Rozprawy
Monografie 102

102 DISSERTATIONS
MONOGRAPHIES

ANDRZEJ TURNAU

Target Control of Nonlinear Systems in Real Time – Intelligent and Time-optimal Algorithms

AGH UCZELNIANE WYDAWNICTWA NAUKOWO-DYDAKTYCZNE KRAKÓW 2002

ROZPRAWY
MONOGRAFIE 102

ANDRZEJ TURNAU

Sterowanie docelowe układami nieliniowymi w czasie rzeczywistym – algorytmy inteligentne i optymalnoczasowe

AGH UCZELNIANE WYDAWNICTWA NAUKOWO-DYDAKTYCZNE KRAKÓW 2002

18,00

Uczelniane Wydawnictwa Naukowo-Dydaktyczne
Akademii Górniczo-Hutniczej im. S. Staszica w Krakowie

Recenzent: *Henryk Górecki*

Praca była finansowana przez Komitet Badań Naukowych w ramach projektu badawczego nr 8T11A 018 18 pt. *Optymalizacja sterowania metodami zmiennej parametryzacji.*

Opracowanie edytorskie: *zespół redakcyjny UWND AGH*

Korekta: *Danuta Harnik*

Druk wykonano ze składu dostarczonego przez Autora

ISSN 0867-6631

Redakcja Uczelnianych Wydawnictw Naukowo-Dydaktycznych
al. Mickiewicza 30, 30-059 Kraków
tel. 617-32-28, tel./fax 636-40-38
e-mail: wydagh@uci.agh.edu.pl
http://galaxy.uci.agh.edu.pl/~wydagh

Spis treści

CZĘŚĆ III. STEROWANIE OPTYMALNOCZASOWE

CZĘŚĆ IV. UODPARNIANIE STEROWANIA OPTYMALNOCZASOWEGO

ANDRZEJ TURNAU

Sterowanie docelowe układami nieliniowymi w czasie rzeczywistym – algorytmy inteligentne i optymalnoczasowe

Streszczenie

Praca jest poświęcona problemom projektowania i realizacji algorytmów sterowania dla obiektów nieliniowych o szybkiej dynamice. Tematem przewodnim jest analiza i synteza takich algorytmów. Skonfrontowano dwa, jakościowo różne, podejścia do projektowania układów sterowania: inteligentne i klasyczne (reprezentowane przez algorytmy optymalnoczasowe). Przeanalizowano warunki związane z praktyczną realizacją sterowań, a więc środowisko programowo-sprzętowe oraz odporność i wrażliwość układu sterowania na zakłócenia i zmiany parametrów.

Część I jest wprowadzeniem do zagadnień sterowania w czasie rzeczywistym, w szczególności zajmuje się oprogramowaniem z użyciem takich narzędzi, jak MATLAB/Simulink, RT-CON i dSPACE. Opisano metody szybkiego prototypowania z przykładami zastosowań (sterowanie siłą manipulatora). Wprowadzono standardowy przykład porównawczy, którym jest laboratoryjny system wahadła na wózku. Jako silnie nieliniowy układ czwartego rzędu, jest on od dawna przedmiotem zainteresowania teorii sterowania.

W II części pracy określono procedury sterowania inteligentnego i opisano wybrane algorytmy uczące się. Autor postawił sobie za cel sprawdzenie znanej tezy inżynierii układów inteligentnych, usiłując odpowiedzieć na pytanie, czy heurystyczne generowanie reguł, wykorzystanie logiki rozmytej i sieci neuronowych w sterowaniu jest konkurencyjne w stosunku do metod ściśle posługujących się matematyczną analizą i syntezą.

Część III jest poświęcona sterowaniu optymalnoczasowemu. Podano warunek konieczny optymalności i wprowadzono równania kanoniczne. Następnie przedstawiono algorytmy numeryczne sterowania optymalnego używane w pracy, w tym oryginalną metodę kontynuacyjną z generacją przełączeń i uzgadnianiem gradientów, a także metodę zmiennej parametryzacji ze swobodnym horyzontem. Zaproponowano sposób konstrukcji regulatora optymalnoczasowego przez utworzenie zbioru reprezentatywnego w przestrzeni stanu. Dla przykładu omówiono syntezę sterowania optymalnoczasowego dla systemu lewitacji magnetycznej. Następnie przedstawiono wyniki sterowania w pętli otwartej systemem rzeczywistym wahadła na wózku; sterowanie to wyliczono na modelu za pomocą optymalnoczasowej procedury numerycznej. Pokazano działanie, na rzeczywistym obiekcie sterowania i w pętli zamkniętej, szybkiego algorytmu napisanego w języku C, będącego wersją procedury stałohoryzontowej z repetycyjną optymalizacją.

W części IV zajęto się uodparnianiem rozwiązań optymalnoczasowych na zakłócenia. Przeprowadzono badania wrażliwości posługując się równaniem wariacyjnym dla układu kanonicznego. Warunki skoku w chwilach przełączeń otrzymano za pomocą odpowiedniego

równania Riccatiego. Pokazano izochrony dotarcia do celu z otoczenia punktu podlegającego zakłóceniom, a leżącego na trajektorii optymalnej. Opisano również regulator adaptacyjny z linearyzującym sprzężeniem zwrotnym wokół trajektorii optymalnej. Badano otoczenie trajektorii optymalnej metodą stochastyczną, w celu przekształcenia trajektorii za pomocą przesunięć, tak by stała się bliską trajektorią ekstremalną odporną na zakłócenia. Przeprowadzono analizę i uodporniono rozwiązania dla kilku typowych zadań optymalnoczasowych.

ANDRZEJ TURNAU

Target Control of Nonlinear Systems in Real Time – Intelligent and Time-optimal Algorithms

Summary

The book is devoted to design and implementation of control algorithms for nonlinear plants with fast dynamics. The attention is focused on the analysis and synthesis of such algorithms. Two qualitatively different approaches are confronted: the intelligent and classical one (represented by time-optimal algorithms). Practical control implementation is discussed, including software-hardware environment and sensitivity of control systems to disturbances and parameter variations.

Part I is an introduction to real-time control, in particular it deals with programming issues connected with such tools as MATLAB/Simulink, RT-CON and dSPACE. Rapid prototyping methods are described with application examples (manipulator force control). A benchmark cart-pendulum system is introduced. As a strongly nonlinear plant of fourth order it has long been an object of interest for control theory.

Part II deals with intelligent control. Selected intelligent and learning algorithms are described. The author's aim is to verify the well-known hypothesis of intelligent systems engineering that heuristic rules generation, fuzzy logic and neural networks are a serious alternative for methods based on classical mathematical tools.

Part III is devoted to time-optimal control. Necessary optimality conditions are given together with canonical equations. Next, the numerical algorithms of time-optimal control used in the book are presented, including the new continuation method with switching generation and gradient matching, as well as the free-horizon method of variable parameterization. A construction of time-optimal regulator by means of representative sets in state space is then proposed. An example of time-optimal control synthesis for a system of magnetic levitation is given. Further, the results of open-loop time-optimal control in a real cart-pendulum system are discussed. The next experiment shows the performance of a fast time-optimal algorithm written in C, in a closed-loop real plant scheme with repetitive optimization.

The robustification of time-optimal solutions to disturbances is the subject of Part IV. The variational canonical equation is used for sensitivity studies. The jump conditions are determined from the solution of an appropriate Riccati equation. The isochrones for reaching the target from a neighborhood of the disturbed state on the optimal trajectory are shown. An adaptive linearized controller is described. The neighborhood of the optimal state trajectory is examined by a stochastic method in order to transform the trajectory, by shifting it, into a neighboring extremal trajectory robust to disturbances. Finally, examples of robustification of time-optimal solutions are shown for several typical control problems.

Wykaz ważniejszych oznaczeń

$A(x,u)^{\cdot}$ – macierz równania sprzężonego

$E(\gamma)$ – wydajność generacji

f – wektor prawych stron równań stanu

c^i – i-ty cel pośredni w metodzie uodparniania trajektorii

f^0, f^1 – funkcje w prawej stronie równania stanu

g – antygradient

$\hat{g}$ – antygradient funkcjonału $\hat{\Sigma}_T$

g^U – rzut antygradientu g na zbiór U

$H(x,\psi,u)$ – hamiltonian

$H_{xx}(x,\psi,u)$ – hesjan hamiltonianu (macierz drugich pochodnych względem stanu)

$J(t)$ – macierz kanonicznego równania wariacyjnego

$K(t)$ – macierz Riccatiego wyrażająca liniową zależność wariacji zmiennej sprzężonej $\delta\psi$ od wariacji stanu δx

p^i – i-ty punkt pośredni w metodzie uodparniania trajektorii

P_{D_i}, P_{N_i} – zbiór zaburzeń dopuszczalnych i niedopuszczalnych na poziomie ε

P_T – stałohoryzontowy problem pomocniczy dla zadania optymalnoczasowego, z horyzontem T

P_k – trajektorie wzorcowe do konstrukcji zbioru reprezentatywnego

R_{opt} – regulator optymalny

$S(u,T)$ – wskaźnik jakości w problemie ze swobodnym czasem końcowym

$S^*(u,T)$ – wartość minimalna wskaźnika jakości w problemie ze swobodnym czasem końcowym

$S_q(u,T)$ – wskaźnik jakości z horyzontem T jako zmienną decyzyjną, sparametryzowany przez q

$S_q{}^*(u,T)$ – wartość minimalna wskaźnika jakości w problemie pomocniczym z horyzontem T jako zmienną decyzyjną i parametrem q

$S'_q(\tau,u_0,T)$ – funkcjonał powstały po obcięciu wskaźnika jakości $S_q(u,T)$ do sterowań bangbangowych

t^* – horyzont optymalny

T – ustalony horyzont sterowania

T_d, T_g – przybliżenia dolne i górne horyzontu optymalnego

$^\mathsf{T}$ – znak transpozycji

u – sterowanie

u^T – sterowanie optymalne w problemie P_T, minimalizujące Σ_T przy stałym horyzoncie T

u^* – sterowanie optymalnoczasowe

$\hat{u}$ – sterowanie schodkowe

U – zbiór dopuszczalnych wartości sterowania

U_{ad} – zbiór sterowań dopuszczalnych

$V_1(t)$ – macierz wrażliwości stanu w chwili t na zmiany stanu w chwili końcowej T

$V_2(t)$ – macierz wrażliwości stanu sprzężonego w chwili t na zmiany stanu w chwili końcowej T

$W_{x^0,x^f}(t)$ – macierz wrażliwości optymalnego stanu końcowego na zaburzenia stanu w chwili t dla zadania stałohoryzontowego ze stanem początkowym x^0 i celem końcowym x^f

$W^{T_1+T_{21}+T_{22}}_{x^0,p^1,p^2,x^f}(t)$ – macierz wrażliwości uodpornionego stanu końcowego na zaburzenia stanu w chwili t dla zadania stałohoryzontowego o horyzontach T_1, T_{21} i T_{22}, ze stanem początkowym x^0, punktami pośrednimi p^1, p^2 i celem końcowym x^f

x – zmienna stanu

x^T – trajektoria optymalna w problemie P_T, przy stałym horyzoncie T

x^0, x^f – stan początkowy i stan docelowy (cel końcowy sterowania)

x^* – trajektoria optymalnoczasowa

X – trajektoria kanoniczna

$x^{T_1+T_2}_{x^0,p^1,x^f}$ – trajektoria sklejona z optymalnych trajektorii stałohoryzontowych, przechodząca przez punkt pośredni p^1

$x^{T_1+T_{21}+T_{22}}_{x^0,p^1,p^2,x^f}$ – trajektoria sklejona z optymalnych trajektorii stałohoryzontowych, przechodząca przez punkty pośrednie p^1 i p^2

$Z_{i\mp}$ – macierz nieciągłości wariacji stanu podczas przełączenia

$\alpha_0,...,\alpha_4$ – współczynniki skalowania

γ – niemalejący ciąg dodatkowych czasów przełączeń w generacji

δX – wariacja trajektorii kanonicznej

$\delta\tau_i$ – wariacja i-tego czasu przełączenia

$\overline{\Delta}^{(j)}$ – rzut antygradientu $\hat{g}^{(j)}$ na zbiór dopuszczalny

$\Delta x^l(t_1)$ – l-te zaburzenie trajektorii w chwili t_1

$\Lambda_{i\pm}$ – wektor współczynników wariacji stanu w wyrażeniu na wariację i-tego czasu przełączenia

ψ – zmienna sprzężona

ψ^T – optymalna trajektoria sprzężona w problemie P_T

ψ^0, ψ^f – wartości brzegowe zmiennej sprzężonej

ψ^* – sprzężona trajektoria optymalnoczasowa

Π_i – współczynniki wzmocnienia zlinearyzowanego regulatora przełączeń, dla i-tego przełączenia

$\Sigma_T(u)$ – wskaźnik jakości w problemie pomocniczym z ustalonym horyzontem T

Σ_T^* – minimum wskaźnika jakości w problemie pomocniczym z ustalonym horyzontem T

$\hat{\Sigma}_T(\hat{u})$ – wskaźnik jakości w problemie pomocniczym z ustalonym horyzontem T i sterowaniem schodkowym

$\tau_1,\tau_2,...,\tau_m$ – chwile przełączeń sterowania bangbangowego (czasy przełączeń)

$\tau \sim \gamma$ – złożenie ciągów niemalejących τ i γ

ϕ, ϕ^* – funkcja przełączająca i optymalna funkcja przełączająca

$\Phi(t,s)$ – macierzowe rozwiązanie fundamentalne równania wariacyjnego

$\nabla_x f(x,u)$ – pochodna funkcji f względem zmiennej x

$\nabla S_q'(\tau,u_0,T)$ – gradient S_q' względem wektora $\mathrm{col}(\tau_1,\tau_2,...,\tau_m,T)$

Wstęp

W pracy przedstawiono problemy projektowania i realizacji algorytmów sterowania dla obiektów o szybkiej dynamice. Przeanalizowano warunki związane z praktyczną realizacją sterowań, a więc: środowisko programowo-sprzętowe, błędy pomiarów i sterowań, odporność i wrażliwość układu sterowania na zakłócenia i zmiany parametrów modelu. Poświęcono uwagę praktycznemu rozwiązaniu postawionych zadań sterowania.

Tematem przewodnim jest analiza i synteza algorytmów sterowania układami nieliniowymi. Wyróżniono dwa, jakościowo różne, podejścia do projektowania układów sterowania: inteligentne oraz klasyczne (reprezentowane przez algorytmy optymalnoczasowe), z zamiarem, by przeciwstawienie obu podejść stało się widoczne i dostarczyło wyczerpującego materiału porównawczego do oceny przedstawionych technik.

Aby uniknąć niejasności terminologicznych, związanych z pojęciem procedury inteligentnej, rozróżniono materiał zawarty w odrębnych częściach w następujący sposób: procedury inteligentne charakteryzuje bezpośrednie, intuicyjne naśladowanie postępowania żywych istot inteligentnych, natomiast procedury klasyczne (szczególnie optymalnoczasowe) wyróżnia wykorzystanie zaawansowanych matematycznych metod analizy i syntezy przy budowie algorytmów sterowania. Naśladownictwo istot żywych może dotyczyć albo fizycznych czynności, albo mechanizmów instynktownych, bądź też rozumowych metod wnioskowania. Cechy działania właściwe istotom żywym, przeniesione w uproszczonej formie na grunt procedur algorytmicznych, określa się mianem „sztucznej inteligencji".

Część I jest wprowadzeniem do sterowania w czasie rzeczywistym. Przy projektowaniu algorytmów sterowania obiektami rzeczywistymi należy już na wstępie mieć na względzie ich zastosowanie. Algorytmów nie buduje się w oderwaniu od środowiska sprzętowo-pomiarowego zakładając, że ich późniejsza implementacja będzie zabiegiem czysto technicznym. Wydaje się, że teza ta jest dobrze rozumiana, także przez teoretyków (Pesch 1989a, b, 1994), pracujących nad rozwojem numerycznych algorytmów rozwiązywania problemów sterowania optymalnego. Rozdział 1 pracy jest poświęcony oprogramowaniu do sterowania w czasie rzeczywistym. Platformą projektową i zarazem wykonawczą algorytmów jest szeroko dostępny komputer osobisty, pracujący pod systemem operacyjnym MS Windows NT lub MS Windows 2000. Zachowanie reżimów czasu rzeczywistego jest podstawowym wymogiem poprawności działania algorytmu w przypadkach, gdy dane pomiarowe i sterowania są próbkowane z częstotliwością większą niż 10 Hz. Im szybsza dynamika sterowanego obiektu, tym większe wymagania stawia się środowisku czasu rzeczywistego. Dlatego

w pracy używa się narzędzia programowego RT-CON, rozszerzającego system operacyjny, lub stosuje się dodatkową kartę procesorową dSPACE. Narzędzia i metody szybkiego prototypowania są tematem rozdziału 2. W przykładzie dotyczącym budowy algorytmu sterowania siłą manipulatora uwagę zwraca się przede wszystkim na część komunikacyjną algorytmu sterowania, tzw. sterowniki urządzeń. Ta warstwa programowa, nie używana w symulacji, staje się nieodzowna w eksperymentach rzeczywistych. Obecność warstwy projektowo-wykonawczej środowiska czasu rzeczywistego uwidoczniono także w dalszej części pracy (podrozdziały 6.2, 9.3 i rozdział 10), wszędzie tam, gdzie ma miejsce implementacja algorytmu. Konkurencyjność podejść do budowy układu regulacji jest najlepiej widoczna, gdy dotyczy tego samego obiektu sterowania – pewnego standardu laboratoryjnego. Z tego względu obszerną część pracy poświęcono procedurom sterowania systemem wahadła na wózku. W rozdziale 3 opisano ten system porównawczy. Jako nietrywialny, silnie nieliniowy układ czwartego rzędu, będący reprezentantem szerokiej klasy układów dynamicznych, jest od dawna przedmiotem zainteresowania wielu autorów: Michie'a, Chambersa (1968); Morie'a, Nishihary, Furuty (1976); Furuty, Ochai, Ono (1984); Furuty, Kaiwary, Kosugi (1988); Meiera, Farwiga, Unbehauena (1990); Kołka, Turnaua (1990); Zhanga, Edmundsa (1992); Gevy, Sitte (1993); Kratochwila, Engelbrechta, Jörgla (1993); Lindena, Lambrechtsa (1993); Kołka, Tabakowskiego, Turnaua (1993); Turnaua (1993, 1994, 1995, 1997, 1998); Chung Choo Chunga, Hausera (1995); Qifeng Wei'a, Dayawansy, Levine'a (1995); Lina *et al.* (1996); Turnaua, Korytowskiego 1996; Kempy *et al.* (1997); Turnaua, Korytowskiego 1997; Gevy, Hanga, Zhanga (1998); Gregi, Turnaua (1998); Korytowskiego, Szymkata, Turnaua (1998, 2001); Kajiwary, Apkariana, Gahineta (1999); Medrano-Cerdy (1999); Szymkata, Korytowskego, Turnaua (1999a, b, 2000); Tsachouridisa (1999); Turnaua, Korytowskiego, Szymkata (1999); Åströma, Furuty (2000).

Część II pracy dotyczy sterowania inteligentnego. W rozdziale 4 określono procedury inteligentne i opisano wybrane algorytmy uczące się. Rozdziały 5 i 6 służą sprawdzeniu znanej tezy inżynierii układów inteligentnych i usiłują odpowiedzieć na pytanie, czy heurystyczne generowanie reguł, wykorzystanie logiki rozmytej i sieci neuronowych w sterowaniu jest konkurencyjne w stosunku do metod ściśle posługujących się matematyczną analizą i syntezą. Właśnie procedury optymalnoczasowe, z trzeciej części pracy, powstają przy użyciu metod klasycznych, są zatem dobrym odniesieniem dla porównań z technikami inteligentnymi. Okazuje się, że przedstawiony w rozdziale 5 algorytm regułowy (Turnau 1993, 1994) można częściowo wykorzystać, aby wspomóc procedury optymalnoczasowe i uzyskać niezawodne działanie optymalnoczasowego regulatora rzeczywistego, zbudowanego w oparciu o przybliżony model.

Treścią III części pracy jest sterowanie optymalnoczasowe. W rozdziale 7 sformułowano problemy sterowania optymalnoczasowego, podano warunek konieczny optymalności i wprowadzono równania kanoniczne. W rozdziale 8 przedstawiono optymalnoczasowe algorytmy numeryczne używane w pracy. Przegląd rozpoczęto od prostej metody *gradientów sprzężonych w przestrzeni sterowań*. Stosowano metodę aproksymacji „schodkowej" z uwagi na jej niskie wymagania odnośnie regularności aproksymowanych funkcji i dobrą ilustracyjność. W kolejnej metodzie Szymkata, Korytowskiego, Turnaua (1999a), nazwanej *kontynuacyjną z generacją przełączeń i uzgadnianiem gradientów*, optymalizację w ustalonej przestrzeni decyzyjnej (czasów przełączeń) oparto na gradientowych metodach zmiennej metryki, modyfikując wymiar przestrzeni decyzyjnej przez generację i redukcję przełączeń. W tej metodzie optymalizację prowadzi się dla stałego horyzontu. Po osiągnięciu mi-

nimum wydłuża się horyzont przez kontynuację i ponawia się optymalizację przy jego nowej wartości. Ostatnia z pokazanych metod Szymkata, Korytowskiego, Turnaua (2000) to *zmienna parametryzacja ze swobodnym horyzontem*. Minimalizuje się w niej pomocniczy funkcjonał zależny od horyzontu i sparametryzowany współczynnikiem wagowym horyzontu. Zbędne, w odróżnieniu od poprzednich algorytmów, staje się wyposażanie algorytmu w zabezpieczenia przeciw przekroczeniu horyzontu optymalnego, gdyż wartość nowego funkcjonału dla horyzontów większych od optymalnego pozostaje liczbą dodatnią. Podczas optymalizacji zachowany jest niski wymiar przestrzeni decyzyjnej i szeroki zakres zbieżności algorytmu. Rozdział 9 zawiera wybrane problemy syntezy sterowań optymalnoczasowych dla modeli: systemu wahadła na wózku i lewitacji magnetycznej. Poszukiwanie analitycznego opisu *powierzchni przełączeń* dla układu o rzędzie większym od dwóch staje się zadaniem trudnym. Regulator optymalnoczasowy buduje się przez utworzenie zbioru reprezentatywnego w przestrzeni stanu. Zbiór taki zawiera wyselekcjonowane punkty trajektorii optymalnoczasowych i odpowiadające im wartości sterowania, odpowiednio gęsto pokrywające czterowymiarową przestrzeń stanu (Turnau, Korytowski 1997). Dla modelu lewitacji magnetycznej syntetyzuje się sterowanie optymalnoczasowe, a następnie stosuje się je podczas eksperymentu w pętli otwartej, prowadzonego w czasie rzeczywistym, z kulką lewitującą w polu magnetycznym. Sterowanie przemieszcza kulkę między dwoma poziomami w możliwie najkrótszym czasie. Rozdział 10 zawiera wyniki istotne z punktu widzenia celów pracy. Pierwsza jego część przedstawia sterowanie w systemie rzeczywistym, w pętli otwartej, wyliczone na modelu za pomocą optymalnoczasowej procedury numerycznej (Turnau, Korytowski, Szymkat 1999). Powodzenie takich eksperymentów świadczy o dopasowaniu modelu do rzeczywistego obiektu sterowania i otwiera pole do adaptacji sterowania na podstawie pomiaru stanu rzeczywistego układu, wykonywanego podczas eksperymentu. W części drugiej rozdziału pokazano w działaniu, na rzeczywistym obiekcie sterowania, szybki algorytm napisany w języku C. Algorytm ten jest nieco zmodyfikowaną stałohoryzontową procedurą opisaną w podrozdziale 8.2. Rozwiązuje problem optymalnoczasowy w ciągu jednego, dwóch, a w najgorszym przypadku kilku kroków próbkowania. Warunkiem koniecznym powodzenia metody jest szybkość i skuteczność znajdowania minimów globalnych przez gradientowe algorytmy numeryczne zaimplementowane w środowisku programowo-sprzętowym czasu rzeczywistego komputera oraz odpowiednia szybkość obliczeniowa tego środowiska. Warunki, o których mowa, udaje się coraz lepiej realizować. Powstaje możliwość bezpośredniej (w czasie rzeczywistym) implementacji algorytmów optymalnoczasowych.

Badania eksperymentalne potwierdzają znany z teorii sterowania optymalnego fakt, że rozwiązania optymalnoczasowe cechuje duża wrażliwość na zakłócenia i na zmianę parametrów układu. Dlatego w części IV pracy zajęto się uodparnianiem rozwiązań optymalnoczasowych. W rozdziale 11 przeprowadzono badania wrażliwości. Posłużono się równaniem wariacyjnym, by uzyskać formuły na wrażliwość stanu, z uwzględnieniem warunków końcowych. Praktycznie interesująca jest wrażliwość stanu w chwili końcowej na zmiany stanu w chwili wcześniejszej. Warunki skoku w chwilach przełączeń można otrzymać za pomocą rozwiązania odpowiedniego równania Riccatiego. Pokazano izochrony dotarcia do celu z otoczenia punktu podlegającego zakłóceniom, a leżącego na trajektorii optymalnej. W ostatnich dwóch rozdziałach korzystano z rozwiązań wariacyjnych z rozdziału 11. W 12 opisano regulator adaptacyjny z linearyzującym sprzężeniem zwrotnym wokół trajektorii

optymalnej (Korytowski, Szymkat, Turnau 2001). To podejście do sterowania porównuje się z metodą powtarzanych poprawek (ang. *repeated corrections*) – implementacji sterowań optymalnoczasowych (Pesch 1989b; Maurer, Pesch 1994). W rozdziale 13 przedstawiono eksperymenty uodparniania trajektorii optymalnej na zakłócenia. Badano otoczenie trajektorii optymalnej metodą stochastyczną, celem wyszukania kierunków przesunięcia trajektorii, tak by stała się bliską trajektorią ekstremalną odporną na zakłócenia. Przebadano i zilustrowano graficznie kilka typowych zadań optymalnoczasowych dla porównawczego systemu wahadła na wózku. W efekcie otrzymano sterowania niewrażliwe, o czasie sterowania nieznacznie dłuższym od optymalnego. Z punktu widzenia założonych celów pracy wyniki uodparniania trajektorii są równie istotne, jak eksperymentalne rezultaty rozdziału 10, dotyczące regulacji optymalnoczasowej. Wyniki te są najbardziej znaczące dla dalszych badań, gdyż dotychczasowe eksperymenty pokazują, że nawet jeżeli wygenerowane oryginalne sterowanie optymalnoczasowe uda się zastosować w systemie sterowania – dzięki odpowiednim środkom technicznym i środowisku czasu rzeczywistego – to może okazać się ono nieskuteczne w zadaniach o bardzo dużej wrażliwości. Można wówczas uodpornić trajektorię przez przesunięcie, a tym samym zmniejszyć jej wrażliwość na zakłócenia, pozostając w klasie rozwiązań bliskich optymalnoczasowym.

Autor bardzo serdecznie dziękuje Panu Profesorowi Henrykowi Góreckiemu, wieloletniemu opiekunowi naukowemu, za stworzenie warunków do powstania tej pracy, Adamowi Korytowskiemu i Maciejowi Szymkatowi za współpracę naukową i wiele cennych merytorycznych uwag przy pisaniu pracy, Krzysztofowi Kołkowi za współpracę przy aplikacjach czasu rzeczywistego i wykonanie tłumaczeń na język C, Stanisławowi Fuksie za cenne uwagi, Adamowi Piłatowi za współpracę przy uruchamianiu sterowania lewitacją magnetyczną i Annie Stengl za korektę tekstu pracy.

Część I
ELEMENTY STEROWANIA W CZASIE RZECZYWISTYM

1. Oprogramowanie do sterowania w czasie rzeczywistym

Przystępując do budowy algorytmu sterowania, należy w projekcie od początku uwzględnić specyfikę środowiska, w którym docelowo algorytm będzie stosowany. Istotny jest sposób komunikacji z obiektem podlegającym sterowaniu, trzeba zatem określić, które zmienne stanu są mierzone, jaka jest dokładność pomiaru i jaka jest częstotliwość zbierania pomiarów. Podobne pytania dotyczą sterowań.

- Czy jest zachowany reżim czasu rzeczywistego, a więc czy momenty czasowe zbierania pomiarów i wysyłania sterowań są zdeterminowane – czy można określić przedziały czasowe opóźnień obsługi tych zdarzeń w systemie?
- Jaką przyjęto warstwę programową (sterowniki urządzeń – ang. *device drivers*) i sprzętową (karty I/O) do komunikacji komputera z obiektem sterowanym?
- Jakie są przewidywane błędy odczytu pomiarów zmiennych stanu i błędy przetwarzania sygnału sterowania przy przejściu z poziomu programu w komputerze do poziomu sygnału wykonawczego w urządzeniu?
- Jak odtwarzane są niemierzalne zmienne stanu?
- Czy nie zachodzi potrzeba filtracji sygnałów, kompensacji niepożądanych zjawisk w systemie itp.?
- Czy czas obliczeń algorytmu nie przekracza czasu próbkowania komputera?
- Czy system operacyjny komputera winien być rozszerzony do pracy w czasie rzeczywistym?
- Czy reżimy czasowe sterowania nie wymuszają użycia specjalizowanego systemu operacyjnego czasu rzeczywistego?

Jeżeli zignoruje się odpowiedzi na postawione pytania przy budowie algorytmu, to może się okazać, że nawet bardzo wyrafinowany algorytm ma znaczenie czysto modelowe. Zatem rodzi się potrzeba uwzględniania w algorytmie warunków pracy w czasie rzeczywistym już na etapie jego powstawania (Kołek *et al.* 1995; Grega, Kołek, Turnau 1997, 1998a, b; Carew, Prince 1997).

1.1. Środowisko i system czasu rzeczywistego

Pod pojęciem środowiska rozumie się sprzęt i oprogramowanie. Wybór lub dostosowanie środowiska do pracy w czasie rzeczywistym jest podstawowym zadaniem przed przystąpieniem do projektowania i w czasie późniejszej realizacji układu sterowania.

Procesy wolnozmienne (o stałych czasowych od kilkunastu sekund do kilkudziesięciu minut) pozostawiają dużą swobodę wyboru środowiska czasu rzeczywistego. Opóźnienia w generacji sterowań rzędu kilkunastu milisekund praktycznie nie wpływają na zachowanie układu o kilkunastosekundowych stałych czasowych. Dla procesów o takich stałych czasowych stosuje się do sterowania i regulacji programowalne sterowniki logiczne i regulatory funkcyjne (Trybus 1992). Właśnie one służą jako sprzętowo-programowe środowisko czasu rzeczywistego i są powszechnie używane w zastosowaniach przemysłowych.

Dla systemów sterowania procesami o szybkiej dynamice ruchu zawężają się możliwości wyboru środowiska czasu rzeczywistego. Czas próbkowania procesorowych urządzeń pomiarowo-sterujących mierzy się wówczas w milisekundach i dopuszcza pewne opóźnienia w generacji sterowań, rzędu kilkudziesięciu mikrosekund. Dla pracy w czasie rzeczywistym istotne jest zapewnienie determinizmu zdarzeń czasowych. Im dokładniej system potrafi przewidzieć czasy opóźnień własnej reakcji na występujące zdarzenia, tym lepiej zachowuje reżimy pracy w czasie rzeczywistym.

Środowiskiem sprzętowym jest przeważnie komputer z pamięcią i procesorem lub sieć komputerów z kartami wejść/wyjść, wzmacniaczami, interfejsem pomiarowo sterującym, czujnikami pomiarowymi i elementami wykonawczymi. Środowiskiem programowym jest system operacyjny, oprogramowanie wspomagające projektowanie systemów sterowania, oprogramowanie sterujące i programowe narzędzia pomocnicze (kompilatory, graficzne interfejsy użytkownika, symulatory itp.).

Dla projektanta systemu sterowania nie bez znaczenia jest to, gdzie i jak powstaje projekt sterownika. Najlepiej, jeżeli platforma źródłowa staje się platformą docelową, to znaczy sterownik powstaje w tym samym komputerze, w którym później wykonywany jest kod wynikowy sterujący systemem. Co więcej, te same narzędzia powinny być wykorzystywane na etapie projektowania, testowania i końcowej aplikacji. Takie środowisko nazywa się *zintegrowanym* środowiskiem sterowania (Schmid 1992; Grega 1996). Winno być ono *otwarte*, *przyjazne* dla użytkownika i *konfigurowalne*.

Otwartość jest rozumiana jako wykorzystanie standardów sprzętowych, standardowych systemów operacyjnych i narzędzi programowych do projektowania i testowania algorytmów sterowania. Środowisko przyjazne dla użytkownika uzyskuje się poprzez wykorzystanie obiektowo zorientowanych narzędzi do tworzenia interfejsu użytkownika i przez użycie narzędzi wizualizacji (obiekty graficzne o hierarchicznej strukturze, konfigurowalne okna, animacje, techniki multimedialne). Środowisko konfigurowalne oznacza umożliwienie zmian struktur i parametrów testowanych regulatorów, sposobów obserwacji, wartości zadanych, zakłóceń itp., podczas trwania eksperymentu. Konfigurowalne środowisko może także korzystać z automatycznie generowanego kodu wynikowego dla regulatorów.

Według standardu IEEE/ANSI (Szmuc 1998) pod pojęciem *systemu czasu rzeczywistego* rozumie się system komputerowy, w którym obliczenia są wykonywane współbieżnie z procesem zewnętrznym w celu sterowania, nadzorowania lub terminowego reagowania na zdarzenia występujące w tym procesie.

System czasu rzeczywistego winien być:

- ciągły w działaniu,
- zależny od procesu zewnętrznego,
- współbieżny z procesem zewnętrznym,
- przewidywalny,
- punktualny.

Proces zewnętrzny przebiega bez przerw, generując dane lub zdarzenia wymagające obsługi przez system czasu rzeczywistego. Dlatego obliczenia w systemie są uzależnione od zdarzeń i danych procesu zewnętrznego, które pojawiają się na ogół w przypadkowych momentach czasu i często współbieżnie żądają obsługi. System powinien więc reagować na zewnątrz w przewidywalny, a więc deterministyczny sposób na żądania jednoczesnej obsługi. Wyniki obliczeń, wykonane zgodnie z użytymi algorytmami, muszą być nie tylko poprawne, ale dodatkowo muszą być dostarczone do procesu zewnętrznego punktualnie, a więc nie za późno i również nie zbyt wcześnie. Cechy charakteryzujące system czasu rzeczywistego uwidaczniają się we wszystkich elementach tworzących ten system. Dodatkowe wymagania pracy w czasie rzeczywistym stawia się więc językom czasu rzeczywistego, systemom operacyjnym, programom obsługującym urządzenia wejścia/wyjścia – sterownikom urządzeń, itp. (Motet, Szmuc 1998).

I tak język do pracy w czasie rzeczywistym winien zapewnić (Szmuc 1998):

- zdefiniowanie spójnych części programu do obsługi zadań aktywizowanych przez zdarzenia zewnętrzne i upływ czasu,
- mechanizm synchronizacji i wymiany informacji między zadaniami realizującymi wspólny cel sterowania,
- mechanizm uzależnienia działania systemu od czasu – aktywizowanie zadania w określonym momencie czasowym lub przedziale czasowym,
- możliwość reakcji na zdarzenia procesu zewnętrznego i definiowania sterowników urządzeń do obsługi nietypowych wejść i wyjść procesu.

1.2. System operacyjny czasu rzeczywistego

Dla zadań czasu rzeczywistego ideałem byłby system operacyjny z obsługą zadań, określoną przez przewidywane terminy zakończenia zadań (ang. *deadlines*). Taki system jeszcze nie powstał, a obecne systemy obsługują zadania zgodnie z priorytetami.

Podstawowymi cechami systemu operacyjnego czasu rzeczywistego są: wielozadaniowość, wywłaszczanie przerwań, sterowanie na podstawie priorytetów, współbieżność procesów, przewidywalność i punktualność reakcji na zewnętrzne zdarzenia.

Minimalne wymagania stawiane systemowi operacyjnemu czasu rzeczywistego są następujące:

- wielozadaniowość z wywłaszczaniem (szczególnym przypadkiem może być jedno zadanie czasu rzeczywistego obsługiwane przez system operacyjny);
- przydział priorytetów dla zadań i wątków oraz dostępność poziomów priorytetów;

- obsługa mechanizmów przewidywalnej synchronizacji procesów (semafory, wielowątki, zdarzenia itp.);
- obsługa mechanizmów komunikacji między zadaniami;
- zabezpieczenie przed inwersją priorytetów;
- przewidywalne działanie, tzn. winny być określone czasy oczekiwania – opóźnienia: przerwań, przełączników zadań, sterowników urządzeń i dostępu do pamięci; zdarzenia i dane generowane przez otoczenie pojawiają się współbieżnie, wymagają więc jednoczesnej obsługi, co prowadzi do wewnętrznych niedeterministycznych zachowań, które na zewnątrz muszą być postrzegane jako deterministyczne;
- zdefiniowanie maksymalnego czasu reakcji na wszystkie możliwe obciążenia systemu, czyli tzw. metryki systemu.

Każda aplikacja nakłada na system operacyjny czasu rzeczywistego własne wymagania co do szybkości obsługi. Na przykład, jak przedstawiono w dalszej części pracy, by sterować systemem wahadła na wózku, trzeba generować sterowania co 10 ms, zaś by podtrzymywać lewitującą kulkę w polu magnetycznym, należy generować sterowania co 1 ms lub co 0,678 ms przy nadążaniu za sygnałem wartości zadanej w procedurze optymalnoczasowej.

Co więcej, aby system operacyjny czasu rzeczywistego zapewniał krótkie (w odniesieniu do czasu próbkowania) czasy wykonywania instrukcji, ważna jest też szybkość jego reakcji na wystąpienie zewnętrznego zdarzenia. Mówimy o różnego typu opóźnieniach. Pierwszym z nich jest *czas oczekiwania po wystąpieniu przerwania*, który upływa od wykonania ostatniej instrukcji przerwanego zadania do pierwszej instrukcji obsługi przerwania. Inny jest *czas oczekiwania podczas powrotu z poziomu obsługi przerwań do poziomu zadań*. Aplikacja czasu rzeczywistego wymaga na ogół kilku zadań, które dzielą czas między siebie. Czas poświęcony przez system operacyjny na przełączenie nazwiemy *czasem zmiany kontekstu*. To opóźnienie różni się od opóźnień wymienionych poprzednio.

Poza szybkością wykonywania instrukcji i szybkością reagowania na zdarzenia, ważne jest, by przy dużym i małym obciążeniu systemu czasy zużywane przez system na obsługę takich samych zdarzeń były podobne. Rozróżnia się systemy o twardych wymaganiach czasowych, które mieszczą się z obsługą zdarzeń i operacjami na danych w określonych okienkach czasu, oraz systemy o miękkich wymaganiach czasowych, które na ogół mieszczą się z obsługą w ściśle określonych okienkach czasu, a zdarzające się wyjątkowo niedotrzymanie któregoś z terminów nie powoduje zawieszenia dalszej obsługi (Timmerman, Van Beneden, Uhres 1998a, b).

1.3. MS Windows NT i MS Windows 2000

Największy wpływ na poprawną pracę układu w czasie rzeczywistym ma system operacyjny. Profesjonalne systemy czasu rzeczywistego, na przykład Vx-Work, OS-9 i QNX, są kosztowne. Dlatego do prowadzenia eksperymentów sterowania wykorzystywano w tej pracy komputer osobisty działający pod kontrolą systemu operacyjnego MS Windows NT lub MS Windows 2000. Te systemy są platformami o otwartej architekturze. Otwartość

umożliwia dostęp do wielu aplikacji. Graficzny interfejs użytkownika (GUI) w systemach MS Windows stał się standardem. Obok licznych korzyści wynikających z używania tego standardu istnieją też poważne wady. Po pierwsze, MS Windows NT i MS Windows 2000 nie są systemami czasu rzeczywistego. W MS Windows używa się odkładanych na stos wywołań procedur, o priorytecie niższym od priorytetu wywołań systemowych i zadań użytkownika. Chodzi tu o zminimalizowanie czasu obsługi przerwań. Dlatego czasy obsługi wywołań odkładanych na stosie metodą FIFO są zmienne i nieprzewidywalne, przy zmieniającym się obciążeniu systemu.

Obecnie środkami programowymi lub sprzętowymi rozszerza się MS Windows NT lub MS Windows 2000, tak by wyposażyć je w cechy systemu operacyjnego czasu rzeczywistego. Zabiegi takie uzasadnia fakt istnienia standardu Win32 API z największą biblioteką programów, graficznym interfejsem o dużej popularności, gotowymi rozwiązaniami komunikacyjnymi i oprogramowaniem pisanym pod systemami MS Windows NT i 2000. Nie bez znaczenia jest okoliczność, że MS Windows NT i MS Windows 2000 są dobrze znane programistom i inżynierom. Wygoda polega na posługiwaniu się tym samym środowiskiem programowym do budowy algorytmów sterujących, do ich uruchamiania w czasie rzeczywistym, a także do raportowania uzyskanych wyników. Systemy MS Windows NT i MS Windows 2000, w celu przystosowania ich do prowadzenia pomiaru i sterowania w czasie rzeczywistym, można rozszerzyć na kilka sposobów (Timmerman, Van Beneden, Uhres 1998a, b).

Pierwszy polega na wywłaszczaniu niemaskowalnych przerwań.

Drugi – na zaimplementowaniu biblioteki Win32 API w komercyjnym systemie operacyjnym czasu rzeczywistego. Niedogodnością tego rozwiązania jest to, że standardowe aplikacje MS Windows NT i MS Windows 2000 nie mogą być wówczas wykonywane.

Trzeci umożliwia współdziałanie MS Windows NT lub MS Windows 2000 i systemu operacyjnego czasu rzeczywistego na tym samym procesorze. Dokonuje się tego przez modyfikację tak zwanego HAL (ang. *hardware abstract layer*), wywłaszczając przerwania i wprowadzając niewielkie jądro czasu rzeczywistego lub system operacyjny czasu rzeczywistego, lub wykonuje się MS Windows jako jedno z zadań prowadzących nadzór nad systemem operacyjnym czasu rzeczywistego. Modyfikacja HAL polega na skierowaniu przerwań sprzętowych i takim uszeregowaniu plików wynikowych, by ominąć jądro systemu MS Windows. Aplikacja użytkownika i system operacyjny dzielą czas procesora między siebie. Jądro MS Windows zawłaszcza przerwania, to znaczy procedura jądra, która dezaktywuje wskaźnik IE (ang. *interrupt enable*) nawet na tak długi czas jak 2 ms, dopóty nie pozwala na obsługę przez procesor główny przychodzących przerwań, dopóki sama nie aktywuje ponownie wskaźnika IE. W wielu aplikacjach w niniejszej pracy używano systemów operacyjnych MS Windows NT lub MS Windows 2000 do pracy w czasie rzeczywistym. Posłużono się oryginalnym oprogramowaniem o nazwie Real-Time Connection (RT-CON) (Grega, Kołek, Turnau 1997), które rozszerza możliwości przybornika Real-Time Workshop (RTW) programu MATLAB. Za pomocą RTW generuje się kod źródłowy w języku C, na podstawie modelu utworzonego w programie Simulink. Procedury sprzęgające, wchodzące w skład oprogramowania RT-CON, umożliwiają kompilację uzyskanego kodu oraz jego wykonanie w warunkach czasu rzeczywistego w systemach MS Windows NT lub MS Windows 2000. Wymagana jest przy tym instalacja oprogramowania w formie sterownika urządzenia pracującego w trybie jądra systemu operacyjnego, tak aby zapewnić dostęp do

przestrzeni adresowej urządzeń wejścia/wyjścia mikroprocesora. Przepływ danych między programem napisanym w MATLAB-ie a zadaniem czasu rzeczywistego odbywa się albo przez wywołanie specjalizowanych funkcji do wymiany informacji z poziomu programu MATLAB, albo przez wykonanie wygenerowanego automatycznie zadania w trybie czasu rzeczywistego.

Czwarty sposób ma charakter sprzętowy. Używa się handlera maskowalnych lub niemaskowalnych przerwań. Handler jest instalowany w generatorze przerwań znajdującym się na karcie wejść/wyjść. Jest to sposób na przechwycenie wszystkich przerwań maskowalnych i wygenerowanie w ich miejsce jednego przerwania niemaskowalnego. Procedura obsługi przerwania niemaskowalnego wykonuje się natychmiast, niezależnie od priorytetu aktualnie wykonywanego zadania. Test pokazuje, że przerwanie pojawiające się co 40 μs jest przesunięte w czasie najwyżej o kilka mikrosekund.

Piąty polega na zastosowaniu środowiska wieloprocesorowego. Tak działa na przykład środowisko dSPACE'a (Hanselmann 1992, 1993), które stosuje odrębną kartę z procesorem sygnałowym firmy Texas Instruments.

Programy dSPACE, MS Windows NT i 2000 z rozszerzeniem RT-CON, używane są w pracy jako systemy operacyjne czasu rzeczywistego w eksperymentach sterowania układami rzeczywistymi. Systemy są zintegrowane z programami MATLAB, Simulink i RTW.

Przeważnie do sterowania układami rzeczywistymi ustala się w komputerze stały okres próbkowania. W czasie jednego okresu muszą się zmieścić następujące akcje: pomiar, wyliczenia algorytmiczne nowych wartości sterowań i przesłania sterowań do członów wykonawczych. Dla poprawnego działania akcje te muszą się zacząć i skończyć w tym samym okresie próbkowania. Nie jest to zadanie trywialne, szczególnie dla komputerów działających w środowisku MS Windows NT lub MS Windows 2000, nie przeznaczonych zasadniczo przez konstruktorów do celów sterowania w czasie rzeczywistym.

2. Szybkie prototypowanie algorytmów sterowania

Rozwój techniki komputerowej umożliwia odmienne niż dawniej podejście do projektowania układów regulacji. Proces implementacji algorytmu sterującego obiektem rzeczywistym można zautomatyzować (Lambrechts 1993; Smith, Elbs 1999). Przy budowie algorytmu sterowania stosuje się cztery metody wymienione w tabeli 2.1.

Tabela 2.1
Metody budowy algorytmu sterowania

Lp.	Metoda	Regulator	Obiekt
1	Regulator rzeczywisty strojony bezpośrednio na obiekcie	rzeczywisty	rzeczywisty
2	Wirtualny obiekt sterowania i wirtualny regulator	symulowany	symulowany
3	Model obiektu sprzężony zwrotnie z regulatorem rzeczywistym – *hardware-in-the-loop*	rzeczywisty	symulowany
4	Regulator symulowany sprzężony zwrotnie z obiektem sterowania – *szybkie prototypowanie*	symulowany	rzeczywisty

Metoda pierwsza jest pozornie najprostsza, gdyż korzysta bezpośrednio z rzeczywistego obiektu i regulatora, niestety praktycznie trudna w realizacji. Dlatego przy projektowaniu symuluje się obie dynamiki razem (metoda druga), albo tylko dynamikę obiektu (metoda trzecia), albo tylko dynamikę regulatora (metoda czwarta).

W pracy wykorzystuje się wielokrotnie metodę czwartą, skuteczną w projektowaniu algorytmów. Metoda trzecia służy na ogół do weryfikacji działania układu regulacji i jest z powodzeniem stosowana na przykład w przemyśle samochodowym (Hanselmann 1992) – testowanie nowych rozwiązań ABS itp.

Szybkie prototypowanie to inżynierska droga na skróty prowadząca do zbudowania regulatora. Intuicyjne reguły algorytmu sterowania pochodzą od eksperta. Dzięki narzędziom programowym i sprzętowym istnieje możność natychmiastowej weryfikacji hipotez bezpośrednio na rzeczywistym obiekcie regulacji. Szybkie prototypowanie weszło do automatyki jako naturalna konsekwencja rozwoju środowiska programowo-sprzętowego czasu rzeczywistego (zob. rozdział 1) i narzędzi do budowy algorytmów inteligentnych (zob. rozdziały 5 i 6). Ważną rolę w tym rozwoju odegrało środowisko dSPACE'a. Posłużenie się nim

w przykładzie szybkiego prototypowania nie jest więc przypadkowe. Szerokie możliwości i wygoda projektowania w środowisku dSPACE'a były stymulatorem do powstania RTWT i podobnych narzędzi, jak RT-CON (użyty w eksperymentach opisanych w podrozdziałach 6.2, 9.1 i w rozdziale 10), czy też WIN-CON. W prototypowaniu nie posługujemy się modelem obiektu sterowanego, gdyż dysponujemy rzeczywistym obiektem i modelujemy tylko regulator (Grega, Pauluk, Turnau 2001). Budowanie algorytmu sterowania odbywa się podczas eksperymentów.

Trzeba tu wymienić co najmniej cztery przyczyny, które dzięki rozwojowi technik komputerowych, umożliwiły takie podejście do procesu prototypowania sterowników:

1) integracja środowisk programowych – automatyczne przejście od języka symulacji Simulink, przez tworzenie kodu C, do kodu wynikowego procesora sterującego;
2) strojenie parametrów regulatora z poziomu Simulinka, bez konieczności rekompilacji i łączenia;
3) szybki (kilka minut) proces kompilacji i łączenia, przy modyfikacji struktury sterownika;
4) zabezpieczenie warunków pracy w czasie rzeczywistym, przy częstości przetwarzania w granicach od 100 Hz do 10 kHz.

Program Simulink wykorzystywany jest tutaj do symulacji układu regulacji i budowy regulatora rzeczywistego. Biblioteka procedur Simulinka umożliwia symulację układów dynamicznych ciągłych (Hajduk, Pauluk, Turnau 1995) i dyskretnych (Adamski, Turnau 1998), przy czym dopuszcza się różne czasy próbkowania bloków dyskretnych przy założeniu, że są wielokrotnością podstawowego okresu próbkowania.

2.1. Zadanie przykładowe: sterowanie siłą manipulatora

Przykładem szybkiego prototypowania sterownika jest zadanie sterowania siłą docisku końcówki wykonawczej manipulatora, przy kontakcie z przeszkodą podczas ruchu ramion robota. Manipulator wykonuje ruch według ściśle określonego programu. Algorytm sterujący ruchem powinien być modyfikowany przy zetknięciu z przeszkodą, tak by nie nastąpiło uszkodzenie manipulatora lub napotkanej przeszkody. Na przykład manipulator malujący, powlekający powierzchnie lub klejący, powinien zachowywać stałą siłę kontaktu z malowaną powierzchnią (Aoustin *et al.* 1994).

Poniżej akcent położono na sposób syntezy regulatorów, sam algorytm sterowania jest tutaj tematem drugoplanowym (algorytmom regułowym poświęcono rozdział 5).

Istnieją różne podejścia do sterowania siłą manipulatora czy robota (np. Siciliano, Villani 1996). Tu zbudowano niekonwencjonalny, oryginalny sterownik (Turnau *et al.* 1995), który adaptuje się do zmieniającej się podczas ruchu manipulatora siły na styku pomiędzy efektorem i przeszkodą. Ramiona manipulatora sterowane są przy użyciu serwomechanizmów. Siła powstająca na styku efektora z przeszkodą nie jest mierzona (w wielu zastosowaniach taki pomiar jest trudny lub wręcz niemożliwy). Siła docisku końcówki jest odtwarzana z pomiaru wartości prądu serwomechanizmu. Zaproponowano inteligentny sterownik z wykorzystaniem heurystycznych reguł strojonych bezpośrednio na rzeczywistym

robocie-manipulatorze. Prace prowadzono w Danish Technical University w Lyngby pod Kopenhagą, dzięki uprzejmości profesora Ole Ravna z Servolaboratoriet.

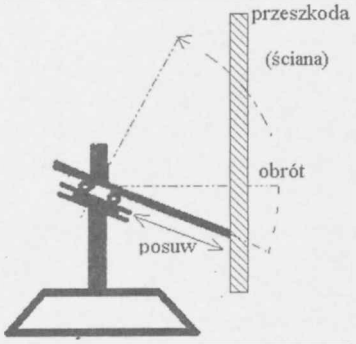
Manipulator wykonuje równocześnie dwa ruchy: obrotowy z dołu ku górze lub na odwrót, w płaszczyźnie pionowej, i posuwisty, polegający na wydłużaniu lub skracaniu ramienia robota. Pary czujników położenia i prędkości dla ruchu obrotowego i posuwistego oraz czujnik prądu serwomechanizmu ruchu posuwistego są źródłem pięciu sygnałów wejściowych sterownika.

