbol{v}) = \sum_{i=1}^{r} \text{proj}_{\tilde{\boldsymbol{b}}_i}(\boldsymbol{v}) = \sum_{i=1}^{r} \langle \boldsymbol{v}, \tilde{\boldsymbol{b}}_i \rangle \tilde{\boldsymbol{b}}_i. \tag{8.1}$$

- Dopełnienie ortogonalne wektora $\boldsymbol{v}$ względem podprzestrzeni liniowej rozpiętej przez $\tilde{\boldsymbol{B}}$ jest określone przez
$$\boldsymbol{v}_{\perp}(\tilde{\boldsymbol{B}}) = \boldsymbol{v} - \boldsymbol{v}_{\parallel}(\tilde{\boldsymbol{B}}). \tag{8.2}$$

- Dopełnienie ortogonalne podprzestrzeni $\boldsymbol{V}$ względem podprzestrzeni liniowej rozpinanej przez $\tilde{\boldsymbol{B}}$ jest zadane jako
$$\boldsymbol{V}_{\perp}(\boldsymbol{B}) = \{\boldsymbol{v} : \boldsymbol{v}_{\perp}(\tilde{\boldsymbol{B}}) \wedge \boldsymbol{v} \in \boldsymbol{V}\}. \tag{8.3}$$

- Kąt między wektorem $\boldsymbol{v}$ a podprzestrzenią liniową rozpinaną przez $\tilde{\boldsymbol{B}}$ jest dany wzorem
$$\beta(\boldsymbol{v}, \tilde{\boldsymbol{B}}) = \arccos \frac{\langle \boldsymbol{v}, \boldsymbol{v}_{\parallel}(\tilde{\boldsymbol{B}})\rangle}{\|\boldsymbol{v}\| \cdot \|\boldsymbol{v}_{\parallel}(\tilde{\boldsymbol{B}})\|}. \tag{8.4}$$

- Objętość zbioru $\boldsymbol{B}$ zawierającego wektory (niekoniecznie niezależne liniowo) $\boldsymbol{b}_i \in \mathbb{R}^n$, rozpinającego r-wymiarową podprzestrzeń $\mathbb{R}^n$, jest obliczana według wzoru
$$\mathrm{vol}(\boldsymbol{B}) = \max_{\boldsymbol{D}\in\mathbb{D}} \mathrm{vol}(\boldsymbol{D}), \quad \boldsymbol{B} = \{\boldsymbol{b}_i : \boldsymbol{b}_i \in \boldsymbol{R}^n, i = 1, \ldots, p\}, \quad \mathrm{rank}(\boldsymbol{B}) = r \leqslant p, \tag{8.5}$$
gdzie $\boldsymbol{D}$ jest r-elementowym podzbiorem $\boldsymbol{B}$, $\mathbb{D}$ – zbiorem wszystkich możliwych zbiorów $\boldsymbol{D}$ oraz
$$\mathrm{vol}(\boldsymbol{D}) = \sqrt{\det(\boldsymbol{M}^T\boldsymbol{M})}, \tag{8.6}$$
gdzie kolumnami macierzy $\boldsymbol{M}$ są wektory należące do zbioru $\boldsymbol{D}$.

Procedurę oceny jakości konfiguracji $\boldsymbol{q}_a$ w perspektywie przemieszczenia do punktu $\boldsymbol{x}_b$ opisano Algorytmem 8.1. Wynik procedury będzie oznaczany jako $lie(\boldsymbol{q}_a, \boldsymbol{x}_b)/lie^{\star}(\boldsymbol{q}_a, \boldsymbol{x}_b)$ w zależności od składowej *contain*, bez/z – odpowiednio.

Uwagi do Algorytmu 8.1:

- Funkcja $lie(\boldsymbol{q}_a, \boldsymbol{x}_b)$ określa kierunkową trudność ruchu i wraz ze wzrostem trudności jej wartość maleje.
- Podprzestrzenie są określane w konfiguracji $\boldsymbol{q}_a$. Dla uproszczenia zapisu argument ten pominięto.
- Wartości współczynnika c_i dla poszczególnych warstw określono za pomocą formuły CBHD jak w rozdziale 5 (strona 55).
- We wzorach (8.7) i (8.10), wyliczających współczynniki c_i są w mianowniku, ponieważ ruch w kierunkach z niższych warstw jest preferowany.
- Macierze $\boldsymbol{F}_i$ ze wzoru (6.4) są złożone z kolejnych warstw bazy algebry Liego (wzór (2.6), strona 18). Użycie wzoru (6.4) na niektórych kolumnach macierzy $\boldsymbol{F}_i^{\mathbb{Q}}$ (np. $\boldsymbol{H}_{\mathbb{Q}}^1$) da odpowiadającą tym kolumną część macierzy $\boldsymbol{F}_i^{\mathbb{X}}$ (oznaczaną w tym przypadku jako $\boldsymbol{H}_{\mathbb{X}}^1$).
- Podprzestrzeń $\boldsymbol{A}$ określa wszystkie możliwe kierunki w danej fazie obliczeń.
- Podczas liczenia wpływu i-tej warstwy na wartość $lie(\boldsymbol{q}_a, \boldsymbol{x}_b)$ (wzory (8.10)-(8.12)) używa się $\boldsymbol{H}_{\mathbb{X}\perp}^i(\boldsymbol{A})$ zamiast $\boldsymbol{H}_{\mathbb{X}}^i$, zakładając, że ruch w kierunkach już znajdujących się w $\boldsymbol{H}_{\mathbb{X}\parallel}^i(\boldsymbol{A})$ może być bardziej efektywnie generowany za pomocą już zbadanych warstw.
- Objętość określana dla zbioru wektorów określa średnią efektywność ruchu w podprzestrzeni rozpinanej przez te wektory. Składowa *manip* jest rzutem tej objętości na konkretny kierunek ruchu, dlatego we wzorach (8.7) i (8.10) jest mnożona przez $\cos(\beta)$.

Algorytm 8.1 Lie-algebraiczna ocena jakości konfiguracji $\boldsymbol{q}_a$ w perspektywie przemieszczenia do punktu $\boldsymbol{x}_b$. Wynikiem algorytmu jest wartość $lie(\boldsymbol{q}_a, \boldsymbol{x}_b)$. (bez składowej *contain*, czyli tożsamościowo $contain = 1$) lub $lie^\star(\boldsymbol{q}_a, \boldsymbol{x}_b)$ (ze składową *contain*).

Dane wejściowe: Konfiguracja oceniana $\boldsymbol{q}_a \in \mathbb{Q}$, układ nieholonomiczny z funkcją wyjścia $\boldsymbol{k}(\boldsymbol{q})$ (6.1), punkt docelowy $\boldsymbol{x}_b \in \mathbb{X}$, horyzont czasowy T.

Krok 1. Określić punkt $\boldsymbol{x}_a$ odpowiadający ocenianej konfiguracji $\boldsymbol{x}_a \leftarrow \boldsymbol{k}(\boldsymbol{q}_a)$. Obliczyć kierunek ruchu $\Delta\boldsymbol{x} \leftarrow \boldsymbol{x}_b - \boldsymbol{x}_a$.

Krok 2. Określić warstwy bazy Ph. Halla, $i = 1, \ldots, i^\star$ w konfiguracji $\boldsymbol{q}_a$ otrzymując podprzestrzeń liniową $\boldsymbol{H}^i_{\mathbb{Q}}$. Wyliczyć jakobian funkcji wyjścia $\boldsymbol{J}(\boldsymbol{q})$ (wzór (6.2), strona 64). Wyliczyć $\boldsymbol{H}^i_{\mathbb{X}}$ ze wzoru (6.4) (zobacz uwagi do Algorytmu 8.1).

Krok 3. Określić podprzestrzeń $\tilde{\boldsymbol{H}}^1_{\mathbb{X}} \leftarrow \mathrm{orthonorm}(\boldsymbol{H}^1_{\mathbb{X}})$. Policzyć, korzystając ze wzorów (8.4)–(8.6), wartość *manip*

$$manip \leftarrow \frac{1}{c_1} \cdot \cos(\beta(\Delta\boldsymbol{x}, \tilde{\boldsymbol{H}}^1_{\mathbb{X}})) \cdot \mathrm{vol}(\boldsymbol{H}^1_{\mathbb{X}}). \tag{8.7}$$

Policzyć wartość *angs* według wzoru

$$angs \leftarrow \cos(\beta(\Delta\boldsymbol{x}, \tilde{\boldsymbol{H}}^1_{\mathbb{X}})). \tag{8.8}$$

Policzyć, korzystając z (8.1) wartość *contain* według wzoru

$$contain \leftarrow \| \operatorname{proj}_{\tilde{\boldsymbol{H}}^1_{\mathbb{X}}}(\Delta\boldsymbol{x}) \| . \tag{8.9}$$

Zainicjować licznik warstw $i \leftarrow 1$ oraz dotychczasowo badaną podprzestrzeń $\boldsymbol{A} \leftarrow \tilde{\boldsymbol{H}}^1_{\mathbb{X}}$.

Krok 4. Zwiększyć o jeden licznik warstw $i \leftarrow i + 1$.
Określić podprzestrzenie $\boldsymbol{H}^i_{\mathbb{X}\perp}(\boldsymbol{A})$ oraz $\tilde{\boldsymbol{H}}^i_{\mathbb{X}\perp}(\boldsymbol{A}) \leftarrow \mathrm{orthonorm}(\boldsymbol{H}^i_{\mathbb{X}\perp}(\boldsymbol{A}))$.

Krok 5. Zaktualizować wartości *manip*, *angs* i *contain*.

$$manip \leftarrow manip + \frac{1}{c_i} \cdot \cos(\beta(\Delta\boldsymbol{x}, \tilde{\boldsymbol{H}}^i_{\mathbb{X}\perp}(\boldsymbol{A}))) \cdot \mathrm{vol}(\boldsymbol{H}^i_{\mathbb{X}\perp}(\boldsymbol{A})). \tag{8.10}$$

gdzie c_i jest rzeczywistym współczynnikiem określającym energię ruchu wzdłuż pola wektorowego i-tej warstwy w czasie $[0, T]$.

$$angs \leftarrow angs + \cos(\beta(\Delta\boldsymbol{x}, \tilde{\boldsymbol{H}}^i_{\mathbb{X}\perp}(\boldsymbol{A}))). \tag{8.11}$$

$$contain \leftarrow contain + \| \operatorname{proj}_{\tilde{\boldsymbol{H}}^i_{\mathbb{X}\perp}(\boldsymbol{A})}(\Delta\boldsymbol{x}) \| . \tag{8.12}$$

Krok 6. Zaktualizować dotychczas badaną podprzestrzeń $\boldsymbol{A} \leftarrow \boldsymbol{A} \oplus \boldsymbol{H}^i_{\mathbb{X}\perp}(\boldsymbol{A})$ oraz ją zortonormalizować $\boldsymbol{A} \leftarrow \mathrm{orthonorm}(\boldsymbol{A})$.

Krok 7. Jeśli $\mathrm{rank}(\boldsymbol{A}) = n$, zwrócić wartość $lie(\boldsymbol{q}_a, \boldsymbol{x}_b) = manip/(angs \cdot contain)$. W innym przypadku przejść do Kroku 4.

- Składowa *manip* premiuje konfiguracje $\boldsymbol{q}_a$, dla których przemieszczenie w kierunku $\boldsymbol{x}_b$ wymaga ruchu wzdłuż pól wektorowych z jak największej liczby warstw. Dzieje się tak ponieważ wartości $\cos(\beta)$ ze wzorów (8.7) i (8.10) spełniają nierówność trójkąta, więc składowa *manip* ma lokalne minima dla $\beta = j\pi/2,\ j \in \mathbb{Z}$.

- Składowa *angs* jest odpowiedzialna za skalowanie rezultatu $lie(\boldsymbol{q}_a, \boldsymbol{x}_b)$ do zbioru wartości $[0, 1/T]$, gdzie $1/T$ odpowiada przypadkowi gdzie przesunięcie $\Delta\boldsymbol{x}$ jest kombinacją liniową ruchu wzdłuż kierunków określanych jedynie przez generatory układu.
- Składowa *contain* jest opcjonalna. Należy jej używać w przypadku, gdy ocena konfiguracji powinna być zależna od odległości euklidesowej pomiędzy konfiguracjami i premiuje mniejsze odległości. Przykładowo, dla jednokołowca oraz identycznościowej funkcji wyjścia, czasu $T = 1$ oraz braku składowej *contain*

$$lie((0,0,0)^T, (1,0,0)^T) = lie((0,0,\pi)^T, (1,0,0)^T) = 1. \tag{8.13}$$

  Przy wprowadzeniu składowej *contain* lepszą ocenę uzyska konfiguracja $(0,0,0)^T$. Użycie składowej *contain* powoduje, że wartości $lie^\star(\boldsymbol{q}_a, \boldsymbol{x}_b)$ nie są w przedziale $[0, 1/T]$.
- Funkcja jakości $lie(\boldsymbol{q}_a, \boldsymbol{x}_b)$ jest tak dobrana, aby premiować ruch w kierunku długich pól wektorowych o niskim stopniu, wskazujących w kierunku punktu $\boldsymbol{x}_b$.
- Algorytm 8.2 zakończy się po maksimum $i^\star$ iteracjach, gdzie $i^\star$ to liczba warstw niezbędnych do spełnienia warunku LARC.
- Algorytm może być stosowany także wtedy, gdy oba argumenty są konfiguracjami w $\mathbb{Q}$ (traktuje się wtedy funkcję wyjścia $\boldsymbol{k}(\boldsymbol{q})$ jako identycznościową). Dlatego w dalszej części pracy podczas oceniania jakości konfiguracji $\boldsymbol{q}_a$ w perspektywie przemieszczenia do konfiguracji $\boldsymbol{q}_b$ będą używane oznaczenia $lie(\boldsymbol{q}_a, \mathbf{id}(\boldsymbol{q}_b))$ oraz $lie^\star(\boldsymbol{q}_a, \mathbf{id}(\boldsymbol{q}_b))$.

Algorytm 8.1 przetestowano na trzech układach bezdryfowych z funkcją wyjścia (6.1): jednokołowcu A.1, samochodzie kinematycznym A.2 oraz integratorze Brocketta A.3. Dla powyższych układów wybrano następujące funkcje wyjścia $\boldsymbol{x} = \boldsymbol{k}(\boldsymbol{q})$ będące podzbiorami wektora konfiguracji $\boldsymbol{q}$

- jednokołowiec: $\boldsymbol{x} = (q_1, q_2)^T = (x, y)^T$,
- samochód kinematyczny: $\boldsymbol{x} = (q_1, q_2)^T = (x, y)^T$ oraz $\boldsymbol{x} = (q_1, q_2, q_3)^T = (x, y, \theta)^T$,
- integrator Brocketta: $\boldsymbol{x} = (q_1, q_3)^T$ oraz $\boldsymbol{x} = (q_2, q_3)^T$.

Przy wyborze funkcji wyjścia pominięto funkcje identycznościowe oraz zawierające jedynie kierunki otrzymywane bezpośrednio z generatorów. W każdej symulacji punkt początkowy był stały i wynosił $\boldsymbol{x}_0 = \mathbf{0}$, natomiast punkt docelowy $\boldsymbol{x}_f$ zależał od konkretnego układu, funkcji wyjścia oraz parametru określającego odległość celu $\Delta_x \in \{0.1, 0.2, 0.5, 0.7, 1, 3\}$

- dla $\boldsymbol{x} = (q_1, q_2)^T = (x, y)^T$: $\boldsymbol{x}_f = \Delta_x(1,0)^T$, $\boldsymbol{x}_f = \Delta_x(0,1)^T$ oraz $\boldsymbol{x}_f = \Delta_x(1/\sqrt{2}, 1/\sqrt{2})^T$,
- dla $\boldsymbol{x} = (q_1, q_2, q_3)^T = (x, y, \theta)^T$: $\boldsymbol{x}_f = \Delta_x(1,0,0)^T$, $\boldsymbol{x}_f = \Delta_x(0,1,0)^T$ oraz $\boldsymbol{x}_f = \Delta_x(1/\sqrt{2}, 1/\sqrt{2}, \pi/(4\Delta_x))^T$,
- dla $\boldsymbol{x} = (q_1, q_3)^T$ oraz $\boldsymbol{x} = (q_2, q_3)^T$: $\boldsymbol{x}_f = \Delta_x(1,0)^T$, $\boldsymbol{x}_f = \Delta_x(0,1)^T$ oraz $\boldsymbol{x}_f = \Delta_x(1/\sqrt{2}, 1/\sqrt{2})^T$.

Dla sprawdzenia użyteczności Algorytmu 8.1 przeprowadzono statystyczne porównanie oceny jakości $lie(\boldsymbol{q}^\star, \boldsymbol{x}_f)$ oraz energii sterowań elementarnego planowania pomiędzy $\boldsymbol{q}^\star$ a $\boldsymbol{x}_f$. Porównanie przeprowadzono przy pomocy współczynników: korelacji liniowej Pearsona [15] oraz rankingowej Spearmana [44] opisanych w podrozdziale 1.6. Kroki pozwalające na określenie korelacji między $lie(\boldsymbol{q}^\star, \boldsymbol{x}_f)$ a energią sterowań zebrano w Algorytmie 8.2. Dla każdego zadania zbiór $\mathbb{S}^\star$ składał się z 300 losowych konfiguracji

$$\boldsymbol{q}_0^\star = (\mathrm{rand}(-\pi, \pi), \ldots, \mathrm{rand}(-\pi, \pi))^T, \tag{8.14}$$

gdzie funkcja rand(a, b) generuje liczbę z przedziału $[a, b]$ według rozkładu jednostajnego. Horyzont czasu ruchu ustalono na $T = 1$.

Algorytm 8.2 Korelacja między Lie-algebraiczną oceną danej konfiguracji a kosztem ruchu.

Dane wejściowe: Układ nieholonomiczny z funkcją wyjścia $\boldsymbol{k}(\boldsymbol{q})$ (6.1), punkty początkowy $\boldsymbol{x}_0 \in \mathbb{X}$ i docelowy $\boldsymbol{x}_f \in \mathbb{X}$, horyzont czasowy T.

Krok 1. Wygenerować losowy (według rozkładu jednostajnego) zbiór $\mathbb{S}^\star \subset \mathbb{Q}$ składający się z konfiguracji $\boldsymbol{q}_0^\star$. Używając algorytmu Newtona, startującego z każdego z punktów $\boldsymbol{q}_0^\star$ utworzyć zbiór $\mathbb{S} \subset \mathbb{Q}$ składający się z konfiguracji początkowych $\boldsymbol{q}_0$ spełniających zależność $\boldsymbol{x}_0 = \boldsymbol{k}(\boldsymbol{q}_0)$.

Krok 2. Dla każdej konfiguracji $\boldsymbol{q}_0^i \in \mathbb{S}$ wykonać Kroki 3-4.

Krok 3. Używając Algorytmu 8.1 dokonać Lie-algebraicznej oceny konfiguracji $\boldsymbol{q}_0^i$ otrzymując $lie(\boldsymbol{q}_0^i)$.

Krok 4. Rozwiązać zadanie planowania ruchu pomiędzy punktami $\boldsymbol{x}_0$ i $\boldsymbol{x}_f$ za pomocą formuły gCBHD inicjalizowanej w konfiguracji $\boldsymbol{q}_0^i$. Z otrzymanych sterowań $\boldsymbol{u}(\cdot)$ otrzymać energię ruchu $en(\boldsymbol{q}_0^i)$.

Krok 5. Określić korelację zbiorów $\boldsymbol{EN} = \{en(\boldsymbol{q}_0^i)\}$ oraz $\boldsymbol{LIE} = \{lie(\boldsymbol{q}_0^i)\}$.

Każde zadanie planowania ruchu w Algorytmie 8.2 rozwiązywano za pomocą metody Lie-algebraicznej wykorzystującej formułę gCBHD (paragraf 2.2.4.2) przystosowanej do układu z funkcją wyjścia (rozdział 6). Otrzymane zadanie kinematyki odwrotnej układu nieholonomicznego rozwiązywano algorytmem Newtona z optymalizacją w przestrzeni zerowej (podrozdział 1.4). Optymalizowaną funkcją była energia ruchu (sterowań)

$$\boldsymbol{E}(\boldsymbol{u}(\cdot)) = \int_0^T \boldsymbol{u}^T(t)\boldsymbol{u}(t)dt. \tag{8.15}$$

Sterowania parametryzowano za pomocą bazy harmonicznej (podrozdział 1.5). Dla trójwymiarowych układów (jednokołowiec, integrator Brocketta) sterowania zawierały harmoniczne do pierwszej włącznie, z wektorem parametrów $\boldsymbol{p} = (p_{1,1}, p_{1,2}, p_{1,3}, p_{2,1}, p_{2,2}, p_{2,3})^T$. Natomiast dla czterowymiarowego układu (samochód kinematyczny) – do drugiej włącznie, i wektorze parametrów $\boldsymbol{p} = (p_{1,1}, p_{1,2}, p_{1,3}, p_{1,4}, p_{1,5}, p_{2,1}, p_{2,2}, p_{2,3}, p_{2,4}, p_{2,5})^T$. Początkowymi wartościami wektora parametrów $\boldsymbol{p}$, niezbędnymi do inicjalizacji algorytmu Newtona z optymalizacją przestrzeni zerowej, były

$$\begin{cases} p_{i,j} = 1 & \text{dla} \quad j = 1, \\ p_{i,j} = 0 & \text{dla} \quad j \neq 1. \end{cases} \tag{8.16}$$

Wyniki działania Algorytmu 8.2 zostały zebrane w tab. 8.1 i 8.2. Tab 8.1 zawiera wyniki dla modeli robotów mobilnych (nienilpotentnych), natomiast tab. 8.2 dla nilpotentnego integratora Brocketta. Przykładowe optymalne ścieżki dla niektórych zadań jednokołowca i samochodu kinematycznego przedstawiono na rys. 8.1 i 8.2.

Na podstawie uzyskanych wyników symulacji można wyciągnąć następujące wnioski:

- W zdecydowanej większości przypadków z tab. 8.1 współczynniki korelacji są wysokie i ujemne, więc funkcja $lie(\boldsymbol{q}_0)$ może być używana do oceny kosztu energetycznego ruchu.
- W przypadku samochodu kinematycznego, im większa różnica pomiędzy wymiarowością przestrzeni konfiguracyjnej $\mathbb{Q}$ i zadaniowej $\mathbb{X}$, tym mniejsza maksymalna korelacja, bowiem podprzestrzeń $\boldsymbol{H}^i_{\mathbb{X},\perp}$ z Kroku 4, Algorytmu 8.1 niesie ze sobą mniej informacji.

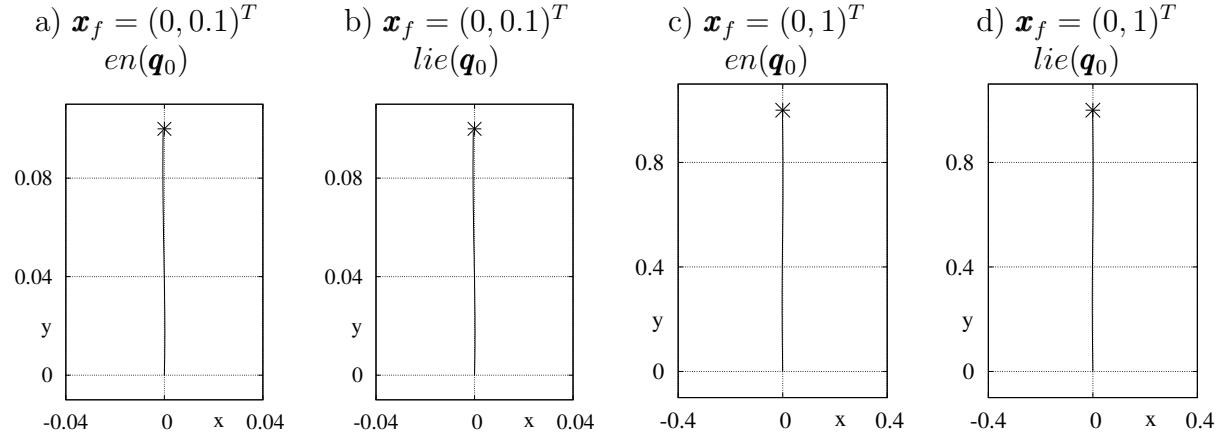
Tabela 8.1 Współczynniki korelacji pomiędzy zbiorami wartości ***LIE*** oraz ***EN*** dla jednokołowca i samochodu kinematycznego.

układ	jednokołowiec			samochód kinematyczny					
$\boldsymbol{x}$	(q_1, q_2)			(q_1, q_2)			(q_1, q_2, q_3)		
$\boldsymbol{x}_f$	$\begin{pmatrix}1\\0\end{pmatrix}$	$\begin{pmatrix}0\\1\end{pmatrix}$	$\begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}}\end{pmatrix}$	$\begin{pmatrix}1\\0\end{pmatrix}$	$\begin{pmatrix}0\\1\end{pmatrix}$	$\begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}}\end{pmatrix}$	$\begin{pmatrix}1\\0\\0\end{pmatrix}$	$\begin{pmatrix}0\\1\\0\end{pmatrix}$	$\begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}}\\\frac{\pi}{4\Delta_x}\end{pmatrix}$
Δ_x	Współczynniki korelacji rankingowej Spearmana								
0.1	-1.00	-1.00	-1.00	-0.71	-0.72	-0.73	-0.96	0.09	-0.56
0.2	-1.00	-1.00	-1.00	-0.70	-0.72	-0.56	-0.92	-0.06	-0.46
0.5	-1.00	-1.00	-1.00	-0.72	-0.76	-0.60	-0.81	-0.12	0.99
0.7	-1.00	-1.00	-1.00	-0.66	-0.70	-0.77	-0.72	-0.05	0.99
1	-1.00	-1.00	-1.00	-0.50	-0.70	-0.58	-0.63	0.05	0.99
3	-0.99	-1.00	-1.00	-0.59	-0.72	-0.57	-0.57	0.09	0.96
	Współczynniki korelacji liniowej Pearsona								
0.1	-0.98	-0.97	-0.97	-0.68	-0.68	-0.69	-0.95	0.25	-0.41
0.2	-0.98	-0.97	-0.98	-0.69	-0.71	-0.57	-0.92	-0.06	0.04
0.5	-0.98	-0.97	-0.98	-0.74	-0.76	-0.66	-0.84	-0.06	0.66
0.7	-0.98	-0.98	-0.98	-0.68	-0.69	-0.76	-0.77	-0.51	0.68
1	-0.98	-0.98	-0.98	-0.51	-0.72	-0.61	-0.69	0.64	0.69
3	-0.98	-0.97	-0.97	-0.64	-0.76	-0.63	-0.54	0.63	0.71

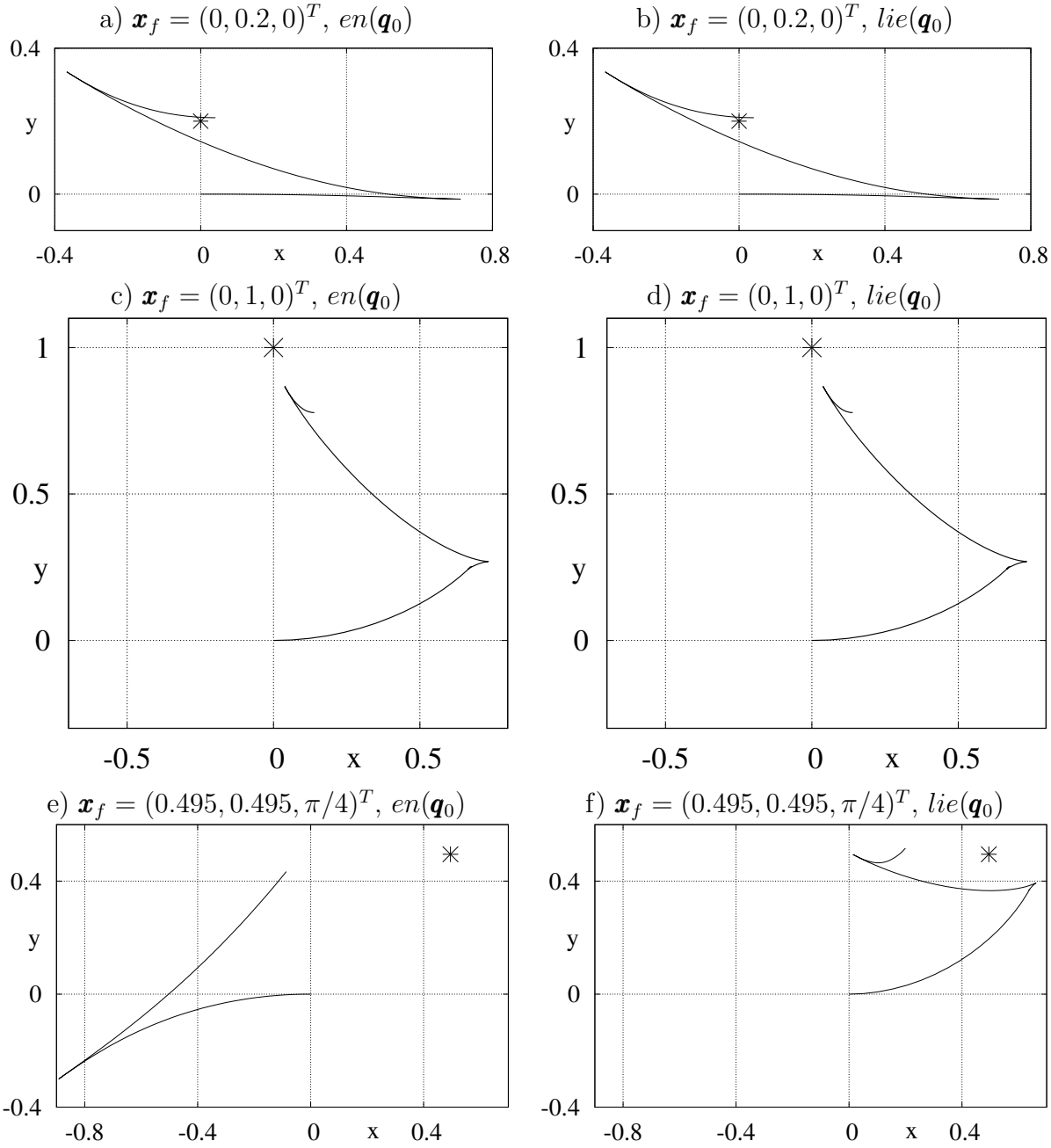
- Wraz ze wzrostem odległości do celu Δ_x maleje korelacja. Jest to spowodowane większym wpływem warstw bazy Ph. Halla nie uwzględnionych w macierzy $\boldsymbol{F}_i$ (zobacz wzór (2.54)). Dokładność realizacji położenia punktu końcowego trajektorii otrzymanej za pomocą formuły gCBHD również maleje ze wzrostem Δ_x.
- Podobnie dla punktu docelowego $\boldsymbol{x}_f = \Delta_x(1/\sqrt{2}, 1/\sqrt{2}, \pi/4\Delta_x)^T$. Dokładność realizacji położenia punktu końcowego trajektorii maleje bardzo szybko, co można zauważyć na rys. 8.2ef. Warto zaznaczyć, że mimo zaistnienia takiej sytuacji na poglądowym rysunku, optymalna trajektoria według funkcji $lie(\boldsymbol{q}_0)$ nie zawsze ma dokładniejszy punkt końcowy niż ta według funkcji $en(\boldsymbol{q}_0)$. Sposoby na polepszenie dokładności położenia punktu końcowego zostały opisane w rozdziale 10.
- Dla samochodu kinematycznego i punktu docelowego $\boldsymbol{x}_f = (0, 1, 0)$ energia ruchu była stała (z dokładnością do błędów numerycznych) dla wszystkich wylosowanych punktów ze zbioru $\mathbb{S}$. Z tego powodu wartości współczynników korelacji nie zawierają żadnych przydatnych informacji. Podobnie dla integratora Brocketta, punktu docelowego $\boldsymbol{x}_f = (0, 1)$ i obu funkcji wyjścia.
- W przypadku integratora Brocketta współczynniki korelacji bardzo rzadko przyjmują znaczące wartości ($|\rho_{\boldsymbol{EN},\boldsymbol{LIE}}| > 0.5$). Dla tego układu Algorytm 8.1 nie jest odpowiednim narzędziem do określania jakości konfiguracji początkowej $\boldsymbol{q}_0$. Przedmiotem dalszych badań jest ustalenie, czy jest to cecha tego układu, czy wszystkich układów nilpotentnych.

Tabela 8.2 Współczynniki korelacji pomiędzy zbiorami wartości ***LIE*** oraz ***EN*** dla integratora Brocketta.

układ	integrator Brocketta					
$\boldsymbol{x}$	(q_1, q_3)			(q_2, q_3)		
$\boldsymbol{x}_f$	$\begin{pmatrix}1\\0\end{pmatrix}$	$\begin{pmatrix}0\\1\end{pmatrix}$	$\begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}}\end{pmatrix}$	$\begin{pmatrix}1\\0\end{pmatrix}$	$\begin{pmatrix}0\\1\end{pmatrix}$	$\begin{pmatrix}\frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}}\end{pmatrix}$
Δ_x	Współczynniki korelacji rankingowej Spearmana					
0.1	0.37	0.79	-0.23	0.18	0.76	-0.15
0.2	0.28	0.94	-0.29	0.15	0.85	-0.07
0.5	0.16	0.69	-0.38	0.19	0.65	-0.25
0.7	0.29	0.70	-0.31	0.14	0.59	-0.10
1	-0.08	0.43	-0.20	0.29	0.54	-0.13
3	0.29	-0.13	-0.28	0.20	0.09	-0.06
	Współczynniki korelacji liniowej Pearsona					
0.1	-0.08	0.75	-0.28	0.26	0.74	0.01
0.2	-0.18	0.92	-0.27	0.39	0.79	-0.04
0.5	-0.16	0.67	-0.35	0.33	0.65	-0.16
0.7	-0.09	0.70	-0.28	0.48	0.59	0.03
1	-0.36	0.44	-0.23	0.36	0.50	-0.00
3	-0.02	-0.41	-0.28	0.36	-0.15	0.02



Rysunek 8.1 Ścieżki ruchu jednokołowca z optymalnej konfiguracji $\boldsymbol{q}_0$ dla funkcji jakości $en(\boldsymbol{q}_0$ lub $lie(\boldsymbol{q}_0)$ i dwóch punktów docelowych $\boldsymbol{x}_f$ oznaczonych gwiazdką.



Rysunek 8.2 Rzuty na płaszczyznę xy ścieżek ruchu samochodu kinematycznego z optymalnej konfiguracji $\boldsymbol{q}_0$ dla funkcji jakości $en(\boldsymbol{q}_0)$ lub $lie(\boldsymbol{q}_0)$, trzech punktów docelowych $\boldsymbol{x}_f$ oznaczonych gwiazdką i funkcji wyjścia $\boldsymbol{x} = (q_1, q_2, q_3)$.

Rozdział 9

Przedwczesna zbieżność

Pożądaną własnością każdego algorytmu planowania ruchu jest zbieżność, która gwarantuje osiągnięcie dowolnego punkt docelowego. Dla układów sterowania opisywanych za pomocą równań różniczkowych zwyczajnych najczęściej stosowaną techniką dowodzenia zbieżności jest podejście bazujące na konstrukcji funkcji Lapunowa [66]. Unimodalna, nieujemna funkcja Lapunowa osiąga wartość zero w stanie docelowym, a zasada określająca aktualne sterowania wymusza malejącą monotoniczność tej funkcji wzdłuż planowanej trajektorii. Niestety, takie podejście jest niewystarczające dla układów nieholonomicznych, gdyż dla tych układów mogą nie istnieć sterowania zmniejszające wartość funkcji Lapunowa (np. odległości do punktu docelowego) w sposób monotoniczny. Z tego powodu warunek określający zbieżność powinien być osłabiony, a jednym ze sposobów jest wybór ciągu kolejnych podcelów planowania w taki sposób, by funkcja Lapunowa malała monotonicznie dla tego ciągu.

Przedwczesna zbieżność jest terminem znanym z algorytmów ewolucyjnych i określa sytuację, gdy populacja punktów przeszukujących przestrzeń w poszukiwaniu globalnego optimum nieznanej funkcji skupia się w okolicy jednego z optimów lokalnych, traci różnorodność i nie może opuścić tego otoczenia [86]. Przedwczesna zbieżność jest właściwością dalece niepożądaną, więc istnieją literaturowe techniki modyfikacji algorytmów ewolucyjnych pozwalające na zachowanie różnorodności w populacji i uniknięcie przedwczesnej zbieżności [43]. W niniejszym rozdziale przedstawiono analogon omawianego zjawiska występujący w planowaniu ruchu metodą Lie-algebraiczną i zaprezentowano algorytm planowania ruchu unikający przedwczesnej zbieżności.

Z definicji układ nieholonomiczny jest sterowalny, więc zaprojektowanie algorytmu planowania ruchu zapewniającego zbieżność nie jest teoretycznie trudne. Jednak w konkretnej konfiguracji początkowej $\boldsymbol{q}_0$ ruch w różnych kierunkach może mieć diametralnie różny stopień trudności (np. energię sterowań, patrz rozdział 7). Przedwczesną zbieżnością dla algorytmów planowania ruchu określamy sytuację, gdy bezwzględna zbieżność w i-tym lokalnym planowaniu może zostać osiągnięta jedynie poprzez ruch w kosztownych energetycznie kierunkach. Aby uniknąć takiej sytuacji należałoby wybrać odpowiednie podcele dla poprzednich $i-1$ lokalnych planowań. W literaturze robotycznej problem znalezienia sterowań optymalnych energetycznie został rozwiązany dla prostych [32] lub małowymiarowych, nilpotentyzowanych [2] układów. Proponowane w tym rozdziale podejście może być zastosowane do ogólnych bezdryfowych układów o dowolnej wymiarowości, jednak otrzymane sterowania są jedynie suboptymalne.

Procedura oceny jakości konfiguracji początkowej $\boldsymbol{q}_a$ w kontekście ruchu do punktu $\boldsymbol{x}_b$, opisana Algorytmem 8.1, może być również wykorzystana do oceny trudności ruchu między konfiguracjami $\boldsymbol{q}_a$ oraz $\boldsymbol{q}_b$ generując wartość $lie(\boldsymbol{q}_a, \mathbf{id}(\boldsymbol{q}_b))$. Kroki pozwalające na pla-

nowanie ruchu z unikaniem przedwczesnej zbieżności zebrano w Algorytmie 9.1.

Algorytm 9.1 Algorytm lokalnego planowania ruchu przeciwdziałający przedwczesnej zbieżności.

Dane wejściowe: Bezdryfowy układ nieholonomiczny (2.1), konfiguracje początkowa $\boldsymbol{q}_0 \in \mathbb{R}^n$ oraz docelowa $\boldsymbol{q}_f \in \mathbb{R}^n$, dokładność osiągnięcia celu $\rho > 0$, wartość progowa funkcji jakości δ, maksymalny zakres pojedynczego planowania Δ oraz czas pojedynczego ruchu T.

Krok 1. Podstawić konfigurację początkową jako aktualną $\boldsymbol{q}_c \leftarrow \boldsymbol{q}_0$. Obliczyć wszystkie pola wektorowe z bazy Ph. Halla do k-tej warstwy, gwarantujące sterowalność w krótkim czasie (STLC) układu (2.1). Zbiory pól wektorowych z danej warstwy oznaczyć $\boldsymbol{H}^1, \ldots, \boldsymbol{H}^k$. Zainicjować pustą trajektorię wynikową, z czasem początkowym $t^\star \leftarrow 0$

Krok 2. Obliczyć $d \leftarrow \|\boldsymbol{q}_c - \boldsymbol{q}_f\|$ i sprawdzić warunek stopu $d < \rho$. Jeśli warunek został spełniony – zakończyć algorytm i zwrócić trajektorię wynikową. Jeśli $d < \Delta$ zmniejszyć $\Delta \leftarrow d$.

Krok 3. Dla konfiguracji $\boldsymbol{q}_c$ obliczyć val $\leftarrow lie(\boldsymbol{q}_c, \mathbf{id}(\boldsymbol{q}_f))$ używając Algorytmu 8.1. Jeśli val $\geqslant \delta$ za następny podcel $\boldsymbol{q}^\star$ przyjąć $\boldsymbol{q}_f$, $\boldsymbol{q}^\star \leftarrow \boldsymbol{q}_f$ i przejść do Kroku 6.

Krok 4. Wybrać zbiór konfiguracji testowych $\mathbb{S} = \{\boldsymbol{q}_1, \ldots, \boldsymbol{q}_r\}$. Ocenić każdą z konfiguracji testowych $\boldsymbol{q}_i \in \mathbb{S}$, $i = 1. \ldots, r$ za pomocą Algorytmu 8.1

$$f(\boldsymbol{q}_i) \leftarrow lie^\star(\boldsymbol{q}_c, \mathbf{id}(\boldsymbol{q}_i)) \cdot lie(\boldsymbol{q}_i, \mathbf{id}(\boldsymbol{q}_f)). \tag{9.1}$$

Przypisać uzyskaną ocenę do konkretnej konfiguracji testowej.

Krok 5. Wybrać najlepszą z konfiguracji testowych jako nowy podcel $\boldsymbol{q}^\star$

$$\boldsymbol{q}^\star : \quad f(\boldsymbol{q}^\star) \leftarrow \max_{\boldsymbol{q}_i \in \mathbb{S}} f(\boldsymbol{q}_i). \tag{9.2}$$

Krok 6. Przeprowadzić pojedyncze planowanie ruchu z konfiguracji $\boldsymbol{q}_c$ do konfiguracji $\boldsymbol{q}^\star$ z maksymalnym zakresem planowania Δ w czasie T. Wynikowym sterowaniem jest $\boldsymbol{u}(\cdot)$.

Krok 7. Zastosować sterowania na układzie (2.1) w konfiguracji $\boldsymbol{q}_c$. Otrzymać rzeczywistą konfigurację końcową dla pojedynczego planowania $\tilde{\boldsymbol{q}}^\star$ oraz trajektorię częściową pomiędzy $\boldsymbol{q}_c$ i $\tilde{\boldsymbol{q}}^\star$. Dokleić otrzymaną trajektorię częściową do wynikowej na przedziale $[t^\star, t^\star + T]$. Przyjąć czas początkowy następnej trajektorii częściowej $t^\star \leftarrow t^\star + T$.

Krok 8. Określić nową konfigurację aktualną $\boldsymbol{q}_c \leftarrow \tilde{\boldsymbol{q}}^\star$ i przejść do Kroku 2.