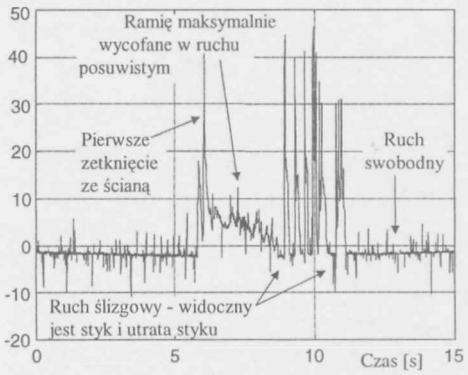
Model sterownika zbudowano z bloków programu Simulink, z użyciem bloków sterowników karty dSPACE'a DS-1102 zmiennoprzecinkowego mikrokontrolera z wbudowanym procesorem Texas Instruments TMS320E32. Dzięki programowi Real-Time Workshop (RTW) automatycznie wygenerowany zostaje kod C i przetłumaczony przez oprogramowanie dSPACE na binarny, wykonywalny kod procesora TMS320E32. Parametry sterownika są dostępne z animacyjnego programu COCKPIT, zaś wizualizacja sygnałów odbywa się przy użyciu programu TRACE. Oba narzędzia programowe stanowią wyposażenie programu dSPACE. Sterownik jest zaimplementowany bezpośrednio w serwomechanizmach sterujących ruchem robota. Strojenie parametrów sterownika odbywa się podczas eksperymentów z rzeczywistym obiektem bez pośrednictwa modelu obiektu.

Zadanie sterowania można sformułować następująco. Koniec ramienia dwusegmentowego robota śledzi dowolny zaprogramowany ruch na płaszczyźnie pionowej. Pierwszy segment porusza się ruchem obrotowym wokół poziomej osi. Drugi segment jest złożony równolegle z segmentem pierwszym, a więzy tego zespolenia ograniczają przemieszczanie się segmentu drugiego do ruchu posuwistego w przód lub w tył, względem pierwszego segmentu. Jeżeli na drodze ruchu końcówki ramienia robota wystąpi przeszkoda, to serwomotor ruchu posuwistego steruje tak, by końcówka wykonywała ruch ślizgowy po powierzchni przeszkody, z niewielką siłą docisku pomiędzy końcówką i przeszkodą. Przez cały czas robot śledzi wartości zadane położeń końcówki. Przy napotkaniu przeszkody wykonuje ruch, tak aby siła docisku końca ramienia robota do przeszkody przyjmowała wartość mniejszą od z góry zadanej wartości progowej. Po ustąpieniu przeszkody robot powraca do zadanego ruchu.

Opiszemy algorytm sterujący i proces jego powstawania, odnosząc się do następującego eksperymentu. Robot wykonuje ruch po okręgu (rys. 2.1) i napotyka pionową ścianę. Na rysunku 2.2 pokazano siłę na styku końcówki ze ścianą, zmierzoną czujnikiem siły. Ten dodatkowy pomiar służył do upewnienia się o słuszności hipotezy pomiaru prądu serwomotoru celem obserwacji siły kontaktu z przeszkodą (bez pomiaru bezpośredniego).

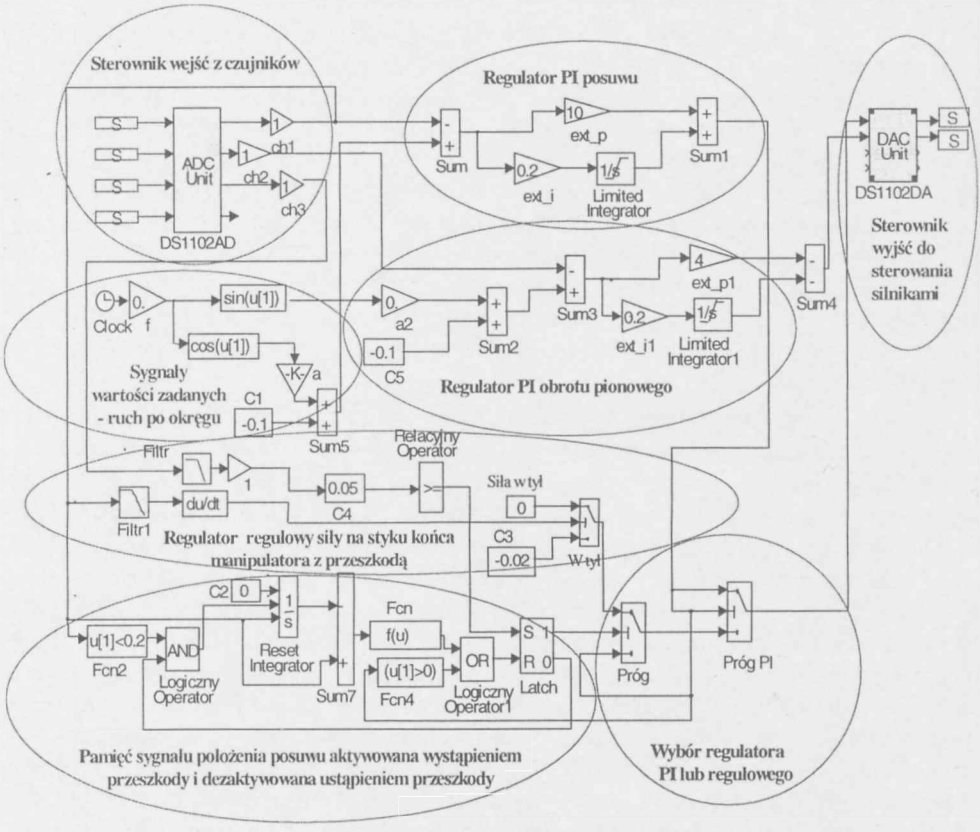
Model regulatora zaprojektowanego w Simulinku przedstawiono na rysunku 2.3. Kontakt z przeszkodą jest wykrywany przez nagły wzrost prądu (charakterystyczne szpilki prądu na rysunku 2.4), sterującego serwomechanizmem posuwu (rys. 2.5). Ruch w płaszczyźnie pionowej odbywa się po okręgu. Dwa regulatory PI służą do śledzenia sygnałów wartości zadanej obrotu i posuwu ramienia robota. Kąt ramienia i jego wysuw są mierzone i porównywane z wartościami zadanymi. Sygnały błędu porównania są podawane jako wejściowe dla regulatorów PI. W przypadku gdy koniec ramienia robota natrafi na przeszkodę, następuje nagły wzrost sygnału błędu. Akcja regulatora PI musi zostać przerwana, gdyż wzrastająca siła kontaktu końca manipulatora z przeszkodą może doprowadzić do uszkodzenia robota lub przeszkody. Po przekroczeniu założonego progu błędu wyłącza się regulator PI i przekazuje działanie regulatorowi regułowemu.



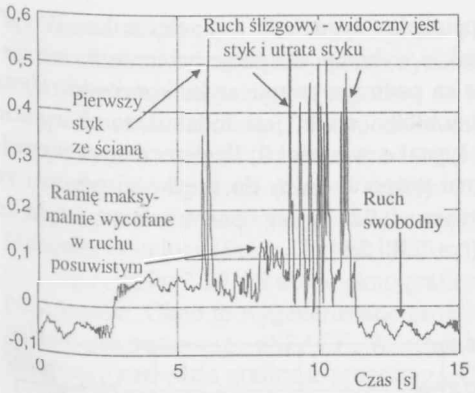


Rys. 2.1. Pole eksperymentu manipulatora

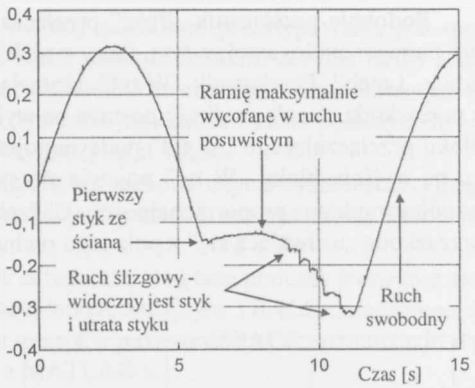
Rys. 2.2. Siła styku między końcówką i ścianą [N]



Rys. 2.3. Regułowy regulator manipulatora wykorzystujący Simulinka, przybornik RTW i bloki RTI sterowników urządzeń w oprogramowaniu dSPACE

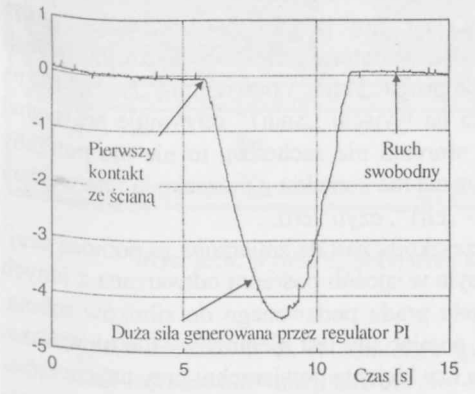


Rys. 2.4. Prąd serwomotoru posuwu [A] (po filtracji) – sygnał „ch3"

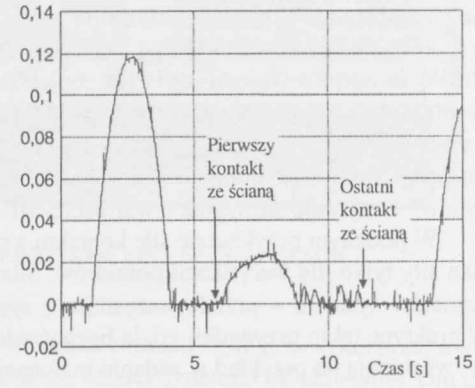


Rys. 2.5. Położenie końcówki robota w posuwie [m] – sygnał „ch1"

W układzie występują trzy przełączniki sterowane progowo. Są to odpowiednio: „Próg PI", „Próg" oraz „W tył". Przełącznik „Próg PI" wyłącza akcję regulatora PI po przekroczeniu wartości progu PI. Regulator PI generuje sygnał proporcjonalny do wartości uchybu (rys. 2.6). Regulator PI jest aktywny, jeśli wartość z wyjścia regulatora posuwu „Sum1" jest większa od wartości progu przełącznika „Próg" i mniejsza od wartości progu przełącznika „Próg PI" lub jeśli bezwzględna wartość sygnału prądu „ch3" odfiltrowanego filtrem dolnoprzepustowym jest mniejsza od wartości „C4" równej 0,05 A. Niespełnienie powyższych warunków oznacza, że powinien zadziałać regulator regułowy. Decyduje o tym stan przerzutnika „Latch". Do włączenia regulatora regułowego siły (S = 1 w przerzutniku „Latch") wystarcza, by sygnał z wyjścia regulatora posuwu „Sum1" (rys. 2.6) był niedodatni lub wartość sygnału na wyjściu pamięci analogowej „Sum7" (rys. 2.7) nie przekroczyła wartości progu ustawionego w „Fcn".

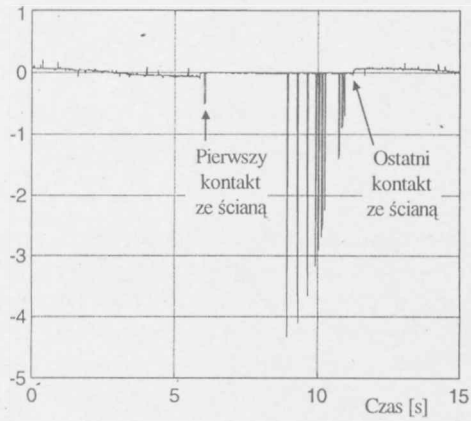


Rys. 2.6. Sygnał z regulatora PI „Sum1" [V]

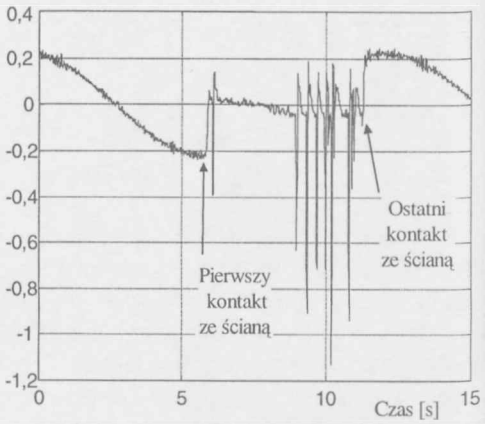


Rys. 2.7. Sygnał z bloku pamięci „Sum7" [m]

Podobnie przełącznik „Próg" przełącza pomiędzy sygnałami z przełącznika „W tył" lub pamięci analogowej; o tym, który z nich będzie wybrany, decyduje przerzutnik zatrzaskowy „Latch". Przełącznik „W tył" odpowiada za podtrzymywanie styku końcówki robota z przeszkodą. Jeżeli prędkość posuwu na wyjściu bloku „du/dt" jest dodatnia, to na wyjście bloku przełączającego „W tył" podawany jest sygnał o wartości 0. W przeciwnym wypadku na wyjściu bloku „W tył" pojawia się sygnał proporcjonalny do prędkości posuwu ze współczynnikiem proporcjonalności „C3" równym –0,02. W ten sposób robot podąża za przeszkodą „usuwającą się" z pola jego ruchu (rys. 2.8 i 2.9).



Rys. 2.8. Sygnał sterowania serwomotorem z wyjścia „Próg PI" [V]



Rys. 2.9. Sygnał na wyjściu „du/dt" – prędkość posuwu [m/s]

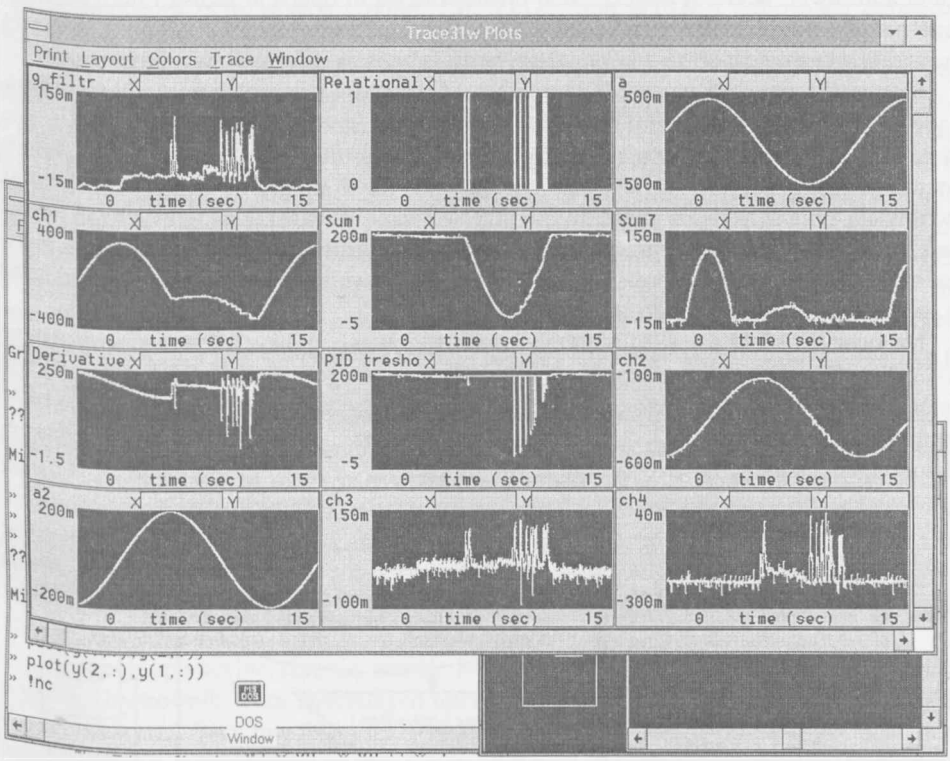
Realizuje się to dzięki blokom „Reset Integrator", „Latch" i dwóm „Operatorom logicznym" w pętlach sprzężenia zwrotnego. „Reset Integrator" wraz z „Sum7" pracują jako pamięć analogowa sygnału położenia „ch1". Oznaczamy przez „ch1(–)" wartość „ch1" w momencie czasu, dla którego logiczna wartość operatora „AND" była równa 1. Jeśli chwilowa wartość sygnału „ch1" jest mniejsza od progu „Fcn2" i przerzutnik „Latch" ma na swym wejściu R (reset) wartość logiczną 1, to na wyjściu „Sum7" otrzymuje się sygnał o wartości „ch1" – „ch1(–)". Jeżeli powyższe warunki nie zachodzą, to nie ma potrzeby zapamiętywania sygnału położenia posuwu, gdyż nie ma kontaktu z przeszkodą i na wyjściu „Sum7" otrzymuje się sygnał o wartości „ch1" – „ch1", czyli zero.

W podanym przykładzie siła kontaktu z przeszkodą została zmierzona za pomocą czujnika siły tylko dla weryfikacji pomiarów. Siła była w sposób pośredni odtwarzana z innych sygnałów systemu – przede wszystkim z sygnału prądu podawanego do silników robota. W praktyce takie przypadki, gdzie bezpośredni pomiar siły jest niemożliwy lub niewskazany, występują na przykład w zadaniu malowania czy klejenia powierzchni przy użyciu robota. W takich przypadkach budowa sterownika z wykorzystaniem rzeczywistego robota i rzeczywistych przeszkód, wydaje się uzasadniona i najszybciej prowadzi do skonstruowania

algorytmu sterowania. Charakterystyczne dla metody szybkiego prototypowania jest ominięcie żmudnej ścieżki identyfikacji dynamiki ruchu manipulatora. Model taki byłby składnąd wątpliwej jakości, gdyż używany do eksperymentów manipulator posiada trudne do zamodelowania elementy. Do zamodelowania dynamiki jego ruchu można posłużyć się pomocniczymi metodami (Szymkat *et al.* 1995; Ravn, Szymkat 1995).

Jedną z propozycji detekcji ruchów manipulatora stanowi użycie systemu wizyjnego stereo i markerów naklejanych w charakterystycznych punktach manipulatora, takich jak przeguby i ramiona (Fuksa, Turnau 1996).

Na rysunku 2.10 pokazano okno graficzne ekranu, zapamiętane podczas jednego z eksperymentów. Okno to wygenerowano programem wizualizacyjnym TRACE dołączanym do pakietu programowego dSPACE. W ostatnich wersjach pakietu dSPACE wizualizacja wykorzystuje narzędzia graficzne programu GUI z MATLAB-a.



Rys. 2.10. Wyniki eksperymentu wizualizowane przez program TRACE

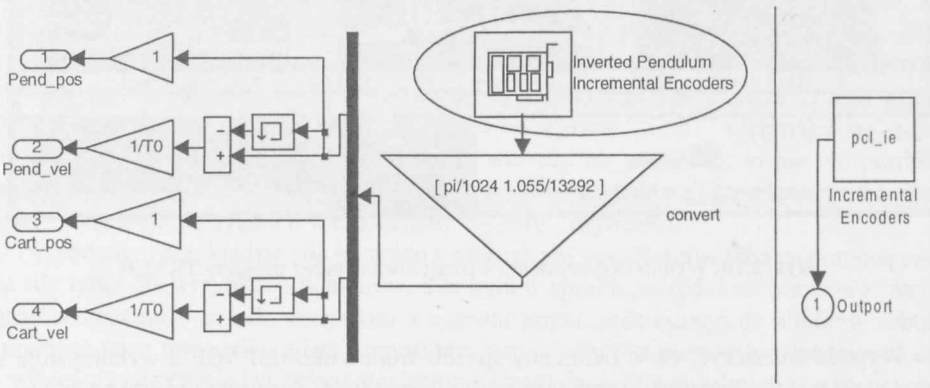
Wypada zaznaczyć, że w odmienny sposób środowisko dSPACE'a wykorzystuje się w metodzie trzeciej, nazwanej *hardware-in-the-loop*. Model dynamiki obiektu służy tam do symulacji, a regulator rzeczywisty zamyka pętlę sprzężenia zwrotnego (Krafka, Kunze 1994).

2.2. Część komunikacyjna algorytmu sterowania – sterowniki urządzeń

W projekcie regulatora rzeczywistego, prócz części algorytmicznej, musi zostać uwzględniona część komunikacyjna z urządzeniami do pomiaru i sterowania obiektem. Mowa tu o sterownikach urządzeń (ang. *device drivers*), które odpowiadają za wymianę informacji między procesorem a urządzeniami wejścia/wyjścia. Urządzenia wejścia/wyjścia, którymi są na ogół przetworniki analogowo-cyfrowe, przetworniki cyfrowo-analogowe oraz wejścia i wyjścia cyfrowe, zajmują określoną liczbę położeń w przestrzeni adresowej pamięci lub w przestrzeni adresowej definiowanej specjalnie dla tych urządzeń. Zmiana stanu wyjścia lub odczyt stanu wejścia wymagają wykonania operacji zapisu i odczytu. Operacje te są ściśle określone przez specyfikę urządzeń zewnętrznych. Zależnie od rodzaju przestrzeni adresowej, w której umieszczono urządzenie, są to operacje zapisu i odczytu dotyczące pamięci lub przestrzeni adresowej urządzeń wejścia/wyjścia. Dostępność rodzajów przestrzeni adresowych oraz wybór operacji wejścia/wyjścia zależą od typu procesora.

Sterowniki urządzeń wykonują te operacje, udostępniając oprogramowaniu wykorzystującemu sterowniki odpowiednie funkcje. Przykładowo, sterowniki karty graficznej w systemie MS Windows udostępniają funkcje zmiany informacji wyświetlanej na ekranie komputera. Sterowniki wymagane przez przybornik RTW programu MATLAB/Simulink umożliwiają zmianę napięcia wyjściowego przetworników cyfrowo-analogowych lub odczyt wartości napięcia na wejściu przetwornika analogowo-cyfrowego. Wszelkie operacje związane ze współpracą z przetwornikami zawarte są w sterowniku urządzenia i są niewidoczne dla aplikacji odwołującej się do sterownika.

Na rysunku 10.22 (s. 138) pokazano okno sterowania aplikacji czasu rzeczywistego RT-CON w Simulinku. Jest to aplikacja regulatora opisanego w podrozdziale 10.2. W tym miejscu zwrócimy jedynie uwagę na jej część komunikacyjną (rys. 2.11), mianowicie bloki Simulinka „PENDULUM PCL-812 Sensors" i „Actuators", będące interfejsem między regulatorem a układami pomiarowo-sterującymi obiektu.



Rys. 2.11. Wnętrze bloku sterownika urządzeń „PENDULUM PCL-812 Sensors"; po prawej – wnętrze bloku „Inverted Pendulum Incremental Encoders"

Podstawowym elementem tych bloków jest *S*-funkcja utworzona w języku C, zawierająca instrukcje bezpośredniej komunikacji z kartą wejściowo/wyjściową komputera. By choć w skrócie opisać działanie warstwy komunikacyjnej, trzeba by zniżyć się do poziomu programu w asemblerze lub w języku C (program „pcl_ie" z rysunku 2.11) i skorzystać z instrukcji, które bezpośrednio komunikują się z określonym adresem w przestrzeni adresowej urządzeń wejścia/wyjścia lub z określoną komórką pamięci. *S*-funkcja spełnia dwa zadania: komunikuje się ze sprzętem i udostępnia lub przyjmuje dane w formacie zgodnym z wymogami Simulinka. Sygnałami wejściowymi i wyjściowymi *S*-funkcji są na przykład stany chwilowe liczników enkoderów, oraz wejścia do przetwornika cyfrowo-analogowego. W obu tych przypadkach sygnał jest liczbą całkowitą. W sterowniku urządzeń (zbudowanym z bloków Simulinka) ma miejsce konwersja liczb całkowitych na wielkości fizyczne – sterownik urządzenia o nazwie „PENDULUM PCL-812 Sensors" przetwarza zliczone impulsy na kąt. Liczba zliczonych impulsów na wyjściu z bloku „Incremental Encoders" zostaje przemnożona przez współczynniki: 1,055/13292 lub $\pi/1024$ i odpowiada pomiarowi drogi mierzonej w metrach i kąta mierzonego w radianach.

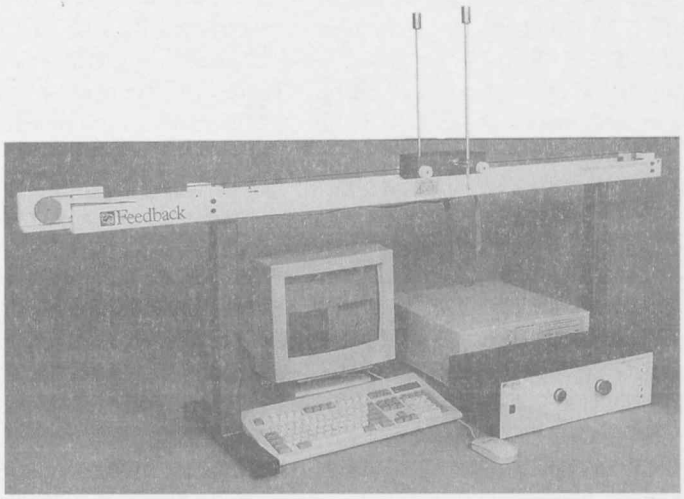
Na rysunku 10.22 zaznaczono elipsą blok „PENDULUM PCL-812 Sensors", którego wnętrze pokazano na rysunku 2.11 z lewej strony. Po sięgnięciu głębiej do wnętrza bloku o nazwie „Inverted Pendulum Incremental Encoders" (zakreślonego też elipsą), otrzymuje się blok „pcl_ie", pokazany na rysunku 2.11 po prawej stronie. Zawiera on program wykonawczy, napisany źródłowo w języku C, do obsługi czytania impulsów z enkodera pomiaru kąta obrotu i przetwarzania ich na postać bitową rejestru karty PCL-812 komputera. W modelu z rysunku 10.22 widoczne są cztery bloki przycisków „RESET ENCODERS PCL-812", „BaseAddr=528", „START" i „STOP". Służą one do wyzerowania stanu rejestru enkoderów, ustawienia adresu bazowego karty wejściowo/wyjściowej komputera, ładowania i wystartowania skompilowanego modelu lub do zatrzymania i wyrzucenia modelu z pamięci komputera.

Typowy sterownik urządzenia jest programem, złożonym na ogół z trzech części: inicjalizacji urządzenia, komunikacji z urządzeniem i zakończenia pracy urządzenia. Część inicjująca odpowiada za ustawienie trybu pracy urządzenia (na przykład może chodzić tu o ustawienie zakresów napięć pomiarowych lub sterujących). Część komunikacyjna realizuje zapis lub odczyt informacji z urządzenia (ustawia napięcia wyjściowe i odczytuje napięcia wejściowe). Zakończenie pracy z urządzeniem to przede wszystkim pozostawienie wyjść urządzenia w stanach bezpiecznych dla sterowanego obiektu. Szczegóły dotyczące implementacji sterownika urządzenia określa aplikacja używająca sterownika (RTW).

Systemy operacyjne chronią zasoby komputera, między innymi urządzenia wejścia/wyjścia. Użytkownik – bez specjalnych uprawnień – nie ma do nich dostępu. Na przykład w systemach MS Windows NT oraz MS Windows 2000 dostęp do przestrzeni urządzeń wejścia/wyjścia realizowany jest przez jądro systemu za pomocą specjalnych sterowników (ang. *kernel mode device driver*). Kiedy użytkownik zażąda wykonania instrukcji wejścia/wyjścia, w systemie próba jej wykonania może spowodować wygenerowanie tak zwanego wyjątku, związanego z brakiem odpowiednich uprawnień. Przeważnie skutkuje to zakończeniem wykonania programu.

3. Porównawcze zadanie sterowania: wahadło na wózku

System wahadła na wózku jest nieliniowym układem czwartego rzędu, z jednym sterowaniem i nieskończoną liczbą punktów równowagi stabilnej i chwiejnej. Wykorzystuje się go szeroko jako standardowy przykład porównawczy do badania algorytmów sterowania, porównania technik uczenia się, testowania środowisk programowo-sprzętowych czasu rzeczywistego itp.: Kwakernaak, Sivan (1972); Boone (1997); Grega, Turnau (1999); Bloch, Leonard, Marsden (2000); Napoleon, Hoshino, Furuta (2000). W niniejszej pracy zadania sterowania są stawiane i rozwiązywane dla dwóch laboratoryjnych systemów wahadła na wózku. Na rysunku 3.1 pokazano laboratoryjny model II.



Rys. 3.1. System laboratoryjny wahadła na wózku, sterowany z komputera – model II

Wahadło zamocowane jest obrotowo na wózku, który jeździ po poziomej szynie w płaszczyźnie obrotu wahadła. Układ jest poruszany silnikiem elektrycznym prądu stałego, który ciągnie wózek poprzez giętką, zębatą linkę opasującą dwa kółka zębate, umieszczone na końcach szyny.

Pewne zjawiska w systemie są trudne do zamodelowania, na przykład tarcie wózka i naprężenia w lince napędowej. Tarcie można w części skompensować. Efekty związane z tarciem linki i asymetrią tarcia w zależności od kierunku ruchu wózka względem silnika, są funkcją wstępnego naciągu linki i mogą być zamodelowane jako niesymetryczne tarcie statyczne wózka (zobacz algorytm regułowy). Dynamikę silnika można pominąć, utrzymując proporcjonalność prądu płynącego przez silnik do napięcia zasilającego na drodze kompensacji sygnałem prędkości obrotowej silnika (zobacz algorytm optymalnoczasowy).

Metody sterowania inteligentnego przedstawione w II części pracy dotyczą tego samego przykładu porównawczego. Modele laboratoryjne wahadła na wózku są wykorzystywane w rozdziale 4 do przedstawienia algorytmów uczących się, a w rozdziale 5 – do przedstawienia algorytmów regułowych. Na tych modelach w rozdziale 6 porównuje się algorytmy używające logiki rozmytej i sieci neuralnych. W częściach III i IV przy użyciu modeli wahadła na wózku pokazuje się w działaniu algorytmy optymalnoczasowe i metody uodparniania. Dlatego w części I, która stanowi wprowadzenie do sterowania w czasie rzeczywistym, zamieszczono opis systemu porównawczego, przywoływany dalej wielokrotnie.

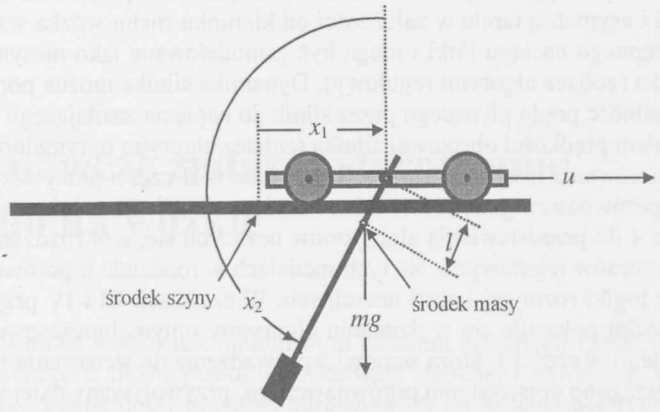
3.1. Parametry modeli i równania ruchu

Parametry systemów laboratoryjnych zestawiono w tabeli 3.1.

Tabela 3.1
Parametry modeli laboratoryjnych

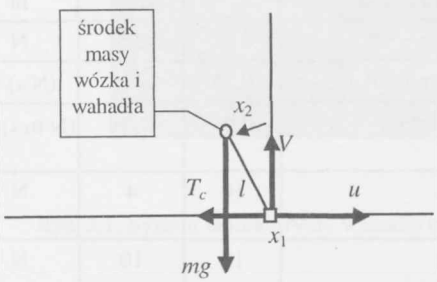
Parametr	Znaczenie	Model I	Model II	Jednostki
m	masa wózka z wahadłem	2,13	1,22	kg
m_c	masa wózka	2,04	1,1	kg
m_p	masa wahadła	0,09	0,12	kg
l	odległość pomiędzy środkiem masy układu a osią obrotu wahadła	0,01479	0,03	m
L	odległość środka masy wahadła od osi obrotu	0,35	0,32	m
F_s	tarcie statyczne wózka	1	0,5	N
f_c	współczynnik tarcia w ruchu liniowym wózka	0,25	0,25	(N·s)/m
f_p	współczynnik tarcia w ruchu obrotowym wahadła	0,01	0,01	(N·m·s)/rad
$u_{\max}^{POST}$	maksymalna siła sterująca w zadaniu postawienia wahadła	10	4	N
$u_{\max}^{STAB}$	maksymalna siła sterująca w zadaniu stabilizacji wahadła	10	10	N
J	moment bezwładności układu względem środka masy	0,01423	0,01197	kg·m^2
J_p	moment bezwładności wahadła względem osi obrotu	0,01468	0,0123	kg·m^2

Schemat wózka z wahadłem pokazano na rysunku 3.2.

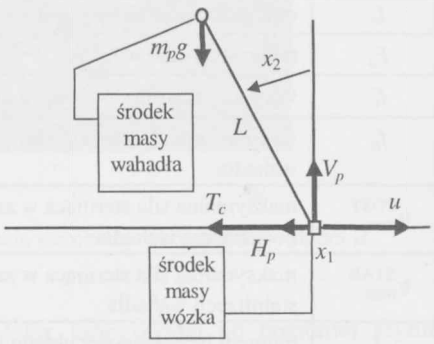


Rys. 3.2. Schemat systemu

Stan systemu jest wektorem czterowymiarowym $x = \mathrm{col}(x_1, x_2, x_3, x_4)$, gdzie x_1 jest położeniem wózka mierzonym (w metrach) od środka szyny; x_2 jest kątem (w radianach) między kierunkiem „pionowo do góry" a wahadłem. Dolnemu położeniu ramienia wahadła odpowiada $x_2 = \pm\pi$ rad; x_3 jest prędkością wózka (mierzoną w metrach na sekundę); x_4 jest prędkością kątową wahadła (w radianach na sekundę); u jest siłą sterującą przyłożoną do wózka (w niutonach), przyjmującą wartości z przedziału $[-u_{\max}, u_{\max}]$. Można przyjąć z niewielkim błędem, że siła u jest proporcjonalna do momentu obrotowego silnika elektrycznego, a moment jest proporcjonalny do prądu płynącego przez uzwojenie wirnika. Zatem silnik przetwarza proporcjonalnie prąd na siłę; m jest sumaryczną masą wahadła i wózka, l jest odległością pomiędzy środkiem masy układu a osią obrotu wahadła, która jest równocześnie środkiem masy wózka. Na rysunkach 3.3 i 3.4 pokazano rozkład sił układu.



Rys. 3.3. Ruch środka masy systemu



Rys. 3.4. Ruch środków mas wahadła i wózka

Pierwszy rysunek przedstawia siły działające na środek masy systemu, drugi – siły działające na środek masy wahadła i środek masy wózka, traktowane oddzielnie.

Zmienne siły i momenty zaznaczone na rysunkach 3.3 i 3.4 to:

V – siła reakcji szyny na wózek, działająca prostopadle do szyny,

V_p – składowa siły reakcji wózka na oś wahadła, działająca prostopadle do szyny,

H_p – składowa siły reakcji wózka na oś wahadła, działająca równolegle do szyny,

T_c – siła tarcia wózka,

D_p – moment tarcia ruchu obrotowego.

Ruch środka masy układu opisują równania poniżej po lewej stronie. Odpowiednie równania dla ruchu środków mas wahadła i wózka podano poniżej po prawej stronie.

$$m\frac{\mathrm{d}^2}{\mathrm{d}t^2}(x_1 - l\sin x_2) = u - T_c \qquad m_c\frac{\mathrm{d}^2}{\mathrm{d}t^2}x_1 = u - T_c + H_p$$

$$m_p\frac{\mathrm{d}^2}{\mathrm{d}t^2}(x_1 - L\sin x_2) = -H_p \tag{3.1}$$

$$m\frac{\mathrm{d}^2}{\mathrm{d}t^2}l\cos x_2 = V - mg \qquad m_p\frac{\mathrm{d}^2}{\mathrm{d}t^2}L\cos x_2 = V_p - m_p g \tag{3.2}$$

$$J\frac{\mathrm{d}^2}{\mathrm{d}t^2}x_2 = (u - T_c)l\cos x_2 + \qquad J_p\frac{\mathrm{d}^2}{\mathrm{d}t^2}x_2 = -H_p L\cos x_2 +$$

$$+Vl\sin x_2 - D_p \qquad +V_p L\sin x_2 - D_p \tag{3.3}$$

Wprowadźmy oznaczenia

$$a = l^2 + \frac{J}{m} = \frac{J_p}{m}, \qquad \mu = ml \tag{3.4}$$

gdzie $J = J_p - \mu l$ jest momentem bezwładności układu względem środka masy, gdy wózek traktuje się jako masę punktową skupioną w osi obrotu. Z równań (3.1)–(3.3), opisujących dynamikę środka masy układu otrzymuje się:

$$\ddot{x}_1 = \frac{a\,(u - T_c - \mu\dot{x}_2^2\sin x_2) + l\cos x_2(\mu g\sin x_2 - D_p)}{J + \mu l\sin^2 x_2} \tag{3.5}$$

$$\ddot{x}_2 = \frac{l\cos x_2(u - T_c - \mu\dot{x}_2^2\sin x_2) + \mu g\sin x_2 - D_p}{J + \mu l\sin^2 x_2} \tag{3.6}$$

Niech $x_3 = \dot{x}_1$, $x_4 = \dot{x}_2$ oraz niech $x = \mathrm{col}(x_1, x_2, x_3, x_4)$ oznacza kolumnowy wektor stanu. Ruch systemu opisany jest równaniem stanu

$$\dot{x} = f(x,u) = f^0(x) + f^1(x)u \tag{3.7}$$

gdzie:

$$f^0(x) = \begin{bmatrix} x_3 \\ x_4 \\ \dfrac{v_2(x) l \cos x_2 - a v_1(x)}{d(x)} \\ \dfrac{v_2(x) - v_1(x) l \cos x_2}{d(x)} \end{bmatrix}, \qquad f^1(x) = \frac{1}{d(x)} \begin{bmatrix} 0 \\ 0 \\ a \\ l \cos x_2 \end{bmatrix},$$

$$v_1(x) = \mu\, x_4^2 \sin x_2 + T_c, \qquad v_2(x) = \mu\, g \sin x_2 - D_p,$$

$$d(x) = J_p - \mu\, l \cos^2 x_2 = J + \mu\, l \sin^2 x_2.$$

Mianownik $d(x)$ jest zawsze dodatni. Sterowania dopuszczalne są ograniczone

$$|u| \le u_{\max} \tag{3.8}$$

Podane w tabeli 3.1 współczynniki F_s, f_c i f_p służą do określenia tarcia. Tarcie w ruchu wózka wyrażono formułą $T_c = T_s + f_c x_3$. Tarcie statyczne

$$T_s = \begin{cases} u + H_p & \text{dla } |u + H_p| \le F_s \\ F_s & \text{poza tym} \end{cases} \tag{3.9}$$

jest kompensowane w algorytmie regułowym (zob. tab. 5.1, reguła 4). W pewnych wypadkach pomija się je w równaniach stanu (3.7), przyjmując

$$T_c = f_c x_3 \quad \text{oraz} \quad D_p = f_p x_4 \tag{3.10}$$

W algorytmach regułowych przy modelowaniu zjawiska tarcia posługiwano się również modelem bardziej rozbudowanym niż przedstawiony za pomocą formuł (3.9) i (3.10). Na przykład do testowania algorytmów regułowych wykorzystywano model dynamiki systemu z rysunku 5.1. Zależność funkcyjną tarcia wózka od prędkości zamodelowano jako nieliniową funkcję w bloku Simulinka o nazwie *Tarcie*. W algorytmach optymalnoczasowych, w II części pracy, tarcie modelowano korzystając z formuł (3.10). Tarcie statyczne kompensowano bez uwzględnienia go w modelu. W jednym z modeli (model simulinkowy) wprowadzono bloki stref nieczułości o nazwach *DZ-Vc* (strefa nieczułości na zmiany prędkości wózka), *DZ-U* (strefa nieczułości na zmiany sterowania), oraz *DZ-Vp* (strefa nieczu-

łości na zmiany prędkości wahadła). Badaniu i modelowaniu zjawiska tarcia poświęca się wiele uwagi w literaturze (Carli de, Cong, Matacchioni 1994; Canudas de Wit *et al.* 1995; Lischinsky, Canudas de Wit, Morel 1999).

Punkty równowagi układu (3.7) w przestrzeni stanu są postaci $\mathrm{col}(\xi, k\pi, 0, 0)$, gdzie ξ jest dowolną liczbą rzeczywistą, a k – całkowitą. Nieparzyste k odpowiadają dolnemu, stabilnemu położeniu równowagi, parzyste zaś – górnemu, niestabilnemu. W układzie wózka z wahadłem występują charakterystyczne symetrie. Załóżmy, że ξ jest dowolną liczbą rzeczywistą, k – całkowitą, $\varepsilon = +1$ lub $\varepsilon = -1$, u jest dowolnym sterowaniem, a x – odpowiadającym mu rozwiązaniem równania stanu (3.7). Wówczas każda funkcja postaci $\varepsilon x + \mathrm{col}(\xi, 2k\pi, 0, 0)$ stanowi rozwiązanie równania stanu, wygenerowane przez sterowanie εu. Inaczej mówiąc, zbiór trajektorii stanu jest niezmienniczy względem jednoczesnej zmiany znaku stanu i sterowania, a także względem dowolnego przesunięcia wózka i obrócenia wahadła o dowolną wielokrotność kąta pełnego.

W rzeczywistym modelu laboratoryjnym wózka z wahadłem mierzone są dwie z czterech zmiennych stanu, a mianowicie położenie kątowe wahadła i położenie wózka na szynie. Pomiar pierwszy otrzymuje się bezpośrednio z 11-bitowego enkodera kąta, pomiar drugi z identycznego enkodera, lecz w sposób pośredni, ponieważ ruch obrotowy silnika jest zamieniany na ruch posuwisty wózka za pomocą linki transmisyjnej.

Dwa enkodery kąta, zamontowane na osi obrotu wahadła i osi silnika, mają rozdzielczość 2048 działek na obrót. Z tego wynika, że kąt wahadła jest mierzony z rozdzielczością 0,003068 rad (kąt pełny 2π jest podzielony na 2048 części), a położenie wózka z rozdzielczością 0,076 mm (jeden obrót silnika odpowiada przesunięciu wózka o 0,156 m).

Parametry określające dokładność pomiaru i jego błąd w stosunku do rzeczywistych wielkości zebrano w tabeli 3.2.

Tabela 3.2
Parametry istotne dla dokładności pomiarów

Rozdzielczość enkodera	2048 działki/obrót
Krok kwantyzacji kąta wahadła	0,003068 rad ($2\pi/2048$)
Krok kwantyzacji położenia wózka	0,076 mm
Czas próbkowania komputera	0,01 s
Częstotliwość próbkowania sterowania i pomiarów	100 Hz

W modelu (3.7), (3.8) stosuje się skalowanie, wprowadzając zmienne bezwymiarowe. Uzyskuje się postać bardziej odpowiednią do obliczeń, dzięki normalizacji i zmniejszeniu liczby mnożeń potrzebnych do wyliczenia wartości prawych stron. Niech

$$t = \alpha_0 t', \quad x_i(t) = \alpha_i x_i'(t'), \quad i = 1,\ldots,4, \quad \Xi = \mathrm{diag}(\alpha_1,\ldots,\alpha_4)$$
$$u(t) = \gamma u'(t') \tag{3.11}$$

Aby zachować prostą postać argumentu funkcji trygonometrycznych i równań (3.7) i (3.8), przyjmuje się $\alpha_1 = \alpha_0\alpha_3$, $\alpha_2 = \alpha_0\alpha_4 = 1$. Założenie $l\alpha_3 = a\alpha_4$ pozwala zachować identyczność mianowników i równość współczynników liczbowych przy sterowaniu u' w przekształconych równaniach. Zatem

$$\alpha_0 = \frac{1}{\alpha_4}, \quad \alpha_1 = \frac{a}{l}, \quad \alpha_3 = \frac{a}{l}\alpha_4 \tag{3.12}$$

Oznaczając $x' = \mathrm{col}(x'_1,..., x'_4)$ i $f' = \mathrm{col}(f'_1,..., f'_4)$, otrzymuje się ogólną postać przeskalowanego modelu układu:

$$\dot{x}' = f'(x',u') \tag{3.13}$$

$$f'_1(x',u') = x'_3$$

$$f'_2(x',u') = x'_4$$

$$f'_3(x',u') = \frac{v'_1(x',u') + c_3 v'_2(x')\cos x'_2}{d'(x')} \tag{3.14}$$

$$f'_4(x',u') = \frac{v'_1(x',u')\cos x'_2 + c_4 v'_2(x')}{d'(x')}$$

$$|u'(t')| \le u'_{\max} \tag{3.15}$$

$$v'_1(x',u') = u' - b_1 x'^2_4 \sin x'_2 - b_2 x'_3$$

$$v'_2(x') = \sin x'_2 - b_3 x'_4$$

$$d'(x') = b_0 + b_1 \sin^2 x'_2 = b - b_1 \cos^2 x'_2$$

$$u'_{\max} = \frac{u_{\max}}{\gamma} = g_1 \tag{3.16}$$

$$b = \frac{J_p}{\gamma l}\alpha_4^2, \quad b_0 = \frac{J}{\gamma l}\alpha_4^2, \quad b_1 = \frac{\mu}{\gamma}\alpha_4^2, \quad b_2 = \frac{J_p f_c}{\gamma \mu}\alpha_4$$

$$b_3 = \frac{f_p}{g\mu}\alpha_4, \quad c_3 = \frac{g\mu^2}{\gamma J_p}, \quad c_4 = \frac{g\mu}{\gamma l}$$

Aby zminimalizować liczbę mnożeń przy wyliczaniu prawych stron w (3.7), wybiera się

$$\gamma = \frac{g\mu^2}{J_p}, \quad \alpha_4 = \sqrt{\frac{g\mu}{J_p}} \tag{3.17}$$

Liczba mnożeń przez stałe maleje w tym wypadku do trzech, ponieważ $b_1 = c_3 = 1$. Korzysta się też ze związku $b = c_4 = b_0 + 1$.

Wartości liczbowe współczynników skalowania i parametrów modelu, wykorzystywane w równaniach poniżej, podano w tabeli 3.3. Współczynniki skalowania α_i, $i = 0,...,4$, wyliczono na podstawie parametrów modelu II z tabeli 3.2 i wzorów (3.11), (3.12), (3.16) i (3.17). Zmienne podane są w postaci przeskalowanej z zachowaniem 15 miejsc znaczących. Procedury rozwiązujące równania stanu i równania sprzężone narzucają wysoką precyzję obliczeń z powodu dużej wrażliwości numerycznej rozwiązań.

Tabela 3.3
Współczynniki skalowania i wartości liczbowe parametrów modelu II

Współczynniki skalowania		**Wartość**
α_0	czasu	0,18521951450651
α_1	położenia wózka	3,36544494514998
α_2	kąta wahadła	1,0
α_3	prędkości wózka	1,81700343730884
α_4	prędkości wahadła	5,39899914252753
Parametry modelu		
b_2		0,42560841106040
b_3		$3,156490651146276 \cdot 10^{-4}$
c_4		11,21355409827388
g_1		3,93517214056836

Wszędzie dalej, dla lepszej czytelności wzorów, pominięto znak prim odróżniający zmienne skalowane od nieskalowanych. Liczba parametrów w równaniach ogranicza się do trzech, są nimi stałe: b_2, b_3 i c_4. Czwartym parametrem jest stała g_1 – przeskalowane sterowanie maksymalne.

Przeskalowane równania stanu z ograniczeniem na sterowanie zapisujemy w postaci

$$\dot{x} = f(x,u) \tag{3.18}$$

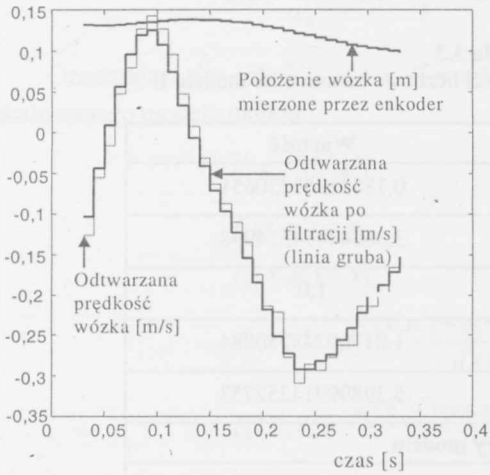
$$f_1(x,u) = x_3, \quad f_2(x,u) = x_4, \quad f_3(x,u) = \frac{v_1(x,u) + v_2(x)\cos x_2}{d(x)}$$

$$f_4(x,u) = \frac{v_1(x,u)\cos x_2 + c_4 v_2(x)}{d(x)}, \quad d(x) = c_4 - \cos^2 x_2 \tag{3.19}$$

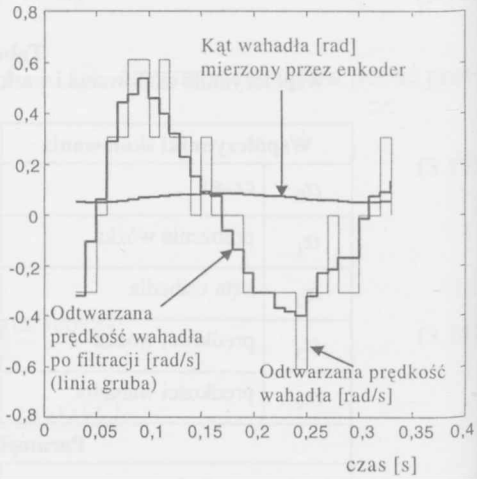
$$v_1(x,u) = u - x_4^2 \sin x_2 - b_2 x_3, \quad v_2(x) = \sin x_2 - b_3 x_4$$

$$|u| \leq g_1 \tag{3.20}$$

Rysunki 3.5 i 3.6 pokazują sygnały z czujników położenia wózka i kąta wahadła. Ze zmierzonych położeń odtwarza się sygnały prędkości w układzie i poddaje je filtracji.



Rys. 3.5. Sygnał z czujnika położenia wózka i odtwarzana prędkość wózka przed i po filtracji



Rys. 3.6. Sygnał z czujnika kąta wahadła i odtwarzana prędkość wahadła

Przy modelowaniu dynamiki systemu należy pamiętać o uwzględnieniu efektów kwantyzacji i próbkowania (Brown, Schneider, Mulligan 1992; Lemkin *et al.* 1995; Pauluk, Turnau 1997). Zniekształcenia siły przyłożonej do wózka, wskutek tarcia, naciągów linki napędowej i dynamiki wprowadzanej przez silnik, można kompensować za pomocą dodatkowych sprzężeń zwrotnych. Siłę działającą na wózek można identyfikować mierząc przyspieszenie bezpośrednio na wózku, używając czujnika bezwładnościowego (Turnau, Rosół, Piątek 2001).

3.2. Zadania sterowania i warunki ich realizacji w czasie rzeczywistym

Dla wahadła na wózku są budowane i testowane algorytmy sterowania wywodzące się z technik klasycznych i inteligentnych (Smith, Doyle 1992; Spong 1994; Wei, Dayawansa, Levine 1995; Shr-Shiung Hu, Bor-Chin Chang 1998; Tsachouridis 1999). Warto scharakteryzować problemy sterowania licznie prezentowane w literaturze dla tego standardowego systemu. W ciągu tylko jednego roku pojawia się kilkadziesiąt prac na temat sterowania wahadłem na wózku (Milam, Mushambi, Murray 2000).

Dla realizacji celu sterowania w systemie rzeczywistym należy rozważyć następujące zagadnienia:

- wyboru modelu dynamiki i identyfikacji jego parametrów;
- dokładności pomiaru zmiennych wyjściowych;
- odtwarzania nie mierzonych zmiennych (budowy obserwatorów);
- filtracji zakłóceń;
- generacji sterowań (wyboru sterownika, interfejsu, wzmacniaczy itp.);
- wyboru środowiska programowego symulacji i sterowania układem rzeczywistym;
- wyboru algorytmu sterowania;
- odporności systemu rzeczywistego na zmianę parametrów i zakłócenia.

Lista czynności niezbędnych dla uruchomienia poprawnego eksperymentu jest długa. W mniej złożonych algorytmach sterowania można postępowanie uprościć. Na przykład stabilizacja wahadła w górnym położeniu równowagi nie wymaga modelu dynamiki. Liniowy regulator na stanie, w ujemnym sprzężeniu zwrotnym, może spełnić to zadanie. Gdyby chodziło jednak o poszerzanie obszaru atrakcji, czyli strefy stabilizacji, czy o syntezę regulatora przy zadanym wskaźniku jakości, to korzystne byłoby użycie modelu (Isidori, Byrnes 1990). W systemie rzeczywistym, w odróżnieniu od jego modelu, algorytm sterujący otrzymuje wartości zmiennych stanu w postaci sygnałów z czujników pomiarowych, a nie na drodze symulacji. Pomiary takie obarczone są błędami próbkowania i kwantyzacji. Zakłócenia oraz trudne do zamodelowania efekty (na przykład naciągu linki poruszającej wózek czy tarcia suchego) zmniejszają zgodność modelu z systemem. Trzeba być po pierwsze świadomym występowania takich zjawisk, po drugie jeżeli nie można ich uniknąć, należy ich skutki minimalizować przez kompensację, filtrację, odtwarzanie stanu (budowę obserwatorów). Ostatecznością byłoby wprowadzenie modeli statystycznych dynamiki, a to z uwagi na nikłą ich użyteczność przy budowie sterowników.

W literaturze wyróżnia się na ogół dwa problemy sterowania dla wahadła na wózku. Pierwszy – oznaczany dalej STAB – to problem stabilizacji wahadła w górnym, niestabilnym punkcie równowagi z wykorzystaniem modelu liniowego. Drugi – oznaczany dalej POST – to nieliniowy problem postawienia wahadła, czyli przeprowadzenia z dolnego położenia równowagi w górne. Naturalne jest postawienie takich zadań, w których celem sterowania jest osiągnięcie równowagi w układzie (Mitkowski 1991).