Uwagi do Algorytmu 9.1:

- Celem algorytmu 9.1 jest znalezienie ciągu podcelów kolejnych lokalnych planowań, minimalizujących trudność ruchu wynikowego.
- W Kroku 3. korekcja położenia podcelu następuje w przypadku ruchu w zbyt kosztownym kierunku.
- Aby zachować zbieżność algorytmu, zbiór testowych konfiguracji $\mathbb{S}$ powinien zawierać konfiguracje zmniejszające odległość euklidesową do celu.
- W Kroku 6. do planowania ruchu można wykorzystać metodę bazującą na gCBHD (strona 26).
- Kolejne podcele (jak również konfiguracje testowe) powinny znajdować się odpowiednio blisko aktualnej konfiguracji, tak aby błąd związany z przybliżeniem formuły gCBHD był mały.

- W Kroku 7. osiągnięta konfiguracja $\tilde{\boldsymbol{q}}^\star$ prawdopodobnie nie będzie równa założonemu podcelowi $\boldsymbol{q}^\star$. Wielkość błędu $\|\tilde{\boldsymbol{q}}^\star - \boldsymbol{q}^\star\|$ może być zmniejszona poprzez zmniejszenie parametru Δ.
- Liczba konfiguracji testowych może być różna dla każdej iteracji.
- Zasadę generowania konfiguracji testowych określa projektant algorytmu. Jedną z propozycji jest losowy wybór punktów z sąsiedztwa aktualnej konfiguracji.

Symulacje ilustrujące działanie Algorytmu 9.1 przeprowadzono dla modelu jednokołowca A.1. Zdefiniowano dwa zadania polegające na zaplanowaniu ruchu układu z konfiguracji początkowej $\boldsymbol{q}_0 = \boldsymbol{0}$ do konfiguracji docelowej $\boldsymbol{q}_f = (0, 10, 0)^T$ (Zadanie 1) lub $\boldsymbol{q}_f = (7, 7, \pi/2)^T$ (Zadanie 2). Zadanie 1 jest szczególnie interesujące, gdyż wymaga ruchu w najtrudniejszym kierunku i wywołuje przedwczesną zbieżność dla podstawowej metody Lie-algebraicznej. Zadanie 2 spełnia rolę przypadku typowego. Dla obu zadań sterowania sparametryzowano za pomocą bazy harmonicznej (podrozdział 1.5), wykorzystując harmoniczne do pierwszej włącznie

$$\begin{cases} u_1 = \frac{1}{\sqrt{T}}(p_{1,1} + p_{1,2}\sqrt{2}\sin(2\pi t) + p_{1,3}\sqrt{2}\cos(2\pi t)), \\ u_2 = \frac{1}{\sqrt{T}}(p_{2,1} + p_{2,2}\sqrt{2}\sin(2\pi t) + p_{2,3}\sqrt{2}\cos(2\pi t)). \end{cases} \tag{9.3}$$

Czas planowania pojedynczego zadania planowania wynosił $T = 1$. Zbiór konfiguracji testowych $\mathbb{S}$ składał się z 10000 punktów otrzymywanych każdorazowo za pomocą formuły

$$\boldsymbol{q}_i = \boldsymbol{q}_c + \Delta\frac{\boldsymbol{q}_{rand}}{\|\boldsymbol{q}_{rand}\|}, \quad \boldsymbol{q}_{rand} = \begin{pmatrix} \mathrm{rand}(-1,1) \\ \mathrm{rand}(-1,1) \\ \mathrm{rand}(-1,1) \end{pmatrix}, \tag{9.4}$$

gdzie funkcja $\mathrm{rand}(a, b)$ generuje liczbę z przedziału $[a, b]$ według rozkładu jednostajnego ciągłego, oraz spełniających warunek zbieżności

$$\|\boldsymbol{q}_f - \boldsymbol{q}_i\| < \|\boldsymbol{q}_f - \boldsymbol{q}_c\|. \tag{9.5}$$

Dla każdego z zadań wybrano następujący zestaw wartości parametrów algorytmu: $\delta \in \{0, 0.5, 0.7, 0.8, 0.9, 0.95\}$ oraz $\Delta = \{0.1, 0.2, 0.5, 1\}$. Przypadek $\delta = 0$ odpowiada podstawowej metodzie Lie-algebraicznej i służy celom porównawczym.

Do oceny trajektorii wynikowych obliczono całkowitą energię sterowań oraz długość wynikowej ścieżki (uwzględniającą jedynie współrzędne pozycyjne, ignorując kątowe). Wyniki symulacji zebrano w tab. 9.1. Oznaczenia #z.p. oraz %z.p. oznaczają, odpowiednio, liczbę i procent zmodyfikowanych podcelów. Na rys. 9.1 i 9.2 przedstawiono ścieżki wynikowe dla Zadania 1 i Zadania 2, odpowiednio. Ścieżkami planowanymi są ścieżki uzyskane za pomocą formuły gCBHD (linie proste pomiędzy kolejnymi punktami $\tilde{\boldsymbol{q}}^\star$). Ścieżki rzeczywiste powstały poprzez zastosowanie sterowań uzyskanych z formuły gCBHD do układu (2.1) i scałkowanie numeryczne powstałych równań. Na rys. 9.3 przedstawiono wartości sterowań w czasie dla Zadania 1, $\Delta = 1$ i różnych parametrów δ.

Na podstawie wyników symulacji można wyciągnąć kilka interesujących wniosków:

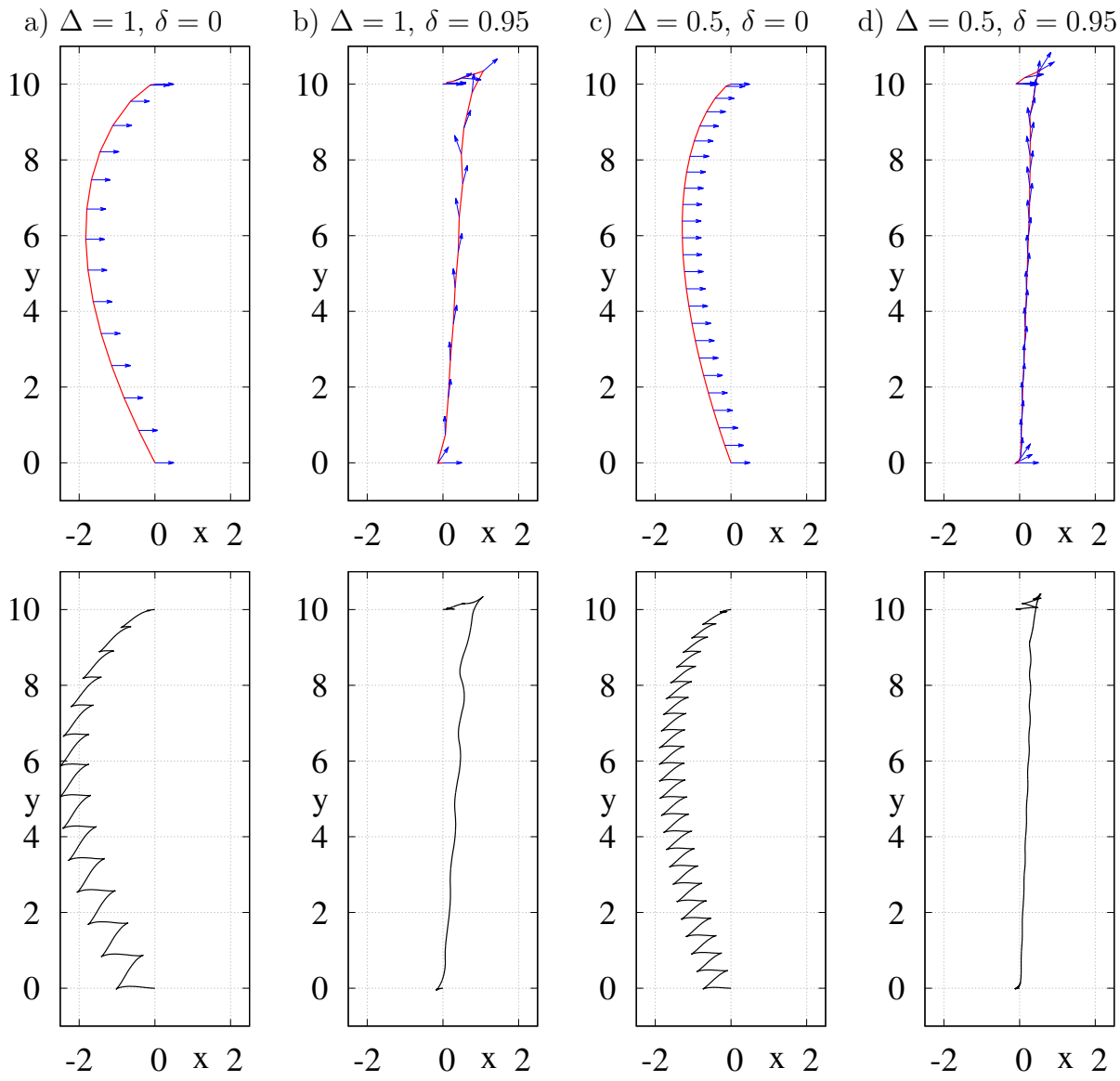
- Energia ruchu otrzymanego z podstawowej metody Lie-algebraicznej w bardzo małym stopniu zależy do maksymalnego zakresu pojedynczego planowania Δ.
- W przypadku użycia Algorytmu 9.1 długość ścieżki i energia ruchu silnie zależy od maksymalnego zakresu pojedynczego planowania Δ oraz progu δ. Wraz ze wzrostem Δ, maleje długość ścieżki. Dla energii zależność od Δ nie jest monotoniczna. Dla różnych

Tabela 9.1 Energia ruchu, długość ścieżki, liczba (#z.p.) oraz procent (%z.p.) zmodyfikowanych podcelów dla jednokołowca i Algorytmu 9.1 o różnych parametrach.

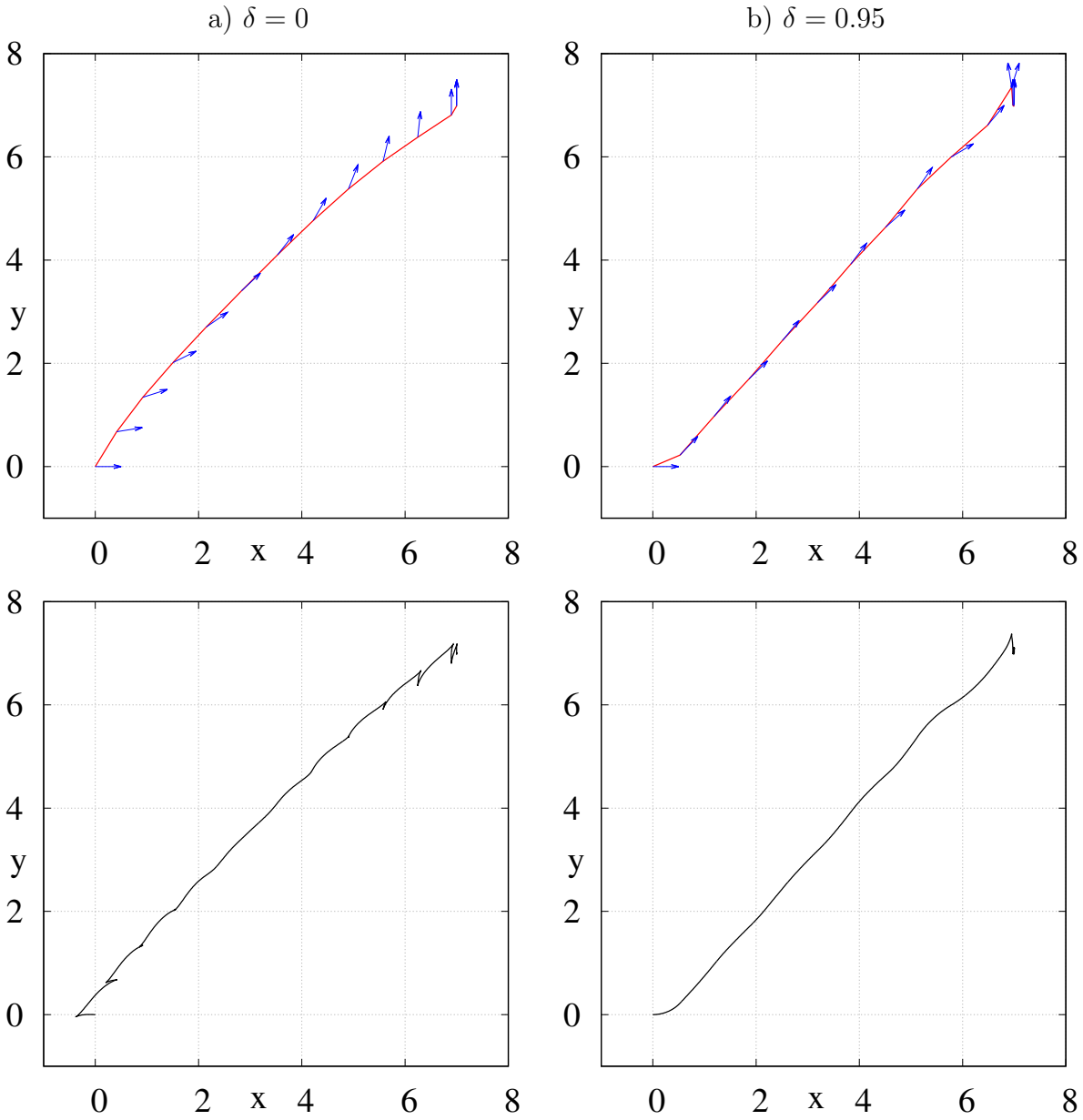
		Zadanie 1, $\boldsymbol{q}_f = (0, 10, 0)^T$				Zadanie 2, $\boldsymbol{q}_f = (7, 7, \pi/2)^T$			
δ	Δ	energia	długość	#z.p.	%z.p.	energia	długość	#z.p.	%z.p.
0	1	131.91	25.20	0	0.00	41.05	13.32	0	0.00
	0.5	133.48	34.59	0	0.00	40.13	16.96	0	0.00
	0.2	136.79	53.93	0	0.00	42.67	26.39	0	0.00
	0.1	139.29	76.01	0	0.00	45.35	38.48	0	0.00
0.95	1	15.73	12.66	12	66.67	11.75	11.17	10	66.67
	0.5	11.49	12.87	19	61.29	7.34	10.80	12	48.00
	0.2	4.84	12.30	39	51.32	3.31	10.74	24	38.10
	0.1	4.10	13.00	69	44.23	3.16	12.05	39	29.77
0.9	1	19.59	12.93	12	63.16	16.00	11.85	9	52.94
	0.5	14.30	13.38	17	53.12	8.92	11.60	12	42.86
	0.2	9.45	13.65	33	41.77	7.10	12.73	19	28.79
	0.1	15.97	30.76	62	40.26	9.12	17.71	32	24.81
0.8	1	28.14	14.64	10	52.63	21.01	12.60	10	50.00
	0.5	22.63	14.44	13	39.39	17.14	12.73	7	25.93
	0.2	37.73	32.17	27	36.49	19.44	18.43	13	20.63
	0.1	41.33	45.29	57	38.26	23.73	28.30	22	18.03
0.7	1	37.19	14.80	7	38.89	30.47	12.60	4	28.57
	0.5	46.98	18.76	10	32.26	28.50	14.60	4	15.38
	0.2	60.77	34.00	23	30.67	34.81	24.29	6	10.34
	0.1	68.43	56.15	49	34.27	38.43	35.48	10	8.93
0.5	1	99.03	22.52	3	20.00	41.05	13.32	0	0.00
	0.5	112.87	31.18	2	7.41	40.13	16.96	0	0.00
	0.2	111.90	49.60	14	22.22	42.67	26.39	0	0.00
	0.1	115.82	70.26	27	21.77	45.35	38.48	0	0.00

wartości δ minimalna energia ruchu przypada na różne wartości Δ, co tłumaczymy istnieniem dwóch przeciwstawnych czynników wpływających na energię. Im większe δ tym mniejsza energia elementarnych ruchów dla "dostatecznie dobrych" ($lie(\boldsymbol{q}_c, \mathbf{id}(\boldsymbol{q}_f)) \geqslant \delta$) konfiguracji. Wraz ze wzrostem Δ rośnie jednak szansa, że ruch z "dostatecznie dobrej" skończy się w konfiguracji "trudnej", tj. nie spełniającej powyższego warunku. Również im większe Δ, tym elementarne ruchy z "trudnej" konfiguracji są bardziej kosztowne energetycznie.

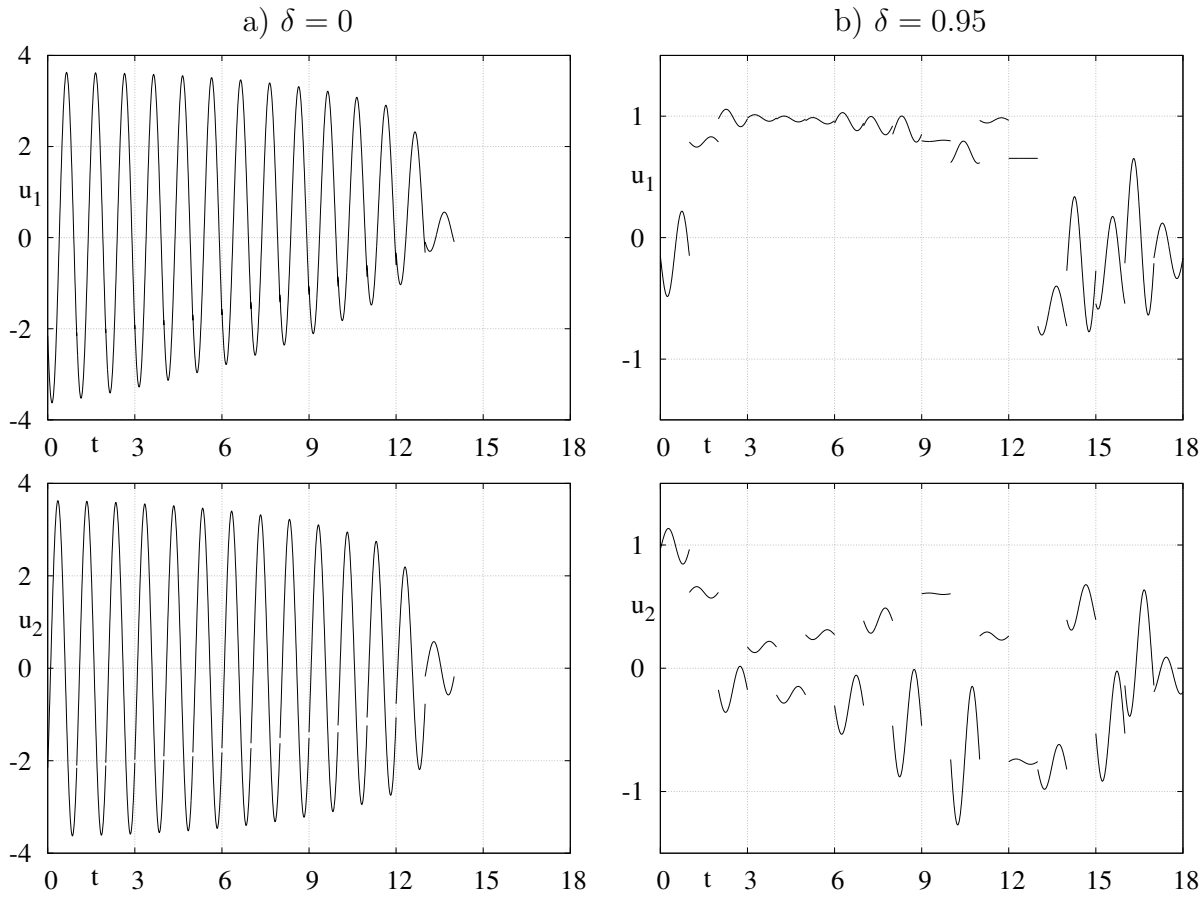
- W przypadku małych wartości progu δ, część Algorytmu 9.1 odpowiedzialna za przeciwdziałanie przedwczesnej zbieżności jest bardzo rzadko wykorzystywana.
- Próg δ powinien mieścić się w przedziale $[0, 1/T]$, gdzie T jest czasem ruchu dla elementarnego planowania. Efektywna wartość progu δ jest zależna od modelu.
- W przypadku gdy Algorytm 9.1 skutecznie przeciwdziała przedwczesnej zbieżności trajektorie wynikowe są gładsze, a obszerność ruchu mniejsza.
- Podstawowa metoda Lie-algebraiczna (Algorytm 9.1 przy $\delta = 0$) wybiera podcele interpolując liniowo punkty między $\boldsymbol{q}_0$ i $\boldsymbol{q}_f$. Dla Zadania 1 oznacza to zachowanie orientacji ($\theta = 0$) dla każdego podcelu i ruch wzdłuż najbardziej kosztownego pola $[\boldsymbol{g}_1, \boldsymbol{g}_2]$.



Rysunek 9.1 Ścieżki jednokołowca wygenerowane Algorytmem 9.1. Zadanie 1 z różnymi wartościami parametrów δ i Δ. Górne rysunki – ścieżka planowana z oznaczoną orientacją, dolne – ścieżka rzeczywista.



Rysunek 9.2 Ścieżki jednokołowca uzyskane Algorytmem 9.1. Zadanie 2, $\Delta = 1$ oraz kilka wartości parametrów δ. Górne rysunki – ścieżka planowana z oznaczoną orientacją, dolne – ścieżka rzeczywista.



Rysunek 9.3 Sterowania dla jednokołowca uzyskane z Algorytmu 9.1 dla Zadania 1, $\Delta = 1$ i różnych wartości parametrów δ.