W tabeli 3.4 podano zbiorcze zestawienie zadań rozwiązywanych w celu porównawczym. Wiele z nich wykorzystano dalej, między innymi do prezentacji algorytmów optymalizacyjnych, badań wrażliwości rozwiązań optymalnych, uodparniania trajektorii itp.

Zadania podane w tabeli 3.4 to:

- SUWN (ruch suwnicy) – przejazd wózka między dwoma punktami, przy czym na początku i na końcu ruchu wahadło znajduje się w dolnym położeniu równowagi;
- AKRO (ruch przypominający przemieszczanie się akrobaty po poziomej linie) – wózek przejeżdża z jednego położenia do drugiego, na początku i na końcu ruchu wahadło znajduje się w górnym położeniu równowagi;
- OBRG – wahadło z dolnego położenia równowagi wykonuje pełny obrót, wózek na początku i na końcu ruchu spoczywa w tym samym punkcie;
- OBRD – wahadło z górnego położenia równowagi wykonuje pełny obrót, wózek na początku i na końcu ruchu spoczywa w tym samym punkcie;

– OPUS – wahadło z górnego położenia równowagi opuszczane jest do dolnego położenia równowagi, wózek na początku i na końcu ruchu spoczywa w tym samym punkcie.

Tabela 3.4
Zadania porównawcze dla systemu

Nazwa zadania sterowania	Stan początkowy	Stan docelowy
STAB (stabilizacja)	x^0	–
POST (postawienie)	$x^0 = \mathrm{col}(0,\pi,0,0)$	$x^f = \mathrm{col}(0,0,0,0)$
OPUS (opuszczenie)	$x^0 = \mathrm{col}(0,0,0,0)$	$x^f = \mathrm{col}(0,\pi,0,0)$
OBRD (obrót dolny)	$x^0 = \mathrm{col}(0,2\pi,0,0)$	$x^f = \mathrm{col}(0,0,0,0)$
OBRG (obrót górny)	$x^0 = \mathrm{col}(0,\pi,0,0)$	$x^f = \mathrm{col}(0,-\pi,0,0)$
AKRO (ruch akrobaty)	$x^0 = \mathrm{col}(x_1^0,0,0,0)$	$x^f = \mathrm{col}(x_1^f,0,0,0)$
SUWN (suwnica)	$x^0 = \mathrm{col}(x_1^0,\pi,0,0)$	$x^f = \mathrm{col}(x_1^f,\pi,0,0)$
POST+AKRO (postawienie z ruchem akrobaty)	$x^0 = \mathrm{col}(x_1^0,\pi,0,0)$	$x^f = \mathrm{col}(x_1^f,0,0,0)$
OPUS+SUWN (opuszczenie z ruchem suwnicy)	$x^0 = \mathrm{col}(x_1^0,0,0,0)$	$x^f = \mathrm{col}(x_1^f,\pi,0,0)$

Wprowadza się różne kryteria jakości sterowania. Niekiedy kryterium nie jest formułowane, a chodzi tylko o algorytm, za pomocą którego można osiągnąć cel. Najtrudniejsza, ale najbardziej interesująca jest synteza sterowania optymalnoczasowego, której poświęcono część III pracy. W celu testowania numerycznych algorytmów optymalnoczasowych definiuje się nie tylko zadania polegające na przeprowadzaniu systemu między punktami równowagi, ale także inne zadania sterowania docelowego, z niezerowymi prędkościami wózka i wahadła na początku lub na końcu ruchu. Lista zadań testowych powiększa się wówczas ośmiokrotnie (Szymkat, Turnau, Korytowski 1997).

Część II
STEROWANIE INTELIGENTNE

4. Układy inteligentne i uczące się

Rozwój techniki komputerowej umożliwia budowanie systemów sterowania o wysokim stopniu autonomii. Do autonomicznego działania niezbędne są mechanizmy podejmowania decyzji w obecności zakłóceń i w warunkach niepewności. Takiemu wyzwaniu chce sprostać sterowanie nazywane inteligentnym, które naśladuje metody rozumowania ludzkiego i funkcje zwykle spełniane przez człowieka lub organizmy żywe. Do algorytmów sterowania inteligentnego włącza się procedury uczące się na podstawie przeszłych doświadczeń oraz planujące reakcje na przewidywalne zdarzenia. By wyeliminować lub – w mniej korzystnym przypadku – zredukować funkcje spełniane dotychczas przez operatora systemu, wprzęga się do prototypowania algorytmów automatyczną identyfikację, diagnostykę błędów i sztuczną inteligencję.

Nauka sterowania ma doprowadzić do dotarcia z wyjściem układu do celu. Uczenie rozpoczyna się od obserwacji reakcji wyjścia układu na przypadkowe wejście. Tworzy się algorytmiczny mechanizm gromadzenia wiedzy – zapamiętywania par (sterowanie/wyjście) sprzyjających dotarciu do celu. Równolegle buduje się mechanizm regulacji – sterowania są generowane na podstawie już zgromadzonej wiedzy o systemie. Uczenie się powoduje, że generowane sterowanie zmienia charakter z przypadkowego na uporządkowany. Duża sprawność takiej nauki zależy od szybkości zbieżności użytych algorytmów.

4.1. Pojęcie sterowania inteligentnego

W najbardziej ogólnym podejściu, sterowanie inteligentne ma strukturę hierarchiczną i dzieli się na powiązane ze sobą warstwy: zarządzania, koordynacji i wykonania. W niniejszej pracy zajęto się warstwą wykonania, uzupełniając ją o warstwę koordynacji wszędzie tam, gdzie występuje proces uczenia się algorytmu.

Słowo „inteligencja" kojarzy się przede wszystkim z naszą zdolnością do abstrakcyjnego rozumowania, umiejętnością syntezy i wykorzystania zbieranej i zapamiętanej informacji do koncepcyjnego myślenia i rozwiązywania złożonych problemów. Znamiona inteligencji, pomimo braku zdolności do abstrakcyjnego myślenia, są widoczne wśród organizmów ży-

wych. Idąc dalej, można nazwać inteligentnym organizm biologiczny lub system nieożywiony, który ma „możliwość zbierania i wykorzystywania gromadzonej wiedzy". Inteligentny będzie też algorytm lub procedura generowane przez „podglądanie natury". A więc rozciągnięcie pojęcia „inteligentny" na systemy nieożywione i dodanie przymiotnika „inteligentny" w celu opisania działania sterownika, regulatora, algorytmu, systemu sterowania itp. odnosi się do metodologii sterowania.

Trudności z definicją systemu inteligentnego są naturalną konsekwencją niejednoznaczności pojęcia „sztuczna inteligencja". Różne definicje można znaleźć na przykład w pracach Uhla, Szymkata (1992), Arzena (1995), Gupty, Sinhy (1996) i Turnaua (1998). Sterowanie inteligentne to stosunkowo nowy obszar automatyki, z ciągle rozwijającą się, lecz nadal niepełną teorią. Podejmowane są próby opisu podstawowych pojęć automatyki, ugruntowanych dla sterowników konwencjonalnych (na przykład regulatorów PID). Przy konstrukcji sterownika zasadnicze znaczenie ma utworzenie mechanizmu gromadzenia wiedzy o procesie, korzystania z tej wiedzy i poprawiania jej, a więc uczenia się.

Mechanizmy podejmowania decyzji w sterowniku inteligentnym i konwencjonalnym są podobne. Poszukiwana jest najlepsza decyzja w odniesieniu do użytego kryterium. A więc regulator zbudowany w logice rozmytej posiada mechanizm wyostrzania, który ze zbioru rozmytego, powstałego z agregowanych reguł, w optymalny sposób dla danego celu sterowania generuje ostrą wartość wyjścia regulatora. Podobnie, wagi sieci neuronowej w regulatorze neuronowym są zmieniane w trakcie uczenia się sieci tak, by zminimalizować zadany wskaźnik jakości. Z kolei algorytm genetyczny wypełnia regułę „przeżyć dzięki dopasowaniu".

Jeżeli konwencjonalny system sterowania staje się bardziej złożony lub mniej dokładnie znany, a więc nasza baza wiedzy o systemie jest niekompletna lub niepewna, wówczas zdefiniowanie optymalnego sterowania względem zadanego kryterium może się nie udać. Jeżeli etap definicyjny, mimo trudności, zakończy się pomyślnie, to nadal nie mamy pewności, czy dostępne środki komputerowe (programowe i sprzętowe) pozwolą na uzyskanie rozwiązań. Wówczas pojawia się naturalna potrzeba zastosowania technik określanych jako inteligentne.

Wydaje się, że idealny algorytm inteligentny powinien składać się z części konwencjonalnej, w której bazę wiedzy o systemie stanowi model matematyczny, powstały na podstawie równań dynamiki obiektu i układu regulacji, i z części inteligentnej, wykorzystującej bazę wiedzy, utworzoną podczas eksperymentów na modelu rzeczywistym, a w przypadku niemożności lub trudności przeprowadzenia eksperymentów identyfikacyjnych dla rzeczywistego procesu – wiedzę zebraną z symulacji modelu procesu.

Przymiotnik „inteligentny" w odniesieniu do sterowania może oznaczać, że ma miejsce adaptacja, samostrojenie regulatora czy też uczenie się. Termin „uczenie się" w systemie inteligentnym dotyczy na ogół uczenia się sztucznej sieci neuronowej. Uczenie się może się odbywać pod nadzorem lub bez nadzoru. W pierwszym przypadku sam system uczący się otrzymuje informację (najlepiej w każdym kwancie czasu sterowania), czy działa dobrze i czy jest zdolny do poprawy działania w przyszłości. W drugim przypadku system uczy się w sposób przypadkowy, ponieważ nie otrzymuje informacji o swoim działaniu.

Te dwa przeciwstawne sposoby uczenia się występują rzadko w czystej formie podczas uczenia się systemów rzeczywistych. Na ogół system nadzorujący (może być nim też

człowiek) informuje system uczący się, w wybranych chwilach czasu, czy sytuacja rozwija się we właściwym kierunku. Za pozytywne wyniki uczenia się generowana jest nagroda, a za negatywne – kara. Te metody określa się w literaturze jako *reward/punishment* (Widrow, Gupta, Maitra 1973). Nagroda lub kara generowane są co pewną liczbę akcji sterujących, a więc nie wiadomo która z akcji miała wkład pozytywny, a która negatywny do osiągnięcia celu. By to sklasyfikować, przypisuje się sterowaniom punkty kredytowe.

Reasumując, pojęcie systemu inteligentnego jest określane na dwa sposoby. Jest to układ:

1) powstały przy użyciu technik wywodzących się z ludzkich lub biologicznych metod gromadzenia wiedzy, form reprezentacji tej wiedzy i podejmowania decyzji;
2) realizujący funkcje, wykonywane przez człowieka, zwierzę czy organizm biologiczny.

W pierwszym przypadku chodzi o podpatrzenie metod, technik i mechanizmów: ludzkiego, zwierzęcego lub biologicznego działania, i zastosowanie ich w postaci sztucznej inteligencji w układzie. Z tego względu maszyna wnioskująca zbudowana w technice rozmytej, lub sieć neuralna albo regulator regułowy – użyte do generacji sterowań – określane są jako sterowniki inteligentne.

W drugim przypadku użyte techniki sterowania są konwencjonalne, ale układ sterowania realizuje zadania, które zwykle wykonuje człowiek czy inny organizm biologiczny. Do tej grupy zaliczamy roboty, które powinny chociaż w części być układami autonomicznymi, tak aby mogły, podobnie jak człowiek, reagować na zmiany środowiska, w którym pracują, i na zakłócenia występujące w układach sterowania (Gupta, Sinha 1996).

4.2. Algorytmy uczące się

Układ uczący się może w różnym stopniu wykorzystywać model matematyczny dynamiki obiektu. Niekiedy jest to model niepełny. W innych przypadkach pełny, ale niepewny. Najbardziej ambitnym zadaniem jest postawienie problemu „czarnej skrzynki", gdzie na podstawie ciągu sterowań i odpowiadającego mu ciągu wartości wyjścia buduje się przepis sterowania. Należy odpowiedzieć na pytanie, jaką informację należy zapamiętać podczas eksperymentów i jak wykorzystać zapamiętaną wiedzę. Trzeba także ustalić, jak ma wyglądać mechanizm zapamiętywania i zapominania, dopasowujący się do zmian parametrów systemu czy do zmian środowiska sterowania i zakłóceń sygnałów (Cypkin 1973).

Ujmując rzecz historycznie, wahadło na wózku jako standard porównawczy do badania mechanizmu uczenia się sterowania systemem dynamicznym, zostało użyte po raz pierwszy w algorytmie BOXES Michie'a, Chambersa (1968). Algorytm uczy się rozwiązywać zadanie balansowania wahadłem w górnym położeniu (na szynie o skończonej długości) tak, aby jak najdłużej je podtrzymać, nim opadnie w dół. Przestrzeń stanu dzieli się na czterowymiarowe prostopadłościany (stąd pochodzi nazwa algorytmu). Każdy z prostopadłościanów jest niezależnym elementem uczącym się. Z prostopadłościanem związana jest specyficzna dla niego akcja sterująca. Polega ona na wygenerowaniu sterowania $u_{\max}$ lub $-u_{\max}$, a więc chodzi o to, czy pchnąć wózek po szynie w prawo, czy w lewo. Nauka odbywa się metodą prób i błędów. Na początku przyporządkowanie wartości sterowania prostopadło-

ścianom jest przypadkowe. Podczas nauki zapamiętuje się dane statystyczne dla prostopadłościanów.

Oznaczmy dla każdego z prostopadłościanów przez:

$N_{u_{\max}}$ – ilość zastosowanych sterowań o wartości $u_{\max}$ (od początku eksperymentu),

$T_{u_{\max}}$ – sumaryczny czas poprawnej akcji balansowania (bez upadku) po zastosowaniu sterowania $u_{\max}$.

Podobnie określa się wartości $N_{-u_{\max}}$ i $T_{-u_{\max}}$.

Oryginalny algorytm BOXES wylicza i zapamiętuje lokalne (związane z jednym prostopadłościanem) i globalne wartości (związane ze wszystkimi prostopadłościanami), które pozwalają na introspekcję strategii uczenia się dla generacji reguł sterowania. Algorytm wylicza relatywną korzyść z zastosowania sterowania $u_{\max}$ lub $-u_{\max}$ w prostopadłościanie.

Reguły mają postać:

(i) jeżeli $w_p = w_l$, to wybierz losowo jedną z wartości: $-u_{\max}$, $u_{\max}$, idź do (iii);

(ii) jeżeli $w_p < w_l$, to $u = -u_{\max}$, w przeciwnym przypadku $u = u_{\max}$, idź do (iii);

(iii) koniec.

Wielkość w_p wylicza się ze wzoru

$$w_p = \frac{T_{u_{\max}} + zc}{N_{u_{\max}} + T_{u_{\max}}} \tag{4.1}$$

gdzie

$$c = c_0 + c_1 k\,, \qquad k = \frac{G_T}{G_N} \tag{4.2}$$

G_N jest liczbą wszystkich akcji sterujących, G_T – łącznym czasem balansowania, stałe z, c_0 i c_1 są dobierane eksperymentalnie i służą do wprowadzenia mechanizmu zapominania starych danych statystycznych, podczas uczenia się przy pojawianiu się nowych danych.

Algorytm BOXES w podobny sposób przyporządkowuje prostopadłościanom wartości w_l, zastępując w powyższych wzorach $u_{\max}$ przez $-u_{\max}$.

Formuła wyliczająca parametr c przy szacowaniu wartości w_p i w_l ma postać kompromisu pomiędzy przyjmowaniem nowych podpowiedzi sterowania a korzyścią płynącą z posiłkowania się starymi, globalnymi akcjami sterującymi w prostopadłościanach. System staje się bardziej konserwatywny w podejmowaniu nowych akcji sterujących wraz z globalnym polepszeniem jego działania.

Sammut i Michie (1991) oraz Sammut (1994) zauważyli, że oryginalny algorytm BOXES podczas jednokrotnego wykonania uczy się sterowania, które jest poprawne tylko lokalnie, tzn. niekoniecznie spełnia zadanie stabilizacji przy starcie z innych warunków początkowych systemu. Algorytm wymaga dużej liczby prób. Wprowadzono szereg modyfikacji do pierwotnego algorytmu. Cribb (1989) wprowadził lokalną miarę korzyści ze sterowania $k = \max\left(\frac{T_{u_{\max}}}{N_{u_{\max}}}, \frac{T_{-u_{\max}}}{N_{-u_{\max}}}\right)$, zamiast korzyści globalnej określonej przez (4.2), równocześnie modyfikując regułę sterowania, tak aby stało się widoczne, kiedy algorytm korzysta z zapamiętanej historii sterowań, a kiedy podejmuje nowe akcje sterowania.

Dalsze modyfikacje do reguł sterowania wprowadził Law (1992), proponując algorytm dla prostopadłościanu, w którym aktualnie znajduje się stan systemu, o postaci:

(i) jeżeli $N_{u_{\max}} N_{-u_{\max}} = 0$, to idź do (iv);

(ii) jeżeli $\frac{T_{u_{\max}}}{N^n_{u_{\max}}} \geq \frac{T_{-u_{\max}}}{N^n_{-u_{\max}}}$, to $u = u_{\max}$, idź do (v);

(iii) jeżeli $\frac{T_{u_{\max}}}{N^n_{u_{\max}}} < \frac{T_{-u_{\max}}}{N^n_{-u_{\max}}}$, to $u = -u_{\max}$, idź do (v);

(iv) sterowanie przyjmuje jedną z wartości $-u_{\max}$ lub $u_{\max}$, wybraną losowo, idź do (v);

(v) koniec.

Dla małych wartości wykładnika n algorytm ma cechy konserwatywne. Im większa jest stała n, tym częściej modyfikowane są sterowania przypisywane prostopadłościanom. Aby z wielu różnych przebiegów uczenia się otrzymać jedno, charakterystyczne dla konkretnego prostopadłościanu sterowanie, przypisuje się prostopadłościanom ilość głosów na $u_{\max}$ lub $-u_{\max}$, otrzymanych podczas eksperymentu. Następnie stosuje się test istotności χ^2 takiego głosowania. Akcje sterujące, które dały pozytywny wynik przy przeprowadzonym teście χ^2, zamraża się. W ten sposób dąży się do ekstrakcji kształtu drzewa decyzyjnego algorytmu, tak aby uzyskać lepszą czytelność wzoru na sterowania generowane w algorytmie (Sammut 1994).

Metody uczenia się sterowania dla zadania STAB realizowano za pomocą sieci neuralnych. Guez, Selinsky (1988) zaproponowali ciąg uczący generowany przez człowieka. Tolat, Widrow (1998) użyli ciągu uczącego generowanego przez system rozpoznawania obrazu. Zamiast poszukiwania minimalnej reprezentacji stanu użyli informacji o pikselach na ekranie. W pracy Andersona (1989) kontynuowano idee uczenia algorytmu BOXES z dwoma innowacjami – prostopadłościany zastąpiono 4-wymiarowym wektorem stanu i dodaniem dodatkowej warstwy w sieci neuronowej, dla odwzorowania nieliniowej relacji wejście/wyjście. Barto, Sutton, Anderson (1983) zwiększyli szybkość uczenia sieci neuronowej dzięki strukturze z predykcją. Pomocny okazał się tzw. *krytyk adaptacyjny*.

Uczenie się sterowania niekoniecznie musi dotyczyć balansowania wahadłem. O wiele bardziej interesujący problem sterowania przedstawia zadanie POST. W pracy Kołka,

Tabakowskiego, Turnaua (1993) użyto sieci neuronowej nauczonej *off line* algorytmu heurystycznego, z adaptacyjnym dostrajaniem reguł.

Z uczeniem wiąże się zdefiniowanie funkcji oceny zmian sterowania. W metodzie prób i błędów, ocena – na początku – dotyczy pierwszych, a więc nielicznych prób zmiany sterowania. Ocena jest sukcesywnie wzbogacana doświadczeniem z kolejnych prób. Ma zatem miejsce adaptacja funkcji oceny do nowych ciągów par (x,u) wektora stanu i sterowania. Część regulatora generująca ocenę nazywana jest krytykiem adaptacyjnym. W najprostszych przypadkach jego rola sprowadza się do przestrajania wzmocnienia sygnału w regulatorze. Dla zadania POST funkcja krytyka jest bardziej złożona. Pokazuje to przykład rozwiązany w pracy Kratochwila, Engelbrechta, Jörgla (1993). Uczenie się sterowania przebiega metodą nagroda – kara. Sterowaniem jest siła $\pm u_0 \sin \omega t$, $\omega t \in [0, \pi]$ zmienna w czasie, o stałej amplitudzie u_0 i stałej częstotliwości ω, dobranej do okresu drgań swobodnych wahadła. Amplituda wahań wahadła jest zwiększana przez zmianę zwrotu siły przyłożonej do wózka. O przeprowadzeniu zmiany decyduje osiągnięcie maksimum funkcji oceny. Ocena jest generowana, jeżeli zostanie spełniony następujący warunek

$$x_4 = 0 \quad \text{lub} \quad x_2 = 0 \quad \text{lub} \quad |x_1| = |x_{1\max}| \tag{4.3}$$

a więc w przypadku, gdy wahadło zatrzymuje się i rozpoczyna ruch w kierunku przeciwnym, gdy przechodzi przez górne niestabilne położenie i gdy wózek osiąga brzegi zakresu jezdnego szyny. Ocena jest karą za oddalanie się od celu lub nagrodą za zbliżanie się do celu. Nagroda dotyczy punktu trajektorii $x^* = \text{col}(x_1^*, x_2^*, x_4^*)$, w którym nastąpiła zmiana sterowania. Zachowanie systemu ocenia się bez uwzględniania zmiennej x_3, bo prędkość wózka, jak widać z formuł (4.3), nie jest warunkiem generowania oceny. Nagrodę w punkcie x^* rozciąga się na punkty sąsiadujące z x^*, przy czym im dalszy punkt, tym mniejsza nagroda (proponuje się wykładnicze zanikanie wartości nagrody z odległością od punktu x^*). W ten sposób z nagrody w punkcie tworzy się funkcję oceny sterowania w przestrzeni stanu. Funkcja ta posiada maksimum w punkcie x^*. Każda zmiana sterowania w kolejnych punktach x_i^*, $i = 1, 2, \ldots$, jeżeli jest nagrodzona po pewnym czasie (wyliczenie oceny jest opóźnione w stosunku do zmiany sterowania, gdyż skutków działania nie można oceniać od razu), zmienia ocenę wszystkich dotychczasowych sterowań wraz z ostatnim. Ocena wypadkowa jest superpozycją wszystkich dotychczasowych ocen. I ta właśnie ocena, będąca funkcją odległości euklidesowej od punktów x_i^*, $i = 1, 2, \ldots$, służy tutaj jako krytyk adaptacyjny. Osiągnięcie maksimum oceny w jednym z punktów rozwijającej się trajektorii x decyduje o zmianie sterowania. Okazuje się, że nie wszystkie nagradzane punkty zmiany sterowania, wraz z nagrodami, należy włączać do sumarycznej oceny. Proponuje się selektywny mechanizm zapamiętywania. Dodatkowo, gdy któryś z uprzednio zapamiętanych punktów x_i^*, $i = 1, 2, \ldots$ leży blisko nowego punktu x_k^*, to x_k^* zostaje zapamiętany, a bliski mu punkt, zapamiętany wcześniej, usunięty z pamięci. Zapobiega to powstawaniu cykli granicznych.

Przedstawiona metoda sterowania nie korzysta z żadnego modelu matematycznego systemu. Uwalniamy się od identyfikacji tarcia, dynamiki silnika itp. Wejściem jest napięcie

podawane na silnik, a wyjściem pomiary kąta i prędkości wahadła oraz położenia wózka na szynie. Wyjazd poza pole jezdne szyny generuje karę, a więc para (x_k*, kara) i związane z nią sterowanie nie są zapamiętywane i włączane w funkcję oceny. Do realizacji zadania POST dochodzi się przez zwiększanie amplitudy wahań wahadła, co doprowadza po pewnym czasie do ruchu obrotowego wahadła i, przez zmniejszanie prędkości obrotowej, do osiągnięcia górnego niestabilnego punktu równowagi. Po uruchomieniu eksperymentu, pierwsze akcje sterujące wykonywane są w sposób przypadkowy, celem powstania wstępnej funkcji oceny.

5. Sterowanie regułowe

Podstawowym źródłem powstawania reguł jest obserwacja odpowiedzi modelu lub obiektu na testowe ciągi sterujące. Doświadczenia zebrane podczas eksperymentów testujących przy zastosowaniu reguł heurystycznych zostają poddane analizie i przybierają formę algorytmu. Tworzenie reguł może być wspomagane częściowymi modelami matematycznymi zjawisk fizycznych sterowanego procesu. Można też działać bez modelu. Przykładowo, operator procesu destylacji radzi sobie dobrze z prowadzeniem procesu nie posługując się analizą matematyczną, a jedynie sięga po nieformalny model, którym jest doświadczenie zdobyte przez lata pracy, zapisane w jego pamięci (Byrski, Duda, Turnau 1988). Trzeba zaznaczyć, że w tym przypadku bezpośrednie wykorzystywanie procesu do generowania reguł sterowania jest możliwe z uwagi na jego wolną zmienność.

Budowanie algorytmów regułowych jest w pełni uzasadnione, gdy:

- złożoność, niepewność parametrów, silne nieliniowości modelu matematycznego nie pozwalają na analizę i syntezę regulatora (Gorczyca, Turnau 1998);
- czas rzeczywisty wykonania algorytmu opartego na matematycznych metodach analizy i syntezy przekracza maksymalny dopuszczalny czas sterowania (czas cyklu obliczeń takiego algorytmu jest dłuższy od czasu próbkowania komputera generującego ciągi sterujące);
- nie zachodzi żaden z dwóch pierwszych warunków, a algorytm regułowy wypełnia zadanie sterowania, zachowując w szerokim zakresie odporność na zakłócenia, zmiany parametrów obiektu i zmiany wartości początkowych zmiennych stanu;
- najwyższy priorytet przypisuje się niskiej czasochłonności projektu (Kołek *et al.* 1995).

Zasadność podejścia algorytmicznego, opartego na regułach heurystycznych, pokazano w następnych podrozdziałach, rozwiązując zadania sterowania dla systemu wahadła na wózku. Z przykładów tych wynika, że regulator regułowy wystarcza tam, gdzie prosty regulator konwencjonalnego typu jest nieskuteczny, a budowa regulatora przy użyciu matematycznych metod analizy i syntezy jest zbyt pracochłonna. W algorytmach regułowych dla wózka z wahadłem, przedstawianych w literaturze, kładzie się nacisk przede wszystkim na osiągnięcie celu sterowania. Koszty sterowania, a więc czas trwania sterowania, zużyta energia itp., są do pominięcia, tak jak to pokazano w pracy Goodricha, Stirlinga, Frosta (1996), gdzie definiuje się klasę sterowań przybliżających osiągnięcie celu, lub uznaje się je za mniej istotne od samego powodzenia akcji sterującej, na przykład tak jak jest to opisane

w pracach: Kratochwila, Engelbrechta, Jörgla (1993); Turnaua (1993); Chung Choo Chunga, Hausera (1995); Gevy, Hanga, Zhanga (1998); Medrano-Cerdy (1999).

Prócz matematycznego modelu dynamiki i algorytmu regułowego, poniżej przedstawiono porównanie wyników sterowania regułowego z bangbangowym. Daje to pewne wyobrażenie o jakości procedury inteligentnej, szczególnie wówczas, gdy dodatkowo zostanie ona przetworzona w taki sposób, aby sterowania przybrały postać bangbangową, bez podpowiedzi ze strony algorytmu optymalnoczasowego (Turnau 1995). Zauważmy, że reguły algorytmu można poddać procesowi adaptacji, jeżeli potrafimy wprowadzić ocenę jakości tych reguł z punktu widzenia celu sterowania. Staje się wówczas możliwe zapięcie pętli sprzężenia zwrotnego adaptacji. Algorytm zaczyna przystosowywać się do zmian środowiska sterowania, zmian parametrów obiektu lub zmian klasy zakłóceń.

5.1. Algorytmy regułowe

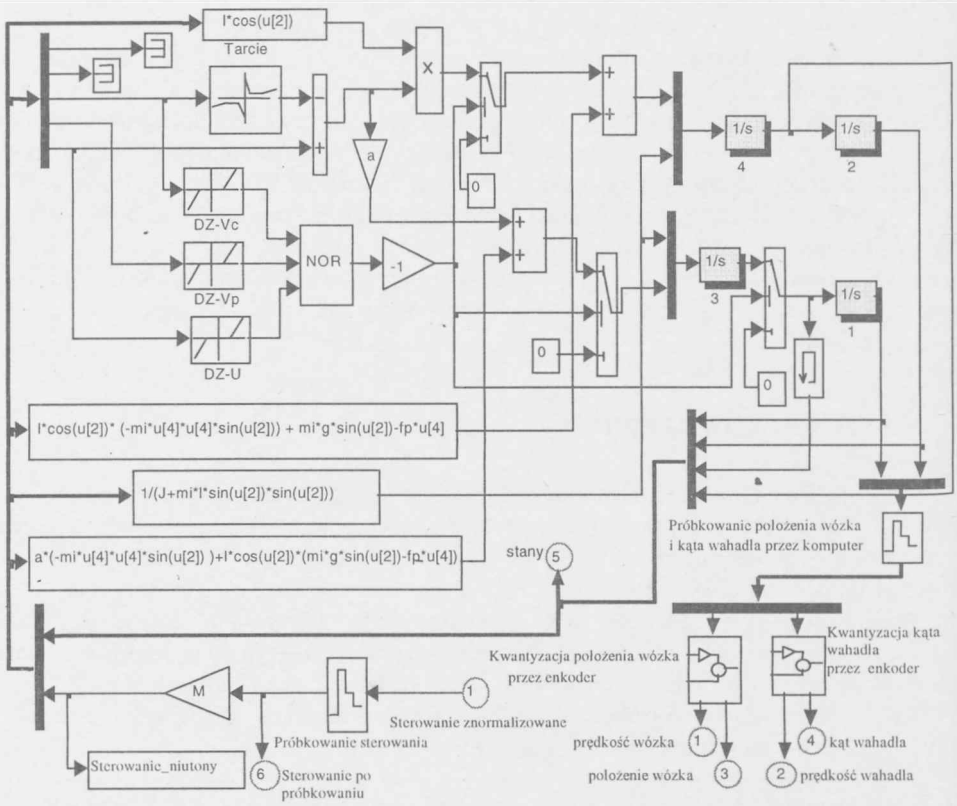
Sterowanie regułowe nie musi być optymalne w sensie kryteriów kwadratowych, czasowych i innych, jest raczej pewną heurystyczną strategią, która ma zapewnić osiągnięcie wyznaczonego celu sterowania. Strategia ta winna być odporna na niewielkie zmiany parametrów modelu i na zakłócenia, które zawsze są obecne w każdym systemie rzeczywistym. Podstawą do napisania reguł sterowania jest znajomość dynamiki systemu poparta eksperymentami i intuicją inżynierską. Przy dobrej praktycznej znajomości systemu taka metoda konstrukcji algorytmów sterowania jest szybka i skuteczna (Turnau 1993, 1994).

Niżej zajmiemy się konstrukcją algorytmów regułowych dla systemu wahadła na wózku. Rozpoczniemy od postawienia zadania.

Zadanie regułowe R1 POST. Z dowolnego stanu początkowego $x^0 \in R^4$ spełniającego $|x_1^0| \le Z$ (Z jest połową zakresu jezdnego szyny), przeprowadzić system w możliwie krótkim czasie do stanu końcowego $x^f = 0$, przy ograniczeniach na sterowanie: $|u| \le u_{\max}^{\mathrm{POST}}$, gdy $|x_2| \ge S$ (wahadło jest poza strefą stabilizacji), oraz $|u| \le u_{\max}^{\mathrm{STAB}}$, gdy $|x_2| < S$ (wahadło jest w strefie stabilizacji) lub zachodzi warunek 6 z tabeli 5.1.

Model obejmujący dynamikę systemu i regulatorów, wygodnie jest napisać w Simulinku. Do całkowania wybrano metodę Bogackiego – Shampine'a o zmiennym kroku, stosowaną do układów ciągłych. Ponieważ sterowanie i pomiar w rzeczywistym układzie mają charakter dyskretny, używa się bloków kwantowania i próbkowania celem uzyskania modelu zgodnego z procesem rzeczywistym.

Na rysunku 5.1 pokazano model dynamiki systemu. Wykorzystano go do budowy algorytmu regułowego. Wprowadzono dwa różne ograniczenia sterowania u. Pierwsze, $|u| \le u_{\max}^{\mathrm{POST}}$, wynika z ograniczonej długości szyny – zwiększenie $u_{\max}$ spowodowałoby przekroczenie zakresu jezdnego w fazie rozhuśtywania wahadła. Drugie ograniczenie, $|u| \le u_{\max}^{\mathrm{STAB}}$, określone przez maksymalną siłę uzyskiwaną w układzie napędowym, stosuje się w stabilizacji, gdzie nie ma niebezpieczeństwa przekroczenia zakresu jezdnego.



Rys. 5.1. Wnętrze bloku „Dynamika"

Za pomocą jednego sterowania należy oddziaływać równocześnie na dwa obiekty: bezpośrednio na wózek i pośrednio na wahadło. Wózek podczas ruchu nie może przekroczyć końców szyny i w chwili końcowej ma zatrzymać się w środku szyny. Wahadło, poza strefą górną stabilizacji, ma zwiększać wahania, a w strefie górnej przeciwnie – zmniejszać, aż do uzyskania stanu spoczynku.

Algorytm regułowy, realizowany za pomocą tego modelu, podano w tabeli 5.1. W celu zapewnienia przejrzystości kąt x_2 sprowadza się do przedziału $[-\pi,\pi]$; przyjęto też, że górny indeks wskazuje na punkt trajektorii, a dolny – na numer współrzędnej stanu.

Parametr Z ustala się na wartości mniejszej niż rzeczywiste pół długości jezdnej szyny. Jeżeli wózek przekroczy założone pole jezdne szyny (warunek 1; **wyjście**), to siła przyłożona do wózka ma zwrot do środka szyny (reguła 1). Jeżeli wahadło znajduje się w otoczeniu górnego punktu równowagi w strefie stabilizacji S (warunek 2), to działa regulator liniowy (reguła 2) z nasyceniem i kompensacją tarcia. $K = [K_1,K_2,K_3,K_4]$ jest macierzą wzmocnień w torach sprzężeń zwrotnych, wyliczoną przez rozwiązanie odpowiedniego problemu liniowo-kwadratowego.

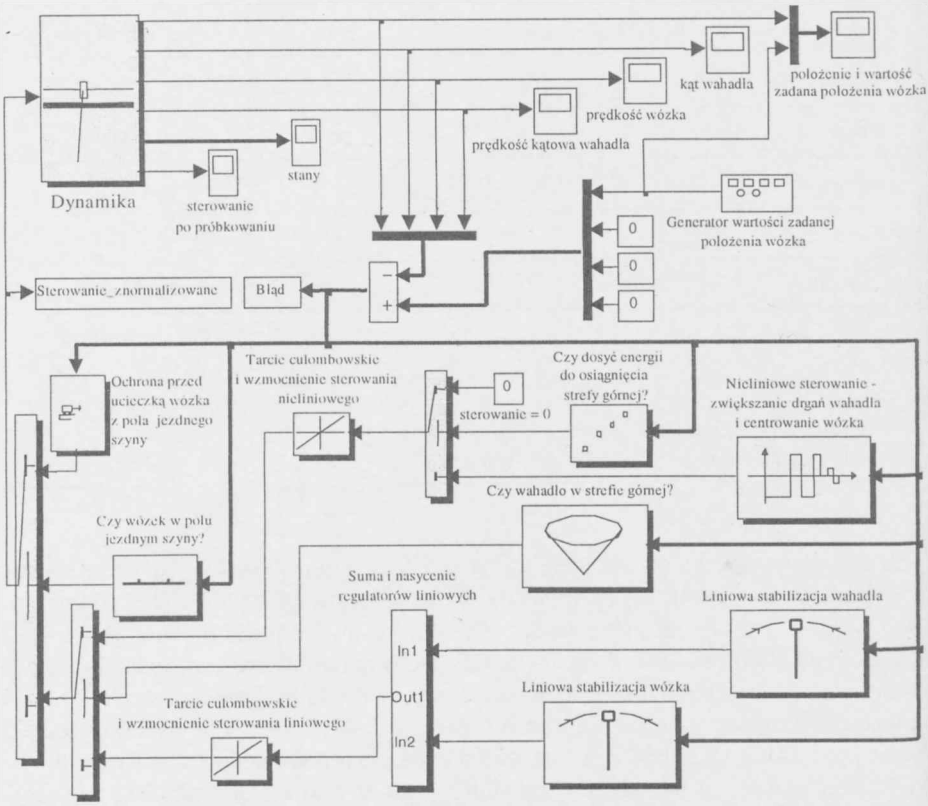
Tabela 5.1
Algorytm regułowy

If $\lvert x_1\rvert - Z > 0$	warunek 1
then $u = -u_{\max}^{STAB}\operatorname{sign} x_1$	reguła 1; **wyjście**
elseif $\lvert x_2\rvert - S < 0$	warunek 2
then $u_r = K_1(x_1 - x_1^f) + K_2x_2 + K_3x_3 + K_4x_4$	reguła 2
if $\lvert u_r\rvert + F_s > u_{\max}^{STAB}$	warunek 3
then $u = u_{\max}^{STAB}\operatorname{sign} u_r$	reguła 3; **wyjście**
else $u = u_r + F_s\operatorname{sign} u_r$ end	reguła 4; **wyjście**
elseif $0{,}5\,x_4^2 + 9{,}81\cdot 3{,}2(\cos x_2 - 1) > 0$	warunek 4
then $u = 0$	reguła 5; **wyjście**
elseif $x_1x_4\left(\lvert x_2\rvert - \frac{1}{2}\pi\right) < 0$	warunek 5
then $u = -u_{\max}^{POST}\operatorname{sign}\left(x_4\left(\lvert x_2\rvert - \frac{1}{2}\pi\right)\right)$	reguła 6; **wyjście**
elseif $\left\lvert \lvert x_2\rvert - \frac{1}{2}\pi\right\rvert < \gamma$	warunek 6
then $u = -u_{\max}^{STAB}\operatorname{sign} x_1$	reguła 7; **wyjście**
else $u_r = -u_{\max}^{POST}\left(1 - \alpha\lvert x_1\rvert - \beta x_3^2\right)\operatorname{sign}\left(x_4\left(\lvert x_2\rvert - \frac{1}{2}\pi\right)\right)$	reguła 8
if $\lvert u_r\rvert + F_s > u_{\max}^{POST}$	warunek 7
then $u = u_{\max}^{POST}\operatorname{sign} u_r$	reguła 9; **wyjście**
else $u = u_r + F_s\operatorname{sign} u_r$ end end	reguła 10; **wyjście**
Z – połowa długości zakresu jezdnego szyny S – połowa kąta strefy stabilizacji górnej wahadła	

Jeżeli sterowanie liniowe dojdzie do ograniczeń (warunek 3), to – by ich nie przekroczyć – podlega nasyceniu (reguła 3; **wyjście**). W układzie występują efekty trudne do zamodelowania – i tak tarcie suche wózka jest kompensowane przez dodanie do sterowania lub odjęcie od sterowania stałej wartości. Gdy sterowanie liniowe jest dodatnie (reguła 4; **wyjście**), to wartość jego zostaje powiększona o stałą F_s, by pokonać tarcie suche. Dla sterowania ujemnego zmniejsza się jego wartość o stałą F_s (reguła 4; **wyjście**). Jeżeli wahadło jest poza strefą stabilizacji, to ma miejsce akcja sprowadzania wahadła do tej strefy (zwiększanie amplitudy wahań). By wahadło osiągnęło górne położenie, bez tendencji do dalszego obrotu, należy zadbać, by prędkość kątowa wahadła w górnym położeniu była bliska zeru. Algorytm sprawdza, w każdym kroku próbkowania, czy energia kinetyczna

wahadła wystarcza do podniesienia środka ciężkości wahadła w jego górne położenie. Jeżeli jest spełniony warunek 4, to sterowanie zostaje wyłączone i przyjmuje wartość zero (reguła 5; **wyjście**). Reguła zwiększania amplitudy wahań ma prostą postać (reguła 6; **wyjście**). Sterowanie ma charakter bang-bang. Czasy przełączeń są funkcją kąta i znaku prędkości kątowej wahadła. Jeżeli nie jest spełniony warunek 5, to oznacza to, że zwiększanie amplitudy drgań wahadła odbywa się kosztem oddalania się wózka od środka szyny. By nie zaburzyć rytmu zwiększania amplitudy wahań wahadła i nie wyjechać poza założony zakres jezdny szyny, zmniejsza się wielkość siły sterującej (reguła 8), proporcjonalnie do położenia wózka x_1 (ze współczynnikiem proporcjonalności α) i proporcjonalnie do kwadratu prędkości wózka x_3^2 (ze współczynnikiem proporcjonalności β). Parametry α i β, służące centrowaniu wózka, są dobierane eksperymentalnie.

Na rysunku 5.2 jest pokazany simulinkowy schemat regulatora regułowego, wykorzystujący algorytm z tabeli 5.1. Regulator ten będzie użyty w symulacyjnych eksperymentach sterowania z modelem „Dynamika", a także w eksperymentach rzeczywistych – po zastąpieniu modelu systemem.



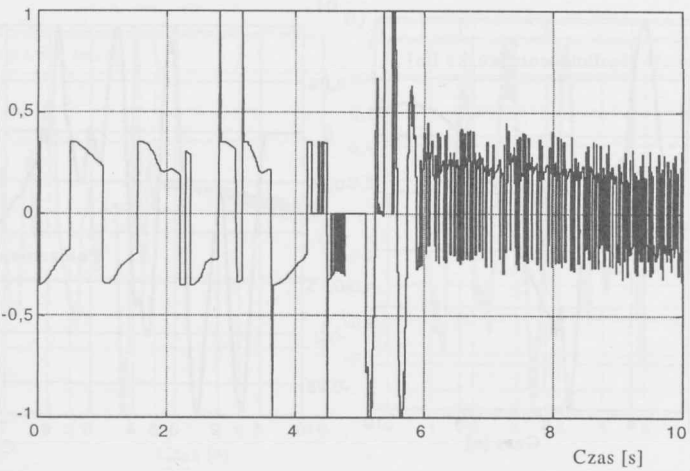
Rys. 5.2. Model sterowania regułowego

Następująca obserwacja zachowania systemu upoważnia do wprowadzenia dodatkowej akcji centrowania wózka, mianowicie ruch wózka po szynie w lewo i w prawo ma nieznaczny wpływ na ruch wahadła, gdy znajduje się ono w małym otoczeniu pozycji poziomej: $|\pi/2 - |x_2|| < \gamma$ (warunek 6). Bez zaburzania ruchu wahadła można wymuszać ruch wózka do środka szyny pełną siłą sterującą (reguła 7; **wyjście**). Sterowanie określone w regule 8 może ulec nasyceniu. Nasycenie sterowania z uwzględnieniem tarcia statycznego F_s, ma miejsce po osiągnięciu granic przedziału $[-u_{\max}^{\mathrm{POST}}, u_{\max}^{\mathrm{POST}}]$. Przy nasyceniu, na wyjściu pojawi się sterowanie o wartości bezwzględnej $u_{\max}^{\mathrm{POST}}$ (reguła 9; **wyjście**), a przy braku nasycenia – sterowanie z przedziału $[-u_{\max}^{\mathrm{POST}}, u_{\max}^{\mathrm{POST}}]$ (reguła 10; **wyjście**).

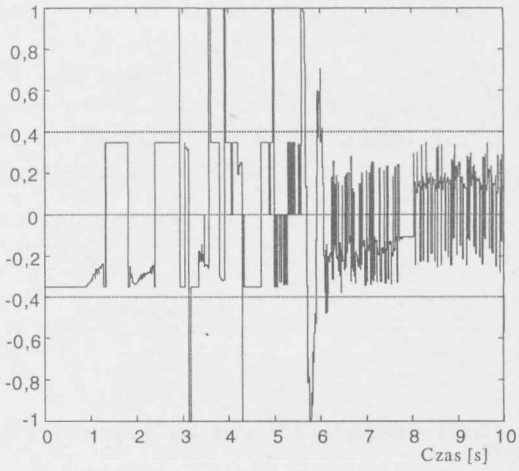
5.2. Eksperymenty symulacyjne i rzeczywiste

Na rysunkach 5.3–5.8 przedstawiono wyniki symulacji prowadzonej na modelu i wyniki eksperymentu sterowania regułowego w rzeczywistym systemie, dla laboratoryjnego modelu II z rysunku 3.1. Algorytm realizuje zadanie POST i po osiągnięciu strefy stabilizacji zadanie STAB. Dla symulacji i sterowania rzeczywistego przyjęto identyczny zestaw parametrów regulacji.

Eksperyment rzeczywisty, jak widać z rysunków, zaczyna się z opóźnieniem około 0,8 sekundy w stosunku do chwili rozpoczęcia zbierania sygnałów. Synchronizacja tych dwóch zdarzeń czasowych lub przeskalowanie osi czasu jest zabiegiem czysto technicznym. Dla uwidocznienia różnicy pomiędzy zbieraniem danych symulacyjnych i rzeczywistych przebiegów zdecydowano się pozostawić dane w oryginalnej formie, w jakiej były zebrane podczas eksperymentu.



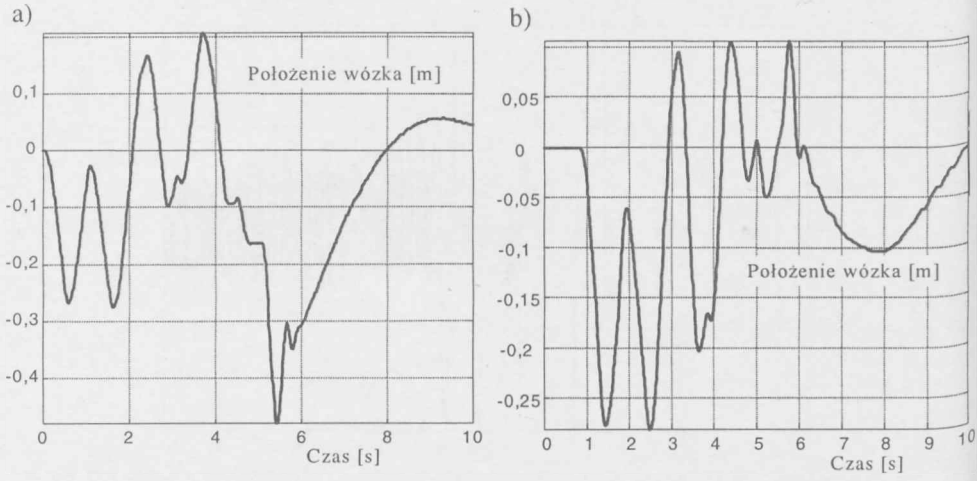
Rys. 5.3. Sterowanie znormalizowane – symulacja ($\alpha = 0{,}8$, $\beta = 0{,}6$, $\gamma = 0{,}2$)



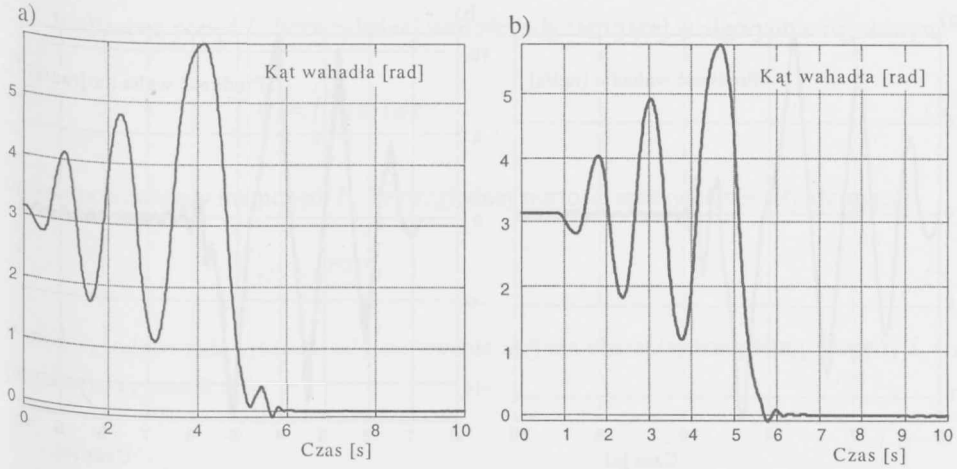
Rys. 5.4. Sterowanie znormalizowane – eksperyment rzeczywisty ($\alpha = 0{,}8$, $\beta = 0{,}6$, $\gamma = 0{,}2$)

Na rysunkach 5.3 i 5.4 można zaobserwować, kiedy sterowanie znormalizowane u przyjmuje wartości z zakresu $[-1, 1]$, a kiedy z zakresu $[-0{,}35, 0{,}35]$. Wartość 1 odpowiada sile $u_{\max}^{STAB}$, a wartość 0,35 sile $u_{\max}^{POST}$.

Położenia wózka (rys. 5.5) i kątowe położenia wahadła (rys. 5.6) pokazano poniżej. Rysunki z lewej strony dotyczą symulacji na modelu, a rysunki z prawej – eksperymentu w systemie rzeczywistym.

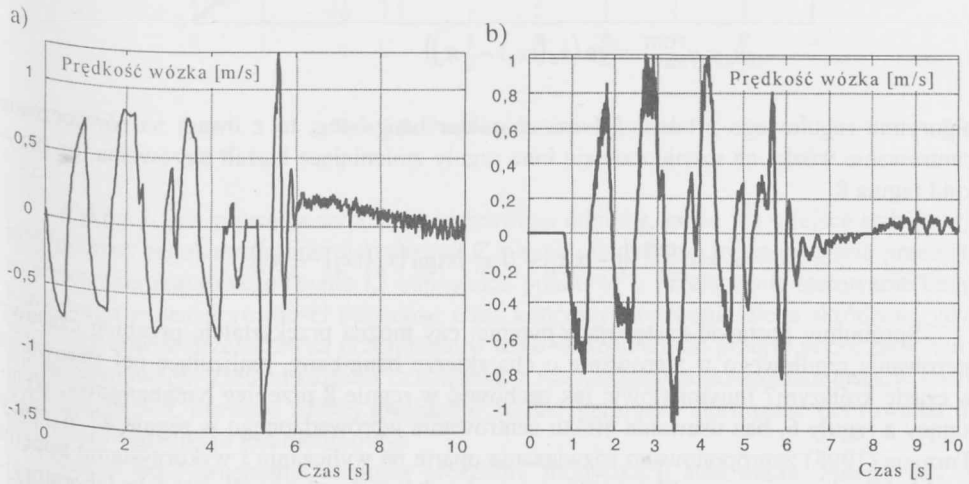


Rys. 5.5. Położenie wózka ($\alpha = 0{,}8$, $\beta = 0{,}6$, $\gamma = 0{,}2$): a) symulacja; b) eksperyment rzeczywisty

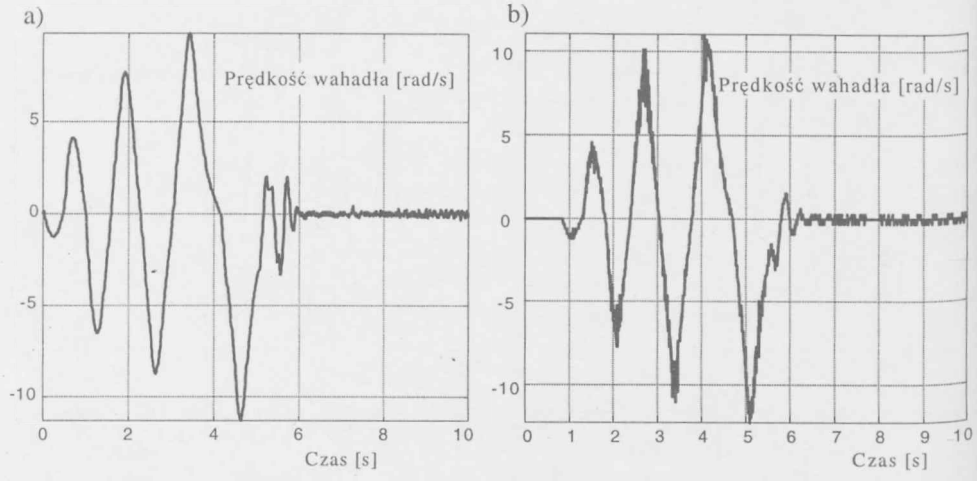


Rys. 5.6. Kąt wahadła ($\alpha = 0{,}8$, $\beta = 0{,}6$, $\gamma = 0{,}2$): a) symulacja; b) eksperyment rzeczywisty

Uwidacznia się duże podobieństwo symulacyjnych i rzeczywistych przebiegów kąta wahadła – w przybliżeniu ten sam czas osiągnięcia strefy stabilizacji, taka sama liczba oscylacji i ta sama amplituda kolejnych oscylacji. Przebiegi położeń wózka są podobne przez pierwsze cztery sekundy, później występują różnice, które nie są znaczące dla osiągnięcia celu sterowania. Można to lepiej zauważyć na rysunkach 5.7 i 5.8, na których zachowując ten sam układ przy prezentacji wyników symulacji i eksperymentu sterowania systemem, pokazano przebiegi prędkości wózka i prędkości kątowej wahadła. Prędkości wózka w obu przypadkach osiągają otoczenie zera około szóstej sekundy licząc od początku sterowania (około siódmej sekundy na rysunku z prawej, z uwagi na opóźniony o sekundę start).



Rys. 5.7. Prędkość wózka ($\alpha = 0{,}8$, $\beta = 0{,}6$, $\gamma = 0{,}2$): a) symulacja; b) eksperyment rzeczywisty



Rys. 5.8. Prędkość kątowa wahadła ($\alpha = 0{,}8$, $\beta = 0{,}6$, $\gamma = 0{,}2$):
a) symulacja; b) eksperyment rzeczywisty

Z zasady maksimum wiadomo, że sterowanie optymalnoczasowe jest bangbangowe (jeśli nie wystąpi osobliwość). Mając algorytm regułowy o sprawdzonym działaniu, zarówno w modelu symulacyjnym, jak i w rzeczywistym systemie, można podjąć próbę skrócenia czasu osiągnięcia celu. Obserwacja przebiegu czasowego sterowania wskazuje na rezerwy niewykorzystanego czasu. Przebieg nie ma charakteru bang-bang (zob. rysunki 5.3 i 5.4), typowego dla sterowań optymalnoczasowych, a więc przy przeprowadzeniu systemu z jednego do drugiego punktu przestrzeni stanu nie wykorzystuje się pełnej mocy sterownika. Chociaż wyjściowa reguła 6

$$u_r = u_{\max}^{\mathrm{POST}} \operatorname{sign}\left(x_4\left(|x_2| - \tfrac{1}{2}\pi\right)\right)$$

algorytmu regułowego z tabeli 5.1 ma charakter bang-bang, to z uwagi na równoczesne centrowanie wózka na szynie działają inne reguły zmieniające kształt sterowania, na przykład reguła 8

$$u_r = u_{\max}^{\mathrm{POST}} \left(1 - \alpha|x_1| - \beta x_3^2\right) \operatorname{sign}\left(x_4\left(|x_2| - \tfrac{1}{2}\pi\right)\right).$$

Spróbujmy postawić następujące pytanie: czy można przekształcić przebieg czasowy sterowania regułowego w sterowanie o charakterze bang-bang, realizujące cel sterowania w czasie krótszym? Innymi słowy: jak zachować w regule 8 przebieg bangbangowy, wynikający z reguły 6, bez usuwania efektu centrowania wprowadzonego w regule 8? W pracy Turnaua (1995) zaproponowano rozwiązanie oparte na wyliczaniu i wykorzystaniu popędu w układzie dynamicznym. Eksperyment symulacyjny prowadzono dla modelu laboratoryjnego I (zob. tab. 3.1).

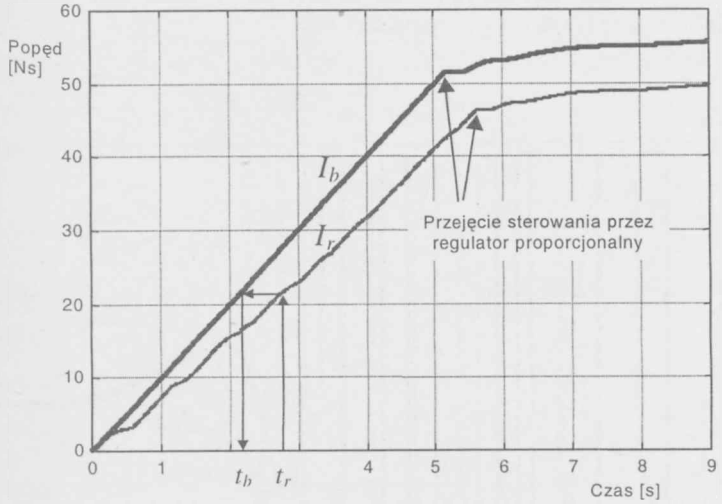
Wyliczono popęd I_r bezwzględnej wartości siły sterującej w algorytmie regułowym

$$I_r(t) = \int_0^t |u(\tau)|\, d\tau \tag{5.1}$$

oraz odpowiadający mu popęd I_b bezwzględnej wartości maksymalnej siły sterującej

$$I_b(t) = u_{\max}^{\mathrm{POST}} t \tag{5.2}$$

Popęd I_b odpowiada sytuacji, gdy sterowanie $u(t)$ ma charakter bang-bang. Popędy I_r i I_b przedstawia rysunek 5.9.



Rys. 5.9. Przebiegi czasowe popędów

Wykres I_b jest prostą (z wyjątkiem końcowego odcinka, gdzie ma miejsce stabilizacja przy pomocy regulatora proporcjonalnego). Z rysunku widać, że chwila przejścia przez zero sterowania regułowego (punkt t_r) odpowiada punktowi t_b przełączenia sterowania bangbangowego w sensie równości popędów. Czas końcowy sterowania ulega skróceniu, gdyż wszystkie chwile przełączeń są przesunięte w lewo w stosunku do przejść przez zero sterowania regułowego. Zmiana kształtu sterowania, z równoczesnym skróceniem czasów kolejnych przełączeń, nie daje żadnej gwarancji, że tak otrzymane sterowanie doprowadzi do celu (sterowany układ jest nieliniowy).

Należy więc uruchomić etapową procedurę modyfikacji sterowania. Polega ona na przesuwaniu kolejnych czasów przejść sterowania przez zero, poczynając od pierwszego. Procedurę generacji sterowania bangbangowego zawarto w tabeli 5.2.

Tabela 5.2
Procedura generacji sterowania bangbangowego

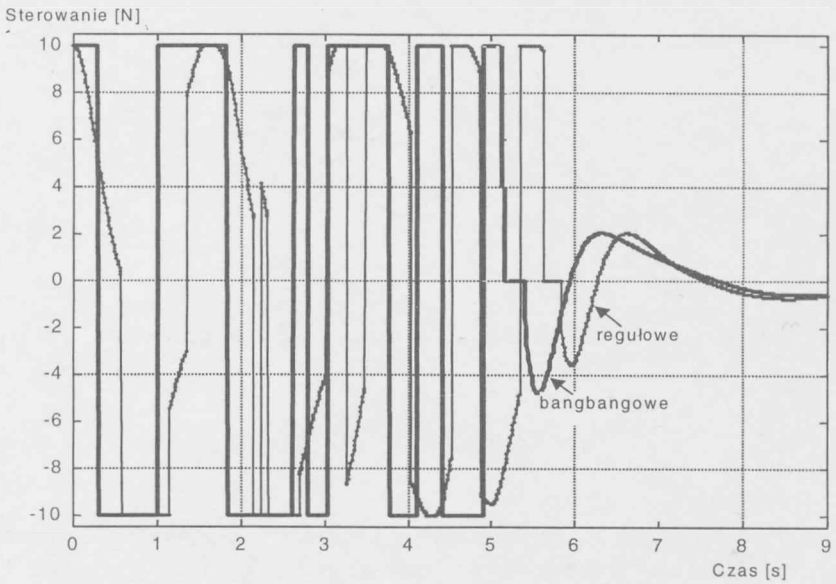
Podstaw $t_b^0 = 0, \quad i = 1.$

1. Wyznacz I_r^i jako sklejenie dwóch sterowań: bangbangowego ($\pm u_{\max}^{\text{POST}}$) w przedziale $]0, t_b^{i-1}[$ i regułowego w przedziale $]t_b^{i-1}, t_r^i[$,

$$t_b^i = \frac{I_r^i(t_r^i)}{u_{\max}^{\text{POST}}}.$$

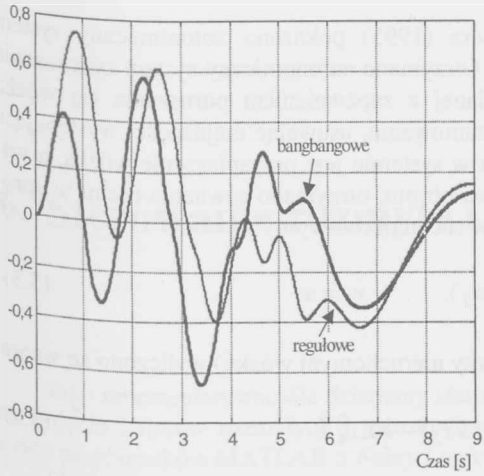
2. Czy osiągnięto cel sterowania?
Tak $\longrightarrow$ koniec, nie $\longrightarrow$ $i = i+1$; idź do 1.

Sterowania regułowe i bangbangowe (powstałe ze sterowania regułowego po zastosowaniu procedury modyfikacyjnej) przedstawia rysunek 5.10.

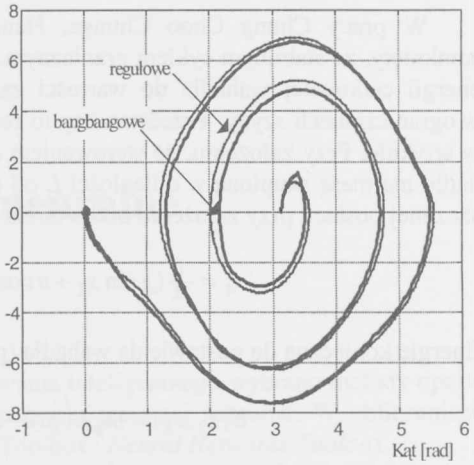


Rys. 5.10. Sterowanie regułowe i uzyskane z niego sterowanie bangbangowe

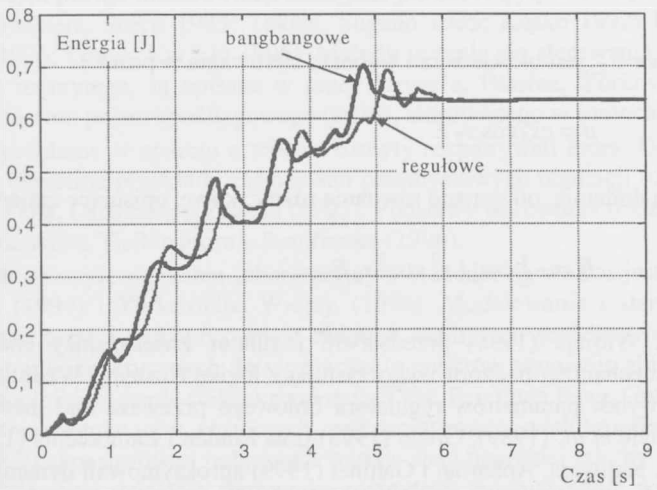
Przeprowadzony eksperyment symulacyjny potwierdza, że istnieje sterowanie o charakterze bang-bang realizujące w krótszym czasie cel sterowania. Położenie wózka na szynie, dla obu sterowań, ilustruje rysunek 5.11. Płaszczyznę fazową wahadła przedstawia rysunek 5.12. Całkowitą energię wahadła pokazano na rysunku 5.13. Wzrost energii wahadła w funkcji czasu nie jest monotoniczny.



Rys. 5.11. Położenie wózka [m]



Rys. 5.12. Prędkość kątowa wahadła [rad/s]



Rys. 5.13. Całkowita energia wahadła [J]

5.3. Sterowanie na podstawie zależności energetycznych

W licznych przykładach spotykanych w literaturze reguły sterowania wahadłem i wózkiem otrzymuje się z obserwacji zmian energii kinetycznej i potencjalnej układu. W pracach Furuty, Ochai, Ono (1984) i Furuty, Kaiwary, Kosugi (1988) podano eksperymentalne wyniki rozwiązania zadania POST. Heurystyczny algorytm energetyczny, z użyciem logiki rozmytej, przedstawiono w pracy Turnaua (1994).

W pracy Chung Choo Chunga, Hausera (1995) pokazano autonomiczny system zamknięty, ze stabilnym cyklem granicznym. Otrzymano autonomiczny system zwiększania energii całkowitej wahadła do wartości zadanej z zapewnieniem poruszania się wózka w ograniczeniach szyny. Przedstawimy to rozumowanie, usuwając niejasności występujące w artykule. Przy założeniu, że sterowaniem u w systemie jest przyspieszenie wózka, a wahadło ma masę skupioną w odległości L od osi obrotu, otrzymano równania ruchu w uproszczonej postaci, przy założeniu braku tarcia w ruchu obrotowym

$$\dot{x}_4 = \frac{1}{2L}(g \sin x_2 + u \cos x_2), \qquad \dot{x}_3 = u \tag{5.3}$$

Energię konieczną do postawienia wahadła (przy nieruchomym wózku) wyliczono ze wzoru

$$E(x_2, x_4) = m_p L\left(g(1-\cos x_2) - \frac{1}{2} L x_4^2\right) \tag{5.4}$$

Łatwo zauważyć, że ma miejsce zależność

$$\dot{E} = \frac{1}{2} m_p L x_4 (g \sin x_2 - u \cos x_2) \tag{5.5}$$

Przy założeniu, że x_2 jest bliskie zera i wyborze sterowania w postaci

$$u = c x_4 \cos x_2 E \tag{5.6}$$

gdzie c jest stałą dodatnią, otrzymano równanie różniczkowe, opisujące zmianę energii

$$\dot{E} = -\frac{1}{2} c m_p L x_4^2 \cos^2 x_2 E \tag{5.7}$$

Yurkovich, Widjaja (1996) przedstawili regulator zwiększający energię wahadła, wzbogacony o mechanizm nadzoru wykorzystujący logikę rozmytą. Według Gevy, Sitte'a (1993) losowy wybór parametrów regulatora liniowego przeważa nad metodami uczenia bez nadzoru. Doyle *et al.* (1989), Green (1993) oraz Linden i Lambrechts (1993) stosowali sterowanie H_∞. Kajiwara, Apkarian i Gahinet (1999) aproksymowali dynamikę systemu za pomocą liniowego modelu o zmiennych parametrach, wykonując liniową cząstkową transformację parametrów lub analizę modelu o parametrach zawartych w wieloboku. Badanie zgodności modelu z systemem opisali Chen i Smith (1998). Hauser i Meyer (1998) pokazali, że sprzężenie zwrotne, tworzące operator rzutowania trajektorii, pozwala na stabilną pracę modelu, mimo zmiany jego parametrów.

6. Sterowanie rozmyte i neuralne

Jako reprezentatywne dla dziedziny sterowania inteligentnego wybrano metody oparte na pojęciu zbiorów rozmytych i metody wykorzystujące sieci neuralne. W obliczeniach użyto przyborników MATLAB-a *Fuzzy Logic Toolbox* i *Neural Networks Toolbox*.

Techniki konstrukcji regulatorów z użyciem logiki rozmytej znajdują coraz częstsze zastosowania. Zbiory rozmyte wprowadził Zadeh (1965). Pierwsze ich zastosowania w automatyce miały miejsce nieco później. W wielu pracach starano się przede wszystkim wyjaśnić, na czym polega modelowanie rozmyte i jak może być wykorzystane do sterowania układami (Maiers, Sherif 1985; Takagi, Sugeno 1985; Kosko 1992; Johansson 1994; Chuang, Lee 1996; Lewis, Kai Liu 1996). Metody uczenia się sterowania, z modelem odniesienia typu rozmytego, są opisane w pracy Layne'a, Passina, Yurkovicha (1993) dla przypadku regulatora przeciwpoślizgowego (ABS), stosowanego w samochodach.

Podobne problemy w oparciu o model rozmyty rozpatrywali Palm, Driankov, Hellendoorn (1996). Obszerne przykłady zastosowań przemysłowych regulacji rozmytej znajdują się w książce Yena, Langariego, Zadeha (1995). Podstawy sterowania rozmytego wyjaśniono w pracy Driankova, Hellendoorna, Reinfranka (1996).

Sterowanie rozmyte modelem laboratoryjnym wahadła na wózku jest opisane w pracach Turnaua (1994) i Yurkovicha, Widjaji (1996). Modelowanie i sterowanie odporne układami niepewnymi z wykorzystaniem zbiorów rozmytych przedstawia Ghalia (1997). Użycie algorytmów genetycznych do wyznaczania punktów skupienia zbiorów rozmytych dla sterowania silnikiem prądu stałego proponują Tzes, Pei-Yuan Peng, Guthy (1998).

Do strojenia regulatorów rozmytych wykorzystywano wielokrotnie sieci neuralne. Chen, Lin (1992) skonstruowali połączenia warstw sieci neuralnej tak, by w nowo powstałej sieci przybliżającej działanie sterownika rozmytego automatycznie wykonywane były etapy rozmywania, wnioskowania i wyostrzania.

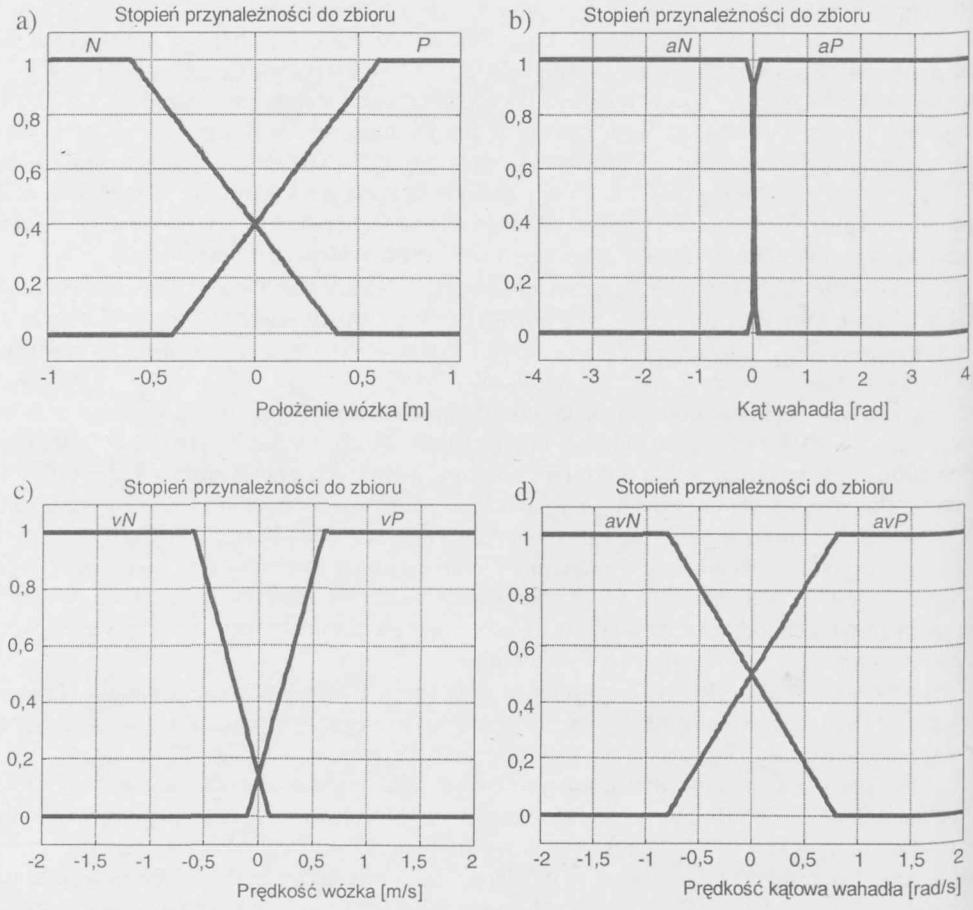
Podstawy użycia sieci neuralnych, między innymi w sterowaniu, podali Nguen, Widrow (1990). Szeroki wykład o sieciach można znaleźć w monografii Tadeusiewicza (1993).

Identyfikacji systemów dynamicznych przez sieć neuralną dotyczy praca Narendry, Parthasarathy (1990). Bogaty przegląd modeli neuralnych do sterowania przedstawiono w pracy Hunta *et al.* (1992). Kołek, Tabakowski, Turnau (1993) zbudowali sterownik neuralny, dla systemu wahadła na wózku, działający jako regulator regułowy. Rozwiązanie to Szymkat, Turnau, Uhl (1995) porównali z neuralną estymacją parametrów dla robota. W pracach Lee, Smytha (1994) i Kempy *et al.* (1997) sieć neuralną stosowano do wygenerowania bangbangowego sterowania optymalnoczasowego.

6.1. Regulatory rozmyte

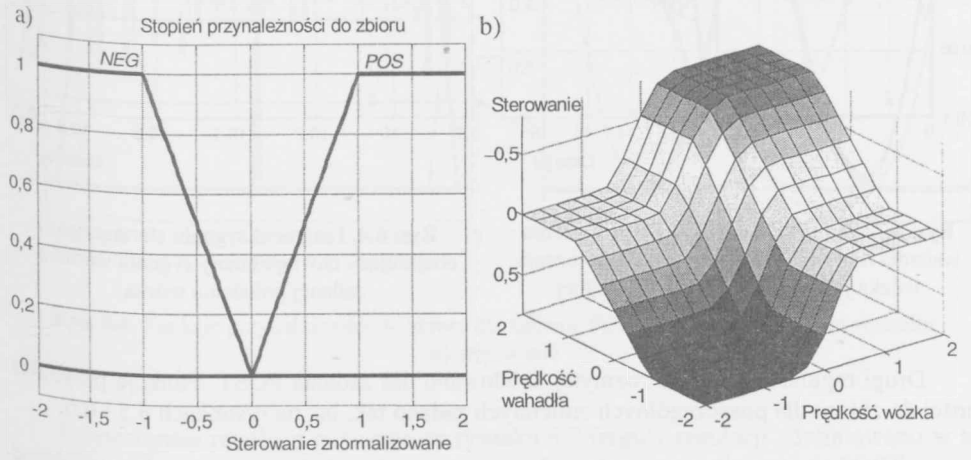
Poniżej pokazano dwa przykłady regulatorów rozmytych dla systemu wahadła na wózku. W obu przypadkach są to klasyczne regulatory rozmyte Mamdaniego, z rozmywaniem sygnałów wejściowych za pomocą funkcji *singleton* i wyostrzaniem wielkości wyjściowych metodą środka ciężkości. Należy zwrócić uwagę, że w obu przykładach zakres zmienności sterowania jest inny niż podany w tabeli 3.1; tutaj jest określony przez przyjęte funkcje przynależności i reguły wyostrzania. Pierwszy regulator rozmyty (rys. 6.1 i 6.2) realizuje w systemie wahadła na wózku zadanie STAB – stabilizacji wahadła w górnym, niestabilnym punkcie równowagi.

Podano cztery funkcje przynależności do zbioru dla czterech rozmytych zmiennych stanu.



Rys. 6.1. Regulator stabilizujący w logice rozmytej – funkcje przynależności zmiennych stanu: a) położenia wózka; b) kąta wahadła; c) prędkości wózka; d) prędkości kątowej wahadła

Funkcje przynależności oznaczono następującymi zmiennymi lingwistycznymi: dla położenia wózka (*position*) zmienne lingwistyczne *N* i *P*, dla prędkości wózka (*velocity*) zmienne lingwistyczne *vN* i *vP*, dla kąta wahadła (*angle*) zmienne *aN* i *aP* oraz dla prędkości kątowej wahadła (*ang-vel*) zmienne lingwistyczne *avN* i *avP*. Funkcje przynależności dla sterowania (*control*) oznaczono przez *NEG* i *POS*. Przyjęto tylko po jednej parze zbiorów dla czterech wejść i jednego wyjścia. Minimalny zestaw składa się z ośmiu reguł sterujących o postaci: jeżeli zmienna wejściowa wskazuje na pewną funkcję przynależności, to wybierz funkcję przynależności sterowania odpowiednią dla tej funkcji przynależności wejścia. Na rysunku 6.2 pokazano jedną z powierzchni regulacji obrazującą sterowanie w funkcji prędkości wózka i wahadła. Powierzchnie dla innych par zmiennych są podobnego kształtu. Regulator zbudowany przy użyciu logiki rozmytej jest układem nieliniowym. W literaturze jest często porównywany do regulatora PID o zmiennych wzmocnieniach.



c)

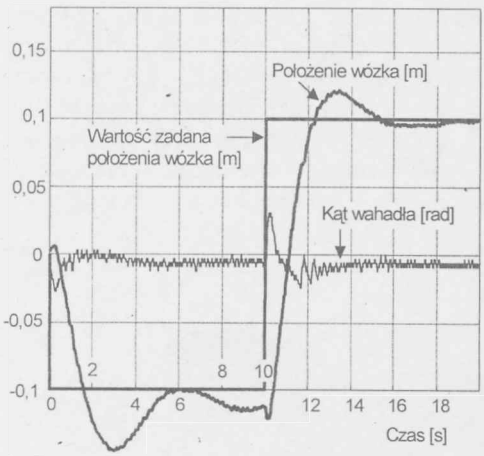
Reguły sterowania

1. If (position is *N*) then (control is *POS*)
2. If (position is *P*) then (control is *NEG*)
3. If (angle is *aN*) then (control is *POS*)
4. If (angle is *aP*) then (control is *NEG*)
5. If (velocity is *vN*) then (control is *POS*)
6. If (velocity is *vP*) then (control is *NEG*)
7. If (ang-vel is *avN*) then (control is *POS*)
8. If (ang-vel is *avP*) then (control is *NEG*)

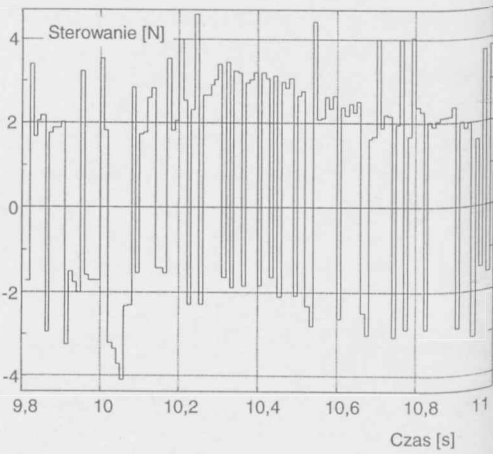
Rys. 6.2. Regulator stabilizujący w logice rozmytej: a) funkcja przynależności sterowania; b) powierzchnia sterowania w funkcji prędkości wózka i wahadła; c) reguły sterowania

Rysunek 6.3 przedstawia odpowiedź układu rzeczywistego wahadła na wózku przy realizacji zadania STAB na sygnał wartości zadanej położenia wózka w formie fali prostokątnej o amplitudzie 0,1 metra. Nagła zmiana wartości zadanej położenia wózka w dziesiątej sekundzie, od wartości –0,1 m do wartości 0,1 m, jest wyraźnie widoczna w przebiegu położenia wózka w czasie. Wahadło, utrzymywane w położeniu górnym, podlega wówczas

największemu zachwianiu równowagi. Warto zwrócić uwagę, że wartości liczbowe próbkowanego sterowania (zob. rys. 6.4) leżą na zewnątrz przedziału $[-F_s, +F_s]$, gdzie F_s oznacza wartość tarcia statycznego wózka, równą 1,5 N.

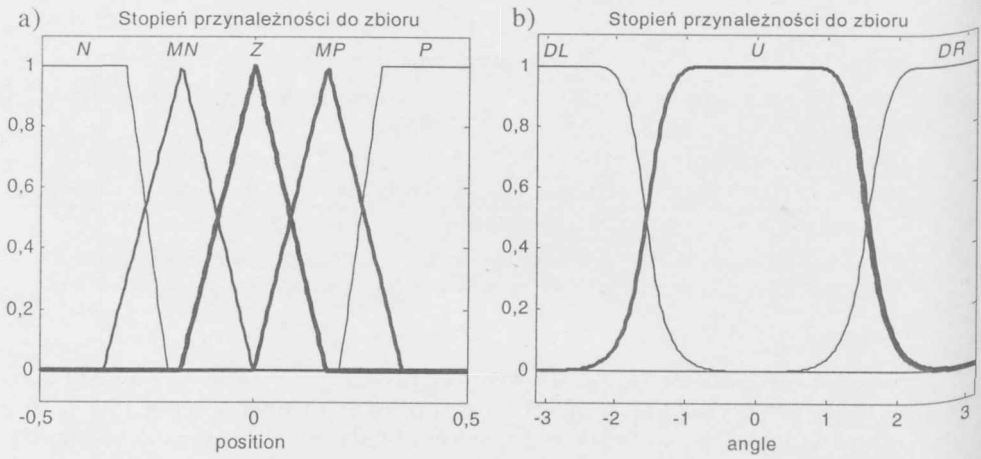


Rys. 6.3. Stabilizacja rzeczywistego systemu wahadła na wózku. Wartość zadana położenia wózka jest sygnałem fali prostokątnej



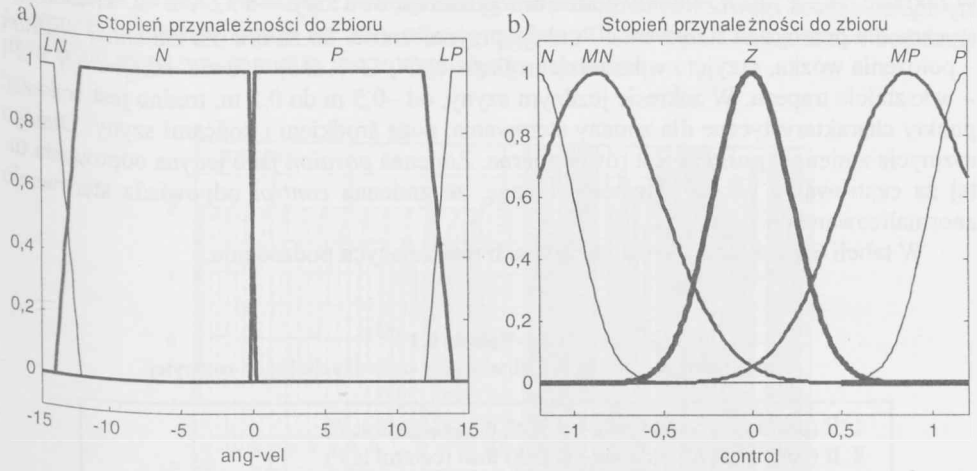
Rys. 6.4. Fragment sygnału sterującego, obejmujący chwilę zmiany sygnału wartości zadanej położenia wózka

Drugi regulator w logice rozmytej zbudowano dla zadania POST. Funkcje przynależności do zbioru dla poszczególnych zmiennych zadano tak, jak na rysunkach 6.5 i 6.6.



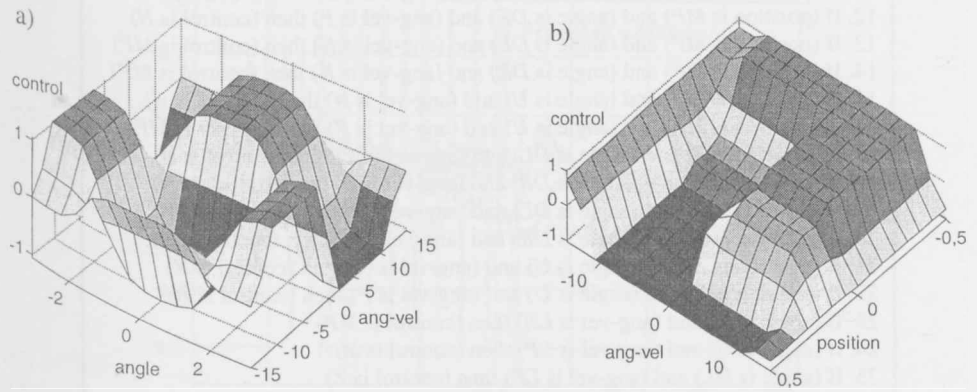
Rys. 6.5. Funkcje przynależności do zbioru dla zadania POST: a) położenie wózka; b) kąt wahadła

Zadanie przeprowadzenia wahadła do górnego niestabilnego położenia równowagi narzuca bardzo złożony kształt powierzchni regulacji. Można je podzielić na podzadania (podobnie jak w przypadku regulacji regułowej z rozdziału 5) i przyporządkowywać im stosowne reguły sterujące. Regulator ma trzy aktywne wejścia i jedno wyjście, są nimi zmienne rozmyte: *position* – położenie wózka, *angle* – kąt wahadła, *ang-vel* – prędkość kątowa wahadła oraz *control* – siła przykładana do wózka.



Rys. 6.6. Funkcje przynależności do zbioru dla zadania POST: a) prędkość kątowa wahadła; b) sterowanie

Powierzchnie regulacji pokazane na rysunku 6.7 (reguły regulacji zdefiniowano w tabeli 6.1) uwidaczniają prawie bangbangowy charakter sterowania dla zadania POST.



Rys. 6.7. Powierzchnie regulacji w zadaniu POST: a) sterowanie w funkcji kąta i prędkości kątowej wahadła; b) sterowanie w funkcji prędkości kątowej wahadła i położenia wózka

Wybór funkcji przynależności nie jest przypadkowy, na ogół łączy się go z wyborem reguł rozmytych sterowania. Jeżeli odwołamy się do heurystycznej reguły zwiększania amplitudy wahań wahadła z regulatora regułowego (tab. 5.1), to zrozumiałe staje się przyjęcie funkcji przynależności do zbioru w postaci przedstawionej na rysunkach 6.5 i 6.6. Przykładowo, dla zmiennej ostrej – kąta wahadła – zdefiniowano trzy zbiory rozmyte o nazwach *DL* (*down left* – dolny w lewo), *U* (*up* – górny) i *DR* (*down right* – dolny w prawo). Podział kąta na trzy zbiory rozmyte wynika z podziału na trzy części kąta pełnego mierzonego w zakresie od $-\pi$ do π. Punkty podziału wyznaczają dwa kąty: $-\pi/2$ i $\pi/2$, charakterystyczne dla przełączeń sterowania. Funkcje przynależności do zbioru dla zmiennej *position* – położenia wózka, przyjęto w kształcie trójkąta (*MN*, *Z* i *MP*) i dwie skrajne (*N* i *P*) – w kształcie trapezu. W zakresie jezdnym szyny, od –0,5 m do 0,5 m, trudno jest wskazać punkty charakterystyczne dla zmiany sterowania, poza środkiem i końcami szyny. Dlatego rozmycie zmiennej *position* jest równomierne. Zmienna *position* jako jedyna odpowiada tutaj za centrowanie wózka. Zwróćmy uwagę, że zmienna *control* odpowiada sterowaniu znormalizowanemu.

W tabeli 6.1 zebrano 28 reguł w grupach realizujących podzadania.

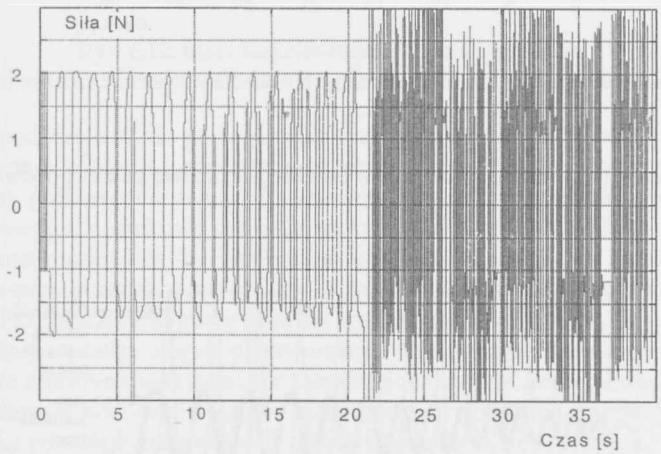
Tabela 6.1

Regulator zadania POST zbudowany przy użyciu logiki rozmytej

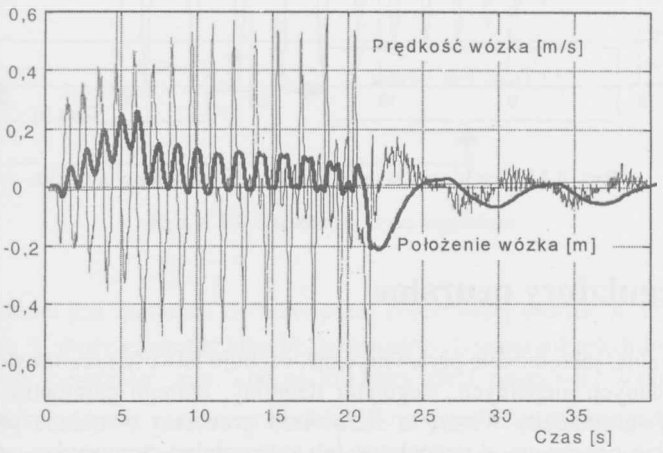
1. If (position is *N*) and (ang-vel is *N*) then (control is *P*)
2. If (position is *N*) and (ang-vel is *P*) then (control is *P*)
3. If (position is *P*) and (ang-vel is *N*) then (control is *N*)
4. If (position is *P*) and (ang-vel is *P*) then (control is *N*)
5. If (position is *MN*) and (angle is *DL*) and (ang-vel is *P*) then (control is *MN*)
6. If (position is *MN*) and (angle is *DR*) and (ang-vel is *P*) then (control is *MN*)
7. If (position is *MN*) and (angle is *DL*) and (ang-vel is *N*) then (control is *P*)
8. If (position is *MN*) and (angle is *DR*) and (ang-vel is *N*) then (control is *P*)
9. If (position is *MN*) and (angle is *U*) and (ang-vel is *N*) then (control is *MN*)
10. If (position is *MN*) and (angle is *U*) and (ang-vel is *P*) then (control is *P*)
11. If (position is *MP*) and (angle is *DL*) and (ang-vel is *P*) then (control is *N*)
12. If (position is *MP*) and (angle is *DR*) and (ang-vel is *P*) then (control is *N*)
13. If (position is *MP*) and (angle is *DL*) and (ang-vel is *N*) then (control is *MP*)
14. If (position is *MP*) and (angle is *DR*) and (ang-vel is *N*) then (control is *MP*)
15. If (position is *MP*) and (angle is *U*) and (ang-vel is *N*) then (control is *N*)
16. If (position is *MP*) and (angle is *U*) and (ang-vel is *P*) then (control is *MP*)
17. If (position is *Z*) and (angle is *DL*) and (ang-vel is *P*) then (control is *N*)
18. If (position is *Z*) and (angle is *DR*) and (ang-vel is *P*) then (control is *N*)
19. If (position is *Z*) and (angle is *DL*) and (ang-vel is *N*) then (control is *P*)
20. If (position is *Z*) and (angle is *DR*) and (ang-vel is *N*) then (control is *P*)
21. If (position is *Z*) and (angle is *U*) and (ang-vel is *N*) then (control is *N*)
22. If (position is *Z*) and (angle is *U*) and (ang-vel is *P*) then (control is *P*)
23. If (angle is *U*) and (ang-vel is *LN*) then (control is *MP*)
24. If (angle is *U*) and (ang-vel is *LP*) then (control is *MN*)
25. If (angle is *DL*) and (ang-vel is *LP*) then (control is *Z*)
26. If (angle is *DR*) and (ang-vel is *LP*) then (control is *Z*)
27. If (angle is *DL*) and (ang-vel is *LN*) then (control is *Z*)
28. If (angle is *DR*) and (ang-vel is *LN*) then (control is *Z*)

Reguły od 1 do 4 mają przeciwdziałać wyjechaniu wózka poza zakres jezdny szyny. Reguły od 5 do 16 służą do zwiększania amplitudy wahań wahadła z równoczesnym sprowadzaniem wózka do środka szyny. Reguły od 17 do 22 wprowadzono tylko dla zwiększania amplitudy wahań wahadła, a reguły od 23 do 28 dodano w celu zapewnienia łagodnego lądowania wahadła w górnej strefie stabilizacji.

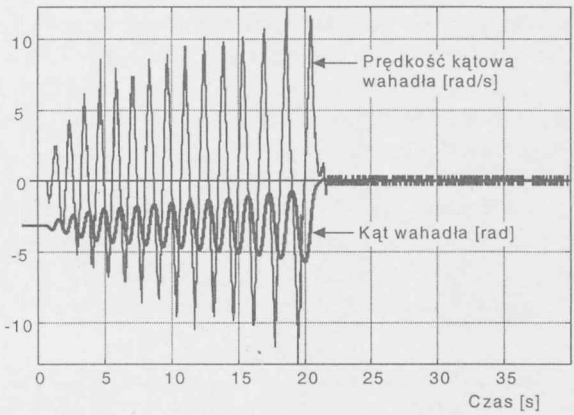
Realizację zadania POST w systemie rzeczywistym pokazano na rysunkach 6.8–6.10. Niewielka amplituda siły sterującej (około 2 N na rysunku 6.8) powoduje wydłużenie procesu dojścia do celu. Po około 22 sekundach górne położenie równowagi zostaje osiągnięte i rozpoczyna się stabilizacja już przy użyciu regulatora liniowego. Dla krótkości będziemy używać takiej nazwy, mimo że jest to regulator z nasyceniem.



Rys. 6.8. Przebieg czasowy sterowania w systemie rzeczywistym

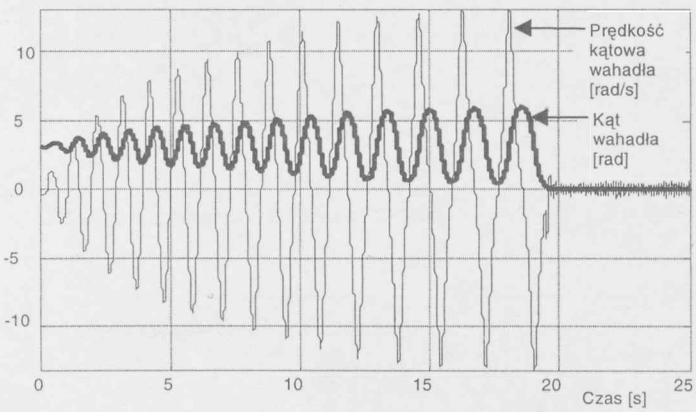


Rys. 6.9. Przebiegi czasowe zmiennych wózka w systemie rzeczywistym



Rys. 6.10. Przebiegi czasowe zmiennych wahadła w systemie rzeczywistym

Na rysunku 6.11 pokazano modelowe przebiegi czasowe celem porównania ich z rzeczywistymi.



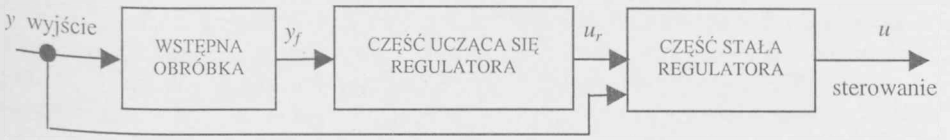
Rys. 6.11. Przebiegi czasowe zmiennych wahadła w modelu

6.2. Regulatory neuralne

Sieci neuralne, jako modele parametryczne, cechuje łatwość *uczenia się* i *generalizacji* na podstawie danych niepełnych. Regulator neuralny, którego parametry (zmienne wagi sięci neuralnej) gromadziły wiedzę o sterowaniu procesem w trakcie procesu uczenia, powinien działać poprawnie ekstrapolując lub interpolując sterowania – *generalizując* – w przypadkach pojawiania się nowych wartości stanów procesu spoza wzorca uczącego.

W regulatorze neuralnym można wyróżnić trzy bloki strukturalne (zob. rys. 6.12), (Sontag 1993):

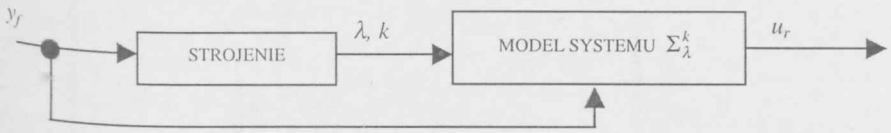
1) blok wstępnej obróbki,
2) część uczącą się,
3) część stałą regulatora.



Rys. 6.12. Bloki funkcjonalne regulatora neuralnego

Adaptacji podczas uczenia podlega tylko *część ucząca się*. Blok *wstępnej obróbki* jest stały dla konkretnej aplikacji i odpowiada za kwantyzację w poziomie sygnałów wyjściowych z systemu, próbkowanie w czasie i konwersję analogowo-cyfrową. W tym bloku są wykonywane również dodatkowe operacje na sygnałach wyjściowych, na przykład wyliczanie transformat numerycznych Fouriera lub wykonywanie procedur symbolicznych (detekcja krawędzi przy przetwarzaniu obrazów, uzyskanie informacji o dopasowaniu sygnału do szablonu). Przetworzone sygnały stają się wejściami części uczącej się. Sygnały z wyjścia części uczącej się także ulegają przetworzeniu, nim staną się sterowaniami. *Część stała regulatora* może reprezentować regulator konwencjonalny lub liniowo-kwadratowy; tu także sygnały podlegają konwersji cyfrowo-analogowej, wygładzeniu, podtrzymaniu itp. Ta część sterownika pozostaje niezmienna w trakcie działania układu regulacji.

W części uczącej się regulatora można odnaleźć strukturę dwuczęściową (rys. 6.13). Składa się ona z bloku strojenia i części reprezentującej dynamikę systemu lub odwzorowanie statyczne.



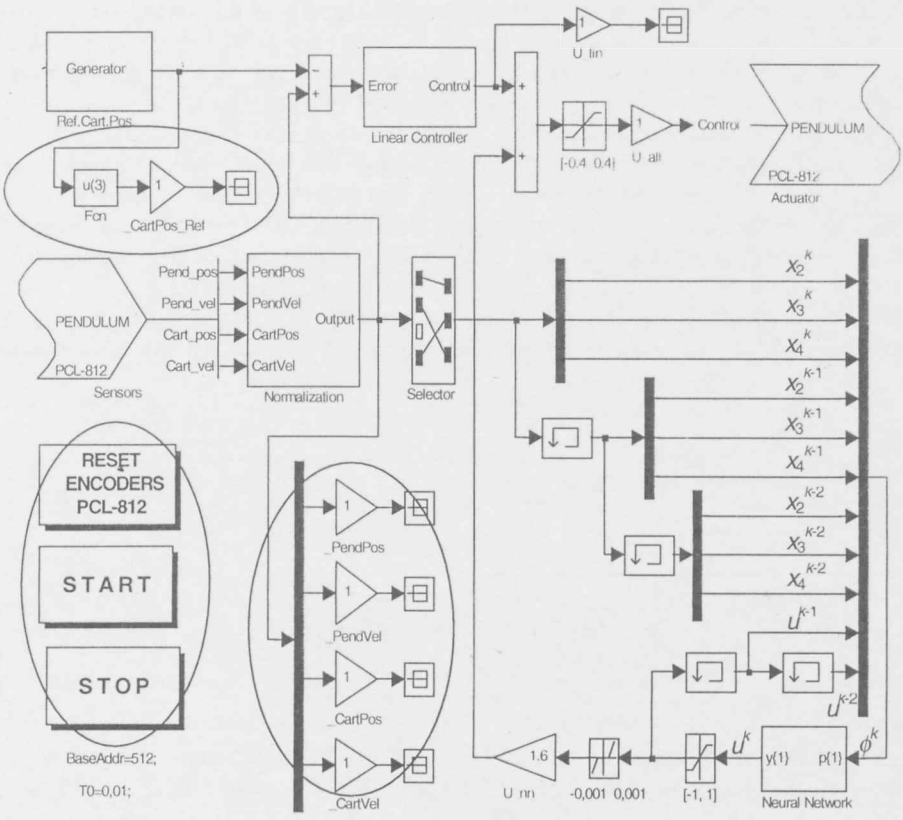
Rys. 6.13. Część ucząca się regulatora

Model systemu jest sparametryzowany przez rzeczywisty wektor λ i dyskretny parametr k. Parametr k może oznaczać albo liczbę zmiennych stanu uwzględnionych w regulatorze, albo liczbę neuronów, albo kod architektury sieci neuralnej. Wektor λ zawiera zmienne parametry poddane uczeniu w modelu. W typowych zastosowaniach Σ_λ^k przedstawia klasę systemów dynamicznych (w identyfikacji) lub statycznych (w rozpoznawaniu obrazów).

Dla systemów dynamicznych używa się modeli zmiennych stanu $\dot{x} = f(x, \lambda, u)$. Funkcja f może być liniową kombinacją zbioru funkcji podstawowych: wielomianów, wielomianów sklejanych z ustalonymi węzłami siatki itp. Zmienne λ są wówczas współczynnikami użytych kombinacji funkcji. Zamiast modeli zmiennych stanu można zastosować sparametryzowane klasy ciągów wejściowo/wyjściowych charakteryzujące dynamiczne zachowanie modeli. Parametry λ odnoszą się wówczas do współczynników transformat.

W przypadku systemu wahadła na wózku dysponujemy dość wiernym modelem i dlatego za pomocą modelu generujemy ciągi wzorcowe przedstawiające działanie liniowego regulatora stabilizującego w zadaniu STAB. Model regułowy (rys. 5.2) jest użyty w poniższym przykładzie do wygenerowania wzorcowych ciągów uczących dla sieci neuralnej. Z modelu wykorzystuje się blok „Dynamika" i regulator liniowy. Sieć jest uczona rozwiązywania zadania STAB.

Na rysunku 6.14 pokazano rzeczywisty regulator stabilizujący z blokiem „Neural Network", który powstaje automatycznie po zakończeniu procesu uczenia sieci.

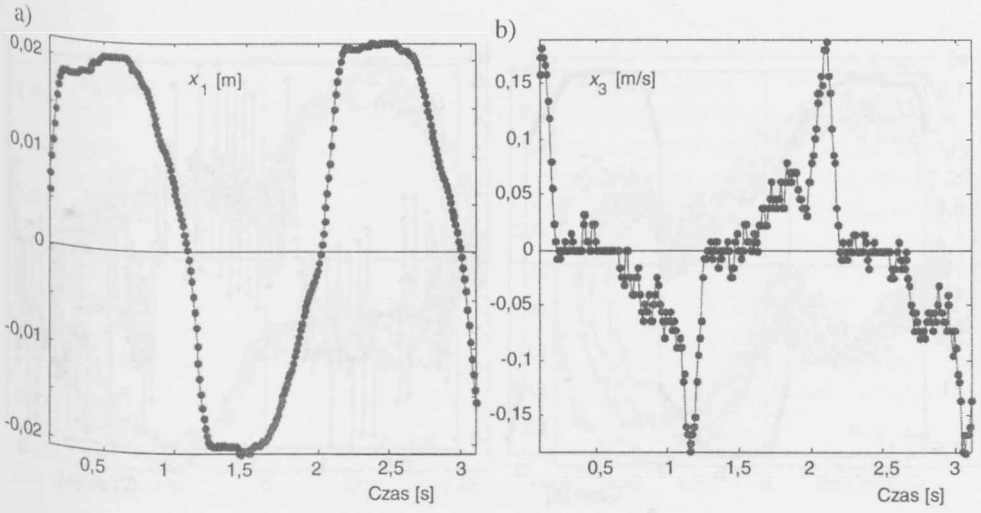


Rys. 6.14. Neuralny regulator stabilizujący dla laboratoryjnego systemu wahadła na wózku

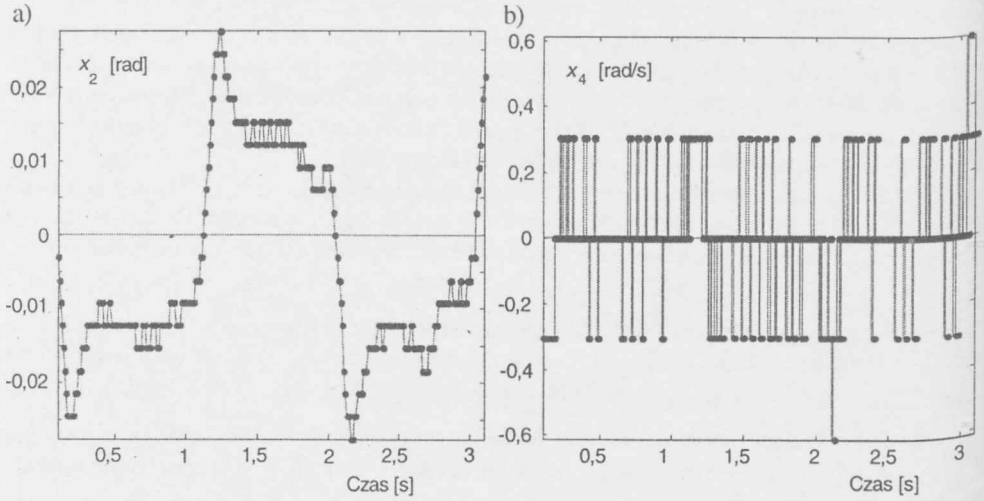
Z systemu laboratoryjnego zbierane są pomiary, a do systemu wysyłane jest sterowanie za pomocą bloków sterowników urządzeń: „Sensors" i „Actuators". Elipsami zakreślono części modelu związane z ewentualną wizualizacją przebiegów oraz części odpowiedzialne za wywołanie lub usunięcie pliku wykonawczego z pamięci komputera i za ustawienie wartości początkowych czujników kąta (zresetowanie enkoderów). Części zaznaczone elipsami nie mają wpływu na przebieg obliczeń algorytmu sterowania.