Rozdział 10

Suboptymalność planowania - algorytm Newtona

W poprzednich rozdziałach przedstawiono, między innymi, badania wpływu wybranych parametrów metody Lie-algebraicznej planowania ruchu wykorzystującej formułę gCBHD na kształt trajektorii wynikowej w przestrzeniach konfiguracyjnej oraz zadaniowej. Wykorzystywana procedura polega na utworzeniu za pomocą formuły gCBHD nieliniowego równania macierzowego (2.56) (strona 29, gdzie macierz $\boldsymbol{F}_{i^\star}(\boldsymbol{q}_0)$ jest równa $\boldsymbol{F}^{\mathbb{Q}}_{i^\star}(\boldsymbol{q}_0)$ bądź $\boldsymbol{F}^{\mathbb{X}}_{i^\star}(\boldsymbol{q}_0)$, w zależności od planowania w przestrzeni konfiguracyjnej czy zadaniowej, odpowiednio.

W celu rozwiązania równania (2.56) najczęściej (poza rozdz. 4 i 7) stosowano algorytm Newtona, lub jego wersję z optymalizacją w przestrzeni zerowej (podrozdział 1.4, zobacz też str. 54). Przemieszczenie układu nieholonomicznego $\boldsymbol{z}(t, \boldsymbol{q}_0, \boldsymbol{u}(\boldsymbol{p}))$ jest funkcją nieliniową, ponadto z przyczyn obliczeniowych jako $\boldsymbol{z}(t, \boldsymbol{q}_0, \boldsymbol{u}(\boldsymbol{p}))$ traktuje się prawą stronę wzoru (2.56). Z tych powodów równanie

$$(\boldsymbol{q}_f - \boldsymbol{q}_0) - \boldsymbol{F}_{i^\star}(\boldsymbol{q}_0)\, \boldsymbol{\alpha}(\boldsymbol{u}(\boldsymbol{p})) = \boldsymbol{0} \tag{10.1}$$

ma więcej niż jedno rozwiązanie ze względu na $\boldsymbol{p}$. Trajektorie uzyskane po zastosowaniu sterowań $\boldsymbol{u}(\boldsymbol{p})$ dla wielu rozwiązań równania (10.1) różnią się oczywiście kształtem. Ważniejszą różnicą między rozwiązaniami (10.1) jest jednak wpływ na wynikową trajektorię pól wektorowych wyższych stopni, nie uwzględnionych w $\boldsymbol{F}_{i^\star}(\boldsymbol{q}_0)$. Jeśli wpływ ten jest duży, to rzeczywisty punkt końcowy $\boldsymbol{q}^\star_f$ może znacząco odbiegać od docelowego $\boldsymbol{q}_f$.

W niniejszym rozdziale zbadano skalę różnic między trajektoriami powstałymi z rozwiązań równania (10.1). Przeanalizowano także możliwości ich wykorzystania do otrzymania suboptymalnych trajektorii według kilku funkcji kryterialnych. Badania przeprowadzono symulacyjnie dla modelu jednokołowca A.1 oraz samochodu kinematycznego A.2, planując zarówno w przestrzeni konfiguracyjnej jak i zadaniowej. Przeprowadzono planowania z punktu początkowego $\boldsymbol{x}_0 = \boldsymbol{0},\ \boldsymbol{q}_0 = \boldsymbol{0}$, do różnych punktów docelowych $\boldsymbol{x}_f$. Uzasadnienie konieczności przyjęcia konfiguracji początkowej $\boldsymbol{q}_0$ można znaleźć na stronie 75.

Jako funkcje wyjścia $\boldsymbol{k}(\boldsymbol{q})$ zastosowano podzbiory właściwe współrzędnych wektora konfiguracji $\boldsymbol{q}$, wybierając współrzędne ważniejsze z punktu widzenia planowania ruchu, jak również funkcję identycznościową odpowiadającą planowaniu w przestrzeni konfiguracyjnej. Dla jednokołowca funkcjami wyjścia były:

- $\boldsymbol{x} = (q_1, q_2)^T$ – współrzędne położenia $(x, y)^T$,
- $\boldsymbol{x} = (q_1, q_2, q_3)^T$ – funkcja identycznościowa.

Natomaist dla samochodu kinematycznego:

- $\boldsymbol{x} = (q_1, q_2)^T$ – współrzędne położenia $(x, y)^T$,
- $\boldsymbol{x} = (q_1, q_2, q_3)^T$ – współrzędne położenia i orientacja tylnej osi $(x, y, \theta)^T$,
- $\boldsymbol{x} = (q_1, q_2, q_3, q_4)^T$ – funkcja identycznościowa.

Dla wszystkich funkcji wyjścia zastosowano minimalną macierz $\boldsymbol{F}_{i^\star}$, $i^\star = 2$ oraz $i^\star = 3$ dla jednokołowca i samochodu kinematycznego odpowiednio (zobacz przykłady 6 i 7).

Za punkty docelowe $\boldsymbol{x}_f$ dla poszczególnych zadań, w zależności od wymiarowości przestrzeni zadaniowej, wybrano

$$\boldsymbol{x}_f = (0, \delta), \quad \boldsymbol{x}_f = (0, \delta, 0), \quad \boldsymbol{x}_f = (0, \delta, 0, 0), \quad \delta \in \{0.05, 0.1, 0.2, 0.5, 1\}, \tag{10.2}$$

gdyż ruch w kierunku pola wektorowego (tożsamego w punkcie $\boldsymbol{q}_0 = \boldsymbol{0}$ do kierunku q_2) o najwyższym stopniu jest najtrudniejszy. Każde zadanie rozwiązywano dla trzech parametryzacji sterowań

$$\begin{cases} u_1 = \frac{1}{\sqrt{T}}(p_{1,1} + p_{1,2}\sqrt{2}\sin(2\pi t) + p_{1,3}\sqrt{2}\cos(2\pi t)), \\ u_2 = \frac{1}{\sqrt{T}}(p_{2,1} + p_{2,2}\sqrt{2}\sin(2\pi t) + p_{2,3}\sqrt{2}\cos(2\pi t)), \end{cases} \tag{10.3}$$

$$u_1 = \frac{1}{\sqrt{T}}(p_{1,1} + p_{1,2}\sqrt{2}\sin(2\pi t)), \quad u_2 = \frac{1}{\sqrt{T}}(p_{2,1} + p_{2,3}\sqrt{2}\cos(2\pi t)), \tag{10.4}$$

$$u_1 = \frac{1}{\sqrt{T}}(p_{1,1} + p_{1,3}\sqrt{2}\cos(2\pi t)), \quad u_2 = \frac{1}{\sqrt{T}}(p_{2,1} + p_{2,2}\sqrt{2}\sin(2\pi t)). \tag{10.5}$$

Jako kryteria optymalizacyjne przyjęto:

1. całkowitą energię sterowań (zobacz (5.6)),
2. długość ścieżki biorąc pod uwagę współrzędne pozycyjne (x, y),
3. odległość pomiędzy punktem docelowym a rzeczywistym punktem końcowym

$$\| \boldsymbol{x}_f - \boldsymbol{x}_f^\star \| . \tag{10.6}$$

Rozwiązania równania (10.1) otrzymano z zastosowaniem algorytmu Newtona (dalej oznaczanego AN) oraz algorytmu Newtona z optymalizacją energii w przestrzeni zerowej (ANOPZ) dla wybranych wartości wektora parametrów początkowych $\boldsymbol{p}_0$. Zbiór $\mathbb{P}_0$ składał się ze 100 wektorów parametrów $\boldsymbol{p}_{0,i}$, z losową wartością każdej współrzędnej, według rozkładu jednostajnego na przedziale $[-1, 1]$. Lokalność algorytmu Newtona umożliwia znalezienia różnych rozwiązań w zależności od przyjętej wartości $\boldsymbol{p}_0$.

W tabelach 10.1, 10.2 zebrano dane określające obliczeniową jakość otrzymanych rozwiązań, a wykorzystane skróty oznaczają:

n.d. – najlepsza dokładność, minimalna wartość (10.6),

d. – dokładność, procent ścieżek wystarczająco dokładnych,

n. – procent zadań dla których algorytm Newtona nie podał rozwiązania z powodu problemów numerycznych z odwracaniem macierzy źle uwarunkowanych,

c.o. – średni czas rozwiązania zadania, zawierajacy: symboliczne generowanie bazy Ph. Halla, określanie $\boldsymbol{z}(t, \boldsymbol{q}_0, \boldsymbol{u}(\boldsymbol{p}))$ i Jakobianu oraz numeryczne wyliczanie trajektorii wynikowej.

Ścieżka jest uznawana za wystarczająco dokładną, gdy jej punkt końcowy $\boldsymbol{x}_f^\star$ jest odpowiednio bliski punktowi docelowemu $\boldsymbol{x}_f$, według wzoru

$$\| \boldsymbol{x}_f - \boldsymbol{x}_f^\star \| < \eta \| \boldsymbol{x}_f - \boldsymbol{x}_0 \|, \tag{10.7}$$

gdzie $\boldsymbol{x}_0 = \boldsymbol{k}(\boldsymbol{q}_0)$, a η to współczynnik dokładności. Wyniki zawarte w tab. 10.1 i 10.2 odpowiadają współczynnikowi $\eta = 0.3$.

Symulacje przeprowadzono wykorzystując autorskie oprogramowanie napisane w pakiecie *Mathematica*, wersja 11.3. Czasy zamieszczone w tab. 10.1 i 10.2 zostały osiągnięte na komputerze osobistym, z procesorem *Intel Core i5-8400, 2.80 GHz* $\times$ *6* i pamięcią 24 GB RAM.

W tab. 10.3 i 10.4 zebrano statystyczne podsumowanie wyników zadań. Znajdują się tam przedziały długości ścieżki oraz energii sterowań dla tych ścieżek wynikowych, które spełniły warunek dokładności (10.2) dla $\eta = 0.3$. Jeśli żadna ze ścieżek nie spełniła warunku dla $\eta = 0.3$, to w pole tabeli wpisany jest myślnik lub cały wiersz został usunięty (w przypadku gdy obie wersje algorytmu Newtona nie zwróciły odpowiednich wyników).

Rys. 10.1 i 10.2 przedstawiają sześć charakterystycznych ścieżek (maksymalne i minimalne wartości długości ścieżki i energii sterowań, najdokładniejsze ścieżki z obu algorytmów Newtona) dla sześciu zadań, po trzy na model. W przypadku samochodu kinematycznego i funkcji $\boldsymbol{k}(\boldsymbol{q}) = (q_1, q_2, q_3)^T$ przedstawiono rzuty ścieżki na płaszczyznę xy. Punkt docelowy $\boldsymbol{x}_f$ na rysunkach został zaznaczony gwiazdką. W tab. 10.5 przedstawiono dane identyfikacyjne przedstawionych zadań.

Na podstawie wyników symulacji można sformułować następujące wnioski:

- Długość ścieżki i energia sterowań silnie zależą od wektora parametrów początkowych $\boldsymbol{p}_0$. Zatem istnieje potencjał optymalizacyjny poprzez równoległy start algorytmu Newtona z wielu punktów początkowych i wybranie najlepszej (według danych kryteriów) trajektorii wynikowej.
- W wielu przypadkach AN generował lepsze rezultaty, także pod względem energetycznym, niż ANOPZ optymalizujący energię. Jednocześnie wariancja wyników ANOPZ jest zdecydowanie mniejsza, przy podobnych wartościach minimalnych długości i energii, w porównaniu do AN. Wydaje się, że AN łatwiej penetruje przestrzeń poszukiwań, natomiast ANOPZ trzyma się dobrych rozwiązań, niekoniecznie znajdując najlepsze.
- Trajektorie powstałe za pomocą ANOPZ mają tendencję do grupowania się w klastry, zawierające podobne trajektorie. Tej właściwości nie zaobserwowano dla AN.
- Procent nieudanych obliczeń z powodów numerycznych jest dużo wyższy dla krótszych parametryzacji. Co ciekawe, dla niektórych zadań obie czteroelementowe parametryzacje ((10.4), (10.5)) mają znacząco różny procent nieudanych obliczeń.
- Czas obliczeń ANOPZ nie jest znacząco wyższy niż dla AN.
- Dla jednokołowca i wyjściowej funkcji identycznościowej zdecydowana większość ścieżek była wystarczająco dokładna. Jest to spowodowane faktem, że układ jest mało skomplikowany, a potencjalnych rozwiązań jest niewiele. Dla funkcji wyjścia mniej wymiarowej liczba rozwiązań drastycznie rośnie, maleje więc odsetek ścieżek wystarczająco dokładnych. Niskowymiarowa funkcja wyjścia jest też mniej restrykcyjna, więc ścieżki dostatecznie dokładne są krótsze i mniej kosztowne energetycznie.
- Trudność zadania rośnie wraz ze wzrostem wymiarowości funkcji wyjścia oraz wektora parametrów. Potrzebny jest więc kompromis pomiędzy złożonością obliczeniową, a jakością otrzymanej ścieżki. Nie dotyczy to funkcji identycznościowej (mającej największą

Tabela 10.1 Dokładność, % nieudanych obliczeń i czas obliczeń dla jednokołowca.

Zadanie			AN				ANOPZ			
$\boldsymbol{k}(\boldsymbol{q})$	$\boldsymbol{u}$	δ	n.d.	d. [%]	n. [%]	c.o. [s]	n.d.	d. [%]	n. [%]	c.o. [s]
id.	(10.3)	0.05	0.000	100	0	1.6	0.001	100	0	1.6
		0.1	0.000	99	0	0.8	0.001	100	0	0.9
		0.2	0.001	97	0	0.8	0.002	100	0	0.8
		0.5	0.010	51	0	0.8	0.011	67	0	0.8
		1	0.037	39	0	0.8	0.046	33	0	0.8
	(10.4)	0.05	0.000	100	0	0.4	0.000	100	0	0.5
		0.1	0.000	100	0	0.4	0.001	100	0	0.4
		0.2	0.000	100	0	0.4	0.002	100	0	0.4
		0.5	0.006	100	0	0.4	0.010	100	0	0.4
		1	0.027	100	0	0.3	0.040	100	0	0.4
	(10.5)	0.05	0.004	99	1	0.4	0.005	100	0	0.5
		0.1	0.008	98	0	0.4	0.017	100	0	0.4
		0.2	0.034	78	0	0.3	0.049	100	0	0.4
		0.5	0.158	0	0	0.3	0.195	0	0	0.3
		1	0.459	0	0	0.3	0.546	0	0	0.3
$\begin{pmatrix} q_1 \\ q_2 \end{pmatrix}$	(10.3)	0.05	0.001	73	0	0.7	0.006	100	0	0.7
		0.1	0.001	65	0	0.7	0.000	99	1	0.8
		0.2	0.002	53	0	0.7	0.012	2	0	0.7
		0.5	0.017	23	0	0.7	0.206	0	1	0.7
		1	0.068	15	0	0.7	0.424	0	1	0.7
	(10.4)	0.05	0.000	58	0	0.3	0.004	81	19	0.5
		0.1	0.001	50	0	0.3	0.023	73	27	0.5
		0.2	0.003	37	0	0.3	0.068	0	21	1.1
		0.5	0.103	5	0	0.3	0.266	0	18	0.3
		1	0.437	0	0	0.3	0.734	0	9	0.3
	(10.5)	0.05	0.000	68	0	0.3	0.005	100	0	0.3
		0.1	0.001	46	2	0.3	0.017	100	0	0.3
		0.2	0.007	47	0	0.3	0.049	100	0	0.3
		0.5	0.016	42	0	0.3	0.195	0	0	0.3
		1	0.056	21	0	0.3	0.546	0	0	0.3

Tabela 10.2 Dokładność, % nieudanych obliczeń i czas obliczeń dla samochodu kinematycznego.

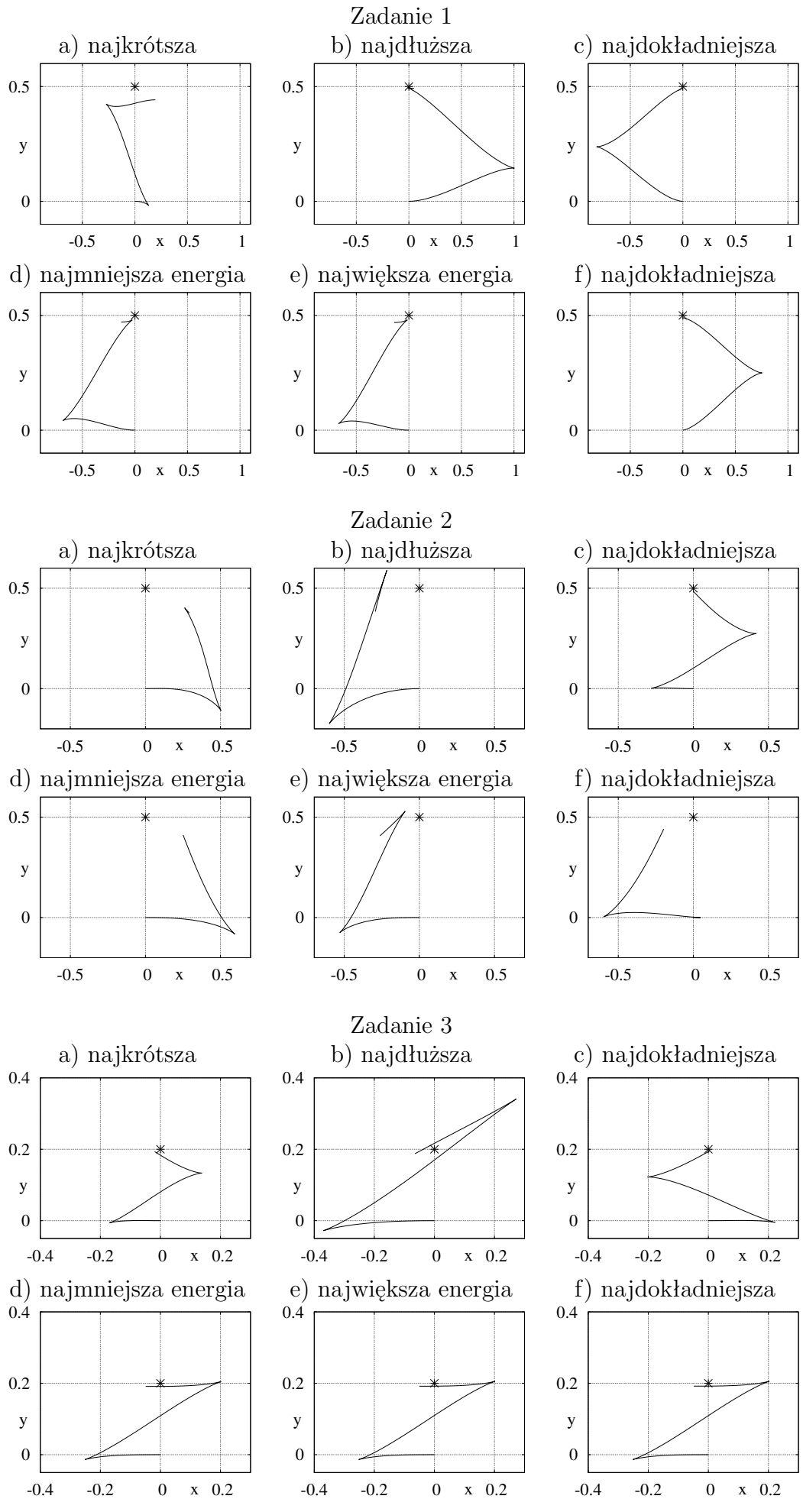
Zadanie			AN				ANOPZ			
$\boldsymbol{k}(\boldsymbol{q})$	$\boldsymbol{u}$	δ	n.d.	d. [%]	n. [%]	c.o. [s]	n.d.	d. [%]	n. [%]	c.o. [s]
id.	(10.3)	0.05	0.003	5	14	35.0	0.003	5	3	35.5
		0.1	0.015	2	14	34.3	0.010	8	5	34.8
		0.2	0.040	1	12	34.6	0.025	4	2	35.2
		0.5	0.047	3	15	37.9	0.019	9	3	38.8
		1	0.182	2	14	33.8	0.018	3	13	34.7
	(10.4)	0.05	0.001	11	1	10.5	0.001	15	5	10.8
		0.1	0.001	3	9	9.4	0.001	18	16	10.0
		0.2	0.244	0	6	9.2	0.000	5	25	9.4
		0.5	0.507	0	16	9.3	0.003	4	34	9.6
		1	0.939	0	30	9.2	0.019	6	44	9.7
	(10.5)	0.05	0.001	56	44	12.2	0.001	56	44	13.1
		0.1	0.001	48	52	13.3	0.001	49	51	14.4
		0.2	0.001	1	99	13.6	0.000	54	46	15.0
		0.5	—	0	100	—	0.001	56	44	16.0
		1	—	0	100	—	0.008	41	59	15.9
$\begin{pmatrix} q_1 \\ q_2 \end{pmatrix}$	(10.3)	0.05	0.003	9	4	36.8	0.006	1	1	36.8
		0.1	0.003	5	6	33.8	0.018	1	1	33.8
		0.2	0.007	4	1	33.9	0.051	1	0	34.0
		0.5	0.127	2	1	33.1	0.172	0	1	33.1
		1	0.328	0	1	35.3	0.381	0	0	35.3
	(10.4)	0.05	0.003	2	27	9.0	0.042	0	0	9.2
		0.1	0.037	0	15	8.3	0.083	0	1	8.4
		0.2	0.095	0	23	8.1	0.164	0	0	8.5
		0.5	0.360	0	6	8.3	0.406	0	0	8.5
		1	0.783	0	4	8.0	0.809	0	0	8.1
	(10.5)	0.05	0.010	3	11	10.7	0.005	67	1	12.7
		0.1	0.028	2	6	10.6	0.017	51	18	10.8
		0.2	0.052	3	9	10.7	0.037	37	6	10.7
		0.5	0.105	2	4	10.7	0.207	0	14	10.8
		1	0.553	0	7	10.6	0.539	0	8	10.7
$\begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix}$	(10.3)	0.05	0.002	66	3	40.9	0.006	93	0	41.0
		0.1	0.005	61	0	42.9	0.018	95	0	42.9
		0.2	0.014	32	4	40.9	0.052	69	0	41.0
		0.5	0.074	16	1	39.7	0.174	0	0	39.8
		1	0.287	1	0	40.4	0.384	0	1	40.6
	(10.4)	0.05	0.007	25	0	14.1	0.007	27	0	14.2
		0.1	0.023	14	0	13.6	0.024	24	0	13.7
		0.2	0.072	0	0	13.6	0.074	0	0	13.7
		0.5	0.303	0	0	13.3	0.304	0	0	13.4
		1	0.788	0	1	13.2	0.815	0	0	13.3
	(10.5)	0.05	0.006	43	1	15.9	0.006	66	0	16.5
		0.1	0.018	10	0	16.4	0.019	65	0	16.6
		0.2	0.070	0	0	15.5	0.056	58	2	15.6
		0.5	0.228	0	0	15.8	0.208	0	2	15.9
		1	0.555	0	0	15.9	0.540	0	1	16.0

Tabela 10.3 Długości ścieżki i energie sterowań dla jednokołowca.