Przed przystąpieniem do uczenia sieci generuje się ciągi zmiennych stanu i sterowań na modelu, tak by w miarę gęsto pokryć strefę górną stabilizacji wahadła. Do ciągu uczącego wybiera się jedenaście elementów wejściowych, a mianowicie wektor wartości zmiennych stanu i sterowania $\phi^k = \left(x_2^k, x_3^k, x_4^k, x_2^{k-1}, x_3^{k-1}, x_4^{k-1}, x_2^{k-2}, x_3^{k-2}, x_4^{k-2}, u^{k-1}, u^{k-2}\right)$. Podobnie postępowali Lairi, Bloch (1999) oraz Lairi, Bloch, Millerioux (1999), ucząc sieć neuralną stabilizacji kulki lewitującej w polu magnetycznym. Dyskretnym wartościom ciągu, obliczanym w krokach czasowych o numerach $k-2, k-1, k$, odpowiada jeden element wyjściowy sieci neuralnej, którym jest sterowanie u^k obliczone w kroku czasowym k. Charakterystyczne jest pominięcie w ciągu uczącym zmiennej x_1 – położenia wózka. Sieć neuralną definiuje się jako dwuwarstwową. Pierwsza warstwa składa się z 11 neuronów. Na wyjściu pierwszej warstwy stosuje się nieliniową funkcję tangens sigmoidalny *tansig*. Druga warstwa składa się z 1 neuronu, którego wyjście podaje się na blok liniowy *purelin*. Do nauki używa się procedury optymalizacyjnej Levenberga – Marquardta. Sieć uznaje się za nauczoną wzorca, gdy błąd SSE (ang. *sum squared error* – sumaryczny błąd średnio-kwadratowy) nie przekracza 10^{-8}.

Na rysunkach 6.15 i 6.16 pokazano przebiegi czasowe zmiennych stanu systemu zebrane w czasie trwania eksperymentu rzeczywistego. Regulator neuralny realizuje zadanie STAB.

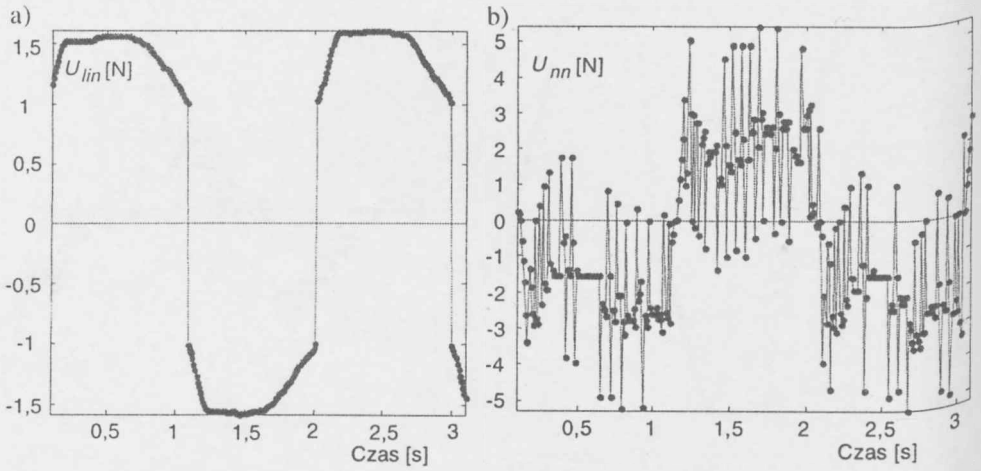


Rys. 6.15. Realizacja zadania STAB w rzeczywistym systemie wahadła na wózku przy użyciu regulatora neuralnego: a) położenie wózka; b) prędkość wózka



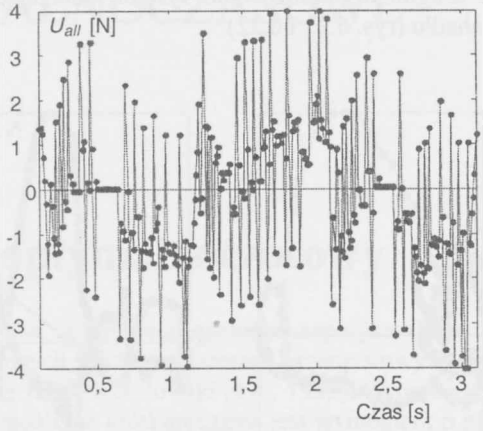
Rys. 6.16. Realizacja zadania STAB w rzeczywistym systemie wahadła na wózku przy użyciu regulatora neuralnego: a) kąt wahadła; b) prędkość kątowa wahadła

Sieć neuralna generuje sterowanie u^k na podstawie podawanego na wejście sieci wektora ϕ^k w dyskretnych chwilach czasu $k\Delta t$ ($\Delta t = 0{,}01$ s). Na rysunku 6.17 po lewej stronie pokazano sterowanie z regulatora liniowego U_{lin}, a po prawej – sterowanie U_{nn} generowane przez sieć neuralną.



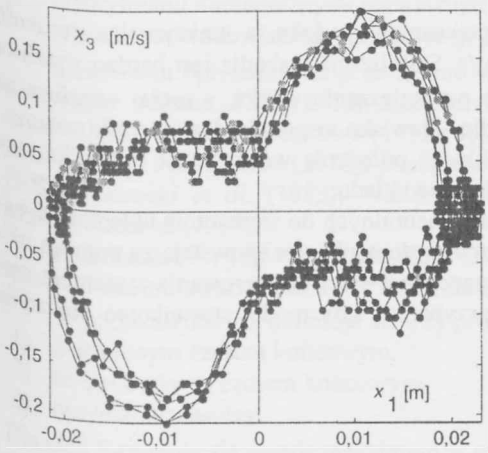
Rys. 6.17. Sterowania w zadaniu STAB:
a) U_{lin} – z regulatora liniowego; b) U_{nn} – generowane przez sieć neuralną

Na rysunku 6.18 przedstawiono wyjściowe sterowanie U_{all} dla zadania STAB będące wynikiem zsumowania sterowań U_{lin} i U_{nn}. Sterowanie U_{lin} jest reakcją na błąd położenia wózka x_1 (por. rys. 6.15a). Sterowanie U_{nn} jest reakcją na błędy pozostałych zmiennych.

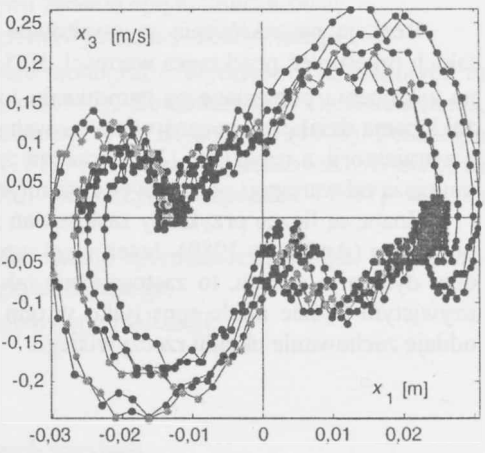


Rys. 6.18. Wyjściowe sterowania w zadaniu STAB U_{all} generowane przez regulator neuralny

Laboratoryjny system wahadła na wózku z neuralnym regulatorem poddano testom. Poza eksperymentem pokazanym niżej na rysunku 6.19, przeprowadzono dwa dodatkowe. W pierwszym zwiększono czterokrotnie masę wahadła. Wyniki przedstawiono na płaszczyźnie fazowej wózka. Można porównać przebiegi z masą nominalną wahadła (rys. 6.19) z przebiegami dla przypadku masy wahadła czterokrotnie zwiększonej (rys. 6.20).



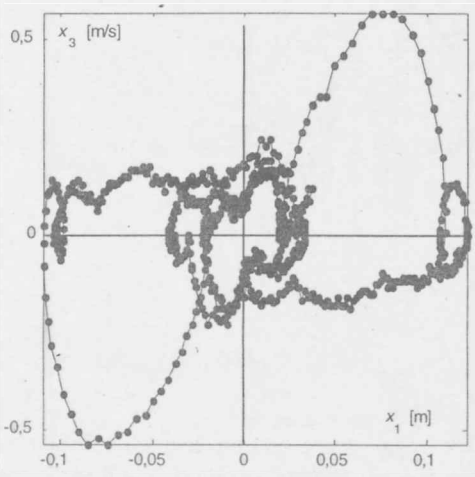
Rys. 6.19. Prędkość wózka x_3 w funkcji położenia wózka x_1; nominalna masa wahadła



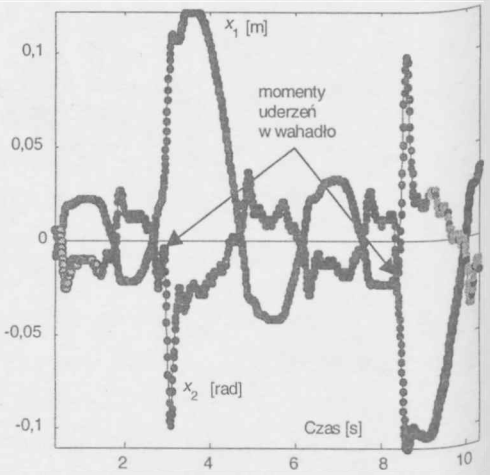
Rys 6.20. Prędkość wózka x_3 w funkcji położenia wózka x_1; czterokrotna masa wahadła

Stabilizator neuralny działa poprawnie, mimo że jest zastosowany w układzie z wahadłem o masie czterokrotnie większej niż ta, dla której był uczony stabilizacji. Na rysunku 6.20 obserwuje się większą amplitudę oscylacji układu podczas stabilizacji, niż dla oryginalnego przebiegu z rysunku 6.19.

W eksperymencie drugim destabilizowano system, uderzając ręką w utrzymywane w górnym położeniu wahadło (rys. 6.21 i 6.22).



Rys. 6.21. Prędkość wózka x_3 w funkcji położenia wózka x_1; wahadło uderzane ręką



Rys. 6.22. Położenie wózka x_1 i kąt wahadła x_2 w funkcji czasu; wahadło uderzane ręką

Reakcją na zakłócenia są gwałtowne przesunięcia wózka, a maksymalna prędkość takich przesunięć przekracza wartości ±0,5 m/s. Stabilizacja wahadła jest bardzo wrażliwa na zakłócenia polegające na hamowaniu lub przyspieszaniu wózka, a mniej wrażliwa na zakłócenia działające bezpośrednio na wahadło. Zjawisko to potwierdzają wyniki zaburzania trajektorii z rozdziału 13. Wariancja zakłóceń położenia wózka musi być kilka razy mniejsza od wariancji pozostałych zmiennych stanu układu.

Znane są liczne przykłady zastosowań sieci neuralnych do sterowania układami dynamicznymi (Anderson 1989). Jeżeli ciągi wzorcowe do nauki sieci powstają za pomocą modelu dynamiki układu, to zastosowanie tak nauczonej sieci do sterowania systemem rzeczywistym będzie miało sens tylko w tym przypadku, gdy model stosunkowo dokładnie oddaje zachowanie układu rzeczywistego.

Część III
STEROWANIE OPTYMALNOCZASOWE

7. Problem optymalnoczasowy

Problemy sterowania optymalnoczasowego napotykamy w wielu dziedzinach inżynierii, badaniach naukowych, także w procesach przemysłowych (Mitkowski, Turnau 1976, 1982; Adamski, Turnau 1998; Korytowski *et al.* 1999). W pewnych przypadkach osiągnięcie celu sterowania w możliwie krótkim czasie jest wymogiem o pierwszorzędnym znaczeniu. Można tu wymienić sterowanie optymalnoczasowe samolotem czy rakietą kosmiczną, reakcją chemiczną, robotem przemysłowym, pozycjonowaniem głowicy czytającej dysku itp. Szybki rozwój techniki komputerowej stwarza realną szansę budowy regulatorów optymalnoczasowych i ich zastosowania w czasie rzeczywistym. Regulator może pracować efektywnie dla układu nieliniowego wysokiego rzędu, jak pokazano to w przypadku trójwymiarowej suwnicy będącej układem rzędu dziesiątego (Pauluk *et al.* 2001).

Do rozwiązania zadania optymalnoczasowego i jego praktycznej realizacji niezbędne jest spełnienie wielu warunków. Należy dysponować:

- realistycznym modelem dynamiki, zgodnym ze sterowanym systemem lub procesem;
- algorytmami numerycznymi rozwiązującymi zadanie optymalnoczasowe;
- właściwym środowiskiem sprzętowo-programowym czasu rzeczywistego.

Sterowaniu optymalnemu poświęcono wiele monografii. Wymieńmy te, z których tu korzystano: Athans, Falb (1966); Lee, Markus (1967); Bołtianski (1971); Kwakernaak, Sivan (1972); Gabasow, Kiriłowa (1973); Rolewicz (1974); Bryson, Ho (1975); Pontriagin *et al.* (1976); Fiedorienko (1978); Sussmann (1979) Findeisen, Szymanowski, Wierzbicki (1980); Górecki *et al.* (1983); Whittle (1996); Jakubczyk, Respondek (1997); Bryson (1999); Fattorini (1999).

Podrozdział 7.1 jest wprowadzeniem do sterowania optymalnoczasowego. Rozpoczyna się od sformułowań zadania i twierdzenia egzystencjalnego.

W podrozdziale 7.2 definiuje się trzy problemy optymalizacji:

1) z ustalonym czasem końcowym,
2) ze swobodnym czasem końcowym,
3) optymalnoczasowy.

Dla nich formułuje się zasadę maksimum w trzech wersjach.

W podrozdziale 7.3 wprowadza się rodzinę problemów pomocniczych i formułuje się twierdzenie dotyczące zadania optymalizacji *ze swobodnym czasem końcowym* i *sparametryzowanym wskaźnikiem jakości*.

7.1. Równanie systemu i własności rozwiązań

Zakładamy, że dynamika obiektu opisana jest następującym równaniem różniczkowym (równaniem stanu)

$$\dot{x}(t) = f(x(t),u(t)) = f^0(x(t)) + f^1(x(t))u(t)$$
$$t \in [0,\Omega], \quad x(0) = x^0 \tag{7.1}$$

gdzie:

$x(t) \in \boldsymbol{R}^n$ – wektor zmiennych stanu,

$\boldsymbol{R}^n$ – przestrzeń stanu,

u, $u(t) \in \boldsymbol{U} \subset \boldsymbol{R}^l$ – sterowanie,

$\boldsymbol{U}$ – zbiór dopuszczalnych wartości sterowania,

$[0,\Omega]$ – przedział czasu, w którym rozpatrujemy rozwiązania równania stanu (dostatecznie duży i ustalany stosownie do naszych potrzeb).

O funkcjach f^0, f^1 i sterowaniu u zakładamy:

$f^0 : \boldsymbol{R}^n \to \boldsymbol{R}^n$ – ciągła i lipschitzowska w całej przestrzeni $\boldsymbol{R}^n$;

$f^1 : \boldsymbol{R}^n \to \boldsymbol{R}^{n\times l}$ – przyjmuje wartości w zbiorze macierzy o wymiarze $n \times l$, których elementy są ciągłe i lipschitzowskie w całej przestrzeni $\boldsymbol{R}^n$;

f^0 i f^1 – klasy $\boldsymbol{C}^1$;

$u : [0,\Omega] \to \boldsymbol{U} \subset \boldsymbol{R}^l$ – funkcja mierzalna.

Dalej zakładamy, że zbiór $\boldsymbol{U}$ jest ograniczony, wypukły, domknięty i ma postać kostki

$$\boldsymbol{U} = [u_{1,\min}, u_{1,\max}] \times [u_{2,\min}, u_{2,\max}] \times \cdots \times [u_{l,\min}, u_{l,\max}] \subset \boldsymbol{R}^l \tag{7.2}$$

gdzie liczby $u_{i,\min} < u_{i,\max}$, $i = 1, ..., l$, reprezentują ograniczenia narzucone na poszczególne składowe wektora $u(t) \in \boldsymbol{U} \subset \boldsymbol{R}^l$.

Oznaczmy przez $\boldsymbol{U}_{\text{ad}}$ zbiór sterowań dopuszczalnych zdefiniowany następująco

$$\boldsymbol{U}_{\text{ad}} = \{u : [0,\Omega] \to \boldsymbol{U},\ u \text{ mierzalne}\}.$$

Przy przyjętych założeniach możemy wypowiedzieć ważne twierdzenie dotyczące istnienia, jednoznaczności i pewnych dodatkowych własności zbioru rozwiązań (trajektorii) równania stanu.

Oznaczmy

$$\nabla_x f(x,u) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_1} \\ \vdots & \vdots & \vdots \\ \frac{\partial f_1}{\partial x_n} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}.$$

Twierdzenie 7.1 (Rolewicz 1974; Pontriagin *et al.* 1976)

Dla każdego warunku początkowego $x^0 \in \boldsymbol{R}^n$ i dla każdego $u \in \boldsymbol{U}_{\text{ad}}$:

- istnieje dokładnie jedno rozwiązanie $x : [0, \Omega] \to \boldsymbol{R}^n$, $x(0) = x^0$ równania stanu (7.1), będące funkcją absolutnie ciągłą i ograniczoną;
- rozwiązanie to, jako wartość odwzorowania $\boldsymbol{R}^n \times \boldsymbol{U}_{\text{ad}} \ni (x^0, u) \mapsto x \in \boldsymbol{C}([0,\Omega];\boldsymbol{R}^n)$, zależy w sposób ciągły od warunku początkowego $x^0 \in \boldsymbol{R}^n$ i sterowania $u \in \boldsymbol{U}_{\text{ad}}$, z metryką w zbiorze $\boldsymbol{U}_{\text{ad}}$ dyktowaną przez dowolną z norm $\|\cdot\|_{L^p}$, $1 \le p \le \infty$, i standardową metryką typu supremum w zbiorze $\boldsymbol{C}([0,\Omega];\boldsymbol{R}^n)$;
- rozwiązanie to jest różniczkowalne ze względu na warunek początkowy x^0 i sterowanie u, z metrykami w odpowiednich zbiorach jak powyżej;
- pochodna trajektorii x względem pary (x^0,u), brana w punkcie (x_0^0,u_0), określona jest relacją (różniczką zupełną)

$$\Delta x(t) = \Phi(t,0)\Delta x^0 + \int_0^t \Phi(t,\tau) f^1(x_0(\tau))\Delta u(\tau) d\tau \tag{7.3}$$

gdzie $\Phi(t,\tau)$, $t,\tau \in [0,\Omega]$, jest macierzowym rozwiązaniem fundamentalnym liniowego równania różniczkowego ze względu na ω

$$\dot{\omega}(t) = (\nabla_x f(x_0(t),u_0(t)))^{\mathsf{T}} \omega(t), \quad t \in [0,\Omega] \tag{7.4}$$

a x_0 oznacza trajektorię wygenerowaną przez sterowanie u_0 ze stanu początkowego x_0^0;

- zbiory $\boldsymbol{C}_t \subset \boldsymbol{R}^n$, $0 \le t \le \Omega$, stanów $x(t)$, osiągalnych w chwili t ze stanu początkowego x^0

$$\boldsymbol{C}_t = \{ y \in \boldsymbol{R}^n : y = x(t),\ u \in \boldsymbol{U}_{\text{ad}},\ x(0) = x^0 \}$$

są ograniczone i domknięte;

– funkcja $[0,\Omega] \ni t \mapsto \boldsymbol{C}_t$ jest ciągła i lipschitzowska w sensie metryki Hausdorffa d_H

$$d_H(\boldsymbol{C}_{t_1}, \boldsymbol{C}_{t_2}) \leq M_H |t_2 - t_1|, \quad t_1, t_2 \in [0, \Omega],$$

z pewną stałą Lipschitza M_H, $M_H \geq 0$. ■

7.2. Problemy sterowania optymalnego i zasada maksimum

Sformułujemy trzy problemy sterowania optymalnego, a następnie podamy warunki konieczne optymalności, nazywane tradycyjnie zasadą maksimum.

Pierwszy problem optymalizacji, *z ustalonym czasem końcowym*, polega na poszukiwaniu minimum funkcjonału o postaci

$$\Sigma_T(u) = F_1(x(T)) \tag{7.5}$$

gdzie:

T – ustalona chwila czasu, $T \in [0,\Omega]$,

F_1 – dana funkcja $\boldsymbol{R}^n \to \boldsymbol{R}^1$,

x – rozwiązanie równania (7.1) wygenerowane przez sterowanie u.

Szukamy sterowania u^T, $u^T \in \boldsymbol{U}_{\text{ad}}$, takiego że

$$u^T = \arg\min_{u \in \boldsymbol{U}_{\text{ad}}} \Sigma_T(u).$$

Wartość minimalną funkcjonału Σ_T oznaczamy przez

$$\Sigma_T^* = \min_{u \in \boldsymbol{U}_{\text{ad}}} \Sigma_T(u).$$

Drugi problem optymalizacji, *ze swobodnym czasem końcowym*, polega na poszukiwaniu minimum funkcjonału

$$S(u,T) = F_2(x(T),T) \tag{7.6}$$

gdzie:

F_2 – funkcja $\boldsymbol{R}^n \times [0,\Omega] \to \boldsymbol{R}^1$,

x – trajektoria stanu systemu (7.1) wygenerowana przez sterowanie u.

Szukamy teraz pary (u^*,T^*) należącej do $U_{ad}\times[0,\Omega]$, takiej że

$$(u^*,T^*) = \arg\min_{(u,T)\in U_{ad}\times[0,\Omega]} S(u,T).$$

Wartość minimalną funkcjonału S oznaczamy następująco

$$S^* = \min_{(u,T)\in U_{ad}\times[0,\Omega]} S(u,T).$$

Trzeci problem optymalizacji to tak zwany problem *optymalnoczasowy*. Jego sformułowanie wymaga nieco dłuższego omówienia. Przez $x(t,u)$ oznaczmy stan w chwili t wygenerowany przez sterowanie u. Dla zadanego wektora $x^f \in \boldsymbol{R}^n$, nazywanego *stanem docelowym*, i sterowania $u\in U_{ad}$ definiujemy zbiór

$$\Theta(u) = \{t\in[0,\Omega]: x(t,u) = x^f\}.$$

Definiujemy wskaźnik jakości $S: U_{ad} \to \boldsymbol{R}^1$ (czas osiągnięcia celu)

$$S(u) = \begin{cases} \min\Theta(u), & \text{gdy } \Theta(u)\neq\varnothing \\ +\infty, & \text{gdy } \Theta(u)=\varnothing \end{cases} \tag{7.7}$$

Sterowanie optymalnoczasowe u^* minimalizuje tak określony wskaźnik jakości. Odpowiedni czas sterowania $t^* = S(u^*)$ nazywamy horyzontem optymalnym. Trajektoria stanu $x^* = x(\cdot,u^*)$ jest trajektorią optymalną.

Podamy teraz warunek konieczny optymalności dla problemu pierwszego. Jest to podstawowa wersja zasady maksimum – dla zadania z ustalonym czasem sterowania.

Twierdzenie 7.2 (Gabasow, Kiriłowa 1973)

Załóżmy, że dla układu dynamicznego (7.1) istnieje sterowanie optymalne u^T, minimalizujące funkcjonał (7.5). Odpowiadającą mu trajektorię optymalną oznaczamy przez x^T. Niech F_1 będzie funkcją różniczkowalną w sposób ciągły. Przy tych założeniach spełniona jest relacja

$$(\psi^f)^{\mathrm{T}}\Phi(T,t)f^1(x^T(t))u^T(t) \geq (\psi^f)^{\mathrm{T}}\Phi(T,t)f^1(x^T(t))v \tag{7.8}$$

dla prawie wszystkich $t\in[0,T]$ i dla wszystkich $v\in U$, gdzie $\Phi(\cdot,\cdot)$ jest rozwiązaniem fundamentalnym równania (7.4), określonym dla trajektorii x^T i sterowania u^T, a wektor ψ^f jest dany przez

$$\psi^f = -\nabla_x F_1(x^T(T)) \ \blacksquare \tag{7.9}$$

Wprowadza się *funkcję sprzężoną* ψ będącą rozwiązaniem liniowego *równania sprzężonego*

$$\dot{\psi}(t) = -\nabla_x f(x(t,u),u(t))\psi(t), \quad t \in [0,\Omega] \tag{7.10}$$

z warunkiem końcowym

$$\psi(T) = -\nabla_x F_1(x(T,u)) \tag{7.11}$$

Będziemy również równanie (7.11) zapisywać w postaci

$$\dot{\psi}(t) = A(x,u)\psi(t), \quad t \in [0,\Omega] \tag{7.12}$$

gdzie

$$A(x,u) = -\nabla_x f(x,u) \tag{7.13}$$

Jeżeli para (x,u) jest optymalna, czyli $x = x^T$, $u = u^T$, to odpowiadającą jej funkcję sprzężoną oznaczymy przez $\psi^T(t)$, otrzymując

$$\psi^T(t) = \Phi(T,t)^{\mathsf{T}}\psi^f, \quad t \in [0,T] \tag{7.14}$$

z warunkiem końcowym

$$\psi^T(T) = -\nabla_x F_1(x^T(T)) \tag{7.15}$$

Definiujemy *funkcję przełączającą*

$$\phi(t) = f^1(x(t))^{\mathsf{T}}\psi(t) \tag{7.16}$$

i *optymalną funkcję przełączającą*

$$\phi^T(t) = f^1(x^T(t))^{\mathsf{T}}\psi^T(t) \tag{7.17}$$

Z relacji (7.8) wynika, dla $k = 1,\ldots,l$ i $t \in [0,T]$

$$u_k^T(t) = \begin{cases} u_{k,\max}, & \text{gdy } \phi_k^T(t) > 0 \\ u_{k,\min}, & \text{gdy } \phi_k^T(t) < 0 \end{cases} \tag{7.18}$$

gdzie $\phi_k^T(t)$ oznacza k-tą składową optymalnej funkcji przełączającej (7.17).

Nierówność (7.8) zapisuje się często w innej postaci

$$H(x^T(t),\psi^T(t),u^T(t)) \geq H(x^T(t),\psi^T(t),v) \qquad \forall v \in \boldsymbol{U},\ \text{p.w. } t \in [0,T] \tag{7.19}$$

gdzie funkcja $H(x,\psi,v)$, zdefiniowana przez

$$H(x,\psi,v) = \psi^{\mathsf{T}} f(x,v) \tag{7.20}$$

jest nazywana hamiltonianem.

Przy założeniu, że żadna z funkcji ϕ_k^T nie zeruje się tożsamościowo na zbiorze o niezerowej mierze, zdefiniujmy funkcję $w: \boldsymbol{R}^n \times \boldsymbol{R}^n \to \boldsymbol{R}^l$

$$w_k(x,\psi) = \begin{cases} u_{k,\max}, & \text{gdy } f^{1,k}(x)^{\mathsf{T}}\psi > 0 \\ u_{k,\min}, & \text{gdy } f^{1,k}(x)^{\mathsf{T}}\psi < 0 \end{cases}, \quad k = 1,\ldots,l \tag{7.21}$$

gdzie $f^{1,k}(x)$ oznacza k-tą kolumnę macierzy $f^1(x)$.

Sterowanie określone przez (7.18) spełnia równość $u^T(t) = w(x^T(t),\psi^T(t))$. Podstawienie funkcji (7.21) do równania stanu i równania sprzężonego daje tak zwany *układ kanoniczny*:

$$\begin{aligned} \dot{x} &= f(x, w(x,\psi)) \\ \dot{\psi} &= A(x, w(x,\psi))\,\psi \end{aligned} \tag{7.22}$$

z warunkami brzegowymi jak w (7.1) i (7.11).

Zauważmy, że relacje (7.8) i (7.18) określają sterowanie optymalne u^T w tych chwilach, dla których wszystkie składowe funkcji ϕ^T są różne od zera.

Jeśli któraś z funkcji ϕ_k^T zeruje się na zbiorze o dodatniej mierze, warunek (7.18) nie pozwala określić odpowiedniej składowej sterowania w tym zbiorze chwil czasu – taki przypadek nazywamy *osobliwym*. Składowa u_k^T przyjmuje wtedy zazwyczaj wartości pośrednie między $u_{k,\min}$ a $u_{k,\max}$. Wynika stąd, że jeżeli nie wystąpi osobliwość, to sterowanie optymalne leży na ograniczeniach – czyli ma charakter bang-bang. Punkty, w których jedna ze składowych funkcji przełączającej zmienia znak, nazywamy *czasami przełączeń* (Górecki, Turowicz 1968).

W drugim problemie, ze swobodnym czasem końcowym, sterowanie optymalne określone jest przez (7.18) – podobnie jak poprzednio. Dla tego problemu podamy zasadę maksimum w następującej postaci.

Twierdzenie 7.3 (Gabasow, Kiriłowa 1973)

Jeśli para (T^*, u^*) i odpowiadająca jej trajektoria x^* są optymalne dla układu dynamicznego (7.1) i funkcjonału (7.6), gdzie F_2 jest funkcją różniczkowalną w sposób ciągły, to spełnione są relacje

$$(\psi^f)^{\mathrm{T}} \Phi(T^*, t) f^1(x^*(t)) u^*(t) \geq (\psi^f)^{\mathrm{T}} \Phi(T^*, t) f^1(x^*(t)) v \tag{7.23}$$

$$-(\psi^f)^{\mathrm{T}} f(x^*(T^*), v) + \nabla_T F_2(x^*(T^*), T^*) \geq 0 \tag{7.24}$$

dla prawie wszystkich $t \in [0, T^*]$ i dla wszystkich $v \in \boldsymbol{U}$, gdzie $\Phi(\cdot,\cdot)$ jest rozwiązaniem fundamentalnym równania (7.4), określonym dla trajektorii x^* i sterowania u^*, oraz wektor ψ^f jest dany przez

$$\psi^f = -\nabla_x F_2(x^*(T^*), T^*) \blacksquare \tag{7.25}$$

Dla trzeciego problemu optymalizacji, nazwanego optymalnoczasowym, mamy następujące twierdzenie.

Twierdzenie 7.4 (Bołtianski 1971)

Dla układu i problemu optymalnoczasowego określonych uprzednio:

- jeśli istnieje choć jedna para $(t, u) \in [0, \Omega] \times U_{\mathrm{ad}}$, taka że $x(0, u) = x^0$ i $x(t, u) = x^f$, to istnieje również para optymalna (t^*, u^*);
- jeśli trójka (t^*, x^*, u^*) jest optymalna, to istnieje niezerowy wektor $\psi^f \in \boldsymbol{R}^n$, taki że zachodzi warunek (7.23), z podstawieniem t^* za T^*, dla prawie wszystkich $t \in [0, t^*]$ i dla wszystkich $v \in \boldsymbol{U}$, gdzie $\Phi(\cdot,\cdot)$ jest rozwiązaniem fundamentalnym równania (7.4), określonym dla trajektorii x^* i sterowania u^*. ■

Jeżeli para (x, u) jest optymalna, to znaczy $x = x^*$, $u = u^*$, to odpowiadającą jej funkcję sprzężoną oznaczymy przez $\psi^*(t)$, otrzymując

$$\psi^*(t) = \Phi(t^*, t)^{\mathrm{T}} \psi^f, \quad t \in [0, t^*] \tag{7.26}$$

z warunkiem końcowym

$$\psi(t^*) = \psi^f \tag{7.27}$$

Optymalna *funkcja przełączająca* jest postaci

$$\phi^*(t) = f^1(x^*(t))^{\mathrm{T}} \psi^*(t) \tag{7.28}$$

Z relacji (7.19) wynika, dla $k = 1,..., l$ i $t \in [0, t^*]$

$$u_k^*(t) = \begin{cases} u_{k,\max}, & \text{gdy } \phi_k^*(t) > 0 \\ u_{k,\min}, & \text{gdy } \phi_k^*(t) < 0 \end{cases} \tag{7.29}$$

gdzie $\phi_k^*(t)$ oznacza k-tą składową optymalnej funkcji przełączającej (7.28).

Nierówność (7.23) zapisuje się często jako warunek maksimum hamiltonianu (7.20)

$$H(x^*(t), \psi^*(t), u^*(t)) \geq H(x^*(t), \psi^*(t), v(t)) \ \forall v \in \boldsymbol{U}, \text{ p.w. } t \in [0, T].$$

7.3. Równoważne ciągi zadań sparametryzowanych

Bezpośrednie wyznaczenie sterowania optymalnoczasowego na podstawie zasady maksimum, czyli z relacji (7.26)–(7.29), jest trudne, głównie z uwagi na to, że nie są z góry znane ani warunki brzegowe na rozwiązanie równania sprzężonego, ani optymalny czas t^*. W związku z tym proponuje się zastąpienie oryginalnego problemu optymalnoczasowego ciągiem zadań pomocniczych. Opiszemy dwa takie podejścia. W pierwszym z nich definiujemy rodzinę P_T problemów ze stałym horyzontem, sparametryzowaną przez horyzont T, $0 \leq T \leq \Omega$. W każdym z nich należy zminimalizować wskaźnik jakości

$$\Sigma_T(u) = \tfrac{1}{2}(x(T) - x^f)^{\mathsf{T}} Q(x(T) - x^f), \quad Q = Q^{\mathsf{T}} > 0 \tag{7.30}$$

gdzie x jest trajektorią wygenerowaną przez sterowanie u, o wartości początkowej x^0.

Oznaczmy przez u^T sterowanie optymalne dla problemu P_T, a przez x^T odpowiadającą mu trajektorię stanu. Oczywiste są związki między oryginalnym problemem optymalnoczasowym a rodziną problemów pomocniczych P_T:

$$\begin{aligned} t^* &= \min\{T \in [0, \Omega] : \Sigma_T(u^T) = 0\} \\ u^* &= u^{t^*} \end{aligned} \tag{7.31}$$

Warunek konieczny optymalności sterowania dla problemu P_T jest natychmiastowym wnioskiem z twierdzenia 7.2.

Twierdzenie 7.5

Rozważmy problem P_T. Jeśli u^T jest sterowaniem optymalnym, x^T – odpowiadającą mu trajektorią stanu, a ψ^T – odpowiednią trajektorią sprzężoną spełniającą warunek końcowy $\psi^T(T) = Q(x^f - x^T(T))$, to dla prawie wszystkich $t \in [0, T]$ zachodzi warunek maksimum hamiltonianu (7.19). ■

Funkcjonał Σ_T jest różniczkowalny. Dla problemu P_T mamy mianowicie

$$\Sigma_T(u+\Delta u)-\Sigma_T(u)=-\int_0^T g(x(t),\psi(t))^{\mathsf{T}}\Delta u(t)\mathrm{d}t+o(\Delta u) \tag{7.32}$$

gdzie x i ψ są rozwiązaniami równania stanu i równania sprzężonego wygenerowanymi przez sterowanie u, z odpowiednimi warunkami brzegowymi. Funkcjonał o, określony na przestrzeni funkcji całkowalnych w kwadracie, jest resztą rzędu wyższego niż pierwszy. Funkcja

$$g(x(t),\psi(t))=\nabla_u H\big|_t=f^1(x(t))^{\mathsf{T}}\psi(t),\quad t\in[0,T] \tag{7.33}$$

nazywana jest *antygradientem*, *funkcją przełączającą* lub *kierunkiem największego spadku*. Wzór (7.32) określa pochodną funkcjonału Σ_T względem sterowania u, traktowanego jako element przestrzeni L_2. Warunek maksimum hamiltonianu można zapisać w postaci

$$u_k(t)=\begin{cases}u_{k,\max}, & \text{gdy } g_k(x(t),\psi(t))>0\\ u_{k,\min}, & \text{gdy } g_k(x(t),\psi(t))<0\end{cases},\quad t\in[0,T],\ k=1,\ldots,l \tag{7.34}$$

Dalej ograniczamy się do przypadku $l=1$. Wówczas zbiór dopuszczalnych wartości sterowania sprowadza się do postaci

$$\boldsymbol{U}=[u_{\min},u_{\max}] \tag{7.35}$$

a wzór (7.34) do

$$u(t)=\begin{cases}u_{\max}, & \text{gdy } g(x(t),\psi(t))>0\\ u_{\min}, & \text{gdy } g(x(t),\psi(t))<0\end{cases},\quad t\in[0,T] \tag{7.36}$$

Rzut antygradientu g na zbiór dopuszczalny $\boldsymbol{U}$ w punkcie u definiuje się następująco

$$g^U(t)=\begin{cases}0, & u(t)=\frac{1}{2}(u_{\max}+u_{\min})+\frac{1}{2}(u_{\max}-u_{\min})\operatorname{sgn} g(x(t),\psi(t))\\ g(x(t),\psi(t)), & \text{poza tym}\end{cases} \tag{7.37}$$

Zgodnie z zasadą maksimum, g^U zeruje się tożsamościowo na sterowaniu optymalnym. Sterowanie ekstremalne (7.36) maksymalizuje hamiltonian (7.20). Po wprowadzeniu oznaczeń

$$X=\begin{bmatrix}x\\ \psi\end{bmatrix},\qquad F(X,u)=\begin{bmatrix}f(x,u)\\ A(x,u)\psi\end{bmatrix} \tag{7.38}$$

równania kanoniczne przybiorą postać

$$\dot{X} = F(X,u), \quad t \in [0,T] \tag{7.39}$$

z warunkami brzegowymi

$$x(0) = x^0, \quad \psi(T) = Q(x^f - x(T)) \tag{7.40}$$

i warunkiem optymalności w postaci (7.36). Zakłada się, że funkcja $[0,T] \ni t \mapsto g(X(t))$ zeruje się w co najwyżej skończonej liczbie punktów $0 < \tau_1 < \tau_2 < \ldots < \tau_m < T$ (nazywanych czasami przełączeń). Sterowanie ma więc charakter bang-bang.

W drugim podejściu do wyznaczenia sterowania optymalnoczasowego, przeszukania osi T dokonuje się inaczej. Definiujemy rozszerzony problem optymalizacji, w którym horyzont jest zmienną decyzyjną, ze wskaźnikiem jakości

$$S_q(u,T) = \Sigma_T(u) + qT \tag{7.41}$$

gdzie q jest nieujemnym parametrem. Poszukiwanie rozwiązania optymalnoczasowego sprowadza się do ciągu minimalizacji wskaźników (7.41) z odpowiednio zmienianym parametrem q. Niech dla dowolnego $q > 0$ para (u_q^*, T_q^*) minimalizuje S_q. Procedura oparta jest na spostrzeżeniu, że $T_q^* \to t^*$ przy $q \to 0^+$, a ponadto u^* jest punktem skupienia zbioru $\{u_q^* : q > 0\}$ według normy w L_2.

Formułujemy więc czwarty problem optymalizacyjny, *ze swobodnym czasem końcowym i sparametryzowanym wskaźnikiem jakości*, podobny do drugiego problemu (7.6)

$$S_q(u,T) = F_3(x(T), q, T) \tag{7.42}$$

gdzie F_3 jest funkcją $\boldsymbol{R}^n \times [0,\infty[\times [0,\Omega] \to \boldsymbol{R}^1$. Szukamy teraz pary (u^*, T^*) należącej do $U_{\text{ad}} \times [0,\Omega]$, takiej że

$$(u^*, T^*) = \lim_{q \to 0^+} \operatorname*{arg\,min}_{(u,T) \in U_{\text{ad}} \times [0,\Omega]} S_q(u,T) \tag{7.43}$$

Wartość minimalną funkcjonału S_q oznaczamy przez

$$S_q^* = \min_{(u,T) \in U_{\text{ad}} \times [0,\Omega]} S_q(u,T) \tag{7.44}$$

Na koniec zdefiniujmy regulator optymalny w problemie optymalnoczasowym jako funkcję $R_{\text{opt}} : \boldsymbol{R}^n \to \boldsymbol{R}^l$, taką że dla każdego sterowania optymalnego u^* i odpowiadającej mu trajektorii optymalnej x^* zachodzi prawie wszędzie

$$u^*(t) = R_{\text{opt}}(x^*(t)) \tag{7.45}$$

8. Metody numeryczne rozwiązywania problemu optymalnoczasowego

Rozwiązanie problemu optymalnoczasowego jest zadaniem złożonym, teoretycznie trudnym i uciążliwym obliczeniowo, i dlatego wymaga posłużenia się algorytmami numerycznymi, korzystającymi z nowych wyników teorii sterowania optymalnego i programowania nieliniowego. Funkcjonały pomocnicze, minimalizowane podczas optymalizacji, są na ogół niewypukłe i mogą mieć wiele minimów lokalnych, które spowalniają proces znajdowania minimum globalnego i często utrudniają jego osiągnięcie, gdy po drodze procedura grzęźnie w którymś z minimów lokalnych. Mimo przeszkód związanych z szybkością i zbieżnością algorytmów rozwiązywania problemów optymalnoczasowych, odnotowuje się wzrost zainteresowania praktycznym wykorzystaniem regulacji optymalnej, co można wytłumaczyć rozwojem możliwości technicznych, przede wszystkim zwiększeniem szybkości procesorów i zrównolegleniem ich pracy. Pojawiają się doniesienia o rozwiązaniach problemów sterowania optymalnoczasowego w czasie rzeczywistym (Steinbach, Bock, Longman 1995; Longman *et al.* 1997).

W typowych podejściach do numerycznego rozwiązywania problemów optymalnoczasowych stosuje się zarówno *bezpośrednie*, jak i *pośrednie* metody obliczeniowe sterowania optymalnego (Stoer, Bulirsch 1987; von Stryk, Bulirsch 1992). Najczęściej używa się wersji przeznaczonych dla zadań ze swobodnym horyzontem; konstruuje się też równoważne ciągi problemów optymalizacji z ustalonym horyzontem.

Metody bezpośrednie polegają na aproksymacji oryginalnego problemu sterowania optymalnego problemem programowania matematycznego o skończonym wymiarze, otrzymanym drogą dyskretyzacji sterowań i równań dynamiki systemu (Bless, Hodges, Seywald 1995), np. przez aproksymację wielomianową w przedziałach. Często stosuje się metodę *strzałów wielokrotnych* lub *kolokacji* (Bock, Plitt 1984; Hargraves, Paris 1987; von Stryk 1993; Andrews 1996; Conway, Larson 1998), a powstałe w ten sposób zadanie optymalizacji rozwiązuje się za pomocą *sekwencyjnego programowania kwadratowego* SQP (von Stryk 1993; Kraft 1994; Hull 1997; Barclay, Gill, Rosen 1998). Metody bezpośrednie nie korzystają z zasady maksimum, są na ogół wolno zbieżne, ale mają duży zakres zbieżności. Ich zaletą jest uniwersalność, pozwalająca w jednolity, stosunkowo prosty sposób traktować różnorodne problemy sterowania.

Metody pośrednie wykorzystują warunki konieczne optymalności w oryginalnej przestrzeni sterowań. Zwykle polegają na rozwiązaniu dwugranicznego układu równań, wynikającego z zasady maksimum Pontriagina (Pontriagin *et al.* 1976). Warunki brzegowe problemu oryginalnego zostają uzupełnione równościami i nierównościami wynikłymi z uwzględnienia warunków koniecznych optymalności. Najskuteczniejszymi narzędziami

obliczeniowymi służącymi do tego celu są *metoda strzałów wielokrotnych* (Pesch 1989a; Lastman 1978) i *metoda kolokacyjna* (Kierzenka, Shampine 1999; Alamir 2000). Wymagają one dobrych punktów startowych – szczególnie ważne jest, aby początkowe przybliżenie trajektorii sprzężonej poprawnie określało strukturę sterowania (ilość przełączeń i wartość początkową sterowania). Takie przybliżenia początkowe uzyskuje się bądź metodami bezpośrednimi (Sontag, Sussmann 1993), bądź przez homotopijne przejście od odpowiednio dobranego problemu, który jest łatwy do rozwiązania (Pesch 1989a). Do metod pośrednich zalicza się również gradientowe metody optymalizacji numerycznej, jeśli gradient wskaźnika jakości jest wyznaczany w oparciu o rozwiązania równania sprzężonego. Wadą podejścia pośredniego jest duży nakład pracy analitycznej podczas przygotowania problemu do obliczeń, stosunkowo mały zakres zbieżności oraz duża wrażliwość na błędy. Pośrednie metody strzałów wielokrotnych i kolokacyjne wykazują jednak bardzo szybką zbieżność w pobliżu rozwiązania optymalnego.

W praktyce metody bezpośrednie mogą posłużyć do wyznaczenia wartości początkowych dla procedur pośrednich (Kumar, Seywald 1996). Jeżeli użyjemy wyłącznie metod pośrednich, to po sparametryzowaniu problemu – z horyzontem sterowania jako parametrem – można go rozwiązać dla krótkiego horyzontu. Uzyskana struktura sterowania staje się podpowiedzią dla strategii kontynuacji – generowania sterowań dla stopniowo wydłużanego horyzontu.

8.1. Algorytm gradientów sprzężonych w przestrzeni sterowań

Rozpocznijmy od algorytmu, w którym używa się aproksymacji sterowań funkcjami przedziałami stałymi, czyli tak zwanej aproksymacji *schodkowej*, ze skończoną liczbą zadanych z góry przedziałów stałości, co prowadzi do problemu optymalizacji ze skończenie wymiarową przestrzenią decyzyjną (Korytowski, Szymkat, Turnau 1998). Aproksymację tę wybrano mimo istnienia bardziej efektywnych metod przybliżania sterowań, z uwagi na jej prostotę ilustracyjną i takie zalety, jak łatwość uwzględnienia ograniczeń czy też niskie wymagania odnośnie regularności funkcji.

Weźmy pod uwagę problem sterowania optymalnego ze stałym horyzontem, polegający na minimalizacji wskaźnika jakości (7.5) na trajektoriach systemu (7.1) ze sterowaniem skalarnym i ograniczeniami (7.35) z $u_{\min} = -u_{\max}$. Antygradient wskaźnika jakości w przestrzeni sterowań wyznacza się z równości (7.33). Załóżmy, że przedział $[0,T]$ został podzielony na p części punktami dyskretyzacji

$$0 = t_0 < t_1 < \ldots < t_p = T \tag{8.1}$$

Na sterowania nakładamy warunek, aby w każdym z przedziałów $\theta_i = [t_{i-1}, t_i)$ były stałe

$$u(t) = \hat{u}_i \quad \forall t \in \theta_i,\ i = 1,\ldots,p \tag{8.2}$$

Przypomnijmy, że zgodnie z (7.35), $\hat{u}_i \in [-u_{\max}, u_{\max}]$. Funkcje czasu spełniające warunek

(8.2) dla ustalonego podziału (8.1) będziemy nazywać *schodkowymi*. Każde sterowanie schodkowe może być utożsamione z wektorem $\hat{u} = \mathrm{col}(\hat{u}_1,...,\hat{u}_p)$ w przestrzeni $\boldsymbol{R}^p$. Funkcjonał $\Sigma_T(u)$ (7.30) definiuje więc funkcję $\hat{\Sigma}_T : \boldsymbol{R}^p \to \boldsymbol{R}$, taką że $\hat{\Sigma}_T(\hat{u}) = \Sigma_T(u)$ dla każdej pary $(\hat{u}, u)$ spełniającej (8.2). Wstawiając (8.2) do (7.32) otrzymujemy

$$\hat{\Sigma}_T(\hat{u} + \Delta\hat{u}) - \hat{\Sigma}_T(\hat{u}) = \sum_{i=1}^{p} \hat{g}_i \Delta\hat{u}_i + o(\Delta\hat{u}) \tag{8.3}$$

gdzie

$$\hat{g}_i = \int_{\theta_i} g(t)\mathrm{d}t, \quad i = 1,...,p \tag{8.4}$$

a funkcjonał o jest resztą rzędu wyższego niż pierwszy na przestrzeni $\boldsymbol{R}^p$. Antygradientem funkcjonału $\hat{\Sigma}_T$ jest zatem wektor $\hat{g} = \mathrm{col}(\hat{g}_1,...,\hat{g}_p)$, określony przez (8.4) i (7.33). Stwierdzenie to pozostaje prawdziwe w przypadku bardziej ogólnym, kiedy $\theta_i, i = 1,...,p$, są dowolnymi rozłącznymi podprzedziałami przedziału $[0,T]$ i sterowanie spełnia warunek (8.2). Przyjmując dyskretyzację określoną wzorami (8.1) i (8.2), oznaczmy przez $\bar{\Delta}$ rzut antygradientu $\hat{g}$ na zbiór dopuszczalnych sterowań w punkcie $\hat{u}$, w którym antygradient został obliczony

$$\bar{\Delta}_i = \begin{cases} 0, & \text{gdy } \hat{u}_i \operatorname{sgn} \hat{g}_i = u_{\max} \\ \hat{g}_i & \text{poza tym} \end{cases}, \quad i = 1,...,p \tag{8.5}$$

Punktem wyjścia w j-tej iteracji metody gradientowej $(j = 1, 2,...)$ jest schodkowe sterowanie dopuszczalne $u^{(j)}$ i jego dyskretna reprezentacja $\hat{u}^{(j)}$, spełniające (8.2). Sterowaniu temu odpowiada trajektoria stanu $x^{(j)}$, będąca rozwiązaniem równania (7.1), oraz trajektoria sprzężona $\psi^{(j)}$ spełniająca (7.10). Z zależności (7.33), (8.4) i (8.5) wyznacza się wektor $\bar{\Delta}^{(j)}$ – rzut antygradientu $\hat{g}^{(j)}$ na zbiór dopuszczalny w punkcie $\hat{u}^{(j)}$. Ponieważ $\bar{\Delta}^{(j)} = 0$, jeśli $u^{(j)}$ jest sterowaniem optymalnym, podstawowym warunkiem zatrzymania algorytmu jest spełnienie nierówności

$$\left\| \bar{\Delta}^{(j)} \right\| \le \varepsilon \tag{8.6}$$

gdzie ε jest z góry zadaną, dostatecznie małą liczbą nieujemną. Jeżeli kryterium zatrzymania algorytmu nie jest spełnione, wyznacza się *kierunek poszukiwania* minimum $\delta^{(j)} \in \boldsymbol{R}^p$. Robi się to różnie w różnych metodach gradientowych, najprościej w *metodzie*

największego spadku, gdzie $\delta^{(j)} = \bar{\Delta}^{(j)}$. *Gradientowe metody kierunków sprzężonych*, na ogół znacznie szybciej zbieżne, używają tego wzoru tylko w pierwszej iteracji i przy każdym odświeżeniu algorytmu, a także zawsze wtedy, gdy wektor $\delta^{(j)}$ wyznaczony według normalnej procedury nie jest kierunkiem poprawy funkcjonału. We wszystkich innych iteracjach kierunek poszukiwania jest określony wzorem $\delta^{(j)} = \Delta^{(j)} + \beta_j \delta^{(j-1)}$ $(j > 1)$. W *metodzie Polaka – Ribière'a*

$$\beta_j = \frac{\left(\bar{\Delta}^{(j)} - \bar{\Delta}^{(j-1)}\right)^T \bar{\Delta}^{(j)}}{\left\| \bar{\Delta}^{(j-1)} \right\|^2}.$$

W drugiej części iteracji wykonuje się *poszukiwanie w kierunku*, to znaczy minimalizuje się funkcję

$$[0,\infty[\ni \lambda \mapsto \hat{\Sigma}_T\left(\overline{W}(\lambda)\right) \tag{8.7}$$

gdzie $\overline{W}(\lambda)$ jest rzutem wektora $W(\lambda) = \hat{u}^{(j)} + \lambda \delta^{(j)}$ na zbiór dopuszczalny

$$\overline{W}_i(\lambda) = \left\{ \begin{array}{ll} -u_{\max}, & \text{gdy } W_i(\lambda) \le -u_{\max} \\ W_i(\lambda), & \text{gdy } W_i(\lambda) \in [-u_{\max}, u_{\max}] \\ u_{\max}, & \text{gdy } W_i(\lambda) \ge u_{\max} \end{array} \right\}, \quad i = 1,\ldots, p.$$

Zauważmy, że dzięki rzutowaniu, działanie algorytmu nie ulegnie zmianie, jeśli we wzorach zastąpimy rzut antygradientu $\bar{\Delta}^{(j)}$ przez antygradient $\hat{g}^{(j)}$. Rezultatem j-tej iteracji jest sterowanie $\hat{u}^{(j+1)} = W(\lambda_j)$, gdzie λ_j oznacza wartość argumentu, dla której funkcja (8.7) osiąga minimum.

W sytuacji kiedy warunek zatrzymania (8.6) nie jest spełniony, zdarza się, że wektor $\bar{\Delta}$ (8.5) nie jest kierunkiem poprawy funkcjonału albo też przy przyjętej dokładności minimalizacja w kierunku $\bar{\Delta}$ nie przynosi poprawy. Powodem może być nieliniowość problemu lub błąd aproksymacji. Zwiększenie dokładności poszukiwania w kierunku (np. przez zwiększenie liczby kroków kontrakcji) powyżej pewnego progu jest nieopłacalne, bo wtedy jeśli nawet poprawę się uzyska, to jest ona pomijalnie mała. Środkiem zaradczym może być natomiast zmienianie przedziału czasu, zawartego w $[0,T]$, w którym prowadzi się optymalizację sterowania. Odpowiedni wybór przedziałów optymalizacji przyspiesza zbieżność i zmniejsza prawdopodobieństwo, że algorytm zatrzyma się daleko od optimum, a także pozwala uniknąć stosowania bardzo dużej dokładności we wczesnym etapie obliczeń. Ogólnie biorąc, wrażliwość funkcjonału Σ_T na zmiany sterowania rośnie z upływem

czasu. Ten argument, a także wygoda obliczeniowa, przemawia za optymalizacją sterowania w końcowych podprzedziałach przedziału $[0,T]$.

Przedstawimy algorytm, w którym zmiana przedziału optymalizacji jest połączona z zaostrzaniem kryterium zatrzymania metody gradientowej w przestrzeni sterowań. Dla $i=1,...,p$, wprowadzamy oznaczenie $\left\|\bar{\Delta}\right\|_i = \left\|\operatorname{col}\left(\bar{\Delta}_i, \bar{\Delta}_{i+1}, ..., \bar{\Delta}_p\right)\right\|$.

Algorytm 8.1 (algorytm gradientów sprzężonych z aproksymacją schodkową)

Dane:

ε – początkowa wartość progu w kryterium zatrzymania (8.6) (np. $\varepsilon = 10^{-4}h$, gdzie $h = T/p$);

ε_f – końcowa wartość progu (np. $\varepsilon_f = 10^{-7}h$);

ε_0 – minimalna długość kroku w minimalizacji kierunkowej (np. $\varepsilon_0 = 10^{-9}$);

$\rho, \rho' \in (0,1)$ – współczynniki redukcji progów (np. $\rho = \rho' = 0{,}3$);

$N_{\max}$ – maksymalna liczba obiegów pętli w kroku 3°;

u_g – liczba z przedziału $]0, u_{\max}]$ – wartość sterowania poza przedziałem $]-u_g, u_g[$ traktuje się jako leżącą na ograniczeniu (np. $u_g = 0{,}999\, u_{\max}$).

1° Metodą gradientową przeprowadź optymalizację sterowania $\hat{u}_k$ w przedziale $1 \le k \le p$.

Kryterium zatrzymania jest podwójne:

(i) zachodzi nierówność $\left\|\bar{\Delta}\right\| \le \varepsilon$

lub

(ii) długość kroku na kierunku poszukiwania $\bar{\Delta}$ jest mniejsza od ε_0.

Jeśli $\left\|\bar{\Delta}\right\| \le \varepsilon_f$, zakończ obliczenia. W przeciwnym razie $\varepsilon := \min\left(\left\|\bar{\Delta}\right\|, \varepsilon\right)$.

Jeśli optymalizacja przyniosła poprawę wskaźnika jakości, podstaw $I_0 := 1$, w przeciwnym razie $I_0 := 0$. Podstaw $k_{\min} := p$.

2° Jeśli nie istnieją i, takie że

$$i \in \{2,...,k_{\min}-1\} \text{ oraz } \left|\hat{u}_i\right| \le u_g, \left|\hat{u}_{i+1}\right| \le u_g, \left|\hat{u}_{i-1}\right| > u_g \tag{8.8}$$

to przejdź do 4°. W przeciwnym wypadku za $k_{\min}$ podstaw największe i spełniające (8.8), a następnie $\varepsilon' := \varepsilon\sqrt{\dfrac{p+1-k_{\min}}{p}}$.

3° W przedziale $k_{\min} \le k \le p$ przeprowadź optymalizację sterowania jak w kroku 1°, zastępując kryterium zatrzymania (i) przez $\left\|\bar{\Delta}\right\|_{k_{\min}} \le \varepsilon'$.

Jeśli wartość funkcjonału uległa poprawie, podstaw $I_0 := 1$.

Jeśli optymalizacja zakończyła się na kryterium zatrzymania (ii) lub po optymalizacji zachodzi

$$\left|\hat{u}_{k_{\min}}\right| > u_g \text{ lub } \left|\hat{u}_{k_{\min}+1}\right| > u_g,$$

to wróć do 2°. W przeciwnym razie podstaw $\varepsilon' := \rho' \| \bar{\Delta} \|_{k_{\min}}$ i wróć do 3° (liczbę obiegów w tej pętli ograniczamy do $N_{\max}$).

4° Jeśli $I_0 = 0$, zakończ obliczenia. Jeśli $I_0 = 1$, wróć do 1° z podstawieniem $\varepsilon := \rho\varepsilon$. ■

8.2. Algorytm kontynuacyjny z generacją przełączeń i uzgadnianiem gradientów

Rozważany tu algorytm ma dwa poziomy: górny i dolny. Na poziomie górnym metodą *kontynuacyjną* określa się rosnący ciąg wartości horyzontu, definiując w ten sposób ciąg zadań sterowania optymalnego P_T ze stałym horyzontem. Dla każdego z nich, algorytmy minimalizacji dolnego poziomu znajdują optimum wskaźnika jakości Σ_T (7.5), oznaczane przez Σ_T^*. Optymalizacja w ustalonej przestrzeni decyzyjnej (czasów przełączeń i sterowań schodkowych) jest oparta na metodach zmiennej metryki i gradientów sprzężonych. Jeśli optymalizacja przebiega w przestrzeni czasów przełączeń, wymiar przestrzeni decyzyjnej jest modyfikowany przez algorytmy *generacji* i *redukcji* (zob. podrozdz. 8.3). Szczególne znaczenie ma nowa metoda (Szymkat, Korytowski, Turnau 1999a, b) zwiększania liczby przełączeń, nazwana metodą *uzgadniania gradientu*. Oparta jest na analizie gradientu wskaźnika jakości w funkcyjnej przestrzeni sterowań. Zapewniona jest tylko zbieżność ciągu rozwiązań problemów ze stałym horyzontem do punktu stacjonarnego w przestrzeni sterowań, chociaż w praktyce brak zbieżności do rozwiązania lokalnie optymalnego występuje rzadko.

Rozpocznijmy opis od algorytmów górnego poziomu – poszukiwania optymalnego horyzontu. By ocenić wielkość horyzontu, stosujemy przybliżenie newtonowskie, oparte na pochodnej optymalnej wartości wskaźnika jakości względem horyzontu. Newtonowska aproksymacja horyzontu optymalnego $N(T) = T + \Delta_N(T)$, gdzie

$$\Delta_N(T) = \begin{cases} -\dfrac{2\Sigma_T^*}{\nabla\Sigma_T^*}, & \text{jeśli } \nabla\Sigma_T^* < 0 \\ +\infty, & \text{poza tym} \end{cases} \tag{8.9}$$

odniesiona jest do zmodyfikowanej funkcji $\sqrt{\Sigma_T^*}$. Symbol $\nabla\Sigma_T^*$ oznacza pochodną Σ_T^* względem T.

Poniżej przedstawiono prosty algorytm *ekspansji* i *bisekcji* z *aproksymacją newtonowską* przeszukujący półprostą (Korytowski, Szymkat, Turnau 1998).

Algorytm 8.2

Niech $]T_d, T_g[$ będzie początkowym przedziałem nieokreśloności horyzontu optymalnego t^*, przy czym $0 \le T_d < T_g \le +\infty$, $\Sigma^*_{T_d} > 0$, $\Sigma^*_{T_g} = 0$ dla $T_g < +\infty$. Przyjmujemy krok początkowy $\Delta \in]0, T_g - T_d[$ i współczynnik ekspansji $\sigma \ge 1$.

1° Podstaw $T := T_d + \Delta$, rozwiąż P_T, znajdź u^T i Σ^*_T.

2° Jeśli $\Sigma^*_T = 0$, podstaw $T_g := T$, $\Delta := \frac{1}{2}(T_g - T_d)$ i wróć do 1°.

3° Jeśli $\Sigma^*_T > 0$, wyznacz $\Delta_N(T)$ z (8.9), a następnie podstaw $T_d := T$, $\Delta := \min\left(\sigma\Delta, \Delta_N(T), \frac{1}{2}(T_g - T_d)\right)$ i wróć do 1°. ■

O funkcji $T \mapsto \Sigma^*_T$ wiemy, że jest ciągła, a ponadto różniczkowalna wszędzie poza tymi punktami $T < t^*$, w których odpowiedni problem P_T ma więcej niż jedno rozwiązanie optymalne.

Bardziej efektywny jest złożony *algorytm kontynuacji*, który w powiązaniu z metodami minimalizacji w przestrzeni czasów przełączeń i uzgadniania gradientu wykazuje dobrą zbieżność. Jego istotnym elementem jest liniowa ekstrapolacja czasów przełączeń przy zmianie horyzontu.

Założenie, że w problemie P_T można wyznaczyć sterowanie optymalne u^T i wartość Σ^*_T, nie musi być prawdziwe. Znane algorytmy optymalizacji pozwalają wyliczać z wymaganą dokładnością tylko przybliżenia minimów lokalnych, których w problemach P_T na ogół jest wiele. Zwiększając czas obliczeń i stosując dobrze zorganizowany przegląd, można zwiększać szansę trafienia na rozwiązanie optymalne. Jeśli zaś w algorytmie 8.2 zastąpić rozwiązanie optymalne przez lokalnie optymalne, to nie zapewni on zbieżności nawet do lokalnego minimum czasu osiągnięcia stanu docelowego.

Niżej opiszemy algorytmy oparte na idei *kontynuacji*, które znacznie lepiej nadają się do praktycznych obliczeń. Odróżniamy w nich sterowanie optymalne u^T od sterowania dopuszczalnego $\hat{u}^T$, uzyskanego w wyniku próby rozwiązania problemu P_T. Oznaczając $\hat{\Sigma}_T = \Sigma_T(\hat{u}^T)$, mamy oczywistą nierówność $\hat{\Sigma}_T \ge \Sigma^*_T$. Aby objaśnić działanie algorytmów kontynuacyjnych, trzeba wziąć pod uwagę, że każde silne minimum lokalne w problemie P_T dla $T \in]0, t^*[$ ma ciągłą kontynuację w pewnym otoczeniu horyzontu T. Niech mianowicie funkcjonał Σ_T ma silne minimum lokalne w punkcie $\tilde{u} \in L_2(0,\infty)$. Istnieje wówczas funkcja ciągła $O_T \ni \tau \mapsto u^\tau \in L_2(0,\infty)$, gdzie O_T jest pewnym otoczeniem T, taka że $u^T = \tilde{u}$ i dla każdego $\tau \in O_T$ funkcjonał Σ_τ ma minimum lokalne w punkcie u^τ. Zbiór wszystkich sterowań silnie lokalnie optymalnych we wszystkich problemach P_T,

$T \in]0,t^*[$, tworzy więc rodzinę Ω funkcji ciągłych postaci $T \mapsto u^T \in L_2(0,\infty)$, z których każda jest zdefiniowana na swoim maksymalnym przedziale określoności w $[0, t^*]$. Sterowanie u^T oznacza tu silnie lokalnie optymalne rozwiązanie problemu P_T.

Przyjmijmy, że rodzina Ω jest skończona i że sterowanie optymalne u^T w problemie P_T jest dla $T \le t^*$ przedziałami ciągłą funkcją horyzontu T. Na przykład w rozważanym systemie wózka z wahadłem liczba przedziałów ciągłości jest zwykle mała, a często wynosi jeden.

W algorytmie *kontynuacji w przód* konstruuje się ciąg sterowań $\hat{u}^{T_i}$, $i = 0,1,2,...$, gdzie $0 \le T_0 < T_1 < ...$ Każde z nich jest wynikiem próby znalezienia sterowania u^{T_i} za pomocą iteracyjnej metody optymalizacyjnej, z początkowym przybliżeniem sterowania optymalnego oznaczonym przez $u_0^{T_i}$. Załóżmy, że $\hat{u}^{T_i}$ jest dostatecznie dobrym przybliżeniem pewnego sterowania u^{T_i}, na którym Σ_{T_i} osiąga silne minimum lokalne, a różnica $T_{i+1} - T_i$ jest na tyle mała, że istnieje ciągła kontynuacja tego minimum lokalnego przy horyzoncie T_{i+1}. Oznacza to, że w zbiorze Ω istnieje funkcja ρ, której przedział określoności zawiera T_i i T_{i+1}, taka że $\rho(T_i) = u^{T_i}$. Zasadnicza idea algorytmu kontynuacyjnego polega na tym, aby na podstawie znanych sterowań $\hat{u}^{T_j}$, $j \le i$, utworzyć $u_0^{T_{i+1}}$ na tyle bliskie $\rho(T_{i+1})$, żeby procedura optymalizacyjna była zbieżna do $\rho(T_{i+1})$ (oczywiście dopuszcza się zbieżność do rozwiązań lepszych). Takie postępowanie ma dwie zalety. Po pierwsze, gdyby sterowanie $u_0^{T_{i+1}}$ było wybierane bez związku z rezultatami wcześniejszych obliczeń, procedura optymalizacyjna mogłaby się zbiegać do minimum lokalnego gorszego niż $\rho(T_{i+1})$. Byłoby to szczególnie prawdopodobne dla dużych wartości T_{i+1}. Po drugie, dzięki bliskości sterowań $u_0^{T_{i+1}}$ i $\rho(T_{i+1})$ zbieżność procedury optymalizacyjnej jest zwykle szybka, bo na ogół $\hat{u}^{T_{i+1}} \approx \rho(T_{i+1})$.

Sterowanie $u_0^{T_0}$ wybiera się dowolnie. Dalsze przybliżenia początkowe określa się na podstawie ekstrapolacji. Bardzo ścisłe trzymanie się idei kontynuacji może być sensowne, gdy funkcja $T \mapsto u^T$ jest ciągła w całym przedziale $[0,t^*]$. Wskazane jest wtedy rozpoczęcie obliczeń od horyzontu T_0 tak małego, żeby problem P_{T_0} miał jedno, dwa minima lokalne. Zwróćmy uwagę, że dla dostatecznie małego horyzontu, sterowania, na których osiągane są minima lokalne, są zwykle stałe. Następnie w kolejnych iteracjach należy ostrożnie powiększać horyzont, odpowiednio małymi krokami, starając się w trakcie całego poszukiwania nie przekraczać horyzontu optymalnego. Skutecznym zabezpieczeniem przed następstwami zerwania ciągłości kontynuacji i przekroczenia horyzontu optymalnego t^* jest uzupełnienie metody *algorytmem kontynuacji wstecznej* oraz systematycznym przeglądem minimów lokalnych dla wybranych wartości horyzontu.

Algorytm kontynuacji w przód pomyślnie kończy działanie, kiedy znajdzie horyzont T_p i sterowanie $\hat{u}^{T_p}$, takie że $\hat{\Sigma}_{T_p} \le \varepsilon_0$; ε_0 jest z góry zadaną, dostatecznie małą wartością progową. Oczywiście t^* może być znacznie mniejsze od T_p.

Algorytm kontynuacji w przód pozwala uwzględnić dolne oszacowania horyzontu optymalnego, podane w (Korytowski, Szymkat, Turnau 1998), które dla dowolnego horyzontu $T \ge 0$, takiego że $\Sigma_T^* > 0$, mają postać $t^* \ge T + \Delta_d(T)$. Wykorzystywany jest także krok $\Delta_N(T)$, wynikający z przybliżenia newtonowskiego (8.9). Jak w poprzednim algorytmie, stosowana jest ekspansja – ze współczynnikiem $\sigma \ge 1$, początkowym krokiem Δ_0 i maksymalnym krokiem $\Delta_{\max}$. Δ_0 jest również maksymalnym krokiem w każdej sytuacji, w której jest prawdopodobne przerwanie ciągłości kontynuacji.

Metoda określenia kroku Δ zależy od wartości $\hat{\Sigma}_T$ i od tego, czy przy aktualnej wartości horyzontu nastąpiła *strukturalna zmiana sterowania*. Załóżmy, że sterowania $u_0^{T_i}$ i $\hat{u}^{T_i}$ są typu bang-bang. Mówimy, że różnią się one strukturalnie, jeśli mają różne liczby przełączeń lub różnią się wartościami w pierwszym przedziale stałości, a w przypadku tej samej liczby przełączeń i tej samej wartości początkowej – jeśli

$$\max_j \frac{\left|\tau_j^0 - \tau_j^i\right|}{T_i} \ge \beta \tag{8.10}$$

gdzie:

τ_j^0 – j-ty punkt nieciągłości sterowania $u_0^{T_i}$,

τ_j^i – j-ty punkt nieciągłości $\hat{u}^{T_i}$,

β – z góry założona, dostatecznie mała liczba dodatnia (na przykład $\beta = 0{,}1$).

Kiedy $\Sigma_T \le \alpha\varepsilon_0$, przyrosty horyzontu dobiera się ostrożniej, ponieważ poszukiwana wartość horyzontu T_p prawdopodobnie jest blisko. Współczynnik α jest ustalany z góry (np. $\alpha = 5 \cdot 10^3$).

Algorytm 8.3 (kontynuacja w przód)

Dane:

ε_0 – tolerancja zera;

$T_g \in [0, \infty]$ – początkowe górne oszacowanie t^* ($\Sigma_{T_g}^* \le \varepsilon_0$, gdy $T_g < \infty$);

α, β – współczynniki progowe;

K – współczynnik korekcji (typowa wartość: $K = 2$);

$\gamma \in [0, 1[$ – współczynnik wagowy (typowa wartość: $\gamma = 0{,}5$);

$\sigma \ge 1$ – współczynnik ekspansji;

Δ_0 – krok początkowy;

$\Delta_{\max}$ – maksymalny przyrost horyzontu w iteracji.

1° Podstaw $T := \Delta_d(0)$. Jeśli $\hat{\Sigma}_T \le \varepsilon_0$, podstaw $t^* := T$ i zakończ algorytm.
W przeciwnym razie podstaw $\Delta := \Delta_0$.

2° Wyznacz $\Delta_d(T)$ i $\Delta_N(T)$. Podstaw $\Delta_g := \min\left(\Delta_N(T), T_g - T\right)$,
$\Delta_m := \Delta_d(T) + \gamma \max\left(0, \Delta_g - \Delta_d(T)\right)$.

3° Podstaw $\Delta := \min\left(\Delta, \Delta_m, \Delta_{\max}, K\Delta_d(T)\right)$.

4° Podstaw $T := T + \Delta$. Jeśli $\hat{\Sigma}_T > \varepsilon_0$, przejdź do 5°.
W przeciwnym razie podstaw $T_g := T$ i przejdź do algorytmu kontynuacji wstecznej.
Jeżeli nie są spełnione warunki takiego przejścia, zakończ obliczenia.

5° Jeśli $\hat{\Sigma}_T \le \alpha\varepsilon_0$, podstaw $\Delta := \min(\Delta_0, \Delta)$ i wróć do 2°.

6° Jeśli dla danego T wystąpiła strukturalna zmiana sterowania, podstaw $\Delta := \Delta_0$.
W przeciwnym razie $\Delta := \sigma\Delta$. Wróć do 2°. ■

Algorytm kontynuacji wstecznej rozwiązuje kolejno problemy P_{t_i}, $i = 0,1,...,q$, przy czym $0 < t_q < ... < t_1 < t_0$, a $t_0 = T_p$ jest wartością horyzontu, na której ostatnio zakończył działanie algorytm kontynuacji w przód. Oznaczmy sterowanie otrzymane w wyniku próby rozwiązania problemu P_{t_i} przez $\hat{v}^{t_i}$, a odpowiednią wartość funkcjonału przez $\hat{\Sigma}_{t_i}$, $\hat{\Sigma}_{t_i} = \Sigma_{t_i}(\hat{v}^{t_i})$. Algorytm zatrzymuje się, kiedy po raz pierwszy uzyska wartość $\hat{\Sigma}_{t_i}$ większą od progu ε_0. Początkowe aproksymacje sterowania w problemach P_{t_i} oznaczamy przez $v_0^{t_i}$. Dla $i > 0$ tworzy się je według schematu ekstrapolacji, tak jak w algorytmie kontynuacji w przód (z uwzględnieniem różnic wynikających ze zmiany kierunku przeszukiwania horyzontu). Natomiast $v_0^{t_0}$ powinno być wybrane w taki sposób, żeby algorytm kontynuacji wstecznej miał szansę wygenerowania w kolejnych krokach sterowań istotnie lepszych niż te, które w ostatnim przebiegu dla podobnych wartości horyzontu generował algorytm kontynuacji w przód. Dlatego w przypadku, gdy $u_0^{T_p}$ istotnie różni się od $\hat{u}^{T_p}$ (w sensie względnej odległości w L_2), przyjmujemy $v_0^{t_0} = \hat{u}^{T_p}$ i pomijamy optymalizację, co daje $\hat{v}^{t_0} = v_0^{t_0}$. W przeciwnym razie stosuje się przegląd sterowań z optymalizacją tak długo, aż uzyska się sterowanie $\hat{v}^{t_0}$ istotnie różne od $\hat{u}^{T_p}$ i spełniające warunek $\Sigma_{t_0}(\hat{v}^{t_0}) \le \varepsilon_0$. Jeśli takiego sterowania nie uda się znaleźć, obliczenia się kończy, przyjmując T_p za horyzont optymalny.

Algorytm 8.4 (kontynuacja wsteczna)

Dane:

ε_0 – tolerancja zera;

$T_g > 0$ – początkowe górne oszacowanie t^* ($\hat{\Sigma}_{T_g} \le \varepsilon_0$);

Δ_0 – krok poszukiwania.

1° Podstaw $T := T_g - \Delta_0$.

2° Jeśli $\hat{\Sigma}_T \le \varepsilon_0$, wróć do 1° z podstawieniem $T_g := T$.

Jeśli $\hat{\Sigma}_T > \varepsilon_0$, przejdź do algorytmu kontynuacji w przód. ■

W algorytmach kontynuacyjnych początkowe przybliżenie sterowania optymalnego $u_0^{T_{i+1}}$ dla $i > 0$ wyznacza się przez ekstrapolację jedno- albo dwukrokową. W schemacie jednokrokowym ($i > 0$)

$$u_0^{T_{i+1}}(t) = \hat{u}^{T_i}(t'), \quad t' = \frac{T_i}{T_{i+1}} t, \quad t \in [0, T_{i+1}] \tag{8.11}$$

Schemat ten stosuje się, gdy nie są spełnione założenia wymagane w schemacie dwukrokowym, a także wtedy, gdy nastąpiła strukturalna zmiana sterowania w trakcie optymalizacji w problemie P_{T_i}.

Schemat dwukrokowy można stosować dla $i > 1$ i bangbangowego sterowania $\hat{u}^{T_i}$. Niech τ_j^i, $j = 1, 2, ..., k$, będzie rosnącym ciągiem wszystkich jego czasów przełączeń, ponadto niech $\tau_0^i = 0$ i $\tau_{k+1}^i = T_i$. Zakładamy $\tau_{j+1}^i - \tau_j^i > \varepsilon$ dla $j = 0, 1, ..., k$, gdzie ε jest z góry ustaloną małą liczbą dodatnią. Dla każdego czasu przełączenia wprowadza się formułę ekstrapolacyjną

$$\tau_j(T) = a_j(T - T_i) + \tau_j^i, \quad j = 0, 1, ..., k+1 \tag{8.12}$$

Przypuśćmy najpierw, że sterowanie $\hat{u}^{T_{i-1}}$ ma dokładnie tę samą liczbę przełączeń co $\hat{u}^{T_i}$ i tę samą wartość w pierwszym przedziale stałości, $\hat{u}^{T_{i-1}}(0_+) = \hat{u}^{T_i}(0_+)$. Czasy przełączeń sterowania $\hat{u}^{T_{i-1}}$ tworzą ciąg rosnący τ_j^{i-1}, $j = 0, 1, ..., k+1$, przy czym $\tau_0^{i-1} = 0$, $\tau_{k+1}^{i-1} = T_{i-1}$, $\tau_{j+1}^{i-1} - \tau_j^{i-1} > \varepsilon$ dla $j = 0, 1, ..., k+1$. Wtedy

$$a_j = \frac{\tau_j^i - \tau_j^{i-1}}{T_i - T_{i-1}}, \quad j = 0, 1, ..., k+1 \tag{8.13}$$

Sterowanie $u_0^{T_{i+1}}$ jest również typu bang-bang z k przełączeniami τ_j^0, $j=1,2,...,k$, otrzymanymi przez liniową ekstrapolację: $\tau_j^0 = \tau_j(T_{i+1})$, $j=0,1,...,k+1$. Zakłada się

$$\tau_{j+1}^0 - \tau_j^0 > \varepsilon, \quad j=0,1,...,k \tag{8.14}$$

Założenie to można zawsze spełnić, wybierając dostatecznie małe T_{i+1}. W pierwszym przedziale stałości $u_0^{T_{i+1}}(t)$ jest równe $\hat{u}^{T_i}(t)$.

8.3. Algorytm zmiennej parametryzacji ze swobodnym horyzontem

Metoda numeryczna rozwiązywania problemu optymalnoczasowego opisana poniżej jest rozwinięciem prac Siriseny (1974) oraz Siriseny, Chou (1979) przez wprowadzenie mechanizmów generacji i redukcji przełączeń sterowania (Korytowski, Szymkat, Turnau 1998; Szymkat, Korytowski, Turnau 1999a, 2000). Algorytm minimalizuje pomocniczy funkcjonał zależny od horyzontu i sparametryzowany współczynnikiem wagowym horyzontu. Charakterystyczne dla tej metody, w odróżnieniu od typowych podejść bezpośrednich, jest zachowanie niskiego wymiaru przestrzeni decyzyjnej optymalizacji, z równoczesnym zapewnieniem zbieżności procedury numerycznej w szerokim zakresie.

Algorytm korzysta z informacji o kształcie gradientu funkcjonału pomocniczego w przestrzeni sterowań. Niezgodność znaków antygradientu i sterowania świadczy o niespełnieniu warunków koniecznych optymalności i wskazuje, jak konstruować reparametryzację przez generację nowych i ewentualną redukcję kolidujących ze sobą przełączeń. Ma to miejsce przed każdą kwazinewtonowską iteracją procedury poszukiwania czasów przełączeń i horyzontu. Ważną właściwością procedury numerycznej jest jej monotoniczna zbieżność na każdym etapie, przy stałej wartości wagi horyzontu w kryterium pomocniczym.

Rozważamy sterowanie optymalnoczasowe w systemie (7.1) ze sterowaniem skalarnym, stanem docelowym x^f, ograniczeniami na sterowanie (7.35) z $u_{\min} = -u_{\max}$. Zbiór sterowań dopuszczalnych $\boldsymbol{U}_{\text{ad}}$ składa się ze wszystkich prawostronnie ciągłych funkcji $u:[0,\infty[\to [-u_{\max}, u_{\max}]$. Wprowadzamy pomocniczy wskaźnik jakości (7.41), zależny od horyzontu $T \geq 0$. Parametr q we wzorze (7.41) jest nieujemny, a macierz Q jest symetryczna i dodatnio określona (zob. (7.30)). Sterowaniem optymalnoczasowym jest sterowanie dopuszczalne u, dla którego $S_0(u,T) = 0$ przy najmniejszym T – zobacz wzór (7.43).

Zajmiemy się parametryzacją, która charakteryzuje wyłącznie sterowania typu bang-bang. Rozwiązania osobliwe i tzw. zjawisko Fullera mogą być rozpatrywane tylko jako przypadki graniczne. Sterowanie dopuszczalne u nazywamy bangbangowym, jeśli istnieją: liczba $u_0 \in \{-u_{\max}, u_{\max}\}$ i skończony ciąg niemalejący $\tau = (\tau_i)_{i=1}^{i=m} \subset [0,\infty[$, takie że

zachodzą równości

$$u(t)=(-1)^i u_0, \quad t\in[\tau_i,\tau_{i+1}[, \quad i=0,1,...,m-1$$
$$u(t)=(-1)^m u_0, \quad t\in[\tau_m,\infty[\tag{8.15}$$

gdzie $\tau_0=0$. Ciąg τ może być pusty; wówczas $m=0$. Sterowanie określone przez (8.15) zapisujemy również jako $u(t;\tau,u_0)$.

Obcięcie wskaźnika jakości (7.41) do sterowań bangbangowych określa w sposób jednoznaczny funkcjonał

$$S'_q(\tau,u_0,T)=S_q(u(\cdot\,;\tau,u_0),T) \tag{8.16}$$

gdzie $\tau=(\tau_i)_{i=1}^{i=m}$ jest skończonym niemalejącym ciągiem w przedziale $[0,T]$.

Przy takich założeniach minimalizacja funkcjonału (7.41) względem bangbangowych sterowań i horyzontu sprowadza się do poszukiwania optymalnej czwórki (m,u_0,τ,T), gdzie m jest nieujemną liczbą całkowitą, $u_0\in\{-u_{\max},u_{\max}\}$, a para (τ,T) należy do zbioru dopuszczalnego Θ_m w $\boldsymbol{R}^{m+1}$

$$\Theta_m=\{\vartheta\in\boldsymbol{R}^{m+1}: 0\le\vartheta_1\le\vartheta_2\le...\le\vartheta_{m+1}\} \tag{8.17}$$

Dyskretne zmienne decyzyjne m oraz u_0 określa się na górnym poziomie minimalizacji algorytmu w procedurach *generacji* i *redukcji*. Na dolnym poziomie optymalizuje się wartości τ i T dla ustalonych m i u_0, przy użyciu metody gradientowej (BFGS).

Gradient S'_q. Dla problemu (7.1), (7.2) definiuje się zmienną sprzężoną $\psi(t)$ (7.10) z warunkiem końcowym $\psi(T)$ (7.40). Jej znajomość pozwala określić antygradient $g(t)$ (7.33) wskaźnika jakości $S_q(u,T)$ (7.41) w przestrzeni sterowań. Gradient S'_q jest rozumiany jako pochodna S'_q względem wektora $\mathrm{col}(\tau_1,\tau_2,...,\tau_m,T)$

$$\nabla S'_q(\tau,u_0,T)=\mathrm{col}(\nabla_\tau S'_q(\tau,u_0,T),\nabla_T S'_q(\tau,u_0,T)) \tag{8.18}$$

$\nabla_\tau S'_q$ jest m-wymiarowym wektorem kolumnowym pochodnych S'_q względem czasów przełączeń

$$\nabla_\tau S'_q(\tau,u_0,T)=\mathrm{col}(\nabla_{\tau_1} S'_q,\nabla_{\tau_2} S'_q,...,\nabla_{\tau_m} S'_q) \tag{8.19}$$

$\nabla_T S'_q$ jest pochodną S'_q względem horyzontu. W pracy Siriseny (1974) wyliczono pochodne (8.19) dla $0<\tau_1<\tau_2<...<\tau_m<T$:

$$\nabla_{\tau_i} S'_q(\tau,u_0,T)=2G(\tau_i), \quad i=1,...,m \tag{8.20}$$

$$\nabla_T S'_q(\tau,u_0,T) = q - \psi(T)^{\mathsf{T}} f(x(T), u(T;\tau,u_0)) = h - G(t) \tag{8.21}$$

gdzie

$$G(t) = u(t;\tau,u_0)g(t) \tag{8.22}$$

$$h = q - \psi(T)^{\mathsf{T}} f^0(x(T)) \tag{8.23}$$

z trajektorią stanu x, trajektorią sprzężoną ψ i antygradientem g wyznaczonymi w punkcie $u(\cdot;\tau,u_0)$.

Definicję gradientu $\nabla S'_q$ (8.18) rozszerza się jednoznacznie w sposób ciągły na cały zbiór domknięty $0 \le \tau_1 \le \tau_2 \le ... \le \tau_m \le T$ w $\boldsymbol{R}^{m+1}$.

Niech $\tau_0 = 0$ i $\tau_{m+1} = T$. Zauważmy, że dla $\tau_{i-1} \le \tau_i < \tau_{i+1}$, $1 \le i \le m$, wzór (8.20) obowiązuje nadal. Jeżeli $\tau_{i-1} < \tau_i = \tau_{i+1}$ i τ_i nie jest równe T lub nie jest równe dowolnemu elementowi τ różnemu od τ_{i+1}, to

$$\nabla_{\tau_i} S'_q(\tau,u_0,T) = -2G(\tau_i) \tag{8.24}$$

Generacje zachowujące sterowanie i generacje istotne. Złożeniem ciągów niemalejących $\tau, \gamma \subset \boldsymbol{R}$ nazywamy ciąg niemalejący $\tau \sim \gamma$ w $\boldsymbol{R}$, taki że dla każdego $r \in \boldsymbol{R}$ liczba wyrazów równych r w $\tau \sim \gamma$ jest równa sumie liczb wyrazów o tej wartości w γ i τ. Dla dowolnego $T \in [0,\infty[$ i $u_0 \in \{-u_{\max}, u_{\max}\}$, niech τ będzie skończonym, ściśle rosnącym ciągiem m czasów przełączeń w przedziale $]0,T[$ (który może być pusty, jeśli $m = 0$).

Generacja przełączeń polega na wyborze pewnego niemalejącego ciągu $\gamma \subset [0,T]$ nowych, dodatkowych czasów przełączeń, z równoczesną ewentualną zamianą u_0 na $u'_0 \in \{-u_{\max}, u_{\max}\}$. W ten sposób powstaje nowe sterowanie bangbangowe $u(\cdot;\tau \sim \gamma, u'_0)$ o m' przełączeniach, $m' > m$. Nowe sterowanie staje się warunkiem początkowym dla optymalizacji dolnego poziomu ze zmiennymi decyzyjnymi $\tau \sim \gamma$ i T. Generacja przełączeń powinna *zachowywać sterowanie*, być *istotną* dla optymalizacji i prowadzić do polepszania wartości minimalizowanego wskaźnika jakości. Żąda się, by generację charakteryzowało *zachowywanie sterowania* (ZS), co oznacza spełnianie warunku

$$u(t;\tau \sim \gamma, u'_0) = u(t;\tau,u_0), \quad \forall t \in [0,T[\tag{8.25}$$

Generacja określona przez γ i u'_0 jest typu ZS wtedy i tylko wtedy, gdy każdy element γ, różny od zera i T, jest nieparzystej krotności, oraz $u'_0 = (-1)^k u_0$, gdzie k jest liczbą zerowych elementów w ciągu γ. Każda generacja typu ZS jest więc jednoznacznie określona przez ciąg γ.

Od generacji typu ZS żądamy dodatkowo, by jej wydajność optymalizacyjna zwiększała się wraz ze wzrostem liczby nowych przełączeń. By określić wydajność generacji, po-

sługujemy się rzutem antygradientu $a = -\nabla S'_q(\tau, u_0, T)$ na zbiór dopuszczalny Θ_m. Dla dowolnego $\vartheta \in \Theta_m$, niech $Z_m(\vartheta)$ oznacza zbiór wszystkich dopuszczalnych kierunków w punkcie ϑ

$$Z_m(\vartheta) = \{z \in \boldsymbol{R}^{m+1} : \exists \bar{\lambda} > 0 \ \forall \lambda \in]0, \bar{\lambda}[, \ \vartheta + \lambda z \in \Theta_m\} \tag{8.26}$$

Rzut a na zbiór Θ_m w punkcie $\vartheta \in \Theta_m$, oznaczony przez $\sigma(\tau, u_0, T)$, jest określony niżej:

(i) $\sigma(\tau, u_0, T) = 0$, jeżeli $a^{\mathsf{T}} z \le 0 \ \forall z \in Z_m(\vartheta)$;

(ii) w przeciwnym wypadku $\sigma(\tau, u_0, T) = (a^{\mathsf{T}} z_0) z_0$, gdzie $z_0 \in Z_m(\vartheta)$, $\|z_0\| = 1$ i

$$a^{\mathsf{T}} z_0 = \max\{a^{\mathsf{T}} z : z \in Z_m(\vartheta), \|z\| = 1\}.$$

Wydajność generacji γ typu ZS jest zdefiniowana wzorem

$$E(\gamma) = \left\|\sigma(\tau \sim \gamma, u'_0, T)\right\|^2 - \left\|\nabla S'_q(\tau, u_0, T)\right\|^2 \tag{8.27}$$

Taką definicję można uzasadnić tym, że $\sigma(\tau \sim \gamma, u'_0, T)$ jest pierwszym kierunkiem optymalizacji dolnego poziomu po generacji γ (z wyjątkiem przypadku, gdy rzut antygradientu równa się zero). Pochodna kierunkowa $S'_q(\tau \sim \gamma, u'_0, T)$ na kierunku σ wyraża się przez $-\left\|\sigma(\tau \sim \gamma, u'_0, T)\right\|^2$, podczas gdy pochodna kierunkowa $S'_q(\tau, u_0, T)$ w kierunku największego spadku przed generacją wyraża się przez $-\left\|\nabla S'_q(\tau, u_0, T)\right\|^2$.

Dla wydajnie działającej procedury optymalizacyjnej pożądane jest, by liczba przełączeń tworzonych podczas generacji była mała, a mimo to efektywnie wpływała na zmniejszanie się wartości wskaźnika jakości. Generacja nie powinna produkować przełączeń bezużytecznych dla zwiększania wydajności procedury. Innymi słowy, winna być *istotna* dla optymalizacji.

Generację γ typu ZS nazywamy istotną, jeżeli antygradient $-\nabla S'_q(\tau \sim \gamma, u'_0, T)$ należy do $\operatorname{int} Z_{m'}(\tau \sim \gamma, T)$, czyli jest skierowany do wnętrza zbioru dopuszczalnego $\Theta_{m'}$.

Można pokazać, że spełnienie pięciu poniższych warunków jest konieczne i wystarczające, by generacja γ, zachowująca sterowanie, była istotna:

(i) Każdy wyraz γ jest różny od wszystkich wyrazów τ.

(ii) Każdy wyraz γ, różny od zera i T, ma krotność 2.

(iii) γ ma co najwyżej jeden element zerowy i co najwyżej jeden element równy T.

(iv) Jeśli γ ma element równy pewnemu $t \in [0, T[$, to $G(t) < 0$ (zob. (8.22)).

(v) Jeśli γ posiada element równy T, to

$$G(T) < -\tfrac{1}{3} h \tag{8.28}$$

Nierówność (8.28) wynika z oczywistego warunku $\nabla_T S'_q(\tau', u_0, T) < \nabla_{\tau'_{m+1}} S'_q(\tau', u_0, T)$, gdzie $\tau' = \tau \sim (T) = (\tau'_1, \tau'_2, ..., \tau'_{m+1})$ (zob. (8.20) i (8.21), po uwzględnieniu koniecznej zmiany znaku).

Każda generacja ZS może być przekształcona w generację istotną po opuszczeniu pewnych elementów z ciągu przełączeń. Dla zadanego $T > 0$, ściśle rosnącego ciągu $\tau \subset]0, T[$ i generacji γ typu ZS, definiujemy $\operatorname{ess}\gamma$ jako maksymalny podciąg γ, taki że generacja $\operatorname{ess}\gamma$ jest istotna. Ważnym spostrzeżeniem jest, że dla każdej generacji γ typu ZS jej wydajność jest identyczna z wydajnością $\operatorname{ess}\gamma$, czyli $E(\gamma) = E(\operatorname{ess}\gamma)$. Z tego powodu będą rozpatrywane tylko generacje istotne.

Wydajność generacji istotnej $\gamma = (\gamma_1, ..., \gamma_\mu)$ można zapisać jako sumę wydajności nowych przełączeń powiększoną o zmianę wydajności horyzontu

$$E(\gamma) = \sum_{i=1}^{\mu} E_i(\gamma) + \Delta E_T(\gamma) \tag{8.29}$$

gdzie

$$\begin{aligned} E_i(\gamma) &= \nabla_{\gamma_i} S'_q(\tau \sim \gamma, u'_0, T)^2 = 4G(\gamma_i)^2 \\ \Delta E_T(\gamma) &= \nabla_T S'_q(\tau \sim \gamma, u'_0, T)^2 - \nabla_T S'_q(\tau, u_0, T)^2 \end{aligned} \tag{8.30}$$

$\Delta E_T(\gamma) = 0$, jeśli γ nie ma wyrazu równego T. W przeciwnym wypadku

$$\Delta E_T(\gamma) = 4hG(T) \tag{8.31}$$

Algorytm generacji. Wydajność generacji istotnej można powiększyć przez dodanie przełączeń w punktach o dużych ujemnych wartościach G. Liczba nowych przełączeń i ich rozmieszczenie muszą podlegać kompromisowemu wyborowi z uwagi na dwa efekty. Po pierwsze, działanie gradientowych procedur optymalizacyjnych bez ograniczeń, na przykład procedury BFGS (Schwartz 1995; Branch, Grace 1996), pogarsza się wraz ze wzrostem wymiaru. Po drugie, małe odległości pomiędzy (różnymi) czasami przełączeń albo czasami przełączeń i końcami przedziału $[0, T]$, zwiększają prawdopodobieństwo trafienia w brzeg zbioru dopuszczalnego Θ_m. Ostatni efekt jest szczególnie szkodliwy, bo każde trafienie w brzeg uaktywnia procedury redukcji i generacji, co prowadzi do częstych, oscylacyjnych zmian liczby czasów przełączeń.

By uniknąć powstawania za dużej liczby przełączeń podczas generacji, zakłada się, że nie więcej niż dwa mogą być dodane w każdym przedziale $[\tau_i, \tau_{i+1}[$, $i = 0, 1, ..., m$, gdzie $\tau_0 = 0$ i $\tau_{m+1} = T$. Dla zwiększenia wydajności optymalizacji nowe przełączenie jest umieszczane tylko w takim punkcie, w którym G osiąga minimum w tym przedziale, i pod warunkiem, że to minimum jest dostatecznie ujemne. Niech p będzie zadaną z góry stałą niedodatnią. Zdefiniujmy $M = p \,\|\, \nabla_\tau S'_q(\tau, u_0, T) \,\|$ dla $m > 0$, i $M = 0$ dla $m = 0$.

Żąda się, by dla każdego $\gamma_i \neq T$

$$G(\gamma_i) < M \tag{8.32}$$

Wygenerowanie przełączenia w $\gamma_\mu = T$ podlega bardziej złożonym regułom. Zakłada się, że nie więcej niż dwa nowe przełączenia mogą zostać wygenerowane w $]\tau_m, T]$. Przypuśćmy na początek, że żaden punkt w $]\tau_m, T[$ nie spełnia podanych wyżej warunków utworzenia w nim pary przełączeń. Wówczas sprawdzane są dwa warunki. Po pierwsze, generacja powinna być istotna, czyli ma być spełniona nierówność (8.28). Po drugie, wzrost wydajności związany z wprowadzeniem przełączenia musi być dostatecznie duży. Wzrost ten jest mierzony w stosunku do gradientu przed generacją

$$E_\mu(\gamma) + \Delta E_T(\gamma) > 4M^2 \tag{8.33}$$

czyli

$$G(T)(h + G(T)) > M^2 \tag{8.34}$$

Przypuśćmy z kolei, że pewien punkt $t \in]\tau_m, T[$ spełnia warunki generacji. Nowe przełączenie jest umieszczane w T (a nie w t), jeśli spełniona jest nierówność

$$G(T)(h + G(T)) > G(t)^2 \tag{8.35}$$

obok warunków (8.28) i (8.34). Jeśli (8.35) nie zachodzi, to para nowych przełączeń zostaje utworzona w t.

To rozumowanie prowadzi do następującego algorytmu generacji. Niech m będzie nieujemną liczbą całkowitą, $\tau = (\tau_i)_{i=1}^{i=m}$ dla $m > 0$, τ pusty dla $m = 0$, $\tau_0 = 0$, $\tau_{m+1} = T$. Ciąg γ składa się wyłącznie z elementów zbioru E budowanego w dwóch etapach. W pierwszym etapie E jest zbiorem wszystkich punktów $t \in [0, T[$ spełniających następujące warunki:

(i) t różni się od wszystkich elementów τ;

(ii) t jest najmniejszym punktem w $[\tau_i, \tau_{i+1}[$, w którym G osiąga minimum w tym przedziale, dla pewnego $i \in \{0, 1, ..., m\}$;

(iii) t spełnia (8.32), czyli

$$G(t) < M \tag{8.36}$$

W drugim etapie definiujemy

$$N = \begin{cases} M, & \text{gdy } E \cap]\tau_m, T[= \varnothing \\ G(t), & \text{gdy } E \cap]\tau_m, T[= \{t\} \end{cases} \tag{8.37}$$

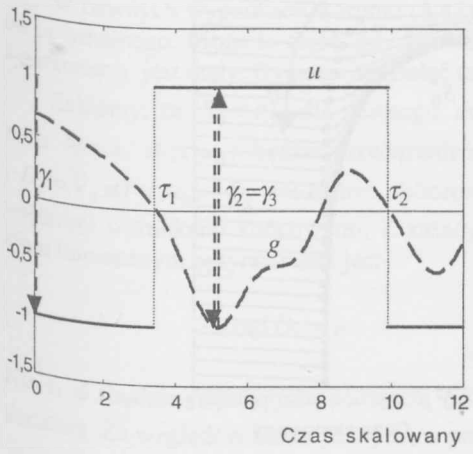
Jeśli warunek (8.28) jest spełniony i

$$G(T)(h+G(T)) > N^2 \tag{8.38}$$

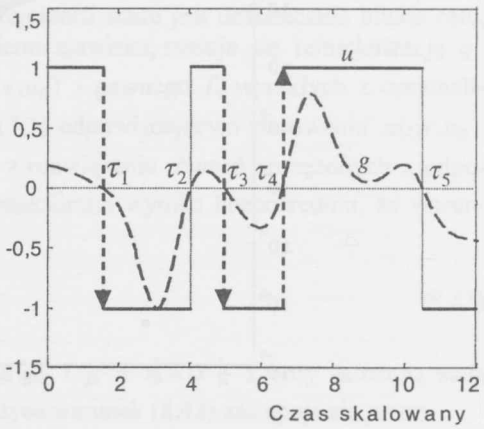
to podstawiamy $\boldsymbol{E} := \boldsymbol{E} \setminus]\tau_m, T[\cup \{T\}$. Dla każdego $t \in \boldsymbol{E}$, takiego że $t \neq 0$ i $t \neq T$, γ ma dokładnie dwa elementy równe t. Dla każdego innego $t \in \boldsymbol{E}$, γ ma dokładnie jeden element równy t.

Idea generacji jest wyjaśniona na rysunkach 8.1 i 8.2. Na rysunku 8.1 przedstawiono sterowanie u (gruba linia ciągła) i antygradient g (linia przerywana) w chwili generacji. Zauważmy, że znaki sterowania i antygradientu są przeciwne, tzn. $G(t) < 0$ dla $t \in [0, \tau_1[$ i w pewnym podprzedziale $[\tau_1, \tau_2[$. Strzałki na rysunku pokazują nowe czasy przełączeń generowane przez $\gamma = (\gamma_1, \gamma_2, \gamma_3)$. Dalsza optymalizacja prowadzi do sytuacji uwidocznionej na rysunku 8.2, ze zmienionymi czasami przełączeń. Przełączenia τ_1, τ_3, τ_4, zapoczątkowane przez γ, są zaznaczone strzałkami.

Sytuacja przedstawiona na rysunku 8.1 odnosi się do stacjonarnego punktu w przestrzeni parametrów (czasów przełączeń), a nie do punktu stacjonarnego w przestrzeni sterowań. Jest to przykład niezgodności sterowania i antygradientu. Rysunek 8.2 pokazuje przypadek antygradientu całkowicie zgodnego ze sterowaniem. Pochodne S'_q względem czasów przełączeń są równe zero, a równocześnie znika rzut antygradientu S_q na zbiór dopuszczalny, wyliczony względem sterowania. Są więc spełnione warunki konieczne optimum w przestrzeni sterowań. Trzeba podkreślić, że sytuacja z rysunku 8.1 na ogół się nie zdarza, gdyż warunki (8.42) i (8.44) zwykle wymusiłyby generację znacznie wcześniej niż w punkcie stacjonarnym w przestrzeni parametrów. Należy też pamiętać, że przejście do końcowej, w pełni zgodnej sytuacji (rys. 8.2), na ogół wymaga większej liczby generacji przedzielanych redukcjami.



Rys. 8.1. Przykład generacji



Rys. 8.2. Przypadek w pełni uzgodniony

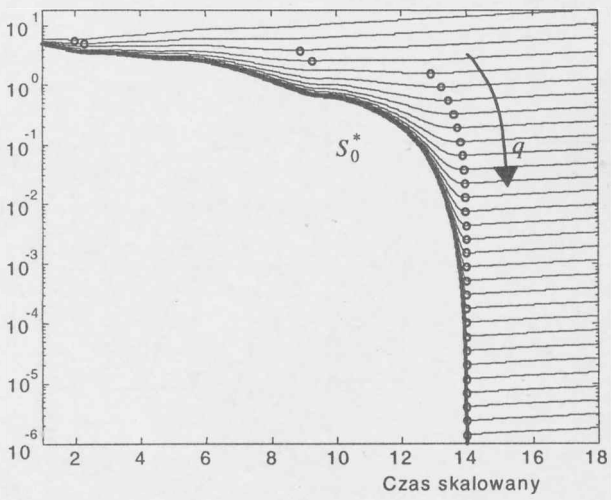
Redukcje. Redukcja przełączeń ma miejsce po każdej iteracji procedury optymalizacyjnej, która kończy się w punkcie (τ, T) leżącym na brzegu zbioru dopuszczalnego Θ_m. Niech τ będzie skończonym, niemalejącym ciągiem czasów przełączeń w $[0, T]$, pochodzącym z optymalizacji, określonym dla dowolnego $T \in [0, \infty[$ i $u_0 \in \{-u_{\max}, u_{\max}\}$.

Redukcja przełączeń polega na przekształceniu τ w skończony, ściśle rosnący ciąg ω w $]0, T[$, w taki sposób, że

$$u(t; \omega, (-1)^k u_0) = u(t; \tau, u_0), \quad \forall t \in [0, T[\tag{8.39}$$

gdzie k jest liczbą zerowych elementów w τ.

Reguły zmieniania parametru q. Dla zbudowania reguł zmieniania parametru q podczas optymalizacji posłużymy się rysunkiem 8.3. Na tym rysunku pokazano przykład ilustrujący zależność optymalnych wartości wskaźnika jakości S_q od horyzontu T. Użyto skali logarytmicznej na osi pionowej. Podane wartości są optymalne względem sterowania dla ustalonych wartości T. Linie ciągłe na rysunku 8.3 po prawej stronie reprezentują optymalne wartości wskaźnika jakości S_q dla malejących q. Kółkami oznaczono położenia minimów. Wykres S_0^* przedstawia optymalne wartości S_0 i tworzy obwiednię tych krzywych. Jest dobrze widoczne, że argumenty minimów zmierzają do wartości optymalnego horyzontu.



Rys. 8.3. Optymalne wartości S_0 i S_q jako funkcje horyzontu

Następujące spostrzeżenie uzasadnia proponowane reguły zmieniania wartości parametru q. Niech

$$S_q^* = \min_{\substack{u\in U_{ad}\\ T\geq 0}} S_q(u,T), \quad \min_{u\in U_{ad}} S_q(u,T_q^*) = S_q^* \tag{8.40}$$

Załóżmy, że istnieje rozwiązanie problemu optymalnoczasowego. Jeśli $q \to 0^+$, to T_q^* zmierza do horyzontu optymalnego t^* z lewej strony.

W algorytmie przedstawionym na schemacie poniżej asymptotyczna optymalność (względem horyzontu) jest zapewniona poprzez krokowe zmniejszanie q. Parametr q, inicjalizowany dodatnią wartością, jest utrzymywany dopóty na tej wartości, dopóki nie zostaną spełnione warunki uaktualnienia q

$$\left\|g^U\right\| < \varepsilon_g \text{ i } \left|\nabla_T S_q'\right| < \varepsilon_T \tag{8.41}$$

gdzie ε_g i ε_T są przyjętymi wartościami progów. Gdy zajdzie warunek (8.41), to wartości q, ε_g i ε_T zostają zastąpione odpowiednio przez $r_q q$, $r_g \varepsilon_g$ i $r_T \varepsilon_T$. Nowe wartości pozostają w użyciu, dopóki warunki (8.41) nie zostaną ponownie spełnione. Dodatnie mnożniki r_q, r_g i r_T są mniejsze od jedności. Cała procedura powtarza się aż do zakończenia algorytmu.

Warunki zakończenia są postaci

$$S_q < \varepsilon_S^f \text{ lub } \left|\nabla_T S_q'\right| < \varepsilon_T^f \text{ i } \left\|g^U\right\| < \varepsilon_g^f \tag{8.42}$$

gdzie ε_S^f, ε_T^f i ε_g^f są wymaganymi tolerancjami zera.

W pewnych wypadkach warunki (8.42) zostają spełnione, kiedy horyzont jest dłuższy od optymalnego. Może to zajść, gdy koniec trajektorii stanu jest dostatecznie blisko celu, a parametr q jest mały. By przeciwdziałać takiemu zjawisku, stosuje się reinicjalizację q.