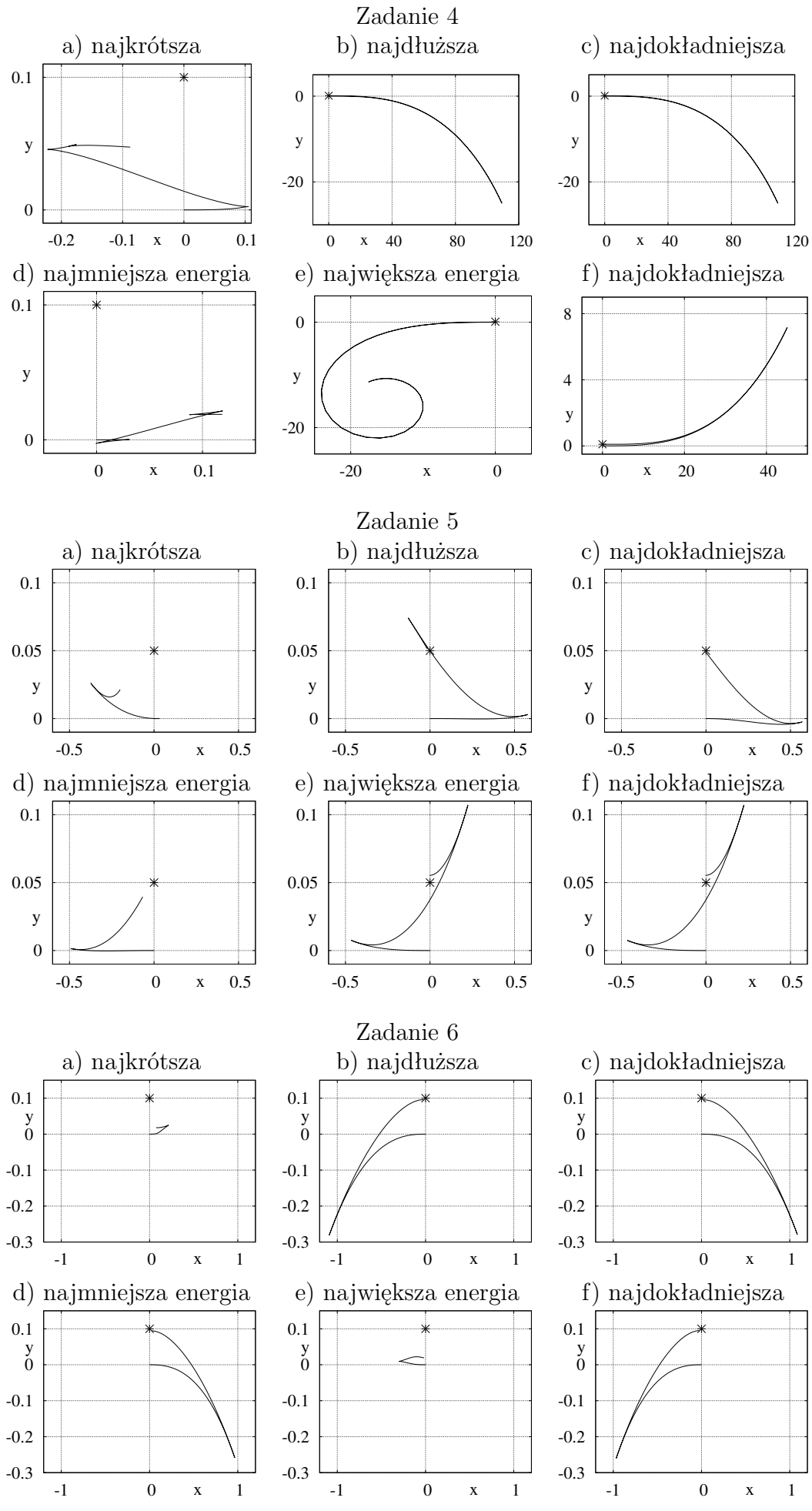
Zadanie			AN		ANOPZ	
$\boldsymbol{k}(\boldsymbol{q})$	$\boldsymbol{u}$	δ	długość	energia	długość	energia
id.	(10.3)	0.05	0.62 ÷ 2.61	0.23 ÷ 1.27	0.62 ÷ 0.65	0.47 ÷ 0.51
		0.1	1.26 ÷ 3.16	0.40 ÷ 1.25	1.24 ÷ 1.26	0.71 ÷ 0.71
		0.2	2.51 ÷ 3.89	0.76 ÷ 1.63	2.50 ÷ 2.51	1.01 ÷ 1.01
		0.5	6.29 ÷ 7.64	1.35 ÷ 2.15	6.27 ÷ 6.28	1.59 ÷ 1.60
		1	12.58 ÷ 14.02	2.05 ÷ 2.75	12.55 ÷ 12.55	2.26 ÷ 2.26
	(10.4)	0.05	0.62 ÷ 1.78	0.28 ÷ 1.18	0.62 ÷ 0.63	0.50 ÷ 0.52
		0.1	1.25 ÷ 3.13	0.33 ÷ 1.26	1.25 ÷ 1.28	0.71 ÷ 0.77
		0.2	2.51 ÷ 3.40	0.73 ÷ 1.52	2.50 ÷ 2.51	1.01 ÷ 1.01
		0.5	6.28 ÷ 6.98	1.28 ÷ 2.02	6.27 ÷ 6.28	1.59 ÷ 1.60
		1	12.57 ÷ 13.56	1.85 ÷ 2.71	12.55 ÷ 12.56	2.26 ÷ 2.26
	(10.5)	0.05	0.62 ÷ 1.83	0.21 ÷ 0.91	0.62 ÷ 0.63	0.50 ÷ 0.52
		0.1	1.25 ÷ 2.91	0.43 ÷ 1.50	1.24 ÷ 1.26	0.71 ÷ 0.71
		0.2	2.51 ÷ 3.27	0.84 ÷ 1.47	2.50 ÷ 2.51	1.01 ÷ 1.01
$\begin{pmatrix} q_1 \\ q_2 \end{pmatrix}$	(10.3)	0.05	0.46 ÷ 2.68	0.25 ÷ 1.02	0.36 ÷ 0.52	0.37 ÷ 0.48
		0.1	0.78 ÷ 3.11	0.36 ÷ 1.13	0.72 ÷ 1.21	0.51 ÷ 0.70
		0.2	1.51 ÷ 4.54	0.60 ÷ 1.35	2.34 ÷ 2.51	0.97 ÷ 1.01
		0.5	4.07 ÷ 7.22	1.18 ÷ 1.61	—	
		1	9.72 ÷ 13.88	1.74 ÷ 2.59	—	
	(10.4)	0.05	0.37 ÷ 1.86	0.22 ÷ 0.81	0.36 ÷ 0.55	0.36 ÷ 0.39
		0.1	0.73 ÷ 2.29	0.33 ÷ 0.90	0.72 ÷ 0.73	0.54 ÷ 0.56
		0.2	1.47 ÷ 2.84	0.60 ÷ 1.01	—	
		0.5	4.46 ÷ 4.98	1.17 ÷ 1.30	—	
	(10.5)	0.05	0.64 ÷ 1.94	0.24 ÷ 0.92	0.62 ÷ 0.63	0.48 ÷ 0.54
		0.1	1.27 ÷ 2.48	0.44 ÷ 1.36	1.24 ÷ 1.26	0.71 ÷ 0.71
		0.2	2.52 ÷ 3.71	0.68 ÷ 1.42	2.50 ÷ 2.51	1.01 ÷ 1.01
		0.5	6.34 ÷ 7.77	1.25 ÷ 2.10	—	
		1	12.83 ÷ 14.10	1.91 ÷ 2.70	—	

Tabela 10.4 Długości ścieżki i energie sterowań dla samochodu kinematycznego.

Zadanie			AN		ANOPZ	
$\boldsymbol{k}(\boldsymbol{q})$	$\boldsymbol{u}$	δ	długość	energia	długość	energia
id.	(10.3)	0.05	12680 ÷ 25975	101 ÷ 145	5272 ÷ 5399	65 ÷ 66
		0.1	55673 ÷ 64650	212 ÷ 228	9680 ÷ 10490	88 ÷ 92
		0.2	45661 ÷ 45661	192 ÷ 192	16746 ÷ 19689	116 ÷ 126
		0.5	44259 ÷ 48472	189 ÷ 198	41128 ÷ 47303	182 ÷ 195
		1	84710 ÷ 121164	262 ÷ 313	82402 ÷ 89131	258 ÷ 268
	(10.4)	0.05	21088 ÷ 26763	130 ÷ 147	4027 ÷ 25032	57 ÷ 142
		0.1	78528 ÷ 79206	252 ÷ 253	8323 ÷ 138741	82 ÷ 335
		0.2	—		21368 ÷ 137355	131 ÷ 333
		0.5	—		43093 ÷ 208641	186 ÷ 411
		1	—		81217 ÷ 154550	256 ÷ 353
	(10.5)	0.05	21129 ÷ 112349	130 ÷ 301	3996 ÷ 76883	56 ÷ 249
		0.1	86171 ÷ 154621	264 ÷ 354	8156 ÷ 101348	81 ÷ 286
		0.2	307720 ÷ 307720	499 ÷ 499	17269 ÷ 90884	118 ÷ 271
		0.5	—		54467 ÷ 177643	210 ÷ 379
		1	—		106612 ÷ 233862	293 ÷ 435
$\begin{pmatrix} q_1 \\ q_2 \end{pmatrix}$	(10.3)	0.05	2.80 ÷ 7.83	0.84 ÷ 1.31	5.28 ÷ 5.28	1.40 ÷ 1.40
		0.1	4.56 ÷ 10.33	1.12 ÷ 1.68	9.00 ÷ 9.00	1.71 ÷ 1.71
		0.2	7.24 ÷ 17.87	1.57 ÷ 1.73	15.87 ÷ 15.87	2.11 ÷ 2.11
		0.5	37.16 ÷ 38.56	2.44 ÷ 2.55	—	
	(10.4)	0.05	2.38 ÷ 2.43	0.99 ÷ 1.08	—	
	(10.5)	0.05	5.32 ÷ 5.65	1.10 ÷ 1.35	5.22 ÷ 5.29	1.40 ÷ 1.41
		0.1	9.38 ÷ 9.52	1.37 ÷ 1.41	8.96 ÷ 9.01	1.70 ÷ 1.71
		0.2	16.12 ÷ 16.27	1.95 ÷ 1.98	15.89 ÷ 15.94	2.11 ÷ 2.21
		0.5	35.80 ÷ 36.05	3.14 ÷ 3.20	—	
$\begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix}$	(10.3)	0.05	5.69 ÷ 20.96	1.16 ÷ 2.56	5.24 ÷ 6.80	1.40 ÷ 1.73
		0.1	9.18 ÷ 41.72	1.31 ÷ 3.78	8.96 ÷ 10.79	1.70 ÷ 2.04
		0.2	16.93 ÷ 45.36	1.67 ÷ 3.39	15.83 ÷ 15.88	2.11 ÷ 2.11
		0.5	36.37 ÷ 54.02	2.22 ÷ 2.81	—	
		1	70.44 ÷ 70.44	2.91 ÷ 2.91	—	
	(10.4)	0.05	6.80 ÷ 8.24	1.20 ÷ 1.95	6.76 ÷ 6.80	1.72 ÷ 1.73
		0.1	10.79 ÷ 11.18	1.74 ÷ 2.29	10.73 ÷ 10.79	2.04 ÷ 2.04
	(10.5)	0.05	5.30 ÷ 5.94	1.00 ÷ 1.44	5.23 ÷ 5.29	1.40 ÷ 1.40
		0.1	9.03 ÷ 9.37	1.40 ÷ 1.81	8.96 ÷ 9.01	1.70 ÷ 1.71
		0.2	—		15.90 ÷ 15.94	2.12 ÷ 2.12



Rysunek 10.1 Rzuty na płaszczyznę xy ścieżek otrzymanych za pomocą algorytmu Newtona (a-c) i algorytmu Newtona z optymalizacją w przestrzeni zerowej (d-f) dla jednokołowca.



Rysunek 10.2 Rzuty na płaszczyznę xy ścieżek otrzymanych za pomocą algorytmu Newtona (a-c) i algorytmu Newtona z optymalizacją w przestrzeni zerowej (d-f) dla samochodu kinematycznego.

Tabela 10.5 Dane określające poszczególne zadania.

Nr zadania	model	$\boldsymbol{k}(\boldsymbol{q})$	$\boldsymbol{u}$	δ
1	jednokołowiec	identyczność	(10.3)	0.5
2	jednokołowiec	$(q_1, q_2)^T$	(10.3)	0.5
3	jednokołowiec	$(q_1, q_2)^T$	(10.5)	0.2
4	samochód kinematyczny	identyczność	(10.3)	0.1
5	samochód kinematyczny	$(q_1, q_2)^T$	(10.3)	0.05
6	samochód kinematyczny	$(q_1, q_2, q_3)^T$	(10.4)	0.1

wymiarowość), jako że w tym przypadku część związana z przeniesieniem zadania do przestrzeni zadaniowej jest pomijana.

- Wyniki otrzymane dla samochodu kinematycznego są szczególnie interesujące, zwłaszcza dla funkcji identycznościowej. Spełnienie warunku $q_4 = 0$ wymaga bardzo obszernego manewru i jest wyjątkowo skomplikowane dla krótkich parametryzacji. Z tego powodu na rys. 10.2 odnoszącego się do Zadania 4 skalowanie osi x i y jest zmienne. Warto jednak zaznaczyć, że mimo tak obszernych ruchów dla przypadków b),c),e),f) rzeczywisty punkt końcowy $\boldsymbol{x}_f^\star$ znajduje się bardzo blisko punktu docelowego $\boldsymbol{x}_f$.
- Może się wydawać dziwne, że dla Zadania 6, rys. 10.2, rzut ścieżki o największej energii sterowań na płaszczyznę xy jest bardzo krótki. W tym przypadku główna część zużytej energii przypada na zmianą orientacji robota (sterowanie u_2). Sterowanie to było wyjątkowo duże, co przy małym sterowaniu postępowym u_1 dało dużą energię sterowań i niewielki ruch na płaszczyźnie xy.

Rozdział 11

Planowanie ruchu w przestrzeni zadaniowej

Zaletą planowania w przestrzeni zadaniowej miast konfiguracyjnej jest zmniejszenie liczby ograniczeń jakie powinna spełniać wynikowa trajektoria. Zatem zarówno czas obliczeń, jak i właściwości trajektorii (np. długość ścieżki, obszerność ruchu czy energia sterowań) mogą ulec poprawie. W praktycznych zastosowaniach często nie wszystkie współrzędne wektora konfiguracji mają znaczenie podczas planowania ruchu, co uzasadnia ograniczenie przestrzeni zadaniowej do jedynie istotnych współrzędnych. Jednym z przykładowych zastosowań jest ruch kołowych robotów mobilnych na płaszczyźnie w środowisku kolizyjnym, gdzie przeszkody $\mathbb{X}_{obst}$ są opisywane jako obszary zabronione w przestrzeni zadaniowej $\mathbb{X} = \mathbb{R}^2$. Zadanie zalezienia ścieżki pomiędzy punktami początkowym $\boldsymbol{x}_0$ i końcowym $\boldsymbol{x}_f$ należącymi do $\mathbb{X}$ w środowisku kolizyjnym rozwiązuje wiele algorytmów planowania geometrycznego opisanych w podrozdziale 2.3, które jednak nie uwzględniają ograniczeń ruchowych robota. Niektóre algorytmy planowania geometrycznego (diagram Woronoja czy graf widoczności) wymaga ograniczonej przestrzeni zadaniowej, którą nazwano sceną i oznaczono jako $\mathbb{X}_S$.

Lie-algebraiczną metodę planowania ruchu trudno zmodyfikować tak, aby uwzględniała bezpośrednio położenie przeszkód. Jednak dzięki własności STLC można kontrolować praktycznie w dowolny sposób (być może skutkujący wielością elementarnych planowań) objętość manewru podczas pojedynczego planowania. Zatem można zastąpić planowanie w środowisku kolizyjnym zadaniem wyznaczenia ruchu do kolejnych punktów pośrednich ścieżki uzyskanych uprzednio z planera geometrycznego. W niniejszym rozdziale przedstawiono algorytm implementujący tę właśnie ideę, którego poszczególne kroki zebrano w Algorytmie 11.1.

Uwagi do Algorytmu 11.1:

- Celem Algorytmu 11.1 jest przekształcenie ścieżki uzyskanej z planowania geometrycznego w trajektorię minimalizującą trudność (i obszerność) ruchu układu nieholonomicznego dzięki generacji, dla kolejno wybranych punktów pośrednich ścieżki geometrycznej w $\mathbb{X}$, najłatwiejszych do osiągnięcia konfiguracji w $\mathbb{Q}$.
- Wpływ wektora początkowego $\boldsymbol{p}_0$ na wynik pojedynczego planowania opisano w rozdziale 10, a zasady jego doboru określa projektant algorytmu.
- Funkcja wyboru konfiguracji $f(\mathbb{Q}^\star, \boldsymbol{q}_{prv}, \boldsymbol{x}_{act}, \boldsymbol{x}_{nxt})$ uwzględnia dwie składowe:
  1. ocenę trudności konfiguracji $\boldsymbol{q}_j^\star$, (zobacz Algorytm 8.1), w perspektywie punktu startowego planowania $\boldsymbol{q}_c$ oraz następnego punktu ścieżki $\boldsymbol{x}_{ref,i+1}$,
  2. odległości między $\boldsymbol{k}(\boldsymbol{q}_j^\star)$ a $\boldsymbol{x}_{ref,i}$.

Algorytm 11.1 Lie-algebraicznego planowania ruchu w środowisku kolizyjnym.

Dane wejściowe: Bezdryfowy układ nieholonomiczny (2.1) z funkcją wyjścia $\boldsymbol{k}(\boldsymbol{q})$, konfiguracja początkowa $\boldsymbol{q}_0 \in \mathbb{R}^n$ oraz punkt docelowy $\boldsymbol{x}_f$, scena $\mathbb{X}_S \subset \mathbb{X}$, przeszkody $\mathbb{X}_{obst}$, maksymalny zakres pojedynczego planowania Δ, czas pojedynczego ruchu T, liczba planowań na krok N_p, funkcja $f(\mathbb{Q}^\star, \boldsymbol{q}_{prv}, \boldsymbol{x}_{act}, \boldsymbol{x}_{nxt})$ wybierająca najlepszą konfigurację jako kolejny podcel. Argumentami funkcji są: $\mathbb{Q}^\star$ - zbiór konfiguracji do wyboru, $\boldsymbol{q}_{prv}$ - konfiguracja początkowa aktualnego planowania, cel $\boldsymbol{x}_{act}/\boldsymbol{x}_{nxt}$ aktualnego/kolejnego planowania.

Krok 1. Obliczyć punkt początkowy $\boldsymbol{x}_0 \leftarrow \boldsymbol{k}(\boldsymbol{q}_0)$ i wyznaczyć bazowa ścieżkę referencyjną $\mathbb{X}_{bref}$ planerem geometrycznym dla określonej sceny $\mathbb{X}_S$, przeszkód $\mathbb{X}_{obst}$ oraz punktów początkowego $\boldsymbol{x}_0$ i docelowego $\boldsymbol{x}_f$.

Krok 2. Uzyskać ścieżkę referencyjną $\mathbb{X}_{ref} = (\boldsymbol{x}_{ref,1} = \boldsymbol{x}_0, \boldsymbol{x}_{ref,2}, \ldots, \boldsymbol{x}_{ref,k} = \boldsymbol{x}_f)$ dyskretyzując bazową ścieżkę referencyjną $\mathbb{X}_{bref}$ i przyjmując maksymalną odległość pomiędzy jej punktami jako Δ.

Krok 3. Zainicjować pusta trajektorię wynikową, z czasem początkowym $t^\star \leftarrow 0$. Za aktualną konfigurację $\boldsymbol{q}_c$ przyjąć konfigurację początkową $\boldsymbol{q}_c \leftarrow \boldsymbol{q}_0$. Ustawić licznik $i \leftarrow 2$.

Krok 4. Jeśli $i > k$ zakończyć algorytm i zwrócić trajektorię wynikową. W przeciwnym przypadku wykonać Krok 5.

Krok 5. Przeprowadzić N_p pojedynczych planowań ruchu metodą bazującą na gCBHD, z konfiguracji $\boldsymbol{q}_c$ do punktu $\boldsymbol{x}_{ref,i}$, w czasie T, stosując algorytm Newtona. Każde z planowań przeprowadzić dla innego (losowego) wektora początkowego $\boldsymbol{p}_{0,j},\ j \in \{1, \ldots, N_p\}$, którego wynikiem jest sterowanie $\boldsymbol{u}_j(\cdot)$ dla kolejnych $j = \{1, \ldots, N_p\}$.

Krok 6. Każde ze sterowań $\boldsymbol{u}_j(\cdot)$ zastosować dla układu (6.1) z konfiguracją początkową $\boldsymbol{q}_c$ i uzyskać zarówno rzeczywistą konfigurację końcową $\boldsymbol{q}_j^\star$ jak i trajektorię między $\boldsymbol{q}_c$ i $\boldsymbol{q}_j^\star$.

Krok 7. Ze zbioru konfiguracji końcowych w kroku $i-$tym $\mathbb{Q}^\star = \{\boldsymbol{q}_j^\star : j \in \{1, \ldots, N_p\}\}$ wybrać najlepszą za pomocą funkcji $f(\mathbb{Q}^\star, \boldsymbol{q}_c, \boldsymbol{x}_{ref,i}, \boldsymbol{x}_{ref,i+1})$ i przyjąć ją jako aktualną konfiguracje $\boldsymbol{q}_c$. Dokleić trajektorię częściową odpowiadającą $\boldsymbol{q}_c$ do trajektorii wynikowej na przedziale $[t^\star, t^\star + T]$. Przyjąć czas początkowy następnej trajektorii częściowej $t^\star \leftarrow t^\star + T$. Zwiększyć licznik $i \leftarrow i + 1$. Przejść do Kroku 4.

W dalszej części rozdziału zaproponowano postaci $f(\cdot)$.

- Dla planowań do punktu końcowego $\boldsymbol{x}_f$ sugeruje się przyjąć funkcję $f(\cdot)$ wybierającą najmniejszą odległość między $\boldsymbol{k}(\boldsymbol{q}_j^\star)$ a $\boldsymbol{x}_f$.

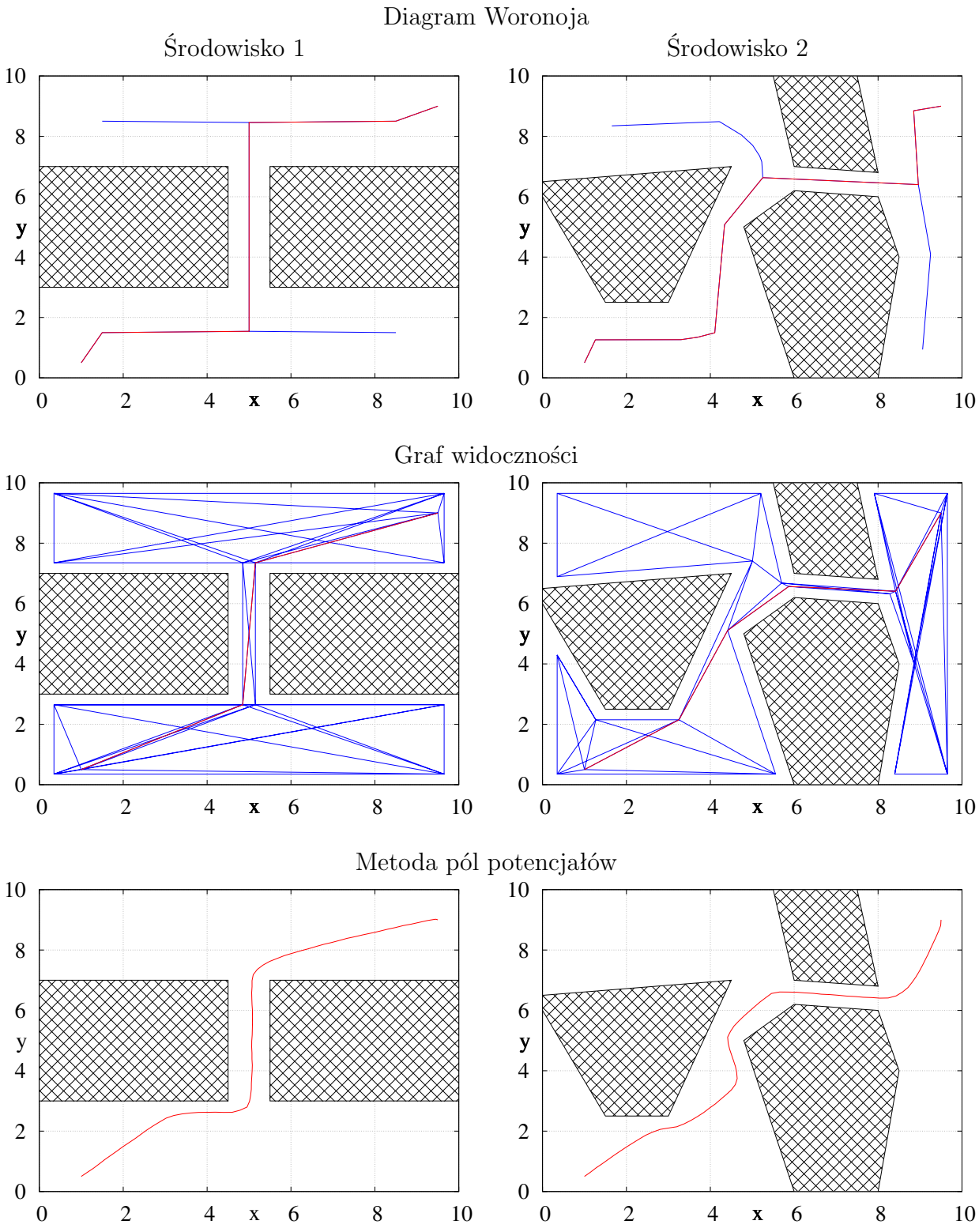
Algorytm 11.1 przetestowano w dwóch środowiskach charakteryzowanych przez:

1. wąskie przejście pomiędzy dwoma przeszkodami,
2. zestaw trzech przeszkód tworzących dwa wąskie przejścia.

W obu środowiskach ograniczono przestrzeń zadaniową do $\mathbb{X}_s = [0, 10] \times [0, 10]$ oraz zastosowano trzy algorytmy planowania geometrycznego między punktami $\boldsymbol{x}_0 = (1, 0.5)^T$ i $\boldsymbol{x}_f = (9.5, 9)^T$:

- diagram Woronoja,
- graf widoczności,
- metodę pól potencjałów.

Środowiska kolizyjne wraz z wynikami wymienionych algorytmów planowania geometrycznego zobrazowano na rys. 11.1. Krok 2 Algorytmu 11.1 zaimplementowano następująco:



Rysunek 11.1 Referencyjne ścieżki uzyskane z algorytmów planowania geometrycznego (czerwone). Kolorem niebieskim oznaczono grafy powstałe w wyniku działania planerów geometrycznych.

1. wyznaczenie charakterystycznych punktów ścieżki geometrycznej (wierzchołki w diagramie Woronoja i grafie widoczności, punkty reprezentujące zakręty w diagramie Woronoja i metodzie pól potencjałów),

2. dyskretyzacja prostych pomiędzy charakterystycznymi punktami.

Symulacje przeprowadzono dla dwóch modeli kołowych robotów mobilnych: jednokołowca A.1 oraz samochodu kinematycznego A.2. W przypadku jednokołowca za konfigurację początkową przyjęto $\boldsymbol{q}_0 = (1, 0.5, 0)^T$, dla samochodu kinematycznego $\boldsymbol{q}_0 = (1, 0.5, 0, 0)^T$. Dla obu modeli przyjęto minimalną macierz $\boldsymbol{F}_{i^\star}$, oraz przeprowadzono symulacje dla różnych maksymalnych zakresów pojedynczego planowania $\Delta = \{0.2, 0.5, 1\}$. Wszystkie symulacje zostały przeprowadzone dla sterowań

$$\begin{cases} u_1 = \frac{1}{\sqrt{T}}(p_{1,1} + p_{1,2}\sqrt{2}\sin(2\pi t) + p_{1,3}\sqrt{2}\cos(2\pi t)), \\ u_2 = \frac{1}{\sqrt{T}}(p_{2,1} + p_{2,2}\sqrt{2}\sin(2\pi t) + p_{2,3}\sqrt{2}\cos(2\pi t)), \end{cases} \tag{11.1}$$

z $T = 1$. Do rozwiązania równania (2.56) zastosowano algorytm Newtona. Liczbę wektorów parametrów początkowych $\boldsymbol{p}_{0,j},\ j \in \{1, \ldots, N_p\}$, Krok 5 Algorytmu 11.1, określono na $N_p = 500$. Współrzędne wektorów $\boldsymbol{p}_{0,j}$ generowano losowo z rozkładem jednostajnym na przedziale $[-1, 1]$.

Przetestowano trzy warianty funkcji wyboru konfiguracji $f(\mathbb{Q}^\star, \boldsymbol{q}_{prv}, \boldsymbol{x}_{act}, \boldsymbol{x}_{nxt})$:

$$f_1(\mathbb{Q}^\star, \boldsymbol{q}_{prv}, \boldsymbol{x}_{act}, \boldsymbol{x}_{nxt}) = \min_{\boldsymbol{q}_j^\star \in \mathbb{Q}^\star} \| \boldsymbol{k}(\boldsymbol{q}_j^\star) - \boldsymbol{x}_{act} \|, \tag{11.2}$$

czyli konfiguracja najbliższa punktowi docelowemu.

$$f_2(\mathbb{Q}^\star, \boldsymbol{q}_{prv}, \boldsymbol{x}_{act}, \boldsymbol{x}_{nxt}) = \max_{\boldsymbol{q}_j^\star \in \mathbb{Q}^\star} (lie^\star(\boldsymbol{q}_{prv}, \mathbf{id}(\boldsymbol{q}_j^\star)) \cdot lie^\star(\boldsymbol{q}_j^\star, \boldsymbol{x}_{nxt})), \tag{11.3}$$

czyli konfiguracja o najlepszych właściwościach Lie-algebraicznych w kontekście ruchu z aktualnej konfiguracji $\boldsymbol{q}_c$ do następnego punktu docelowego.

$$\begin{aligned} &f_3(\mathbb{Q}^\star, \boldsymbol{q}_{prv}, \boldsymbol{x}_{act}, \boldsymbol{x}_{nxt}) = \\ &= \max_{\boldsymbol{q}_i^\star \in \mathbb{QT}^\star} (lie^\star(\boldsymbol{q}_{prv}, \mathbf{id}(\boldsymbol{q}_i^\star)) \cdot lie^\star(\boldsymbol{q}_i^\star, \boldsymbol{x}_{nxt})),\ \mathbb{QT}^\star = \{\boldsymbol{q}_j^\star \in \mathbb{Q}^\star : \| \boldsymbol{k}(\boldsymbol{q}_j^\star) - \boldsymbol{x}_{act} \| < \delta_3\} \end{aligned} \tag{11.4}$$

podobna do f_2 z zawężeniem do konfiguracji spośród wszystkich konfiguracji będących bliżej konfiguracji docelowej niż δ_3.

Dla f_2, f_3 użyto wersję $lie^\star(\boldsymbol{q}_a, \boldsymbol{x}_b)$ zawierającą składową *contain*. $lie^\star(\boldsymbol{q}_a, \mathbf{id}(\boldsymbol{q}_b))$ oznacza że $\boldsymbol{k}(\boldsymbol{q})$ jest funkcją identycznościową, a przestrzeń zadaniowa jest równa konfiguracyjnej. W symulacjach parametr minimalnej dokładności δ_3 ze wzoru (11.4) był zależny od długości planowania według zależności

$$\delta_3 = \delta \cdot \Delta, \qquad \text{gdzie} \quad \delta \in \{0.4, 0.3, 0.2, 0.1, 0.05\}. \tag{11.5}$$

W tabelach 11.1, 11.2 zebrano parametry trajektorii wynikowej i czasu działania Algorytmu 11.1 dla wybranych modeli wykorzystując następujące skróty:

dł. – długość wynikowej ścieżki w przestrzeni zadaniowej $\mathbb{X}$,

en. – energia sterowań generujących trajektorię wynikową,

c.o. – czas obliczeń.

$\mathbb{QT}^\star \equiv \emptyset$ oznacza, że Algorytm 11.1 nie zakończył się sukcesem, gdyż nie istniały konfiguracje spośród testowanych, których obraz $\boldsymbol{k}(\boldsymbol{q}_j^\star)$ był dostatecznie blisko punktu docelowego, wymagane przez funkcję wyboru konfiguracji f_3, zob. wzór (11.4). Jeśli natomiast trajektoria wynikowa była kolizyjna, zamiast parametrów trajektorii w pola tabel wpisano

Tabela 11.1 Długość ścieżki, energia sterowań oraz czas obliczeń dla jednokołowca.

Środowisko 1.										
Δ	f	δ	diagram Woronoja			graf widzialności			metoda pól pot.		
			dł.	en.	c.o. [s]	dł.	en.	c.o. [s]	dł.	en.	c.o. [s]
0.2	f_1		64.46	142.2	32.8	kolizja			62.42	135.7	137.9
	f_2		51.72	100.3	45.8	42.39	77.7	46.3	45.87	90.7	271.6
	f_3	0.4	52.83	104.8	57.3	41.52	76.4	27.2	44.22	93.5	302.3
	f_3	0.3	55.78	118.9	69.0	41.92	84.7	27.9	42.21	87.6	334.5
	f_3	0.2	62.62	128.9	83.7	47.67	87.8	31.3	47.79	100.2	368.3
	f_3	0.1	59.60	124.0	100.4	50.33	99.9	35.6	48.73	106.3	404.4
	f_3	0.05	59.54	112.9	119.7	51.30	90.8	41.3	55.35	113.2	441.3
0.5	f_1		39.64	126.6	12.8	kolizja			36.23	112.9	45.4
	f_2		26.07	56.0	13.2	20.27	41.1	145.4	25.48	53.7	82.2
	f_3	0.4	25.97	55.3	14.8	22.10	42.4	151.5	28.20	60.8	87.5
	f_3	0.3	25.48	53.9	14.5	19.76	37.1	156.5	27.11	61.6	92.3
	f_3	0.2	30.05	66.3	14.8	24.11	55.3	162.6	26.08	59.6	97.5
	f_3	0.1	31.16	78.9	15.8	24.58	54.9	167.5	27.21	66.5	102.6
	f_3	0.05	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
1	f_1		27.47	110.5	18.3	kolizja			kolizja		
	f_2		18.78	45.5	30.7	16.45	37.3	9.8	kolizja		
	f_3	0.4	19.74	51.7	32.0	15.95	37.3	10.4	kolizja		
	f_3	0.3	20.99	62.4	33.3	15.10	38.8	10.6	kolizja		
	f_3	0.2	20.80	57.7	34.5	18.03	53.8	11.0	kolizja		
	f_3	0.1	23.81	83.9	35.5	$\mathbb{QT}^\star \equiv \emptyset$			kolizja		
	f_3	0.05	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		

Środowisko 2.										
Δ	f	δ	diagram Woronoja			graf widzialności			metoda pól pot.		
			dł.	en.	c.o. [s]	dł.	en.	c.o. [s]	dł.	en.	c.o. [s]
0.2	f_1		69.18	151.2	616.1	kolizja			kolizja		
	f_2		56.64	107.0	1254.8	41.20	77.1	74.8	45.47	93.5	514.0
	f_3	0.4	51.50	105.8	1340.1	42.79	74.1	88.0	43.73	89.9	555.5
	f_3	0.3	57.37	110.8	1422.5	41.34	81.4	101.3	47.13	96.0	597.1
	f_3	0.2	61.86	127.6	1512.2	43.75	83.4	116.2	46.03	96.4	654.1
	f_3	0.1	65.10	138.4	1599.9	49.50	100.1	132.6	52.15	109.8	694.3
	f_3	0.05	61.58	129.1	1688.4	48.50	92.8	149.8	49.51	106.5	730.6
0.5	f_1		39.10	124.3	217.6	kolizja			kolizja		
	f_2		27.40	58.6	425.3	20.62	39.1	18.9	26.44	64.6	191.9
	f_3	0.4	23.75	51.9	438.1	20.53	45.1	20.6	29.31	62.7	201.2
	f_3	0.3	28.21	65.4	451.3	24.22	56.7	22.0	26.17	59.4	209.0
	f_3	0.2	31.34	72.1	463.7	23.89	46.4	23.6	28.53	73.6	217.3
	f_3	0.1	30.60	90.3	478.0	23.73	56.1	25.2	31.67	85.1	225.3
	f_3	0.05	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
1	f_1		30.79	127.4	208.1	kolizja			28.27	97.8	135.5
	f_2		19.88	58.9	401.4	16.35	41.2	37.3	22.67	67.2	253.6
	f_3	0.4	20.80	59.4	407.7	16.10	41.1	38.7	21.94	63.9	259.7
	f_3	0.3	22.00	63.7	412.6	16.26	42.0	39.6	22.41	64.5	265.5
	f_3	0.2	21.59	69.0	417.3	17.08	45.3	40.6	21.72	66.1	274.9
	f_3	0.1	26.42	96.2	423.7	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
	f_3	0.05	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		

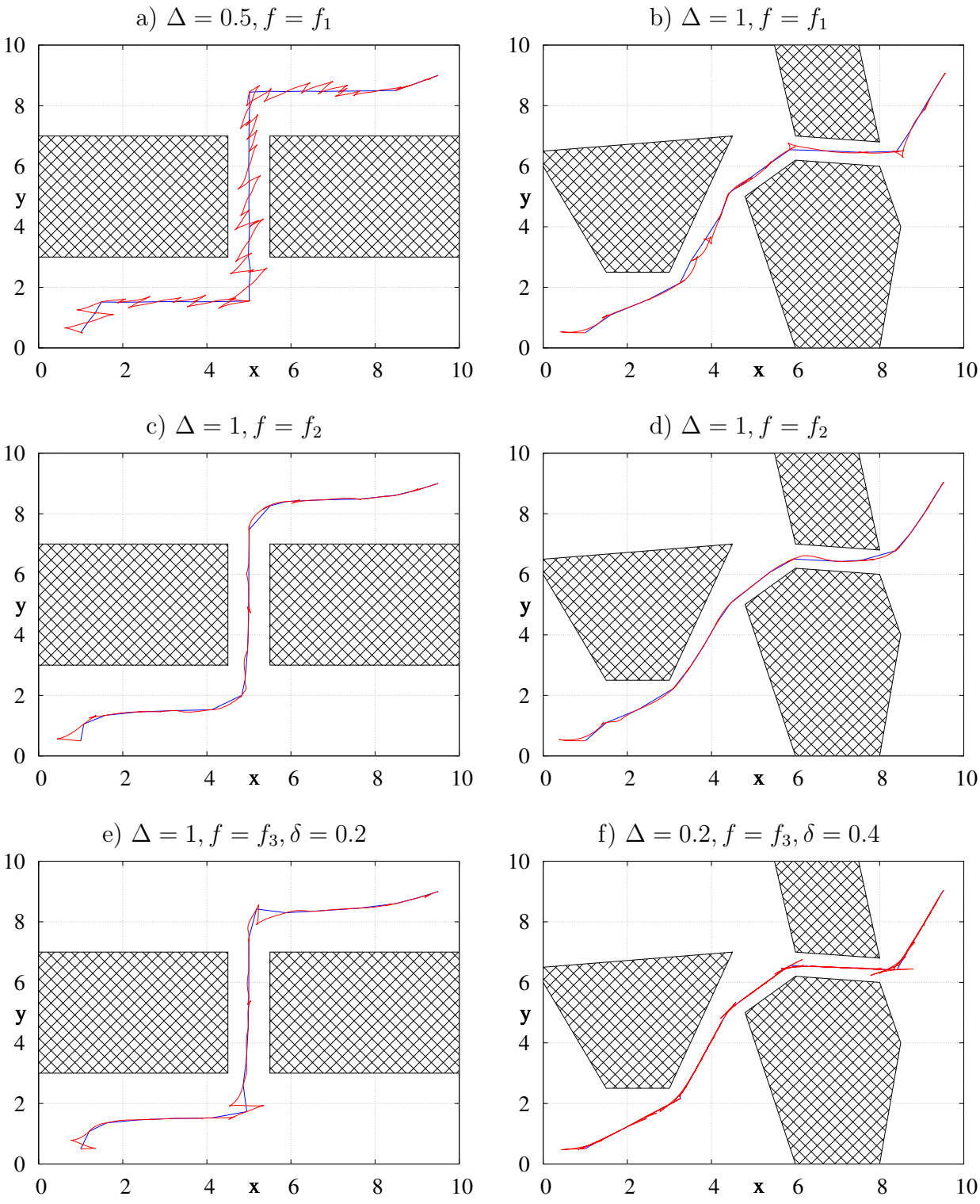
Tabela 11.2 Długość ścieżki, energia sterowań oraz czas obliczeń dla samochodu kinematycznego.