Załóżmy, że $S_0 < \varepsilon_S^f$ dla pewnego $u(\cdot;\tau,u_0)$ i pewnego T, wynikłych z optymalizacji. Niech $x(\cdot;\tau,u_0)$ będzie rozwiązaniem (7.1) odpowiadającym sterowaniu $u(\cdot;\tau,u_0)$ i $D_\tau = \nabla_\tau x(T;\tau,u_0)$. D_τ może być obliczone z rozwiązania równań sprzężonych z odpowiednimi warunkami końcowymi. Z zasady maksimum wynika bezpośrednio, że warunkiem koniecznym optymalności jest

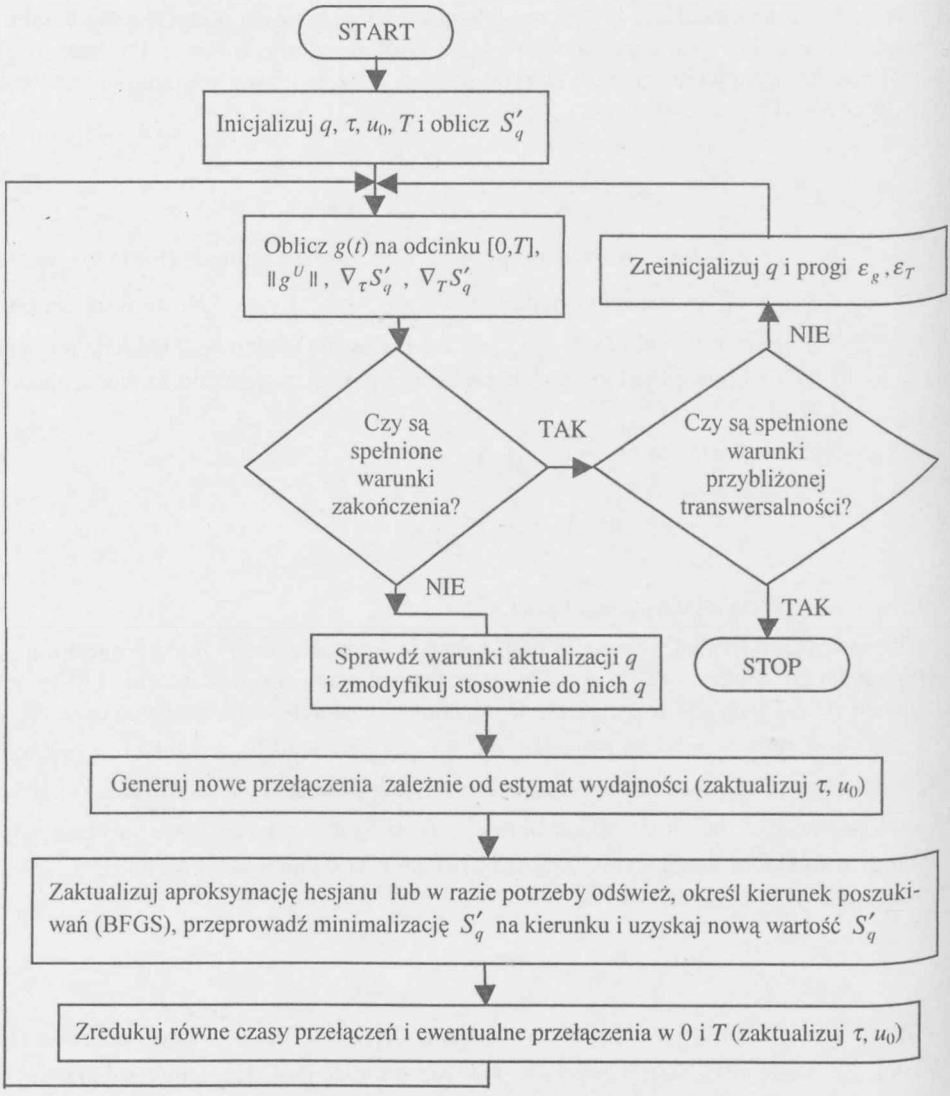
$$\text{rząd } D_\tau < n \tag{8.43}$$

Niech δ będzie najmniejszą wartością osobliwą D_τ, a $\kappa \approx 0$ – z góry założoną stałą dodatnią. Ze względów numerycznych, w praktyce warunek (8.43) zastępuje się przez

$$\delta < \kappa \tag{8.44}$$

Nierówność (8.44) nazywa się *przybliżonym warunkiem transwersalności*. Jeśli nierówność (8.44) jest spełniona, obliczenia się kończą. W przeciwnym razie reinicjalizuje się q: parametrowi q nadaje się dostatecznie dużą wartość q_0, tak by zapobiec przedwczesnemu zakończeniu algorytmu. Jednocześnie reinicjalizuje się progi ε_g i ε_T.

Algorytm 8.5 (zmienna parametryzacja ze swobodnym horyzontem)
Pełny algorytm numeryczny jest przedstawiony na rysunku 8.4.



Rys. 8.4. Schemat pełnego algorytmu zmiennej parametryzacji ze swobodnym horyzontem

Dla uzyskania kolejnych aproksymacji rozwiązania optymalnego w przestrzeni parametrów stosuje się procedurę kwazinewtonowską (BFGS). W każdej iteracji procedury optymalizacyjnej przed rozpoczęciem poszukiwania minimum na kierunku wylicza się g, $\nabla_\tau S'_q$ i $\nabla_T S'_q$. W tym celu używa się rozwiązań zdyskretyzowanych równań stanu i sprzężonych z odpowiednimi warunkami brzegowymi. Wypełnienie warunków koniecznych optymalności osiąga się stopniowo przy coraz dokładniejszych aproksymacjach sterowania optymalnego. Dzieje się tak dzięki procesom generacji, działającym naprzemiennie z minimalizacjami kierunkowymi i redukcjami.

Parametr q jest aktualizowany każdorazowo, gdy zostaną spełnione warunki (8.41), i reinicjalizowany, gdy algorytm przechodzi przez test zakończenia (8.42). Dla stałych wartości q, kryterium (8.16) maleje monotonicznie. Całkowity algorytm, z właściwie dobranymi parametrami, zbiega się przynajmniej do minimum lokalnego kryterium pomocniczego.

Właściwy wybór parametru p w warunkach wydajności, oraz progów ε_g i ε_T w warunkach uaktualniania q, w znaczący sposób polepsza przebieg optymalizacji. Przy za dużej wartości parametru p algorytm zużywa wiele iteracji na dojście do punktu stacjonarnego w przestrzeni parametrów, chociaż porównanie kształtu gradientu i sterowania może jasno wskazywać na potrzebę dodania przełączeń. Gdy p jest za małe, częste generacje produkują nietrwałe przełączenia i w ten sposób pogarszają wydajność numeryczną.

Jeżeli progi ε_g i ε_T są zbyt łatwe do osiągnięcia, mechanizm generacji może przeoczyć optymalną strukturę i dać w wyniku rozwiązanie spełniające warunki zakończenia z za długim horyzontem. Jeżeli progi są trudne do osiągnięcia, to algorytm wykonuje pewną liczbę iteracji na próżno, utrzymując q niepotrzebnie na wartości stałej i w ten sposób hamuje postęp optymalizacji horyzontu.

9. Wybrane problemy syntezy sterowania

Eksperymenty laboratoryjne dotyczące sterowania optymalnoczasowego układami rzeczywistymi opisano w dwóch rozdziałach: 9 i 10. Rozdział 9 ma charakter wstępny. Przedstawione w nim eksperymenty optymalnoczasowe prowadzono dla dwóch systemów sterowania: porównawczego systemu wahadła na wózku opisanego w rozdziale 3 i dla systemu lewitacji magnetycznej. Dla wahadła na wózku rozwiązano na drodze symulacji zadanie syntezy regulatora optymalnego przy użyciu zbioru reprezentatywnego. W celu utworzenia optymalnych ciągów wzorcowych korzystano z algorytmu 8.1. Dla lewitacji magnetycznej, podczas eksperymentu rzeczywistego, użyto sterowania optymalnoczasowego w pętli otwartej. Do wyznaczenia tego sterowania wykorzystano algorytm 8.5.

Najbardziej istotne wyniki eksperymentów w czasie rzeczywistym, dotyczące sterowania wahadłem na wózku w pętli otwartej i zamkniętej, są wyodrębnione w rozdziale 10.

9.1. Problem optymalnoczasowy dla wahadła na wózku

Do równań stanu zapisanych przy użyciu zmiennych przeskalowanych (wzory (3.18), (3.19)) i ograniczenia na sterowanie (3.20), dołącza się równania sprzężone:

$$\begin{aligned} &\dot{\psi}_1 = 0, \quad \dot{\psi}_2 = A_{23}(x,u)\psi_3 + A_{24}(x,u)\psi_4 \\ &\dot{\psi}_3 = -\psi_1 + A_{33}\psi_3 + A_{34}\psi_4, \quad \dot{\psi}_4 = -\psi_2 + A_{43}(x)\psi_3 + A_{44}(x)\psi_4 \end{aligned} \tag{9.1}$$

gdzie:

$$\begin{aligned} A_{23}(x,u) &= (x_4^2 \cos x_2 - \cos^2 x_2 + v_2(x)\sin x_2 + f_3 \sin 2x_2)/d(x) \\ A_{24}(x,u) &= (x_4^2 \cos^2 x_2 - c_4 \cos x_2 + v_1(x,u)\sin x_2 + \\ &+ f_4 \sin 2x_2)/d(x) \\ A_{33} &= b_2/d(x), \quad A_{34} = b_2 \cos x_2/d(x) \\ A_{43}(x) &= (2x_4 \sin x_2 + b_3 \cos x_2)/d(x) \\ A_{44}(x) &= (x_4 \sin 2x_2 + b_3 c_4)/d(x) \end{aligned} \tag{9.2}$$

Wielkości $d(x)$, $v_1(x,u)$, $v_2(x)$, $f_3(x,u)$ i $f_4(x,u)$ są podane wzorami (3.19).

Twierdzenie 9.1

Niech u i $u+\Delta u$ będą dowolnymi ograniczonymi sterowaniami, całkowalnymi w kwadracie na $[0, T]$. Przez x oznaczamy rozwiązanie równania (3.18) wygenerowane przez sterowanie u, a przez ψ – rozwiązanie układu sprzężonego (9.1), odpowiadające sterowaniu u i trajektorii stanu x. Formuła (7.32) dla przeskalowanego systemu wahadła na wózku jest postaci

$$\Sigma_T(u+\Delta u)-\Sigma_T(u)=-\int_0^T g(t)\Delta u(t)\mathrm{d}t+\ o(\Delta u) \tag{9.3}$$

gdzie

$$g=\frac{\psi_3+\psi_4\cos x_2}{d(x)} \tag{9.4}$$

Dla prawie wszystkich $t\in[0,t^*]$ sterowanie optymalne spełnia relację

$$u(t)=\begin{cases} u_{\max}, & \text{jeśli} \quad \psi_3(t)+\psi_4(t)\cos x_2(t)>0 \\ -u_{\max}, & \text{jeśli} \quad \psi_3(t)+\psi_4(t)\cos x_2(t)<0 \end{cases} \blacksquare \tag{9.5}$$

Sterowanie optymalnoczasowe ma zatem charakter bang-bang (jeśli nie wystąpi osobliwość).

9.2. Aproksymacja regulatora optymalnoczasowego metodą zbiorów reprezentatywnych

W układzie drugiego rzędu synteza regulatora optymalnoczasowego sprowadza się do zapamiętania krzywej przełączeń, która dzieli płaszczyznę stanu na zbiory punktów o stałych wartościach sterowania. Dla układów o rzędzie większym od dwóch mamy do czynienia z powierzchniami przełączeń, które rozdzielają przestrzeń stanu. Trudność syntezy polega na wyznaczeniu tych powierzchni. Aproksymacja powierzchni uzyskana numerycznie, staje się nieprzydatna w końcowej fazie sterowania. Dotyczy to sytuacji, gdy trajektoria optymalna leży na tej powierzchni. Wówczas ujawnia się duża wrażliwość rozwiązań na zakłócenia sygnałów i błędy modelowania (zob. część IV).

Idea metody polega na tym, żeby dostatecznie gęsto pokryć przestrzeń stanu trajektoriami optymalnoczasowymi i z podzbioru punktów tych trajektorii utworzyć tzw. zbiór reprezentatywny, będący zbiorem par (n-wymiarowy punkt przestrzeni stanu, odpowiadająca mu wartość sterowania optymalnego). Dla każdego stanu, po odnalezieniu najbliższego punktu w zbiorze reprezentatywnym, odczytuje się stowarzyszone z nim sterowanie i stosuje jako aproksymację sterowania optymalnego.

Gdy trajektoria położy się na powierzchni przełączeń, należy przerwać wyznaczanie sterowania na podstawie zbioru reprezentatywnego trajektorii optymalnych. Od tego momentu można posłużyć się inną metodą, tak jak to zrobiono w tym rozdziale.

Sterowania optymalnoczasowe w pętli otwartej, wyliczone dla zbioru warunków początkowych, stanowią podstawę do syntezy regulatora (7.45). Do wyznaczenia sterowań i trajektorii optymalnych służą algorytmy gradientowe, zamieniające problem optymalnoczasowy na ciąg problemów z ustalonym horyzontem. Minimalizuje się odległość stanu końcowego od zbioru docelowego za pomocą gradientowych metod optymalizacji, wyliczając gradienty przy użyciu równań sprzężonych. Regulator ten jest przedstawiony numerycznie za pomocą pewnego zbioru reprezentatywnego, w czterowymiarowej przestrzeni stanów. Zbiór reprezentatywny konstruuje się dokonując selekcji punktów trajektorii optymalnych, w szczególności wybiera się punkty związane z chwilami przełączeń sterowania.

Poniżej przedstawiono metodę konstrukcji zbioru reprezentatywnego. Algorytm stosowany do generacji wzorcowych trajektorii optymalnoczasowych opisano w podrozdziale 8.1 i w następujących pracach: Turnau, Korytowski (1996, 1997); Korytowski, Szymkat, Turnau (1998).

Rozpatrzmy zadanie POST dla I systemu laboratoryjnego wahadła na wózku, którego parametry podano w tabeli 3.1. Rozważamy problem optymalnoczasowy sterowania systemu (3.7) do zbioru docelowego w przestrzeni $\boldsymbol{R}^4$ za pomocą ograniczonego sterowania (3.8). Regulatorem optymalnoczasowym jest funkcja o postaci (7.45).

Zadanie optymalnoczasowe O1 POST. Z dowolnego stanu początkowego

$x^0 \in \boldsymbol{R}^4$ spełniającego $|x_1^0| \le Z$ (Z jest połową zakresu jezdnego szyny),

przeprowadzić system (model I) w minimalnym czasie do stanu końcowego

$x^f \in \{\mathrm{col}(0,2i\pi,0,0): i = 0,\pm1,\pm2,...\}$, przy ograniczeniach na sterowanie u,

$|u| \le u_{\max} = 10.$

Położenie wózka podano w metrach, kąt wahadła w radianach i sterowanie w niutonach.

Wygenerowano 13 wzorcowych rozwiązań optymalnoczasowych. Każde z wzorcowych rozwiązań składa się ze sterowania optymalnego i trajektorii optymalnej, startującej z wybranego arbitralnie punktu początkowego. Warunki początkowe zostały tak dobrane, by w miarę gęsto pokryć obszar przestrzeni stanu, odpowiadający warunkom systemu rzeczywistego. Z uwagi na dyskretyzację przestrzeni, każda trajektoria wzorcowa składa się z 501 punktów w czterowymiarowej przestrzeni stanu. W rezultacie otrzymuje się około 6500 punktów. Z każdym punktem skojarzona jest wartość sterowania optymalnego $+u_{\max}$ lub $-u_{\max}$. Punkt zbioru wzorcowego określony jest przez piątkę liczb: cztery współrzędne wektora stanu i sterowanie z przedziału $[-u_{\max}, +u_{\max}]$.

Oznaczmy zbiory wzorcowe przez P_k, $k = 1,...,13$. Do zbioru wzorcowego nie włączamy punktów rozwiązań optymalnych z trzech ostatnich przedziałów stałości sterowania optymalnego (dla układu czwartego rzędu końcowa część trajektorii optymalnej, odpowiadająca trzem ostatnim przedziałom stałości sterowania leży na trójwymiarowej powierzchni przełączeń). Z pozostałych punktów trajektorii wybieramy pewien podzbiór celem utworzenia zbioru reprezentatywnego. Warunki początkowe 13 trajektorii zebrano w tabeli 9.1.

Tabela 9.1
Warunki początkowe trzynastu wzorców

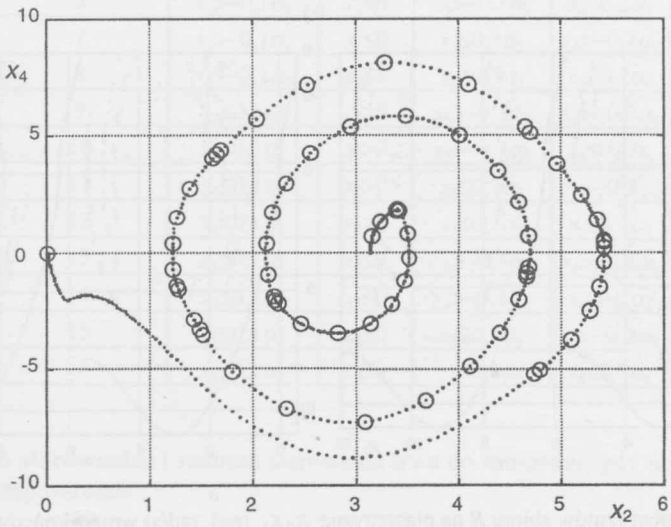
Numer wzorca	1	2	3	4	5	6	7	8	9	10	11	12	13
$x_1(0)$ [m]	0	0	0	0	0	0	0	0	0,8	0	0	0	0,8
$x_2(0)$ [rad]	$\pi/4$	$\pi/4$	π	$\pi/2$	$\pi/2$	$\pi/2$	$\pi/2$	$\pi/4$	$\pi/4$	π	$\pi/4$	$\pi/4$	π
$x_3(0)$ [m/s]	0	0	0	0	0	0,5	0	0	0	0	0	0	0
$x_4(0)$ [rad/s]	–0,8	0,8	0,8	0,8	–0,8	0	0	0,4	0	0	–0,4	0	0

Zbiór reprezentatywny tworzymy rekurencyjnie. W pierwszej iteracji tworzymy zbiór R_0, do którego włączamy wszystkie pary punktów, pomiędzy którymi następuje przełączenie sterowania, a nawet trójki, jeżeli para nie określa dokładnie przełączenia. W dalszych iteracjach tworzymy zbiory R_j, $j = 1,2,...$, które powstają przez dołączanie kolejnych punktów ze zbiorów P_k, $k = 1,...,13$. Punkt p jest dołączany do zbioru R_j, jeśli jego odległość od tego zbioru jest dostatecznie duża

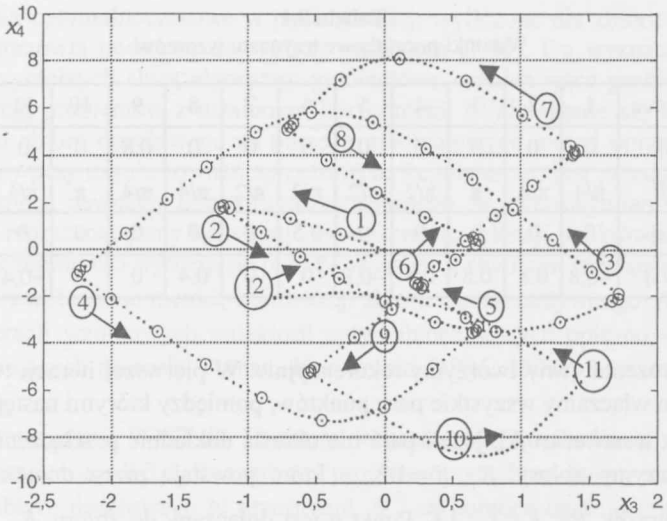
$$d(p, R_j) \geq \varepsilon(1 + 2p_3^2 + 2p_4^2) \tag{9.6}$$

W ten sposób powstaje zbiór R_{j+1}. Zbiór reprezentatywny R otrzymujemy po wyczerpaniu wszystkich punktów ze zbiorów P_k.

Na rysunkach 9.1–9.2 przedstawiono punkty odpowiadające wzorcowi numer 10 (P_{10}).



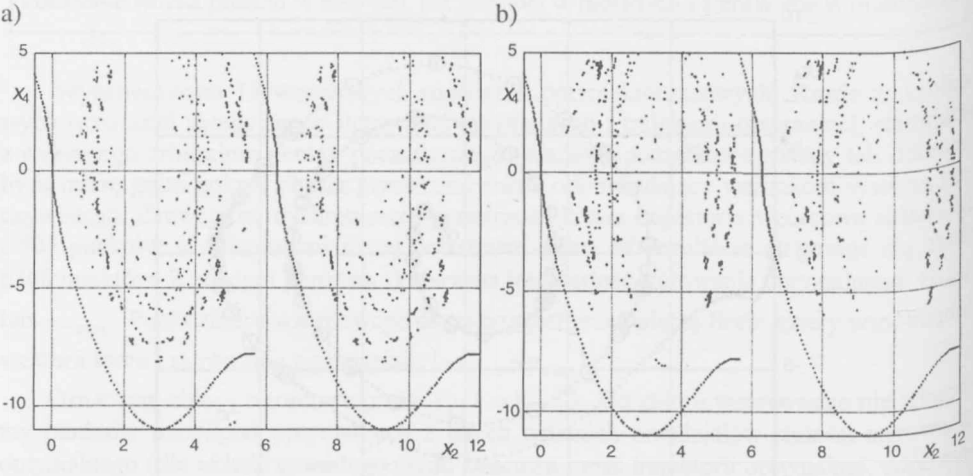
Rys. 9.1. Punkty podzbioru P_{10} włączone do zbioru R; rzut na płaszczyznę fazową wahadła



Rys. 9.2. Punkty podzbioru P_{10} włączone do zbioru R; rzut na płaszczyznę prędkości wahadła i wózka

Każde z przełączeń jest reprezentowane przez dwa lub trzy punkty. Kółkami zaznaczono punkty włączone do zbioru reprezentatywnego R.

Przyjmując w algorytmie sortującym wartość parametru $\varepsilon = 0{,}1$; otrzymujemy zbiór reprezentatywny R złożony z 1125 punktów. Fragmenty rzutów zbioru R na płaszczyznę x_2x_4 pokazano na rysunku 9.3.



Rys. 9.3. Fragment rzutów zbioru R na płaszczyznę x_2x_4 (rad, rad/s) wraz z końcowymi częściami trajektorii optymalnoczasowej; a) sterowanie +10 N; b) sterowanie −10 N

Uwzględnienie dwóch symetrii sterowania optymalnoczasowego prowadzi do czterokrotnego powiększenia ilości punktów zbioru R. Pierwsza symetria dotyczy znaku zmiennych stanu. Należy dołączyć punkty trajektorii o znaku przeciwnym i związane z tymi punktami sterowania, też o znaku przeciwnym. Druga symetria jest związana z przesunięciem kąta wahadła o całkowitą wielokrotność kąta 2π. Otrzymujemy $4 \times 1125 = 4500$ punktów zbioru.

I stadium sterowania. Synteza regulatora optymalnoczasowego ma dać w efekcie algorytmiczną procedurę z możliwością określenia sterowania w czasie krótszym od 0,01 sekundy (jest to okres próbkowania pomiarów i generowania sterowań w komputerze). W podejściu wykorzystującym zbiór reprezentatywny R wartość sterowania dla danego punktu przestrzeni stanu jest taka sama jak dla najbliższego punktu ze zbioru R. Potrzebna jest więc procedura wyszukania tego najbliższego punktu. Aby nie przeglądać wszystkich punktów zbioru R, dzieli się go na 16 podzbiorów (nie całkowicie rozłącznych – o wspólnych częściach w otoczeniach brzegowych). Sposób podziału podano w tabeli 9.2.

Tabela 9.2

Podział zbioru reprezentatywnego na podzbiory; $\alpha_1 = 0{,}4667$, $\alpha_3 = 1{,}504$, $\alpha_4 = 3{,}223$ są współczynnikami skalowania (3.11) i (3.12), opisanymi też w pracy Turnaua, Korytowskiego (1996)

Numer zbioru	Warunki przynależności punktu do zbioru			
1	$x_1 > -0{,}1\alpha_1$	$x_2 > 0$	$x_3 > -0{,}3\alpha_3$	$x_4 > -0{,}2\alpha_4$
2	$x_1 > -0{,}1\alpha_1$	$x_2 > 0$	$x_3 > -0{,}3\alpha_3$	$x_4 \le 0{,}2\alpha_4$
3	$x_1 > -0{,}1\alpha_1$	$x_2 > 0$	$x_3 \le 0{,}3\alpha_3$	$x_4 > -0{,}2\alpha_4$
4	$x_1 > -0{,}1\alpha_1$	$x_2 > 0$	$x_3 \le 0{,}3\alpha_3$	$x_4 \le 0{,}2\alpha_4$
5	$x_1 > -0{,}1\alpha_1$	$x_2 \le 0$	$x_3 > -0{,}3\alpha_3$	$x_4 > -0{,}2\alpha_4$
6	$x_1 > -0{,}1\alpha_1$	$x_2 \le 0$	$x_3 > -0{,}3\alpha_3$	$x_4 \le 0{,}2\alpha_4$
7	$x_1 > -0{,}1\alpha_1$	$x_2 \le 0$	$x_3 \le 0{,}3\alpha_3$	$x_4 > -0{,}2\alpha_4$
8	$x_1 > -0{,}1\alpha_1$	$x_2 \le 0$	$x_3 \le 0{,}3\alpha_3$	$x_4 \le 0{,}2\alpha_4$
9	$x_1 \le 0{,}1\alpha_1$	$x_2 > 0$	$x_3 > -0{,}3\alpha_3$	$x_4 > -0{,}2\alpha_4$
10	$x_1 \le 0{,}1\alpha_1$	$x_2 > 0$	$x_3 > -0{,}3\alpha_3$	$x_4 \le 0{,}2\alpha_4$
11	$x_1 \le 0{,}1\alpha_1$	$x_2 > 0$	$x_3 \le 0{,}3\alpha_3$	$x_4 > -0{,}2\alpha_4$
12	$x_1 \le 0{,}1\alpha_1$	$x_2 > 0$	$x_3 \le 0{,}3\alpha_3$	$x_4 \le 0{,}2\alpha_4$
13	$x_1 \le 0{,}1\alpha_1$	$x_2 \le 0$	$x_3 > -0{,}3\alpha_3$	$x_4 > -0{,}2\alpha_4$
14	$x_1 \le 0{,}1\alpha_1$	$x_2 \le 0$	$x_3 > -0{,}3\alpha_3$	$x_4 \le 0{,}2\alpha_4$
15	$x_1 \le 0{,}1\alpha_1$	$x_2 \le 0$	$x_3 \le 0{,}3\alpha_3$	$x_4 > -0{,}2\alpha_4$
16	$x_1 \le 0{,}1\alpha_1$	$x_2 \le 0$	$x_3 \le 0{,}3\alpha_3$	$x_4 \le 0{,}2\alpha_4$

II stadium sterowania. I stadium sterowania trwa do momentu, gdy po raz pierwszy zostanie spełniony warunek

$$x_4 = 0 \quad \text{i} \quad \cos x_2 > 0{,}6 \tag{9.11}$$

Algorytm sterujący przestaje posługiwać się zbiorem reprezentatywnym. Kończy się stadium I, w którym trajektoria optymalna przebijała jedynie powierzchnię przełączeń mając z nią izolowane punkty wspólne. Wartość 0,6 pochodzi z obserwacji, z których wynika, że dla wszystkich ważnych z praktycznego punktu widzenia trajektorii ostatnie lokalne maksimum funkcji $\cos x_2$ zawiera się w przedziale od 0,64 do 0,71. Formuła sterująca w II stadium, powstała na podstawie obserwacji wzorców optymalnych i suboptymalnych przebiegów opisanych w podrozdziale 5.2.oraz w pracy Turnaua (1995), jest postaci

$$u = u_{\max} \operatorname{sign}(x_4 \cos x_2) \tag{9.12}$$

Taka aproksymacja dobrze przybliża sterowanie optymalnoczasowe w każdym z rozpatrywanych przypadków.

III stadium sterowania. Celem jest osiągnięcie trajektorii optymalnoczasowej, przechodzącej przez początek układu współrzędnych. Na ogół trajektoria generowana w II stadium sterowania nie przecina tej trajektorii. Pewne jest natomiast, że przetną się rzuty na płaszczyznę x_2x_4, trajektorii generowanej i trajektorii optymalnoczasowej, przechodzącej przez początek układu współrzędnych. Gwarantowane jest osiągnięcie punktu $(x_1,0,x_3,0)$. Jeżeli x_1 i x_3 są różne od zera, to przejście do punktu (0,0,0,0) – zadanie nazwane *ruchem akrobaty* – jest realizowane przez regulator proporcjonalny.

Stadium II kończy się, gdy trajektoria stanu spełni warunek

$$\operatorname{dist}(x(t), Q) < 0{,}2 \tag{9.13}$$

gdzie $Q = Q^+ \cup Q^-$ jest powierzchnią w R^4 określoną następująco. Niech

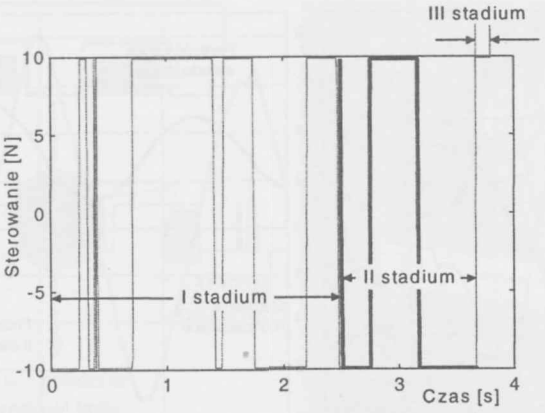
$$y_i = g_i(s, \pm u_{\max}), \quad i = 1,...,4, \quad s \geq 0 \tag{9.14}$$

będzie parametrycznym równaniem trajektorii kończącej się w początku układu współrzędnych, generowanej przez stałe sterowanie $\pm u_{\max}$. Wówczas

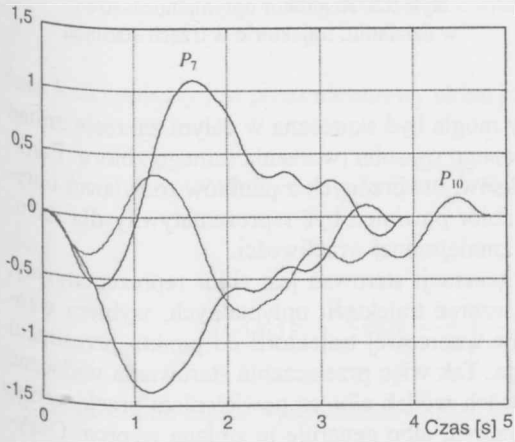
$$Q^{\pm} = \left\{ y \in R^4 : y_2 = g_2(s, \pm u_{\max}),\ y_4 = g_4(s, \pm u_{\max}),\ s \geq 0 \right\} \tag{9.15}$$

Sterowanie przyjmuje wartość $+u_{\max}$ lub $-u_{\max}$, zależnie od tego, czy odległość od zbioru Q: $\operatorname{dist}(x(t), Q)$ została osiągnięta na zbiorze Q^+ czy Q^-. Stadium III trwa tak długo, jak długo spełniony jest warunek $\operatorname{dist}(x(t), Q) < 0{,}2$. Jeżeli z jakichś powodów zostanie on przekroczony, sterowanie wraca do stadium II. (Uwaga ta dotyczy modelu symulacyjnego. W układzie rzeczywistym, by zapobiec drganiom mechanicznym, można by zastosować regulator proporcjonalny).

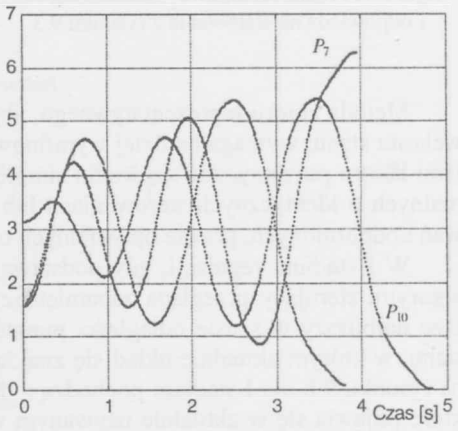
Wszystkie stadia sterowania przedstawiono na rysunku 9.4. Na rysunkach 9.5–9.7 pokazano wzorce P_7 i P_{10} położenia wózka, kąta i prędkości kątowej wahadła oraz odpowiedzi modelu na sterowanie z rysunku 9.4.



Rys. 9.4. Trzy stadia sterowania (II stadium oznaczono linią grubą); $x(0) = (0,\pi/2,0,0,5)$



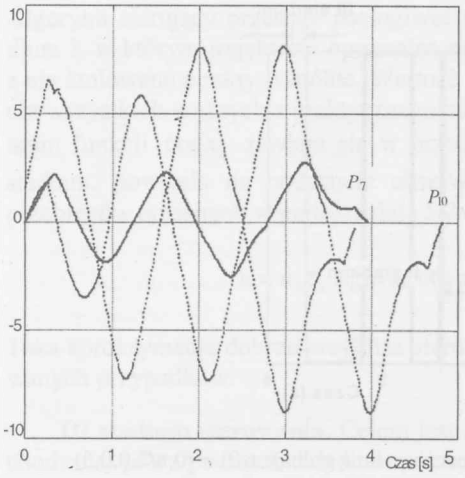
Rys. 9.5. Położenie wózka – wzorce P_7 i P_{10} oraz odpowiedź na sterowanie z rysunku 9.3



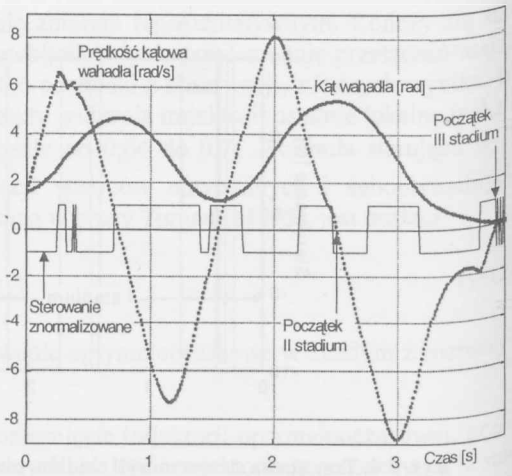
Rys. 9.6. Kąt wahadła – wzorce P_7 i P_{10} oraz odpowiedź na sterowanie z rysunku 9.3

Na rysunku 9.8 pokazano regulator optymalnoczasowy w działaniu. Należy zwrócić uwagę, że podobieństwo generowanych sterowań nie zachodzi ani w stadium I, ani też w stadium III. Koncepcyjne ustalenie w I stadium sterowania reguł, bazujących na pomiarze aktualnego wektora stanu i zbliżonych działaniem do sterowania optymalnoczasowego, jest raczej niemożliwe bez użycia zbioru reprezentatywnego, a więc bez wzorców rozwiązań problemu optymalnego, uzyskanych na drodze optymalizacji.

Wielokrotnie powtórzony eksperyment symulacyjny, dla różnych warunków początkowych, uwiarygadnia tezę postawioną na wstępie rozdziału, a mianowicie, że w I stadium sterowania zbiór reprezentatywny może być z powodzeniem wykorzystany do generacji sterowania optymalnego.



Rys. 9.7. Prędkości kątowe – wzorce P_7, P_{10} i odpowiedź na sterowanie z rysunku 9.3



Rys. 9.8. Regulator optymalnoczasowy w działaniu; trajektorie w trzech stadiach

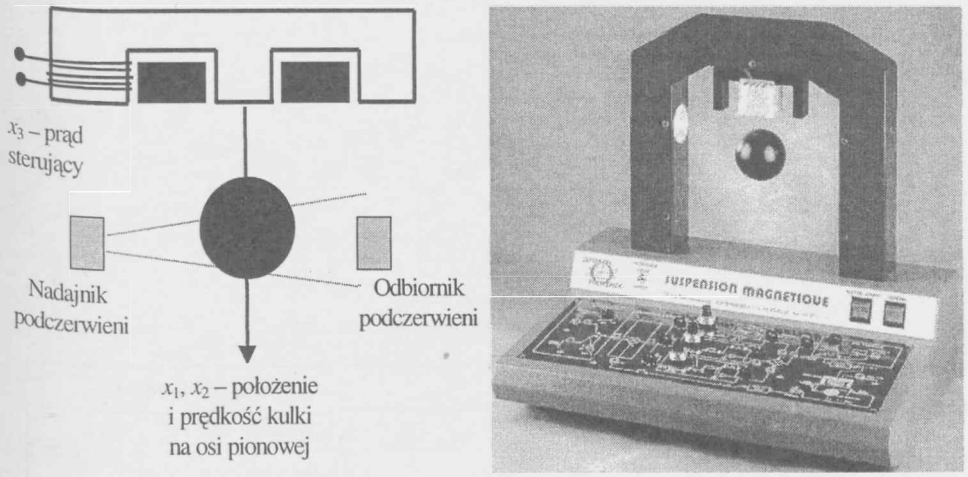
Metoda zbioru reprezentatywnego, aby mogła być skuteczna w całym zakresie zmian wektora stanu, wymaga bardziej wyrafinowanego sposobu tworzenia samego zbioru. Punktami zbioru powinny stać się środki simpleksów, utworzonych z punktów rozwiązań optymalnych o identycznych sterowaniach lub zbiór powinien być reprezentatywny dla sterowań uodpornionych, prawie optymalnych o zmniejszonej wrażliwości.

W I stadium regulacji, gdy podstawą generacji sterowań jest zbiór reprezentatywny, algorytm sterujący przegląda zapamiętane wzorce trajektorii optymalnych, wybiera wzorzec najbliższy w sensie odległości punktów wzorcowej trajektorii od punktu przestrzeni stanu, w którym aktualnie układ się znajduje. Tak więc przełączenia sterowania widoczne na rysunku 9.8 dla I stadium pochodzą z dwóch źródeł: albo są powieleniem przełączenia, które pojawia się w aktualnie używanym wzorcu, albo generuje je zmiana wzorca. Oczywiście, zmiana wzorca nie musi powodować zmiany znaku sterowania.

Metoda zbiorów reprezentatywnych staje się nadmiernie wrażliwa, gdy trajektorie osiągają powierzchnię przełączeń. Zbiór reprezentatywny jest wówczas zbyt rzadki dla dobrej aproksymacji regulatora.

9.3. Przykład: lewitacja magnetyczna

Rozważmy system lewitacji magnetycznej, pokazany na rysunku 9.9. Praktyczna realizacja algorytmu optymalnoczasowego w systemie wymaga zachowania reżimów czasu rzeczywistego. Co okres $\Delta t = 0{,}67$ ms, bez przyspieszeń i opóźnień, zdejmuje się pomiary i generuje sterowanie. Narzędzia programowo-sprzętowe (przybornik RTWT MATLAB-a) umożliwiają takie działanie i pozwalają równocześnie na strojenie na obiekcie rzeczywistym sterowania, otrzymanego uprzednio podczas optymalizacji *off line*.



Rys. 9.9. System lewitacji magnetycznej

Ruch kulki opisany jest przez nieliniowy układ równań

$$\dot{x}_1 = x_2, \qquad \dot{x}_2 = \frac{L'(x_1)}{2m} x_3^2 + g, \qquad \dot{x}_3 = \frac{1}{T}(ku - i_s - x_3) \tag{9.16}$$

gdzie

$$L'(x_1) = -\frac{L_0}{s} \exp\left(-\frac{x_1}{s}\right).$$

Dla uproszczenia zapisu wprowadzamy oznaczenia

$$a = -\frac{1}{s}, \quad b = -\frac{L_0}{2ms}, \quad c = -\frac{1}{T} \tag{9.17}$$

$$v = \frac{ku - i_s}{T}, \quad v_{\min} = \frac{ku_{\min} - i_s}{T}, \quad v_{\max} = \frac{ku_{\max} - i_s}{T} \tag{9.18}$$

Zatem

$$u = \frac{Tv + i_s}{k} \tag{9.19}$$

Równania systemu w nowych oznaczeniach są postaci

$$\dot{x}_1 = x_2, \qquad \dot{x}_2 = bx_3^2 \exp(ax_1) + g, \qquad \dot{x}_3 = cx_3 + v \tag{9.20}$$

Otrzymujemy punkty równowagi przy stałym u

$$x_1 = \frac{1}{a} \ln \frac{-gc^2}{bv^2}, \quad x_2 = 0, \quad x_3 = -v/c \tag{9.21}$$

Parametry modelu zebrano w tabeli 9.3.

Tabela 9.3
Parametry systemu magnetycznej lewitacji

Parametr	Wartość i jednostki
masa kulki	$m = 0{,}06$ kg
przyspieszenie ziemskie	$g = 9{,}81$ m/s^2
stała przetwornika	$k = 0{,}29703779960998$ A/V
stała czasowa obwodu elektrycznego	$T = 0{,}01069365993381$ s
prąd statyczny	$i_s = 1{,}50595851172854$ A
indukcyjność cewki dla $x_1 = 0$	$L_0 = 0{,}10912926393443$ H
punkt przecięcia stycznej indukcyjności z osią x_1	$s = 0{,}00773746617018$ m
zadana wartość minimalna napięcia sterującego	$u_{\min} = 5{,}56$ V
zadana wartość maksymalna napięcia sterującego	$u_{\max} = 9{,}56$ V
przedział zmienności położenia kulki	$x_1 \in [0\ 0{,}023]$ m

Odległość x_1 jest mierzona od dolnej powierzchni elektromagnesu do powierzchni kulki. Zmienna x_2 oznacza prędkość kulki w pionie. Odległość x_1 mierzy się za pomocą detektora podczerwieni. Eksperyment pokazuje, że przetwornik napięciowo-prądowy, którego integralną część stanowi cewka, wprowadza własną dynamikę. Założenie, że prąd x_3 w cewce jest proporcjonalny do napięcia sterującego u, upraszczające syntezę regulatora optymalnoczasowego, nie jest możliwe do spełnienia w przypadku systemu z rysunku 9.9. Uwzględnienie dynamiki obwodu prądowego prowadzi do modelu trzeciego rzędu (Reif *et al.* 1997). W takim przypadku synteza regulatora optymalnoczasowego znacznie się komplikuje (Pao, Franklin 1993). W wyjątkowych przypadkach możliwa jest analityczna synteza regulatora (Li, Bainum 1994; Pao 1996), lecz na ogół jedyną drogą do wyznaczenia sterowania optymalnoczasowego jest optymalizacja numeryczna.

Stawiamy zadanie optymalnoczasowe ze stanem początkowym x^0 i stanem docelowym x^f. Wprowadzamy pomocniczy wskaźnik jakości, zależny od horyzontu T

$$S(v,T) = \tfrac{1}{2}(x(T) - x^f)^{\mathrm{T}}(x(T) - x^f) \tag{9.22}$$

Hamiltonian zapisujemy jako

$$H = \psi_1 x_2 + \psi_2 \left(b x_3^2 \exp(a x_1) + g \right) + \psi_3 (c x_3 + v) \tag{9.23}$$

Warunek maksimum hamiltonianu daje

$$v = \begin{cases} v_{\min}, & \text{gdy } \psi_3 < 0 \\ v_{\max}, & \text{gdy } \psi_3 > 0 \end{cases} \tag{9.24}$$

Otrzymujemy równania sprzężone

$$\begin{aligned} \dot{\psi}_1 &= -ab \exp(a x_1) x_3^2 \psi_2, \qquad \dot{\psi}_2 = -\psi_1 \\ \dot{\psi}_3 &= -2b \exp(a x_1) x_3 \psi_2 - c \psi_3 \end{aligned} \tag{9.25}$$

z warunkiem końcowym

$$\psi(T) = x^f - x(T) \tag{9.26}$$

Równania sprzężone można zapisać w postaci

$$\dot{\psi} = A \psi \tag{9.27}$$

gdzie

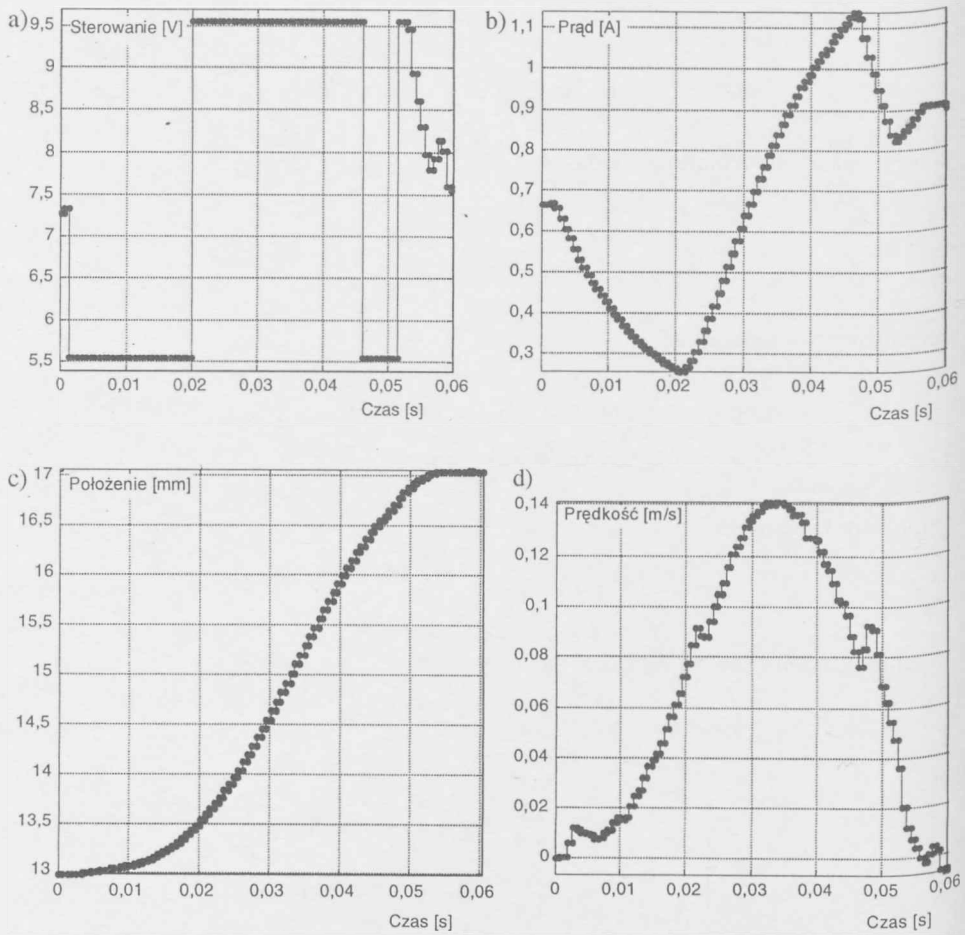
$$A = \begin{bmatrix} 0 & A_{12} & 0 \\ -1 & 0 & 0 \\ 0 & A_{32} & -c \end{bmatrix}, \qquad \begin{aligned} A_{12} &= -ab \exp(a x_1) x_3^2, \\ A_{32} &= -2b \exp(a x_1) x_3. \end{aligned}$$

Równanie wariacyjne dla równań stanu ma postać (zob. rozdział 11)

$$\delta \dot{x} = -A^{\mathrm{T}} \delta x \tag{9.28}$$

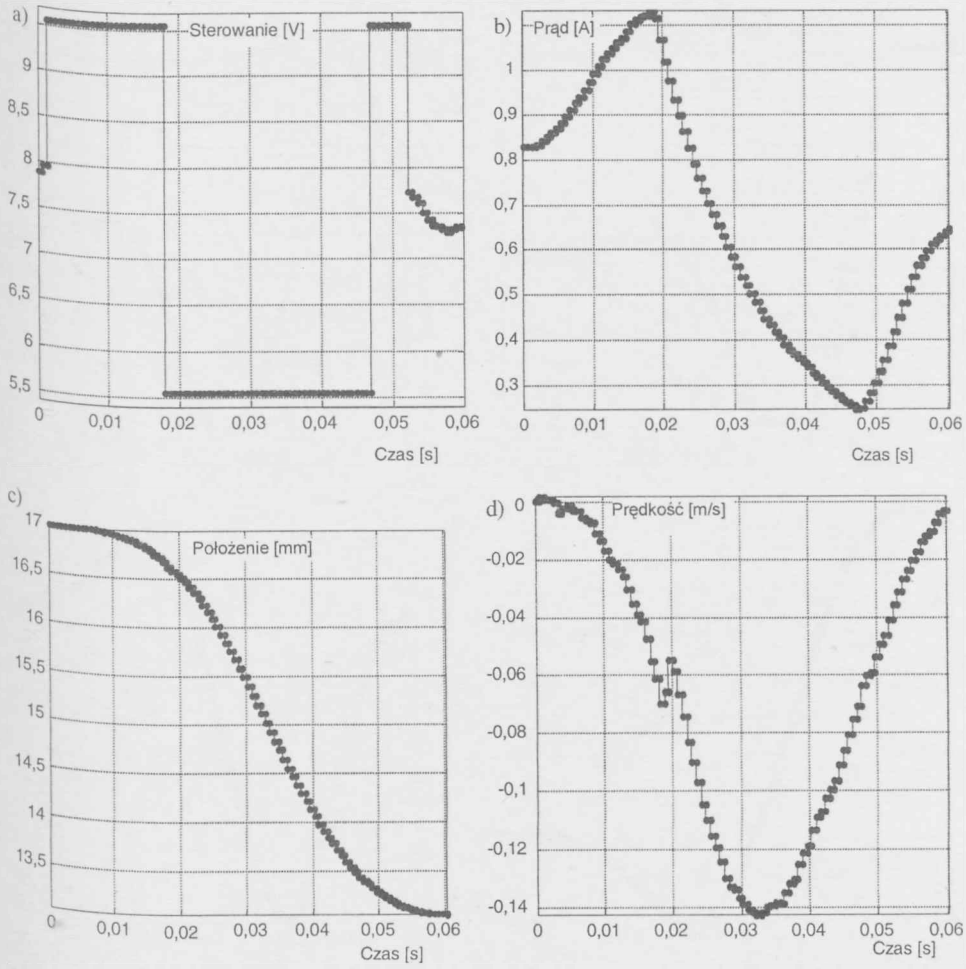
Przy ograniczeniach technicznych systemu lewitacji, w którym nie można sforsować prądu w cewce tak, by uzyskać przebieg zbliżony do bangbangowego, konieczne staje się uwzględnienie dynamiki obwodu prądowego. Wprawdzie w pracy Turnaua, Kołka (1998) przedstawiono regulator optymalnoczasowy, gdzie dynamikę tę pominięto i otrzymano eksperymentalnie krzywą przełączeń traktując układ lewitacji jako system drugiego rzędu, ale wyniki takiej regulacji nie były dokładne.

Eksperymenty optymalnoczasowego sterowania lewitującą kulką przeprowadzono na modelu laboratoryjnym (rys. 9.9). Pierwszy eksperyment polegał na przemieszczeniu układu z położenia równowagi z $x_1(0) = 13$ mm, do położenia równowagi x^f, $x_1^f = 17$ mm (kulka jest opuszczana w dół). Otrzymane przebiegi czasowe sterowania optymalnego, prądu w cewce, położenia i prędkości kulki przedstawiono na rysunku 9.10.



Rys. 9.10. Wyniki eksperymentu optymalnoczasowego przemieszczenia kulki w dół: a) sterowanie (napięcie cewki); b) prąd w cewce; c) położenie kulki; d) prędkość kulki

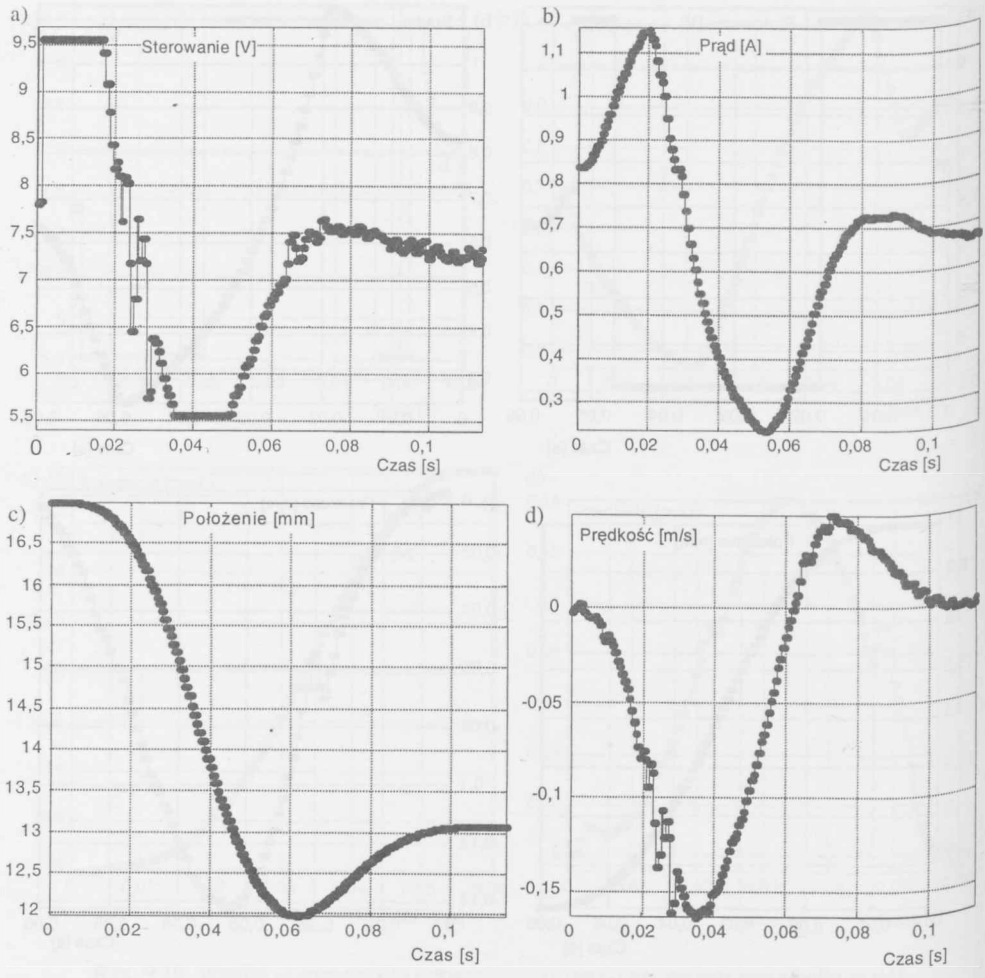
W drugim eksperymencie kulka jest podobnie przemieszczana w górę, od położenia $x_1(0) = 17$ mm do $x_1^f = 13$ mm (rys. 9.11). Lewitująca kulka utrzymywana jest na stałych poziomach 13 lub 17 mm przez regulator liniowo-kwadratowy (z nasyceniem sterowania). Przemieszczenie między poziomami odbywa się pod działaniem sterowania optymalnoczasowego w pętli otwartej. Sterowanie to jest wyliczone przy użyciu algorytmu opisanego w podrozdziale 8.3. Ma ono charakter bang-bang z dwoma czasami przełączeń. Przeprowadzono strojenie wartości tych czasów podczas rzeczywistych eksperymentów z systemem i ostatecznie otrzymano czasy τ_1 i τ_2 nieznacznie różniące się od wyliczonych. Rozbieżności wynikają z dyskretyzacji pomiaru i sterowania, a także z zakłóceń wprowadzanych przez czujnik odległości kulki od cewki. Zakłócenia te można zaobserwować na wykresie prędkości kulki (rysunki 9.10, 9.11 i 9.12). Występują one w chwilach zmian sterowania.