Środowisko 1.										
Δ	f	δ	diagram Woronoja			graf widzialności			metoda pól pot.		
			dł.	en.	c.o. [s]	dł.	en.	c.o. [s]	dł.	en.	c.o. [s]
0.2	f_1		62.48	314.5	136.3	51.87	253.8	60.4	57.62	293.7	333.6
	f_2		68.38	235.5	257.2	63.27	189.4	98.4	63.94	200.8	661.7
	f_3	0.4	76.83	208.0	292.4	61.58	192.5	113.0	67.66	203.4	709.3
	f_3	0.3	76.70	242.9	328.1	67.42	191.1	128.9	66.83	186.6	758.8
	f_3	0.2	77.33	212.9	366.8	62.16	192.2	145.6	67.40	201.1	811.2
	f_3	0.1	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
	f_3	0.05	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
0.5	f_1		kolizja			29.53	150.6	20.5	32.61	277.2	133.1
	f_2		36.41	314.0	81.4	26.71	200.3	28.5	36.17	134.3	251.0
	f_3	0.4	34.20	187.0	86.3	27.07	189.5	30.2	34.58	156.8	259.8
	f_3	0.3	37.63	310.6	91.6	28.00	158.7	32.0	30.58	175.9	269.0
	f_3	0.2	$\mathbb{QT}^\star \equiv \emptyset$			28.12	192.3	33.8	$\mathbb{QT}^\star \equiv \emptyset$		
	f_3	0.1	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
	f_3	0.05	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
1	f_1		kolizja			kolizja			kolizja		
	f_2		29.32	239.3	90.3	kolizja			kolizja		
	f_3	0.4	24.63	163.5	92.1	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
	f_3	0.3	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
	f_3	0.2	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
	f_3	0.1	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
	f_3	0.05	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		

Środowisko 2.										
Δ	f	δ	diagram Woronoja			graf widzialności			metoda pól pot.		
			dł.	en.	c.o. [s]	dł.	en.	c.o. [s]	dł.	en.	c.o. [s]
0.2	f_1		64.13	360.4	44.6	kolizja			kolizja		
	f_2		82.96	236.1	53.4	60.22	187.5	252.4	65.92	210.4	1023.9
	f_3	0.4	78.93	245.4	47.3	61.02	162.8	277.9	69.08	206.3	1052.5
	f_3	0.3	78.10	246.3	49.8	61.46	171.9	304.1	68.25	191.5	1086.0
	f_3	0.2	79.82	238.7	55.4	kolizja			65.07	200.8	1143.3
	f_3	0.1	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
	f_3	0.05	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
0.5	f_1		35.35	244.9	410.9	31.44	250.7	50.2	39.57	320.4	218.1
	f_2		39.29	236.1	802.9	28.69	203.4	87.4	37.16	214.0	423.1
	f_3	0.4	$\mathbb{QT}^\star \equiv \emptyset$			kolizja			35.48	225.7	435.9
	f_3	0.3	$\mathbb{QT}^\star \equiv \emptyset$			31.99	236.4	95.7	$\mathbb{QT}^\star \equiv \emptyset$		
	f_3	0.2	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
	f_3	0.1	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
	f_3	0.05	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
1	f_1		kolizja			22.44	193.7	45.0	kolizja		
	f_2		26.25	203.7	14.5	17.38	96.9	79.7	kolizja		
	f_3	0.4	28.27	288.1	15.3	19.17	188.8	81.7	31.67	229.4	399.7
	f_3	0.3	$\mathbb{QT}^\star \equiv \emptyset$			17.80	182.4	83.3	$\mathbb{QT}^\star \equiv \emptyset$		
	f_3	0.2	$\mathbb{QT}^\star \equiv \emptyset$			16.29	163.4	84.9	$\mathbb{QT}^\star \equiv \emptyset$		
	f_3	0.1	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		
	f_3	0.05	$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$			$\mathbb{QT}^\star \equiv \emptyset$		

"kolizja". Na rys. 11.2 przedstawiono ścieżki wygenerowane Algorytmem 11.1 (kolorem czerwonym) oraz linię łamaną łączącą kolejne punkty (kolorem niebieskim) $\boldsymbol{k}(\boldsymbol{q}_c)$ dla wybranych parametrów algorytmu.

Na podstawie wyników symulacji można wyciągnąć wniosek, że Algorytm 11.1 umożliwia zaplanowanie trajektorii w środowisku kolizyjnym z uwzględnieniem ograniczeń nieholonomicznych. Pewnym zaskoczeniem może być fakt, że dla dłuższych zakresów planowania Δ trajektorie wynikowe uzyskiwały lepsze parametry długości ścieżki, energii sterowań oraz obszerności ruchu. W wyniku planowania przy pomocy funkcji wyboru konfiguracji f_2 uzyskiwano niekiedy trajektorię istotnie oddaloną od otrzymanej z planera geometrycznego. Funkcja wyboru f_3 umożliwia znalezienie odpowiedniego kompromisu między dokładnym osiąganiem punktów referencyjnych z planera geometrycznego (jak w przypadku f_1) ale dobrymi własnościami Lie-algebraicznymi punktów pośrednich. Analizując wynikowe ścieżki można zauważyć, że dla funkcji f_2 lub f_3 zredukowana zostaje liczba planowań dających obszerne ścieżki do tych punktów, gdzie ścieżka z planera geometrycznego skręca gwałtownie.



Rysunek 11.2 Ścieżki wygenerowane Algorytmem 11.1 oraz łamane łączące kolejne punkty $\boldsymbol{k}(\boldsymbol{q}_c)$. Lewa kolumna: Środowisko 1, diagram Woronoja, jednokołowiec. Prawa kolumna: Środowisko 2, graf widoczności, samochód kinematyczny.

Podsumowanie

W dysertacji wzbogacono zastosowanie Lie-algebraicznej metody planowania ruchu układów nieholonomicznych także o układy z funkcją wyjścia. Dotychczas w literaturze przedmiotu metoda była stosowana dla układów nieholonomicznych opisanych jedynie w przestrzeni konfiguracyjnej, więc planowanie odbywało się w tej samej przestrzeni co opis dynamiki układu. W pracy pokazano, że zastosowanie Lie-algebraicznej metody planowania ruchu do planowania trajektorii w przestrzeni zadaniowej jest nie tylko możliwe, ale także poszerza spektrum rozwiązywanych zadań o wiele praktycznych. Lie-algebraiczne planowanie w przestrzeni zadaniowej poszerza także możliwości optymalizacyjne metody względem planowania jedynie w przestrzeni konfiguracyjnej.

Wyniki uzyskane w dysertacji odnoszą się do trzech obszarów tematycznych związanych z planowaniem ruchu: formuły gCBHD wykorzystywanej w planowaniu ruchu metodą Lie-algebraiczną, przeniesienia literaturowych obiektów i pojęć związanych z planowaniem w przestrzeni konfiguracyjnej do przestrzeni zadaniowej oraz samym planowaniem w przestrzeni zadaniowej, będącym praktycznym podsumowaniem i wykorzystaniem spostrzeżeń zgromadzonych podczas realizacji badań w pierwszych dwóch obszarach.

W pierwszym obszarze zainteresowań zbadano wpływ parametryzacji oraz parametrów sterowań na kształt trajektorii planowanej za pomocą formuły gCBHD w przestrzeni konfiguracyjnej. Podczas analizy elementów formuły gCBHD wyrażonej w formie kombinacji liniowej pre-sterowań i elementów bazy Ph. Halla najpierw zauważono, a następnie udowodniono związek między współczynnikami liczbowymi pre-sterowań w niej występujących. Wynikiem praktycznym jest autorskie zaprojektowanie kombinatorycznego algorytmu obliczającego pre-sterowania według formuły gCBHD, którego złożoność obliczeniowa jest liniowa, poprawiając istotnie algorytmy literaturowe o eksponencjalnej złożoność obliczeniowej.

Korzystając istotnie z zasady analogii przeniesiono niezbędne do przeprowadzenia planowania ruchu pojęcia i obiekty, literaturowo zdefiniowane dla przestrzeni konfiguracyjnej, do przestrzeni zadaniowej umożliwiając zdefiniowanie zadania planowania dla układów nieholonomicznych z funkcją wyjścia. Przeanalizowano różnice pomiędzy definicjami w przestrzeni konfiguracyjnej i zadaniowej. Z powodu niejednoznaczności funkcji odwrotnej dla surjektywnej funkcji wyjścia, stwierdzono, że nie jest możliwe planowanie w przestrzeni zadaniowej nie odwołujące się explicite do pojęć z przestrzeni konfiguracyjnej. Zaproponowano jednak dwa (silny i słaby) teoretyczne sposoby uniezależnienia niektórych pojęć z przestrzeni zadaniowej od jawnej zależności od przestrzeni konfiguracyjnej. Pokazano również, że w przypadku planowania w przestrzeni zadaniowej (w odróżnieniu od konfiguracyjnej) mogą występować osobliwości, dla których stwierdzenia zaproponowano szerszą niż dla manipulatorów definicję. Wskazano także potencjalne ich źródła.

Głównym osiągnięciem rozprawy jest zaprojektowanie autorskiego algorytmu oceniającego jakość konfiguracji z przestrzeni konfiguracyjnej pod kątem ruchu w zadanym kierunku w przestrzeni zadaniowej. Algorytm nie jest złożony obliczeniowo, a umożliwia

ocenę konfiguracji pośrednich w przypadku trajektorii złożonej z wyników wielu lokalnych planowań wykorzystujących formułę gCBHD. Użycie takiej oceny pozwoliło na autorskie opracowanie: algorytmu zapobiegającego przedwczesnej zbieżności w planowaniu wieloetapowym oraz algorytmu planowania ruchu w przestrzeni zadaniowej w obecności stacjonarnych przeszkód. Na etapie projektowania drugiego z algorytmów wykorzystano wyniki badań wpływu wektora inicjującego algorytm Newtona rozwiązującego równanie ruchu powstałe na bazie formuły gCBHD. W obu opracowanych algorytmach zbadano symulacyjnie wpływ parametrów algorytmu na planowaną trajektorię. Oba algorytmy porównano z podstawowymi wersjami metody Lie-algebraicznej zastosowanej dla układu z funkcją wyjścia.

Naturalnym kierunkiem dalszych badań nad planowaniem ruchu układów nieholonomicznych w przestrzeni zadaniowej metodą Lie-algebraiczną jest uwzględnienie dynamiki układu, w celu uzyskania trajektorii najłatwiejszej do śledzenia podczas sterowania. Ponadto już na etapie planowania ścieżki holonomicznej, będącej bazą punktów pośrednich dla planowania w przestrzeni zadaniowej, celowym wydaje się uwzględnienie nieholonomiczności układu.

Dodatek A

Modele układów nieholonomicznych

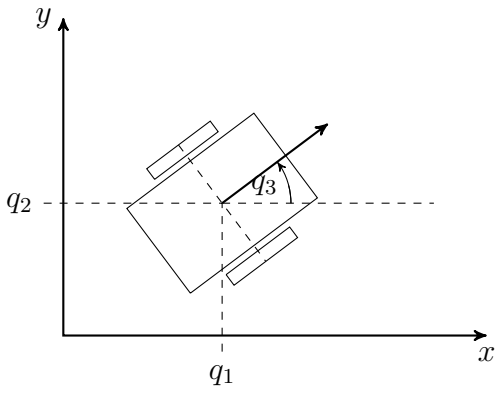
A.1 Jednokołowiec

Jednokołowiec (monocykl) jest jednym z najprostszych układów nieholonomicznych. Na rys. A.1a przedstawiono definicje jego współrzędnych: q_1, q_2 oznaczają, odpowiednio, położenie wzdłuż osi x i y, globalnego układu współrzędnych, natomiast q_3 opisuje orientację robota względem osi x tego układu (oznaczaną tradycyjnie przez θ). Model ten znajduje zastosowanie w opisie kinematyki prostych robotów mobilnych (zwykle dwu- lub czterokołowych) które mają fizyczną możliwość obrotu w miejscu, a jego równania, wynikające z braku poślizgu bocznego, są następujące

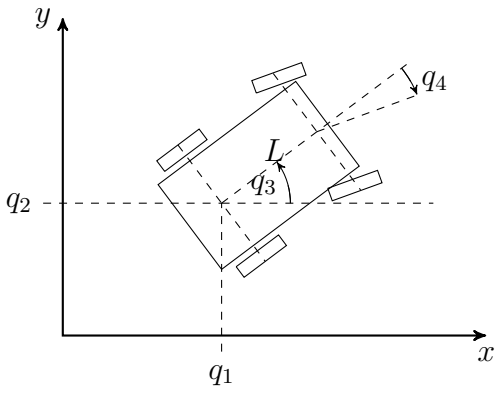
$$\dot{\boldsymbol{q}} = \begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{pmatrix} = \begin{pmatrix} \cos(q_3) & 0 \\ \sin(q_3) & 0 \\ 0 & 1 \end{pmatrix} \boldsymbol{u} = \begin{pmatrix} \cos(q_3) \\ \sin(q_3) \\ 0 \end{pmatrix} u_1 + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} u_2 = \boldsymbol{g}_1(\boldsymbol{q})u_1 + \boldsymbol{g}_2(\boldsymbol{q})u_2, \qquad \text{(A.1)}$$

gdzie sterowania u_1/u_2 posiadają fizyczną interpretację prędkości postępowej/obrotowej pojazdu. Do pokazania sterowalność STLC jednokołowca wystarczy wyliczyć pole wekto-

a) Jednokołowiec (na przykładzie dwu--kołowego robota mobilnego).



b) Samochód kinematyczny (na przykładzie dwuosiowego robota mobilnego).



Rysunek A.1 Jednokołowiec i samochód kinematyczny, rzuty z góry.

rowe generowane nawiasem Liego $[\boldsymbol{g}_1, \boldsymbol{g}_2]$ jako

$$[\boldsymbol{g}_1, \boldsymbol{g}_2] = \begin{pmatrix} \sin(q_3) \\ -\cos(q_3) \\ 0 \end{pmatrix}, \tag{A.2}$$

bowiem macierz złożona z pól $\boldsymbol{g}_1, \boldsymbol{g}_2, [\boldsymbol{g}_1, \boldsymbol{g}_2]$ posiada pełen rząd dla każdego $\boldsymbol{q}$

$$\operatorname{rank} \boldsymbol{F}_2 = \operatorname{rank} \begin{pmatrix} \cos(q_3) & 0 & \sin(q_3) \\ \sin(q_3) & 0 & -\cos(q_3) \\ 0 & 1 & 0 \end{pmatrix} = 3, \tag{A.3}$$

Pole $[\boldsymbol{g}_1, \boldsymbol{g}_2]$ spełniając zależność

$$\langle \boldsymbol{g}_1, [\boldsymbol{g}_1, \boldsymbol{g}_2] \rangle = \boldsymbol{0} \tag{A.4}$$

umożliwia ruch na płaszczyźnie XY w kierunku prostopadłym do pola $\boldsymbol{g}_1$.

Obliczeniowo łatwo pokazać, że elementy wyższych warstw bazy Ph. Halla algebry Liego są liniowo zależne od kolumn macierzy $\boldsymbol{F}_2$ i należą do mało licznego zbioru: $\pm\boldsymbol{g}_1, \pm[\boldsymbol{g}_1, \boldsymbol{g}_2], \boldsymbol{0}$, a zatem układ nie jest nilpotentny. Przykładowo: elementy trzeciej warstwy bazy Ph. Halla są następujące:

$$[\boldsymbol{g}_1, [\boldsymbol{g}_1, \boldsymbol{g}_2]] = \boldsymbol{0}, \quad [\boldsymbol{g}_2, [\boldsymbol{g}_1, \boldsymbol{g}_2]] = \begin{pmatrix} \cos(q_3) \\ \sin(q_3) \\ 0 \end{pmatrix} = \boldsymbol{g}_1.$$

A.2 Samochód kinematyczny

Samochód kinematyczny jest czterowymiarowym układem nieholonomicznym. Na rysunku A.1b przedstawiono wizualizację modelu na płaszczyźnie XY. Współrzędne $\boldsymbol{q}_1$ i $\boldsymbol{q}_2$ oznaczają położenie tylnej osi pojazdu względem osi x i y, odpowiednio, $\boldsymbol{q}_3$ jest orientacją osi tylnej (oznaczaną θ), natomiast $\boldsymbol{q}_4$ – orientacją przedniej osi sterowanej (oznaczaną ψ). Odległość pomiędzy osiami L jest parametrem geometrycznym. Model ten znajduje zastosowanie w opisie kinematyki robotów trójkołowych (jedno przednie koło sterujące) [37], samochodów (biorąc pod uwagę tylko wypadkową prędkość i orientację mechanizmu różnicowego) [52] oraz układu ciągnik (jako jednokołowiec) z wózkiem [54, 90].

Uogólnienie samochodu kinematycznego o kolejne wózki pozwala na modelowanie, przykładowo, pojazdów lotniskowych służących do transportu bagażu [19]. Równania opisujące samochód kinematyczny, będące rezultatem braku poślizgu wzdłużnego i poprzecznego, przyjmują postać

$$\dot{\boldsymbol{q}} = \begin{pmatrix} L\cos(q_3)\cos(q_4) & 0 \\ L\sin(q_3)\cos(q_4) & 0 \\ \sin(q_4) & 0 \\ 0 & 1 \end{pmatrix} \boldsymbol{u} = \begin{pmatrix} L\cos(q_3)\cos(q_4) \\ L\sin(q_3)\cos(q_4) \\ \sin(q_4) \\ 0 \end{pmatrix} u_1 + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} u_2 = \boldsymbol{g}_1(\boldsymbol{q})\, u_1 + \boldsymbol{g}_2(\boldsymbol{q})\, u_2. \tag{A.5}$$

Podobnie jak dla jednokołowca, sterowania u_1/u_2 mają fizyczną interpretację prędkości postępowej/obrotowej pojazdu Przyjmuje się, że napęd samochodu kinematycznego znajduje się w osi przedniej pojazdu. W przypadku napędu na tylną oś, generatory równania (A.5) byłyby w postaci $\boldsymbol{g}_1 = (L\cos(q_3), L\sin(q_3), \operatorname{tg}(q_4))^T$ oraz $\boldsymbol{g}_2 = (0, 0, 0, 1)^T$. Rozwiązanie to skutkuje jednak osobliwością w przedniej osi dla $\boldsymbol{q}_4 = \pm\pi/2$. Często, dla

uproszczenia obliczeń, przyjmuje się wartość $L = 1$.

Aby określić sterowalność układu należy wygenerować pola drugiego i trzeciego stopnia. Pola odpowiadające elementom drugiej i trzeciej warstwy bazy Ph. Halla wyliczamy jako

$$[\boldsymbol{g}_1, \boldsymbol{g}_2] = \begin{pmatrix} L\cos(\boldsymbol{q}_3)\sin(\boldsymbol{q}_4) \\ L\sin(\boldsymbol{q}_3)\sin(\boldsymbol{q}_4) \\ -\cos(\boldsymbol{q}_4) \\ 0 \end{pmatrix}, \quad [\boldsymbol{g}_1, [\boldsymbol{g}_1, \boldsymbol{g}_2]] = \begin{pmatrix} -L\sin(\boldsymbol{q}_3) \\ L\cos(\boldsymbol{q}_3) \\ 0 \\ 0 \end{pmatrix}, \quad [\boldsymbol{g}_2, [\boldsymbol{g}_1, \boldsymbol{g}_2]] = \boldsymbol{g}_1. \tag{A.6}$$

Układ jest sterowalny w krótkim czasie, gdyż macierz złożona z pól $\boldsymbol{g}_1, \boldsymbol{g}_2, [\boldsymbol{g}_1, \boldsymbol{g}_2], [\boldsymbol{g}_1[\boldsymbol{g}_1, \boldsymbol{g}_2]]$ jest pełnego rzędu dla każdego $\boldsymbol{q}$, bowiem

$$\operatorname{rank} \boldsymbol{F}_3 = \operatorname{rank} \begin{pmatrix} L\cos(q_3)\cos(q_4) & 0 & L\cos(q_3)\sin(q_4) & -L\sin(q_3) \\ L\sin(q_3)\cos(q_4) & 0 & L\sin(q_3)\sin(q_4) & L\cos(q_3) \\ \sin(q_4) & 0 & -\cos(q_4) & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} = 4. \tag{A.7}$$

Analogicznie jak dla jednokołowca, wyliczono również elementy bazy Ph. Halla dla warstwy o jeden wyższej niż wymaga warunek LARC

$$[\boldsymbol{g}_1, [\boldsymbol{g}_1, [\boldsymbol{g}_1, \boldsymbol{g}_2]]] = \begin{pmatrix} -L\cos(q_3)\sin(q_4) \\ -L\sin(q_3)\sin(q_4) \\ 0 \\ 0 \end{pmatrix}, \quad [\boldsymbol{g}_2, [\boldsymbol{g}_1, [\boldsymbol{g}_1, \boldsymbol{g}_2]]] = \boldsymbol{0},$$

$$[\boldsymbol{g}_2, [\boldsymbol{g}_2, [\boldsymbol{g}_1, \boldsymbol{g}_2]]] = \begin{pmatrix} -L\cos(q_3)\sin(q_4) \\ -L\sin(q_3)\sin(q_4) \\ \cos(q_4) \\ 0 \end{pmatrix} = -[\boldsymbol{g}_1, \boldsymbol{g}_2],$$

zatem także ten układ nie jest nilpotentny. Kolejne pola wektorowe powielają pola uprzednio wyliczone z dokładnością do znaku.

A.3 Integrator Brocketta

Integrator Brocketta jest układem o trójwymiarowej przestrzeni konfiguracyjnej opisanym równaniami

$$\dot{\boldsymbol{q}} = \begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ -q_2 & q_1 \end{pmatrix} \boldsymbol{u} = \begin{pmatrix} 1 \\ 0 \\ -q_2 \end{pmatrix} u_1 + \begin{pmatrix} 0 \\ 1 \\ q_1 \end{pmatrix} u_2 = \boldsymbol{g}_1(\boldsymbol{q})u_1 + \boldsymbol{g}_2(\boldsymbol{q})u_2. \tag{A.8}$$

Jedyne niezerowe pole wektorowe $[\boldsymbol{g}_1, \boldsymbol{g}_2]$ utworzone z generatorów układu jest następujące

$$[\boldsymbol{g}_1, \boldsymbol{g}_2] = \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix}. \tag{A.9}$$

Pozostałe pola wektorowe odpowiadające elementom bazy Ph. Halla zerują się tożsamościowo, bowiem dla trzeciej warstwy mamy $[\boldsymbol{g}_1[\boldsymbol{g}_1, \boldsymbol{g}_2]] = [\boldsymbol{g}_1[\boldsymbol{g}_1, \boldsymbol{g}_2]] = \boldsymbol{0}$. Integrator

Brocketta jest najprostszym układem nilpotentnym, będąc stopnia drugiego. oraz sterowalnym w krótkim czasie, gdyż rząd macierzy złożonej z pól $\boldsymbol{g}_1, \boldsymbol{g}_2, [\boldsymbol{g}_1, \boldsymbol{g}_2]$ jest pełny i równy 3.

Bibliografia

[1] A. P. Aguiar, A. M. Pascoal. Regulation of a nonholonomic autonomous underwater vehicle with parametric modeling uncertainty using lyapunov functions. *IEEE Conf. on Decision and Control (Cat. No. 01CH37228)*, wolumen 5, strony 4178–4183, 2001.

[2] A. A. Ardentov. Controlling of a mobile robot with a trailer and its nilpotent approximation. *Regular and Chaotic Dynamics*, 21(7-8):775–791, 2016.

[3] D. P. Bertsekas. *Constrained optimization and Lagrange multiplier methods*. Academic Press, 2014.

[4] P. Bhattacharya, M. L. Gavrilova. Roadmap-based path planning-using the voronoi diagram for a clearance-based shortest path. *IEEE Robotics & Automation Magazine*, 15(2):58–66, 2008.

[5] Å. Björck. Numerics of Gram-Schmidt orthogonalization. *Linear Algebra and Its Applications*, 197:297–316, 1994.

[6] B. Bocklund, L. D. Bobbio, R. A. Otis, A. M. Beese, Z.-K. Liu. Experimental validation of Scheil–Gulliver simulations for gradient path planning in additively manufactured functionally graded materials. *Materialia*, 11:100689, 2020.

[7] N. Bourbaki. Lie groups and Lie algebras. *Elements of the History of Mathematics*, strony 247–267. Springer, 1994.

[8] D. Cekus, D. Skrobek. Poszukiwanie optymalnej trajektorii manipulatora SCARA z wykorzystaniem metod RRT i PSO. *Modelowanie Inżynierskie*, 31, 2017.

[9] H. Chen, K. Chang, C. S. Agate. UAV path planning with tangent-plus-Lyapunov vector field guidance and obstacle avoidance. *IEEE Trans. on Aerospace and Electronic Systems*, 49(2):840–856, 2013.

[10] E. Chibrikov. A right normed basis for free Lie algebras and Lyndon–Shirshov words. *Journal of Algebra*, 302(2):593–612, 2006.

[11] W. Chow. Über systeme von linearen partiellen Differentialgleichungen erster Ordnung. *Math. Annalen*, 1(117):98–105, 1939.

[12] R. Daily, D. M. Bevly. Harmonic potential field path planning for high speed vehicles. *American Control Conference*, strony 4609–4614. IEEE, 2008.

[13] A. De Luca, G. Oriolo, P. R. Giordano. Kinematic modeling and redundancy resolution for nonholonomic mobile manipulators. *IEEE Int. Conf. on Robotics and Automation*, strony 1867–1873, 2006.

[14] A. De Luca, G. Oriolo, C. Samson. Feedback control of a nonholonomic car-like robot. *Robot motion planning and control*, strony 171–253. Springer, 1998.

[15] W. J. Dixon, F. J. Massey Jr. *Introduction to statistical analysis.* McGraw-Hill, 1951.

[16] I. Duleba, A. Mielczarek. Evaluation of parameterizations in local Lie-algebraic motion planning. *Advanced, Contemporary Control*, strony 928–940. Springer, 2020.

[17] I. Dulęba. *Algorithms of motion planning for nonholonomic robots.* Wroclaw Univ. of Technology Publishing House, Wroclaw, 1998.

[18] I. Dulęba. *Metody i algorytmy planowania ruchu robotów mobilnych i manipulacyjnych.* EXIT, Akademicka Oficyna Wydawnicza, Warszawa, 2001.

[19] I. Dulęba. Kinematic models of doubly generalized n-trailer systems. *Journal of Intelligent & Robotic Systems*, 94(1):135–142, 2019.

[20] I. Dulęba, W. Khefifi. Pre-control form of the generalized Campbell–Baker–Hausdorff–Dynkin formula for affine nonholonomic systems. *Systems & Control Letters*, 55(2):146–157, 2006.

[21] I. Dulęba, A. Mielczarek. A simulation evaluation of gCBHD formula for driftless nonholonomic systems. *Int. Conf. on Methods and Models in Automation and Robotics*, strony 206–211. IEEE, 2017.

[22] I. Dulęba, A. Mielczarek. Small time local controllability of driftless nonholonomic systems in a task-space. *Archives of Control Sciences*, 29(3):316–322, 2019.

[23] I. Dulęba, J. Sąsiadek. Nonholonomic motion planning based on newton algorithm with energy optimization. *IEEE Transactions on Control Systems Technology*, 11(3):355–363, 2003.

[24] E. Dynkin. Calculation of the coefficients in the campbell–hausdorff formula. *DYNKIN, EB Selected Papers of E.B. Dynkin with Commentary. Ed. by Yushkevich, AA*, strony 31–35, 2000.

[25] W. Findeisen. *Teoria i metody obliczeniowe optymalizacji.* Państwowe Wyd. Naukowe, 1977.

[26] S. Garrido, L. Moreno, M. Abderrahim, F. Martin. Path planning for mobile robot navigation using voronoi diagram and fast marching. *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, strony 2376–2381, 2006.

[27] A. Gasparetto, P. Boscariol, A. Lanzutti, R. Vidoni. Path planning and trajectory planning algorithms: A general overview. *Mechanisms and Machine Science*, 29:3–27, 03 2015.

[28] T. Gawron. *Algorytmizacja ruchu robotów mobilnych z ograniczeniami stanu i wejść sterujących w kontekście metodyki VFO.* Praca doktorska, Politechnika Poznańska, 2019.

[29] R. Gilmore. *Lie groups, physics, and geometry: an introduction for physicists, engineers and chemists.* Cambridge University Press, 2008.

[30] B. Goodwine, J. W. Burdick. Motion planning for kinematic stratified systems with application to quasi-static legged locomotion and finger gaiting. *IEEE Trans. on Robotics and Automation*, 18(2):209–222, 2002.

[31] Y. Guo, J. Mao, Y. Wang, S. Guo, Z. Miao. Adaptive tracking control of nonholonomic mobile manipulators using recurrent neural networks. *Int. Journal of Control, Automation and Systems*, 16(3):1390–1403, 2018.

[32] U. Halder, U. Kalabić. Time-optimal solution for a unicycle path on SE(2) with a penalty on curvature. *IFAC-PapersOnLine*, 50(1):6320–6325, 2017.

[33] M. Hall. *The theory of groups*. Courier Dover Publications, 2018.

[34] R. Horn, C. Johnson. *Matrix analysis*. Cambridge University Press, 1990.

[35] F. Islam, J. Nasir, U. Malik, Y. Ayaz, O. Hasan. RRT*-smart: Rapid convergence implementation of RRT* towards optimal solution. *IEEE Int. Conf. on Mechatronics and Automation*, strony 1651–1656, 2012.

[36] G. Jacob. Lyndon discretization and exact motion planning. *European Control Conf.*, strony 1507–1512, 1991.

[37] J. Jagodzinski. *Metody planowania ruchu układów bezdryfowych bazujące na algorytmie Lafferriera-Sussmanna*. Praca doktorska, Politechnika Wrocławska, 2011.

[38] B. Jakubczyk. Nonholonomic path following with fastly oscillating controls. *Int. Workshop on Robot Motion and Control*, strony 99–103. IEEE, 2013.

[39] J. Jakubiak, K. Tchoń, W. Magiera. Motion planning in velocity affine mechanical systems. *Int. Journ. of Control*, 83(9):1965–1974, 2010.

[40] T. Kai. Mathematical modelling and theoretical analysis of nonholonomic kinematic systems with a class of rheonomous affine constraints. *Applied Mathematical Modelling*, 36(7):3189–3200, 2012.

[41] Y. A. Kapitanyuk, A. V. Proskurnikov, M. Cao. A guiding vector-field algorithm for path-following control of nonholonomic mobile robots. *IEEE Trans. on Control Systems Technology*, 26(4):1372–1385, 2017.

[42] S. Karaman, E. Frazzoli. Sampling-based algorithms for optimal motion planning. *The Int. Journ. of Robotics Research*, 30(7):846–894, 2011.

[43] I. Karcz-Dulęba. The impatience mechanism as a diversity maintaining and saddle crossing strategy. *Int. Journal of Applied Mathematics and Computer Science*, 26(4):905–918, 2016.

[44] M. G. Kendall. *Rank correlation methods.* Griffin, 1948.

[45] W. Khefifi. *Lie-algebraiczne metody planowania ruchu układów afinicznych.* Praca doktorska, Politechnika Wrocławska, 2005.

[46] J. Kitzinger, B. Moret. *The visibility graph among polygonal obstacles: a comparison of algorithms*. Praca doktorska, University of New Mexico, 2003.

[47] J. Komorowski, A. Strasburger. *Od liczb zespolonych do tensorów, spinorów, algebr Liego i kwadryk*. Państwowe Wyd. Naukowe, 1978.

[48] G. Lafferriere, H. J. Sussmann. A differential geometric approach to motion planning. *Nonholonomic Motion Planning*, strony 235–270. Kluwer, 1993.

[49] K. Łakomy, M. M. Michałek. Robust output-feedback VFO-ADR control of underactuated spatial vehicles in the task of following non-parametrized paths. *European Journal of Control*, 58:258–277, 2021.

[50] J.-C. Latombe. *Robot motion planning*, wolumen 124. Springer Science & Business Media, 2012.

[51] J.-P. Laumond, P. E. Jacobs, M. Taix, R. M. Murray. A motion planner for nonholonomic mobile robots. *IEEE Trans. on Robotics and Automation*, 10(5):577–593, 1994.

[52] B. Li, Z. Shao. Simultaneous dynamic optimization: A trajectory planning method for nonholonomic car-like robots. *Advances in Engineering Software*, 87:30–42, 2015.

[53] H. S. W. Liu. Limits of highly oscillatory controls and the approximation of general paths by admissible trajectories. *IEEE Conf. on Decision and Control*, wolumen 1, strony 437–442, 1991. doi: 10.1109/CDC.1991.261338.

[54] O. Ljungqvist, N. Evestedt, D. Axehill, M. Cirillo, H. Pettersson. A path planning and path-following control framework for a general 2-trailer with a car-like tractor. *Journal of Field Robotics*, 36(8):1345–1377, 2019.

[55] G. G. Lorentz. *Bernstein polynomials*. American Mathematical Soc., 2013.

[56] P. Ludwików. *Metody planowania ruchu manipulatorów mobilnych w środowisku kolizyjnym.* Praca doktorska, Politechnika Wrocławska, 2007.

[57] E. Masehian, M. Amin-Naseri. A voronoi diagram-visibility graph-potential field compound algorithm for robot path planning. *Journal of Robotic Systems*, 21(6):275–300, 2004.

[58] A. Mielczarek, I. Dulęba. Development of task-space nonholonomic motion planning algorithm based on Lie-algebraic method. *Applied Sciences*, 11(21):10245, 2021.

[59] A. Mielczarek, I. Dulęba. Theoretical and algorithmic aspects of generating pre-control form of the gCBHD formula. *Int. Conf. on Methods, Models in Automation and Robotics*, strony 905–909, 2018.

[60] A. Mielczarek, I. Dulęba. Premature convergence in motion planning of nonholonomic systems and how to counteract it. *International Conference on Methods and Models in Automation and Robotics*, strony 87–92, 2019.

[61] A. Mielczarek, I. Dulęba. Small radius spheres in output space of nonholonomic systems. *Int. Conf. on Informatics in Control, Automation and Robotics*, wolumen 2, strony 423–436, 2019.

[62] O. Montiel, U. Orozco-Rosas, R. Sepúlveda. Path planning for mobile robots using bacterial potential field for avoiding static and dynamic obstacles. *Expert Systems with Applications*, 42(12):5177–5191, 2015.

[63] R. Mukherjee, D. P. Anderson. Nonholonomic motion planning using stoke's theorem. *IEEE Int. Conf. on Robotics and Automation*, strony 802–809, 1993.

[64] H. Munthe-Kaas, B. Owren. Computations in a free Lie algebra. *Philosophical Trans. of the Royal Society of London. Series A: Mathematical, Physial and Engineering Sciences*, 357(1754):957–981, 1999.

[65] R. M. Murray. *Robotic control and nonholonomic motion planning*. Praca doktorska, University of California, Berkeley, 1991.

[66] R. M. Murray, Z. Li, S. S. Sastry, S. S. Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 1994.

[67] R. M. Murray, S. S. Sastry. Nonholonomic motion planning: Steering using sinusoids. *IEEE Trans. on Autom. Control*, 38(5):700–716, 1993.

[68] Y. Nakamura. *Advanced robotics: redundancy and optimization*. Addison-Wesley Longman Publ. Co., 1990.

[69] Y. Nakamura, W. Chung, O. J. Sordalen. Design and control of the nonholonomic manipulator. *IEEE Trans. Robot. Autom.*, 17(1):48–59, 2001.

[70] Y. Nakamura, R. Mukherjee. Nonholonomic path planning of space robots via bidirectional approach. *IEEE Int. Conf. on Robotics and Automation*, strony 1764–1769, 1990.

[71] J. Nasir, F. Islam, U. Malik, Y. Ayaz, O. Hasan, M. Khan, M. S. Muhammad. RRT*-smart: A rapid convergence implementation of RRT. *Int. Journal of Advanced Robotic Systems*, 10(7):299, 2013.

[72] M. Opałka. *Motion planning of stratified systems*. Praca doktorska, Politechnika Wrocławska, 2014.

[73] M. Otte, E. Frazzoli. Rrtx: Asymptotically optimal single-query sampling-based motion planning with quick replanning. *Int. Journal of Robotics Research*, 35(7):797–822, 2016.

[74] B. S. Park, S. J. Yoo, J. B. Park, Y. H. Choi. A simple adaptive control approach for trajectory tracking of electrically driven nonholonomic mobile robots. *IEEE Trans. on Control Systems Technology*, 18(5):1199–1206, 2009.

[75] I. E. Paromtchik, C. Laugier. Autonomous parallel parking of a nonholonomic vehicle. *IEEE Conf. on Intelligent Vehicles*, strony 13–18, 1996.

[76] D. Popa, J. Wen. Feedback stabilization of nonlinear affine systems. *IEEE Conf. on Decision and Control*, strony 1290–1295, Phoenix, AZ., 1999.

[77] A. Ratajczak, K. Tchoń. Parametryczna i nieparametryczna metoda endogenicznej przestrzeni konfiguracyjnej. *Prace Naukowe Politechniki Warszawskiej. Elektronika*, 2(182):415–424, 2012.

[78] J. Ratajczak, K. Tchoń. Normal forms and singularities of non-holonomic robotic systems: A study of free-floating space robots. *Systems & Control Letters*, 138:1–9, 2020.

[79] N. Ratliff, M. Zucker, J. A. Bagnell, S. Srinivasa. Chomp: Gradient optimization techniques for efficient motion planning. *IEEE Int. Conf. on Robotics and Automation*, strony 489–494, 2009.

[80] C. Reutenauer. Free Lie algebras. *Handbook of algebra*, wolumen 3, strony 887–903. Elsevier, 2003.

[81] P. Rouchon, M. Fliess, J. Lévine, P. Martin. Flatness, motion planning and trailer systems. *IEEE Conf. on Decision and Control*, strony 2700–2705, 1993.

[82] G. Roussos, D. V. Dimarogonas, K. J. Kyriakopoulos. 3d navigation and collision avoidance for nonholonomic aircraft-like vehicles. *Int. Journal of Adaptive Control and Signal Processing*, 24(10):900–920, 2010.

[83] M. Sarfraz, F. u. Rehman, I. Shah. Robust stabilizing control of nonholonomic systems with uncertainties via adaptive integral sliding mode: an underwater vehicle example. *Int. Journal of Advanced Robotic Systems*, 14(5):1729881417732693, 2017.

[84] D. H. Sattinger, O. L. Weaver. *Lie groups and algebras with applications to physics, geometry, and mechanics*, wolumen 61. Springer Science & Business Media, 2013.

[85] J. Sawada, C. Miers, F. Ruskey. Generating Lyndon brackets: a basis for the n-th homogeneous component of the free Lie algebra. *Journal of Algorithms*. Citeseer, 2001.

[86] L. Schaffer, J. Eshelman. Preventing premature convergence in genetic algorithms by preventing incest. *Int. Conf. on Genetic. Morgan Kaufmann, San Mateo, CA*, 1991.

[87] J.-P. Serre. *Lie algebras and Lie groups: 1964 lectures given at Harvard University*. Springer, 2009.

[88] B. Sharma, J. Vanualailai, S. Singh. Motion planning and posture control of multiple n-link doubly nonholonomic manipulators. *Robotica*, 35(1):1–25, 2017.

[89] A. Shirshov. On the bases of a free Lie algebra. *Selected Works of A.I. Shirshov*, strony 113–118. Springer, 2009.

[90] O. J. Sordalen. Conversion of the kinematics of a car with n trailers into a chained form. *IEEE Int. Conf. on Robotics and Automation*, strony 382–387, 1993.

[91] M. Spivak. *Analiza na rozmaitościach*. PWN, Warszawa, 2005.

[92] R. S. Strichartz. The Campbell-Baker-Hausdorff-Dynkin formula and solutions of differential equations. *Journal of Functional Analysis*, 72(2):320–345, 1987.

[93] G. Szegö. Orthogonal polynomials, vol. 23. *American Math. Society Colloquium Publications*, 1975.

[94] Y. Tan, Z. Jiang, Z. Zhou. A nonholonomic motion planning and control based on chained form transformation. *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, strony 3149–3153, 2006.

[95] H. G. Tanner, S. G. Loizou, K. J. Kyriakopoulos. Nonholonomic navigation and control of cooperating mobile manipulators. *IEEE Trans. on Robotics and Automation*, 19(1):53–64, 2003.

[96] K. Tchoń, J. Jakubiak. Endogenous configuration space approach to mobile manipulators: a derivation and performance assessment of jacobian inverse kinematics algorithms. *Int. Journal of Control*, 26(14):1387–1419, 2003.

[97] K. Tchoń, A. Mazur, I. Dulęba, R. Hossa, R. Muszyński. *Manipulatory i roboty mobilne : Modele, planowanie ruchu, sterowanie*. Akademicka Oficyna Wyd. PLJ, Warszawa, 2000.

[98] Y.-P. Tian, S. Li. Exponential stabilization of nonholonomic dynamic systems by smooth time-varying control. *Automatica*, 38(7):1139–1146, 2002.

[99] D. Tilbury, J.-P. Laumond, R. Murray, S. S. Sastry, G. Walsh. Steering car-like systems with trailers using sinusoids. *IEEE Conf. on Robotics and Automation*, 1992.

[100] G. P. Tolstov. *Fourier series*. Courier Corporation, 2012.

[101] J. Vanualailai, B. Sharma, A. Ali. Lyapunov-based kinematic path planning for a 3-link planar robot arm in a structured environment. *Global Journal of Pure and Applied Mathematics*, 3(2):175–190, 2007.

[102] M. Vendittelli, G. Oriolo, J.-P. Laumond. Steering nonholonomic systems via nilpotent approximations: The general two-trailer system. *IEEE Int. Conf. on Robotics and Automation (Cat. No. 99CH36288C)*, wolumen 1, strony 823–829, 1999.

[103] T. L. Vincent, W. J. Grantham. *Nonlinear and optimal control systems*. John Wiley & Sons, 1997.

[104] H. Zhang, Y. Wang, J. Zheng, J. Yu. Path planning of industrial robot based on improved RRT algorithm in complex environments. *IEEE Access*, 6:53296–53306, 2018.

[105] S. Zheng, H. Liu. Improved multi-agent deep deterministic policy gradient for path planning-based crowd simulation. *IEEE Access*, 7:147755–147770, 2019.

[106] Z. Zheng, L. Sun. Adaptive sliding mode trajectory tracking control of robotic airships with parametric uncertainty and wind disturbance. *Journal of the Franklin Institute*, 355(1):106–122, 2018.

[107] Z. Zheng, L. Xie. Finite-time path following control for a stratospheric airship with input saturation and error constraint. *Int. Journal of Control*, 92(2):368–393, 2019.

[108] E. Zhu, J. Pang, N. Sun, H. Gao, Q. Sun, Z. Chen. Airship horizontal trajectory tracking control based on active disturbance rejection control (adrc). *Nonlinear Dynamics*, 75(4):725–734, 2014.

[109] W. Ziller. Lie groups. representation theory and symmetric spaces. 2010.