Rys. 9.11. Wyniki eksperymentu optymalnoczasowego przemieszczenia kulki w górę: a) sterowanie (napięcie cewki); b) prąd w cewce; c) położenie kulki; d) prędkość kulki

Ten pierwszy przykład ukazuje możliwość bezpośredniej syntezy sterowania optymalnoczasowego dla układu niskiego rzędu, w którym nie występuje zjawisko tarcia. Synteza regulatora optymalnoczasowego wymaga użycia efektywnego algorytmu numerycznego.

Dla porównania jakości sterowania, wykonano dodatkowo eksperyment regulacji liniowo-kwadratowej (z nasyceniem sterowania), przy przemieszczeniu kulki w górę z poziomu 17 do poziomu 13 mm. Wyniki tego sterowania pokazuje rysunek 9.12. Czas sterowania w tym przypadku wydłuża się znacznie (prawie dwukrotnie) w stosunku do optymalnego. Przemieszczenie kulki między poziomami trwa w tym przypadku około 90 ms. Regulator proporcjonalny, którym się posłużono w eksperymencie, miał wektor wzmocnień $K = (-1730\ -30{,}7\ 5{,}42)$. Czasy przełączeń i horyzonty sterowania, zebrano w tabeli 9.4. Czasy wyliczone w algorytmie 8.5 poddano dostrajaniu podczas eksperymentów.



Rys. 9.12. Wyniki eksperymentu przemieszczenia kulki w górę przy użyciu regulatora liniowo-kwadratowego: a) sterowanie (napięcie cewki); b) prąd w cewce; c) położenie kulki; d) prędkość kulki

Tabela 9.4
Czasy przełączeń i horyzonty sterowania wyliczone w algorytmie 8.5 i stosowane w eksperymencie

Czasy przełączeń i horyzont	**Ruch kulki w dół od poziomu 13 mm do poziomu 17 mm**		**Ruch kulki w górę od poziomu 17 mm do poziomu 13 mm**	
	wyliczone	zastosowane	wyliczone	zastosowane
τ_1 [ms]	19,51061799067886	18,76	18,16931940014038	16,75
τ_2 [ms]	46,12196262851364	44,89	45,60019740109170	45,56
T [ms]	50,60221160401352	50,25	51,02775770639569	52,26

10. Sterowanie optymalnoczasowe wahadłem na wózku w czasie rzeczywistym

Sterowanie optymalnoczasowe układami nieliniowymi w czasie rzeczywistym jest zadaniem trudnym, wymagającym dokładnej identyfikacji dynamiki sterowanego obiektu i przeprowadzenia wstępnych wyliczeń trajektorii optymalnych. Problem sterowania jest najlepiej widoczny w lotnictwie, gdzie komputery pokładowe wyposaża się w algorytmy, modele i przygotowane off line trajektorie ruchu (Pesch 1994).

Eksperymenty przedstawione w tym rozdziale odnoszą się do sterowania w czasie rzeczywistym. W pierwszej części zawarto wyniki sterowania w pętli otwartej. Podczas eksperymentu i adaptacji otrzymanego sterowania stosowane są algorytmy z podrozdziału 8.2. W drugiej części pokazano użycie algorytmu optymalnoczasowego do sterowania w pętli zamkniętej w czasie rzeczywistym. Przykład ten to jeden z najważniejszych wyników tej pracy.

10.1. Sterowanie w pętli otwartej

Postawiono następujące zadanie optymalnoczasowe.

Zadanie optymalnoczasowe O2 POST + AKRO. Ze stanu początkowego

$$x^0 = \mathrm{col}(-0{,}25, \pi, 0, 0)$$

przeprowadzić system wahadła na wózku (model II), w minimalnym czasie, do stanu

$$x^f = \mathrm{col}(0{,}15, 0, 0, 0),$$

przy ograniczeniach na sterowanie u,

$$|u| \le u_{\max} = 4.$$

Położenie wózka podano w metrach, kąt wahadła w radianach i sterowanie w niutonach.

Model dynamiki ruchu winien być na tyle dokładny, by przybliżał dynamikę wahadła na wózku w szerokim zakresie zmian warunków początkowych i sterowania. Aby zwiększyć szansę powodzenia eksperymentu prowadzonego w układzie otwartym, zastosowano kompensację tarcia statycznego wózka (zob. piąta linia uproszczonego algorytmu regułowego z tabeli 10.2). W modelu pominięto dynamikę silnika. W zamian wprowadzono kompensacyjne sprzężenie zwrotne od prędkości wózka (10.1), co pozwoliło na uzyskanie zadowalającej zgodności modelu z obiektem.

Napięcie wejściowe silnika $v(t)$ wyliczano następująco (zob. rys. 10.22)

$$v(t) = k_0 x_3(t) + k_1 u(t) \tag{10.1}$$

gdzie:

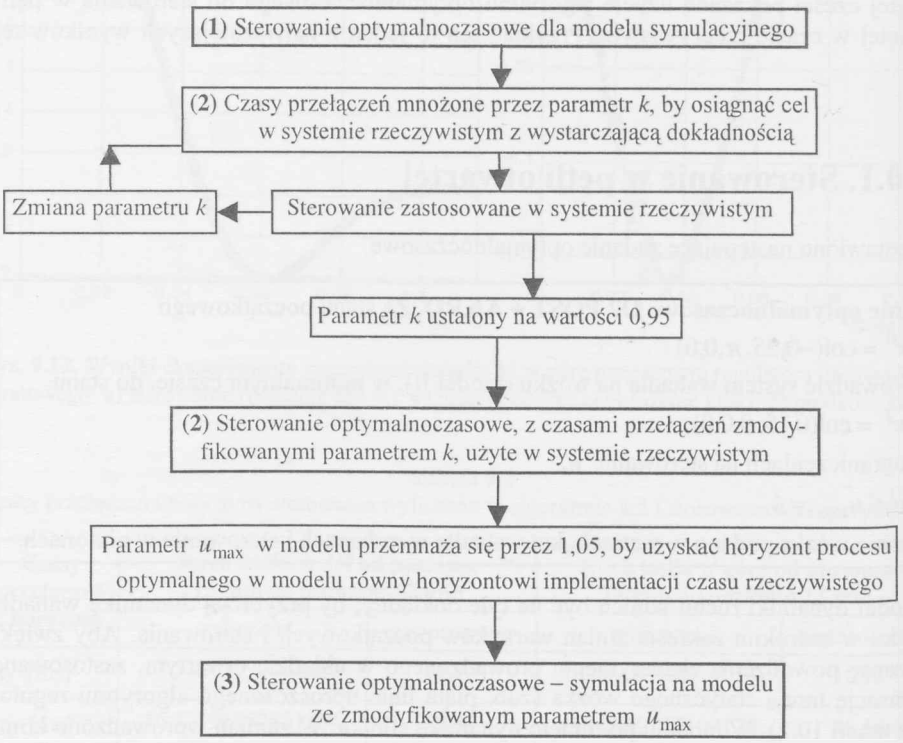
$u(t)$ – skorygowane (według schematu z rysunku 10.1) sterowanie uzyskane z rozwiązania zadania optymalnoczasowego,

$x_3(t)$ – prędkość wózka identyfikowana z pomiarów rzeczywistych,

k_0 i k_1 – stałe określone podczas rzeczywistych eksperymentów sterowania systemem.

Eksperyment realizujący zadanie O2 POST+AKRO powtórzono 100 razy i w 70 przypadkach uzyskano wynik pozytywny, pomimo wysokiej wrażliwości trajektorii optymalnej na zakłócenia, szczególnie w końcowej fazie zadania (Turnau, Korytowski, Szymkat 1999). Zwiększenie pewności działania algorytmu rzeczywistego uzyskuje się przez adaptację czasów przełączeń sterowania. Wyznaczanie nowego sterowania optymalnego odbywa się dla wybranych punktów trajektorii rzeczywistej, które odbiegają od odpowiadających im punktów trajektorii modelowej.

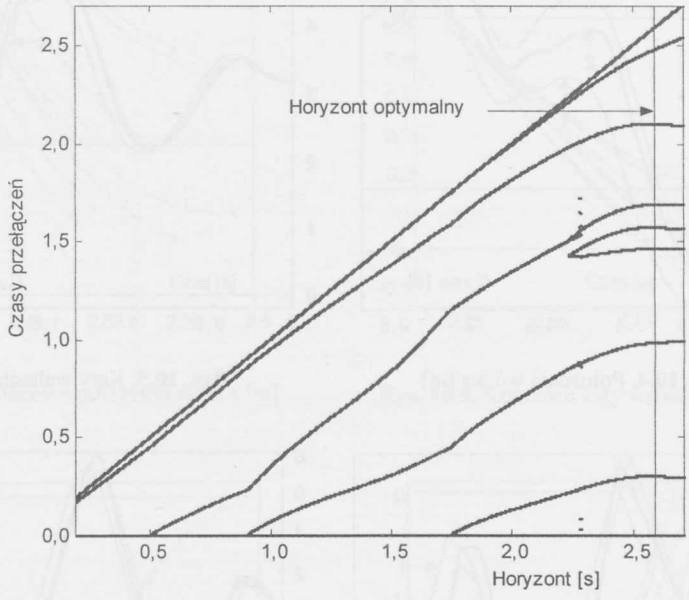
Na rysunku 10.1 podano schemat eksperymentu sterowania w pętli otwartej.



Rys. 10.1. Schemat eksperymentu prowadzonego w systemie rzeczywistym

W kroku (1), korzystając z algorytmu 8.5, opisanego także w pracy (Szymkat, Korytowski, Turnau 1999a), dla przeskalowanego modelu dynamiki systemu wahadła na wózku II wyliczano sterowanie optymalnoczasowe. Sterowanie to stosowano w pętli otwartej w laboratoryjnym systemie wahadła na wózku. Skracano proporcjonalnie czasy przełączeń (zmienny parametr k na rysunku 10.1), tak by wahadło osiągało górny punkt równowagi z wystarczającą dokładnością.

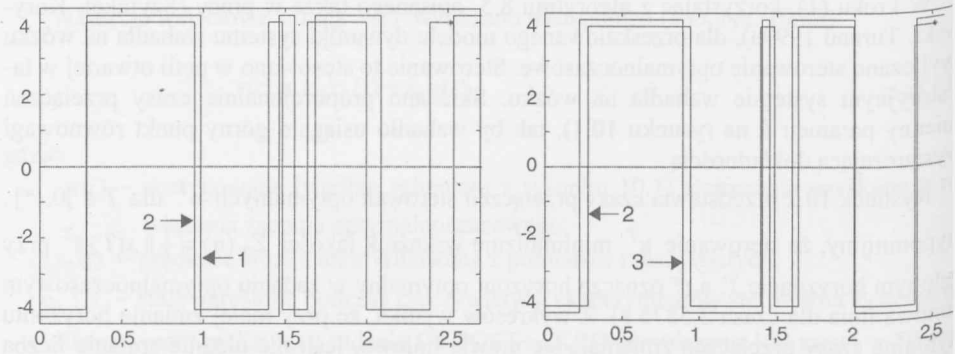
Rysunek 10.2 przedstawia czasy przełączeń sterowań optymalnych u^T dla $T \in]0, t^*]$. Przypomnijmy, że sterowanie u^T minimalizuje wskaźnik jakości $\Sigma_T(u) = \frac{1}{2} \| x(T) \|^2$ przy ustalonym horyzoncie T, a t^* oznacza horyzont optymalny w zadaniu optymalnoczasowym (pionowa linia dla czasu 2,5875 s). Z wykresów wynika, że przy małej zmianie horyzontu optymalne czasy przełączeń zmieniają się prawie liniowo, jeśli nie ulegnie zmianie liczba przełączeń. Na rysunku 10.2 widać generację nowych przełączeń i ich kontynuację przy wzrastajacym horyzoncie. Widać też redukcję takich przełączeń nadmiarowych, które nie zostały zachowane przy kolejnych optymalizacjach.



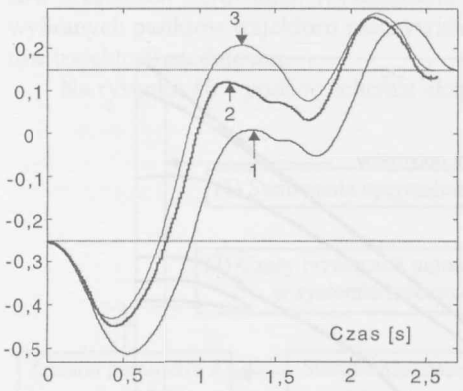
Rys. 10.2. Czasy przełączeń sterowania stałohoryzontowego w funkcji horyzontu

Dopasowanie sterowania z modelu do sterowania obiektem rzeczywistym, polegające na proporcjonalnym skracaniu lub wydłużaniu czasów przełączeń, znajduje więc uzasadnienie, szczególnie, że do strojenia regulatora używa się jednego parametru k. Ostatecznie na drodze eksperymentu rzeczywistego wybrano współczynnik proporcjonalności k o wartości liczbowej 0,95, przez który pomnożono czasy przełączeń uzyskane w kroku (1).

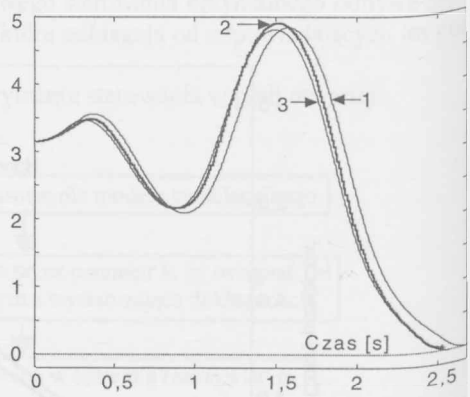
Symulacyjne i rzeczywiste przebiegi podano na rysunkach od 10.3 do 10.7.



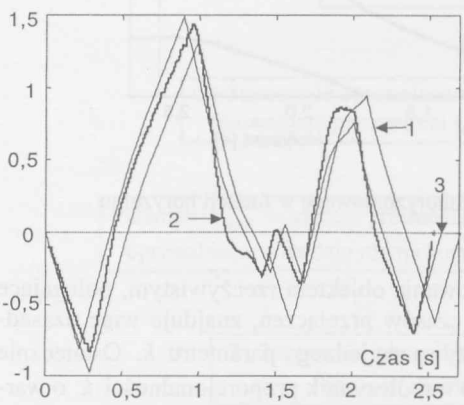
Rys. 10.3. Sterowania (siły) [N] w funkcji czasu [s]



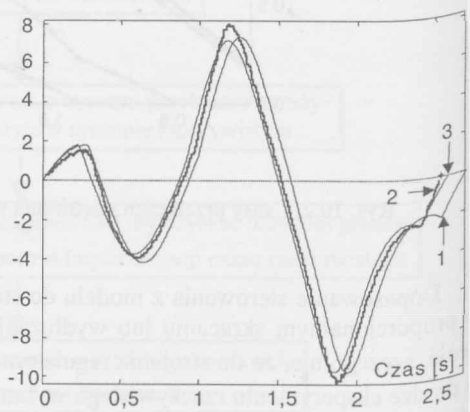
Rys. 10.4. Położenia wózka [m]



Rys. 10.5. Kąty wahadła [rad]



Rys. 10.6. Prędkości wózka [m/s]

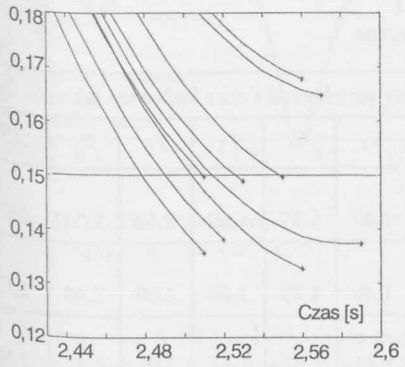


Rys. 10.7. Prędkości wahadła [rad/s]

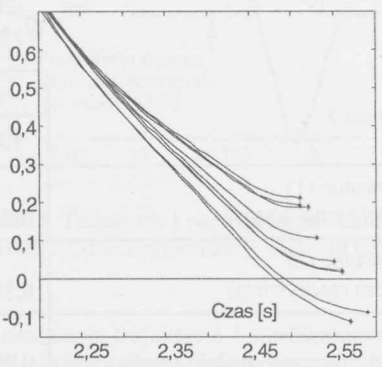
Strzałki z cyframi oznaczają wyniki uzyskane w kolejnych krokach eksperymentu przedstawionego schematycznie na rysunku 10.1. Strzałki z cyfrą 1 wskazują przebiegi w modelu pierwotnym, z cyfrą 3 – przebiegi w modelu zmodyfikowanym na podstawie wyników eksperymentu, i z cyfrą 2 – przebiegi sterowania i zmiennych stanu w systemie rzeczywistym.

Końcowe fragmenty trajektorii dziewięciu wybranych eksperymentów rzeczywistych pokazano na rysunkach 10.8–10.11.

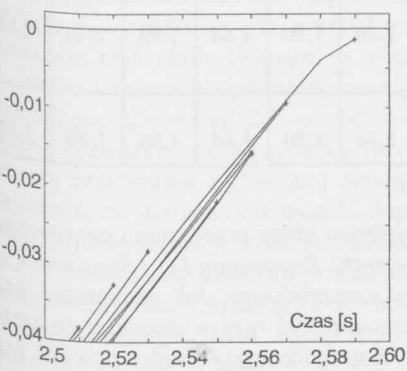
Sterowanie optymalnoczasowe, otrzymane drogą syntezy *off line*, zostało użyte z pozytywnym skutkiem do przeprowadzenia nieliniowego systemu rzeczywistego ze stabilnego punktu równowagi do otoczenia niestabilnego punktu równowagi. Eksperyment powiódł się dzięki kompensacji tarcia wózka, modyfikacji wyliczonych sterowań optymalnoczasowych za pomocą adaptacji i optymalizacji, z rzeczywistym obiektem zamykającym pętlę sterowania, oraz dzięki włączeniu cech systemu czasu rzeczywistego (RT-CON) do systemu operacyjnego Windows NT.



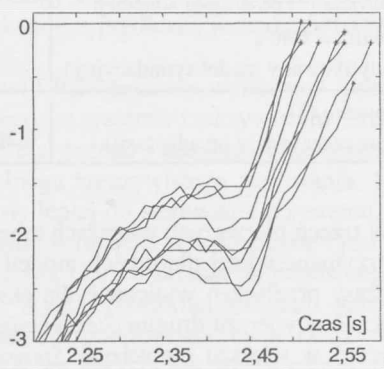
Rys. 10.8. Końcowe położenia wózka [m]



Rys. 10.9. Końcowe kąty wahadła [rad]



Rys. 10.10. Końcowe prędkości wózka [m/s]



Rys. 10.11. Końcowe prędkości wahadła [rad/s]

Adaptacja sterowania. Zapewnienie zgodności dynamik: zbudowanego modelu i systemu rzeczywistego jest trudnym do spełnienia warunkiem. Na ogół model odbiega od rzeczywistości. Eksperyment adaptacyjny polega na ponownym wyliczaniu sterowania optymalnoczasowego z modelu, kiedy rozwijająca się w czasie trajektoria rzeczywista zaczyna znacząco odbiegać od trajektorii modelowej (Kristic, Canellacopoulos, Kokotovic 1995). Ponieważ w każdym kroku próbkowania, lub co pewną liczbę kroków, wyliczamy sterowanie optymalne na modelu i stosujemy tak wyliczone sterowanie do obiektu rzeczywistego, to powstaje pytanie, czy taki algorytm realizuje cel sterowania. Jak pokaże eksperyment przedstawiony poniżej, kolejne adaptacje sterowania prowadzone w celu likwidacji odchyłek, powstających pomiędzy trajektorią rzeczywistą systemu i trajektorią modelu, są skuteczne przy niskiej wrażliwości trajektorii na błędy modelowania i lokalne zakłócenia.

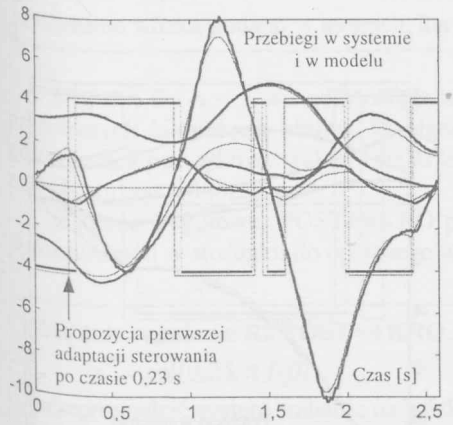
Wyniki eksperymentu, w którym badamy przydatność zmodyfikowanego modelu (oznaczonego poprzednio cyfrą 3) nie tylko w chwili początkowej, ale także w następnych wybranych chwilach czasu, zebrano w tabeli 10.1.

Tabela 10.1
Wyniki numeryczne

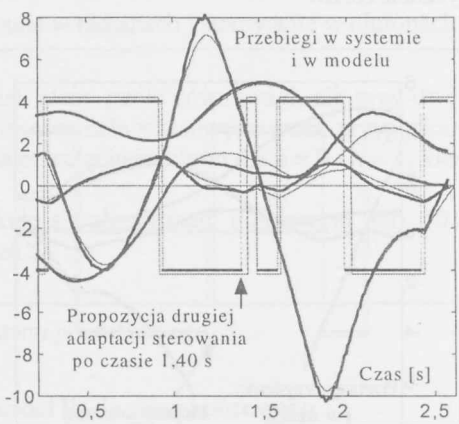
	Czasy przełączeń i czas końcowy [s]							
	τ_1	τ_2	τ_3	τ_4	τ_5	τ_6	τ_7	t^*
Sterowanie (1) (model symulacyjny)	0,29	0,99	1,47	1,57	1,69	2,10	2,54	2,71
Sterowanie (2) (system rzeczywisty)	**0,28**	**0,94**	**1,40**	**1,49**	**1,60**	**2,00**	**2,40**	**2,57**
Sterowanie (3) (zmodyfikowany model symulacyjny)	0,23	0,90	1,41	1,46	1,61	2,01	2,41	2,58
Sterowanie (3) po pierwszej adaptacji w chwili $t = 0{,}23$ s (zmodyfikowany model symulacyjny)	**0,26**	**0,93**	1,43	1,49	1,62	2,03	2,44	2,61
Sterowanie (3) po drugiej adaptacji w chwili $t = 1{,}40$ s (zmodyfikowany model symulacyjny)			**1,40**	**1,50**	**1,64**	**1,98**	**2,40**	**2,58**
Sterowanie (2) (system rzeczywisty po adaptacji)	**0,26**	**0,93**	**1,40**	**1,50**	**1,64**	**1,98**	**2,40**	**2,58**

W trzech pierwszych wierszach tabeli 10.1 wypisano czasy przełączeń i czas sterowania, otrzymane wyżej dla dwóch modeli symulacyjnych: *Sterowanie* (1) i *Sterowanie* (3), oraz czasy przełączeń wyliczone dla eksperymentu rzeczywistego. Jak pamiętamy, czasy przełączeń w wierszu drugim *Sterowanie* (2) są skrócone 0,95 razy w stosunku do czasów przełączeń w wierszu pierwszym *Sterowanie* (1). Zmodyfikowany model powstał przez zwiększenie 1,05 razy wartości $u_{\max}$. Tak więc czasy przełączeń *Sterowanie* (3), otrzymane z procedury optymalizacyjnej użytej do modelu zmodyfikowanego, różnią się od czasów

przełączeń pierwotnego modelu *Sterowanie* (1). Czasy te nie pokrywają się też z czasami w systemie rzeczywistym *Sterowanie* (2). Pierwszy eksperyment rzeczywisty jest pokazany na rysunku 10.12. Pierwsze przełączenie w systemie rzeczywistym *Sterowanie* (2) ma miejsce w chwili 0,28 s. Według modelu *Sterowanie* (3) winno nastąpić wcześniej, a mianowicie w chwili 0,23 s, licząc od początku sterowania. Wyliczamy ponownie sterowanie optymalne. Warunkiem początkowym jest zmierzony wektor stanu systemu w chwili 0,23 s.



Rys. 10.12. Trajektorie i sterowania; start optymalizacji w chwili 0 s



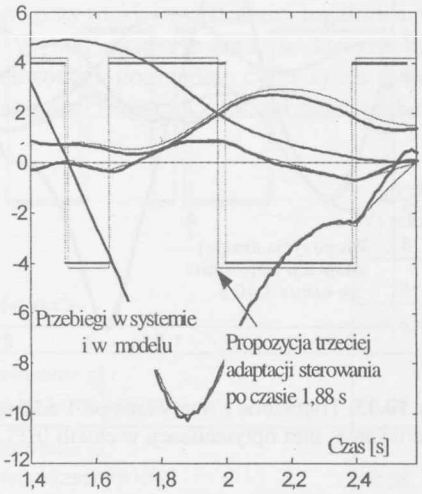
Rys. 10.13. Trajektorie i sterowania po 1 adaptacji sterowania; start optymalizacji w chwili 0,23 s

Na rysunkach 10.12–10.15 linie pogrubione oznaczają trajektorie i sterowania w systemie rzeczywistym (2). Linie cienkie oznaczają trajektorie i sterowania w modelu zmodyfikowanym (3).

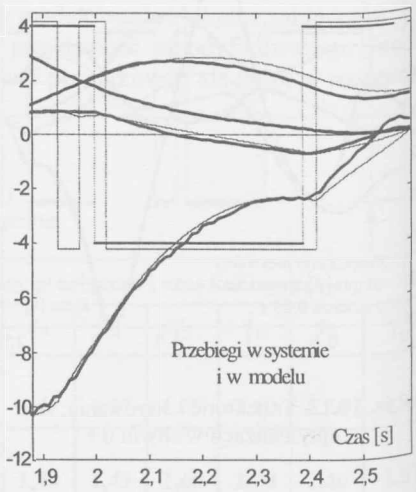
Trajektoria rzeczywista osiąga stan $x(0,23) = \mathrm{col}(-0,340,\ 3,36,\ -0,71,\ 1,53)$ po czasie 0,23 s, a trajektoria modelowa osiąga stan $x(0,23) = \mathrm{col}(-0,345,\ 3,39,\ -0,81,\ 1,83)$ w chwili 0,23 s. Położenia wózka różnią się o 5 milimetrów, kąty wahadła o 0,03 radiana, czyli około 1,8 stopnia, prędkości wózka o 0,01 metra na sekundę, a prędkości wahadła o 0,3 radiana na sekundę, czyli około 18 stopni na sekundę.

Nowe sterowanie optymalne startujące w chwili 0,23 sekundy wyliczamy z modelu, ale punkt początkowy $x(0,23)$ otrzymujemy z pomiaru w systemie rzeczywistym. Przebiegi czasowe przedstawiono na rysunku 10.13. Chwile pierwszego i drugiego przełączenia pokryły się praktycznie z chwilami przełączeń przebiegu rzeczywistego sterowania. Można powiedzieć, że sterowanie z modelu dopasowało się lepiej do sterowania z systemu. Z rysunku 10.13 wyraźnie wynika, że trzecie przełączenie w modelu przesunęło się i jest opóźnione w stosunku do trzeciego przełączenia *Sterowania* (2). To skłania do ponownego wyliczenia sterowania optymalnego w chwili 1,40 s. Wyniki powtórnej adaptacji sterowania przedstawia rysunek 10.14. Nowe sterowanie z modelu podpowiada, by w rzeczywistym wzorcu sterowania pozostawić przełączenie trzecie na dotychczasowej wartości 1,40 s, a czas czwartego opóźnić we wzorcu o jeden okres próbkowania (0,01 s), czyli ustawić na 1,50 s, i przełączenie piąte opóźnić o 0,04 s, czyli ustawić na 1,64 s.

Ponowiono więc eksperyment rzeczywisty w pętli otwartej, z nowym ciągiem czasów przełączeń. Rysunek 10.15 przedstawia wyniki drugiego eksperymentu rzeczywistego. Ostatecznie przeprowadzono jeszcze jedną próbę adaptacji sterowania w chwili 1,88 s, według propozycji z rysunku 10.15. Wyników tej adaptacji modelowej, z uwagi na dużą wrażliwość trajektorii, nie użyto do ponownej adaptacji wzorca rzeczywistego czasów przełączeń, pozostawiając takie czasy przełączeń, jakie wyliczono podczas drugiego eksperymentu rzeczywistego. Końcowy fragment sterowania w systemie pokazuje więc rysunek 10.14, a nie rysunek 10.15.



Rys. 10.14. Trajektorie i sterowania po 2 adaptacji sterowania; start optymalizacji w chwili 1,40 s



Rys. 10.15. Trajektorie i sterowania bez 3 adaptacji sterowania; start optymalizacji w chwili 1,88 s

Eksperyment adaptacji sterowania w systemie rzeczywistym, przy użyciu tego samego modelu (3), polegający na ponawianiu procedury wyliczania sterowania optymalnego, dla której punktami startu są wybrane punkty trajektorii rzeczywistego systemu, pokazuje, że procedura jest efektywna dla niezbyt dużej wrażliwości rozwiązań równań stanu na zmiany punktu początkowego. Trzecia adaptacja sterowania rzeczywistego w chwili 1,88 s, według podpowiedzi z modelu, miałaby skutek przeciwny do zamierzonego, gdyby zastosować ją w systemie rzeczywistym.

Porównanie algorytmu optymalnoczasowego z regułowym. Rozważamy zadania sterowania optymalnoczasowego O3 i regułowego R2 dla modelu laboratoryjnego II (parametry – patrz tabela 3.1). Zadania mają te same warunki początkowe i końcowe oraz ograniczenia na sterowanie. Różnica polega na wprowadzeniu w zadaniu regułowym innego ograniczenia sterowania w strefie stabilizacji. Dlatego porównanie działania algorytmów należy prowadzić w przedziale czasu: od początku sterowania do momentu wejścia trajektorii układu w strefę stabilizacji S.

Zadanie optymalnoczasowe O3 POST+AKRO. Ze stanu początkowego

$x^0 = \text{col}(0{,}25, \pi, 0, 0)$

przeprowadzić system wahadła na wózku (model II), w minimalnym czasie, do stanu

$x^f = \text{col}(-0{,}20, 2n\pi, 0, 0), \quad n = 0 \quad \text{lub} \quad n = 1,$

przy ograniczeniach na sterowanie u,

$|u| \le u_{\max} = 4.$

Położenie wózka podano w metrach, kąt wahadła w radianach i sterowanie w niutonach.

Dla $n = 0$ i $n = 1$ wahadło osiąga ten sam górny punkt równowagi, ale przy dwóch przeciwnych kierunkach obrotu. W algorytmie optymalnoczasowym, poza przypadkami, gdy warunek początkowy wektora stanu leży na przecięciu izochron dla $n = 0$ i $n = 1$, kierunek obrotu jest jednoznaczny.

Zadanie (regułowe) POST+AKRO posiłkuje się algorytmem regułowym (tab. 10.2), uproszczonym w stosunku do opisanego w tabeli 5.1.

Zadanie regułowe R2 POST+AKRO. Ze stanu początkowego

$x^0 = \text{col}(0{,}25, \pi, 0, 0)$

przeprowadzić system wahadła na wózku (model II) do stanu końcowego

$x^f = \text{col}(-0{,}20, 2n\pi, 0, 0), \quad n = 0 \quad \text{lub} \quad n = 1;$

ograniczenia na sterowanie u poza strefą S stabilizacji kąta wahadła są postaci

$|u| \le u_{\max}^{\text{POST}} = 4, \quad \text{dla } x_2 \notin [-S, S], \quad S = 0{,}3,$

ograniczenia na sterowanie u w strefie S są postaci

$|u| \le u_{\max}^{\text{STAB}} = 10 \quad \text{dla } x_2 \in [-S, S].$

Przyczyny, dla których w algorytmie regułowym stosuje się dwa różne ograniczenia sterowania, wyjaśniono w rozdziale 5. Ograniczenie przyjęte w zadaniu optymalnoczasowym wynika z ograniczoności zakresu jezdnego szyny. Kierunek obrotu wahadła w algorytmie regułowym nie musi być zgodny z obrotem optymalnoczasowym (rys. 10.16). W proponowanym algorytmie (tab. 10.2) uproszczenie polega na usunięciu reguł w postaci funkcji kary za zbliżanie się wózka do końców pola jezdnego szyny. W ten sposób zwiększanie amplitudy drgań wahadła (reguła 8) nie jest zakłócane akcją centrowania wózka, aż do osiągnięcia strefy górnej wahadła. Sprowadzenie wózka do położenia x_1^f ma miejsce w drugim etapie sterowania, gdy trwa stabilizacja systemu. Uproszczenie (chodzi o uzyskanie sterowania bangbangowego) wprowadzono, aby porównać algorytm optymalnoczasowy z regułowym, opisanym w rozdziale 5. Usunięcie reguł centrowania wózka zawęża zbiór, z którego można wybierać wektor początkowy x^0, tak by wózek nie przekroczył zakresu jezdnego szyny przez cały okres sterowania, aż do osiągnięcia celu x^f. Wielkość tego zbioru zależy od dwóch parametrów: punktu docelowego x^f i wartości sterowania $u_{\max}^{\text{POST}}$, czyli amplitudy siły poruszającej wózek.

Tabela 10.2
Uproszczony algorytm regułowy

Stabilizacja

If $|x_2| - S < 0$ (1)

then $u_r = K_1(x_1 - x_1^f) + K_2 x_2 + K_3 x_3 + K_4 x_4$ (2)

if $|u_r| + F_s > u_{\max}^{\mathrm{STAB}}$ (3)

then $u = u_{\max}^{\mathrm{STAB}} \operatorname{sign} u_r$ (4)

else $u = u_r + F_s \operatorname{sign} u_r$ (5)

end

Zwiększanie amplitudy wahań

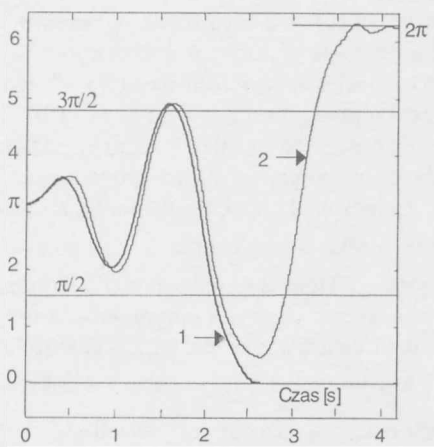
elseif $\frac{1}{2}x_4^2 + 9{,}81 \cdot 3{,}2(\cos x_2 - 1) > 0$ (6)

then $u = 0$ (7)

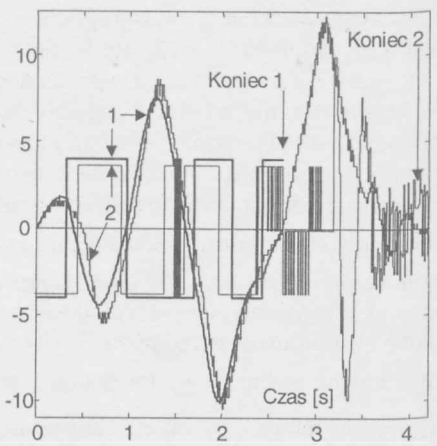
else $u = -u_{\max}^{\mathrm{POST}} \operatorname{sign}\left[x_4 \left(|x_2| - \frac{\pi}{2} \right) \right]$ (8)

end

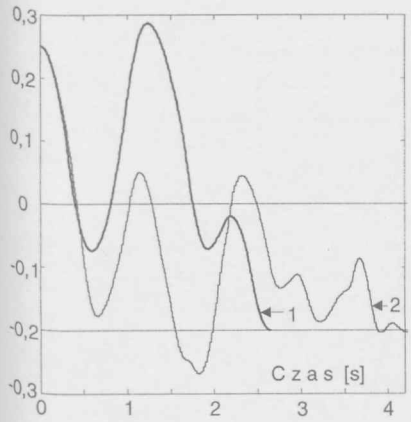
Symulacyjne odpowiedzi dla algorytmu optymalnoczasowego (1) i odpowiedzi systemu pod działaniem algorytmu regułowego (2) przedstawiono na rysunkach 10.16–10.19. Zauważmy, że czas symulowanego sterowania optymalnoczasowego $t^* = 2{,}69$ s jest o wiele krótszy od czasu sterowania rzeczywistym systemem za pomocą algorytmu regułowego $t^{\mathrm{regułowy}} = 4{,}2$ s.



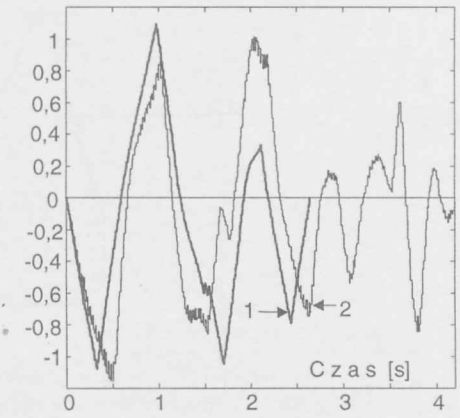
Rys. 10.16. Kąt wahadła [rad]



Rys. 10.17. Prędkość wahadła [rad/s]; sterowanie [N]

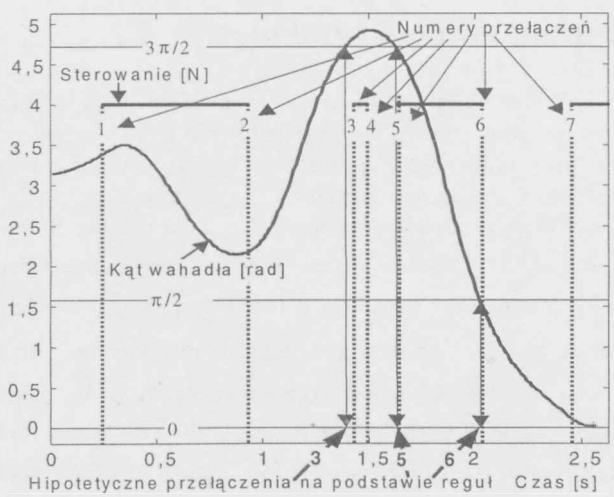


Rys. 10.18. Położenie wózka [m]

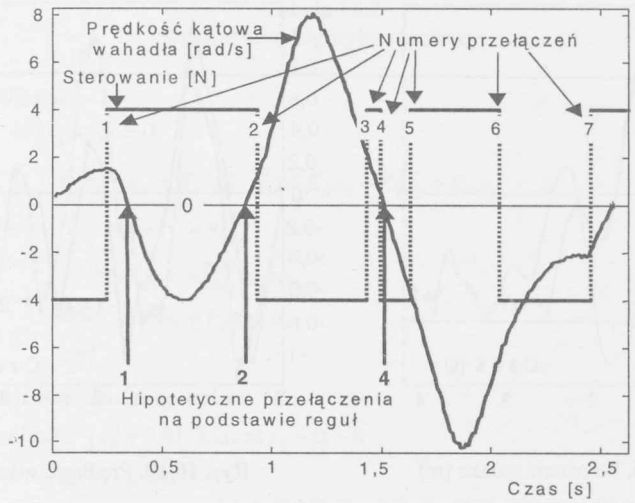


Rys. 10.19. Prędkość wózka [m/s]

Dla oceny „odległości" sterowania optymalnoczasowego od regułowego, na rysunkach 10.20 i 10.21 zaznaczono grubymi strzałkami hipotetyczne chwile przełączeń, które generowałby algorytm regułowy (uwzględniono tylko zmienne fazowe wahadła). Chwile przełączeń 3, 4, 5 i 6 są zbliżone w obu algorytmach. Przy tej ocenie warto przypomnieć, że regulator optymalnoczasowy z podrozdziału 9.1, budowany w oparciu o zbiór reprezentatywny, był aktywny tylko w I stadium sterowania (do chwili czasu odpowiadającej w przybliżeniu piątemu przełączeniu na rysunku 10.20).



Rys. 10.20. Hipotetyczne przełączenia algorytmu regułowego wynikające z optymalnoczasowego przebiegu kąta wahadła



Rys. 10.21. Hipotetyczne przełączenia algorytmu regułowego wynikające z optymalnoczasowego przebiegu prędkości kątowej wahadła

Można wyciągnąć wniosek, że algorytm heurystyczny, utworzony na podstawie obserwacji dynamiki ruchu systemu, nie tylko realizuje zadania sterowania, ale dodatkowo jest zbliżony do algorytmu optymalnoczasowego z wyłączeniem stadiów: początkowego i końcowego.

10.2. Sterowanie w pętli zamkniętej

Aby zastosować regulator optymalnoczasowy do sterowania systemem wahadła na wózku w czasie rzeczywistym, niezbędne jest spełnienie kilku warunków sprzętowo-programowych. Przede wszystkim należy dysponować szybkim algorytmem optymalizacyjnym do wyliczania czasów przełączeń sterowania. Najlepiej byłoby, gdyby czas jednokrotnego wyznaczenia sterowania optymalnoczasowego t_{opt}, przy zadanym warunku początkowym x^0, stanie docelowym x^f i ograniczeniu $u_{\max}$ na sterowanie, mieścił się w jednym okresie próbkowania t_p komputera sterującego. Jeżeli warunki techniczne nie pozwalają na spełnienie nierówności $t_{\text{opt}} < t_p$, to przy niewielkim opóźnieniu w aplikowaniu wyliczonego sterowania ma sens zamknięcie pętli sprzężenia regulacji. W eksperymentach czas wyliczenia sterowania t_{opt} przyjmował wartości od 10 do 50 ms – sporadycznie zdarzało się, że optymalizacja trwała 60 ms (zob. rys. 10.23c, 10.26b i 10.34b). Na ogół $t_{\text{opt}} \leq 2t_p$, szczególnie w końcowych fazach sterowania, kiedy optymalizację prowadzi się na coraz krótszych odcinkach czasu. W eksperymentach rzeczywistych sterowania w pętli otwartej, opisanych w podrozdziale 10.1, nie dysponowano jeszcze tak szybką wersją algorytmu, by

zbudować regulator. Stało się to możliwe po zapisaniu wszystkich procedur algorytmicznych w języku C. Drugim zabiegiem zwiększającym szybkość obliczeń było zrezygnowanie ze stosowania algorytmu 8.5, mimo że za jego pomocą rozwiązuje się problem optymalnoczasowy. Użyto algorytmu stałohoryzontowego 8.3. W praktycznym zastosowaniu taka zamiana celem zwiększenia szybkości obliczeń jest w pełni uzasadniona. Algorytm 8.3, by stał się użyteczny dla regulacji optymalnoczasowej, wymaga kilku modyfikacji. By nie spowalniać algorytmu, nie wykorzystuje się mechanizmu kontynuacji – zwiększania horyzontu i proporcjonalnego wydłużania czasów przełączeń. W zamian prowadzi się optymalizację dla stałego horyzontu. Wartości horyzontu, nieznacznie krótszego od optymalnego, kojarzy się z warunkami początkowymi i tabelaryzuje. Jednej wartości horyzontu, używanej przez algorytm do obliczeń, odpowiada zbiór warunków początkowych z kostki w przestrzeni czterowymiarowej. Zbiór wartości horyzontów, które należy stabelaryzować, nie jest duży. Dla typowych zadań POST i POST +AKRO wystarczy stabelaryzować kilka wartości horyzontu (zmiana horyzontu optymalnego nie przekracza 10%).

Podczas rozwiązywania wybranych 66 testowych zadań sterowania optymalnoczasowego okazało się, że przy ustaleniu punktu docelowego wyodrębnienie dwudziestu wartości horyzontu i skojarzenie z nimi 20 podzbiorów warunków początkowych wystarcza do rozwiązania zadania optymalizacji z dowolnego (praktycznie realizowalnego) warunku początkowego.

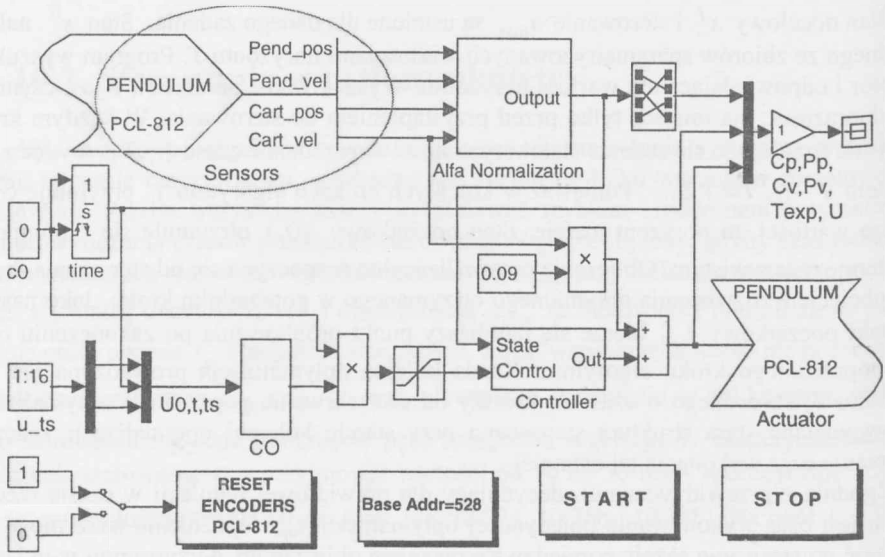
Danymi wejściowymi algorytmu są: stan początkowy x^0, stan docelowy x^f, maksymalna amplituda sterowania $u_{\max}$, horyzont T i przybliżenie sterowania optymalnego, określone przez liczbę czasów przełączeń i ich wartości $\tau_i^{\text{stare}}, i = 1,...,n^{\text{stare}}$ oraz początkową wartość sterowania u_0^{stare}. Na wyjściu algorytmu otrzymuje się obliczone sterowanie optymalne z czasami przełączeń $\tau_i, i = 1,...,n$ i wartością początkową u_0.

Stan docelowy x^f i sterowanie $u_{\max}$ są ustalone dla danego zadania. Stan x^0 należy do jednego ze zbiorów sparametryzowanych wartościami horyzontu T. Program wyszukuje ten zbiór i odpowiadającą mu wartość horyzontu. Wyszukiwanie parametru T jest czynnością jednorazową, ma miejsce tylko przed przystąpieniem do sterowania. W każdym kroku algorytmu rozwiązuje się zadanie stałohoryzontowe w przedziale czasu $[t_i, T]$, a więc z horyzontem $T - t_i$, $i = 1,2,...$. Ponieważ w kolejnych krokach algorytmu t_i przyjmuje coraz większe wartości, to horyzont maleje. Stan początkowy $x(t_i)$ otrzymuje się z pomiarów w systemie rzeczywistym. Obliczenia optymalizacyjne rozpoczyna się od sterowania, będącego obcięciem rozwiązania optymalnego otrzymanego w poprzednim kroku. Jako następny punkt początkowy t_{i+1} bierze się najbliższy punkt próbkowania po zakończeniu obliczeń poprzedniego kroku algorytmu. Każda kolejna optymalizacja prowadzona jest dla horyzontu T, skróconego o czas nie krótszy od czasu trwania poprzedniej optymalizacji. Podpowiedziana stara struktura sterowania przy starcie kolejnej optymalizacji znacznie przyspiesza czas wykonania tej ostatniej.

Zgodnie z przewidywaniami, decydujący dla prawidłowej regulacji w czasie rzeczywistym jest czas wykonywania pojedynczej optymalizacji t_{opt}. Opóźnienia takie mogą powodować powstawanie różnic pomiędzy sterowaniem obliczanym, a używanym w układzie rzeczywistym. Najprostszym sposobem na przyspieszenie algorytmu jest użycie komputera z większą częstotliwością zegara procesora. Innym sposobem jest profilowanie czasów wy-

konania podprogramów algorytmu i wykrycie najbardziej czasochłonnych procedur. Do takich należą procedury obliczania wartości funkcji sinus i cosinus, wykonywane wiele tysięcy razy. Tablicowanie tych funkcji dałoby znaczące skrócenie czasu obliczeń. Mimo że pojedyncza optymalizacja przeciętnie trwa dłużej niż próbkowanie, algorytm można z powodzeniem stosować w regulacji.

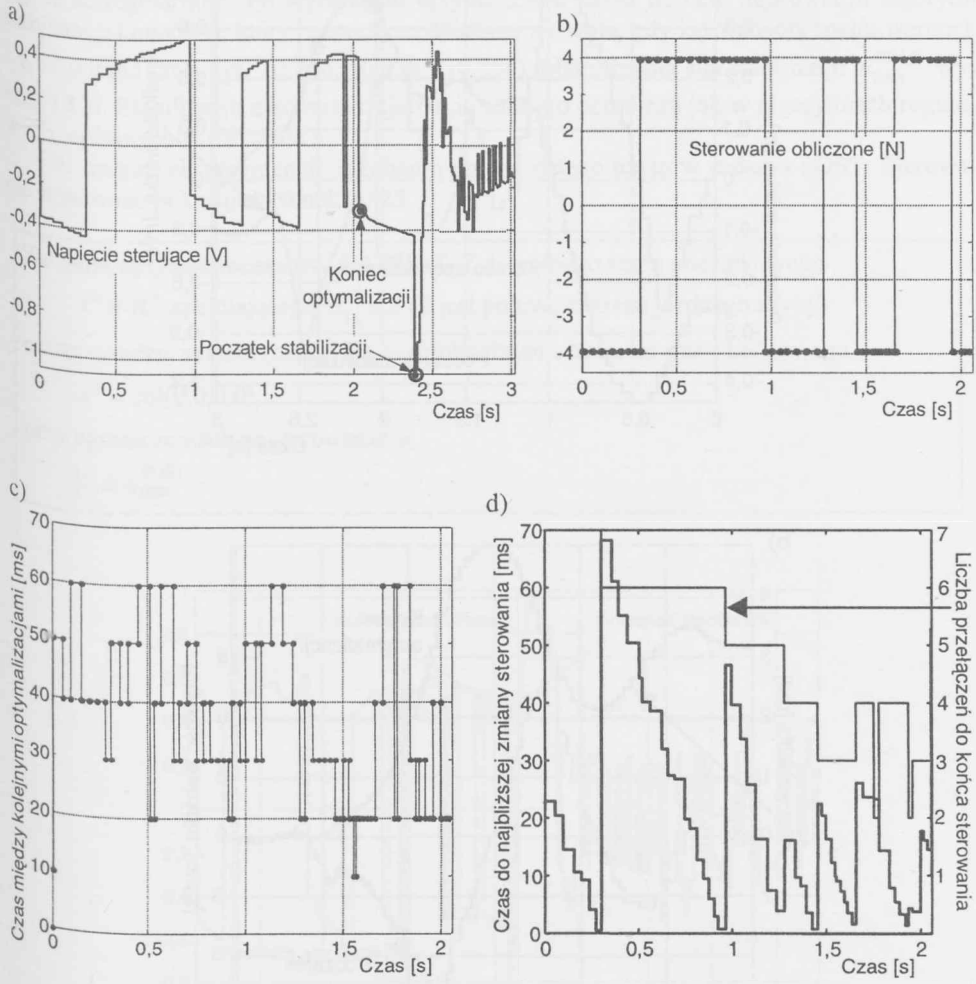
Pierwszy z przeprowadzonych eksperymentów polega na zrealizowaniu w systemie rzeczywistym zadania optymalnoczasowego O2 POST+AKRO ze stabilizacją końcową, zdefiniowanego w podrozdziale 10.1. Podobnie jak w eksperymencie prowadzonym w pętli otwartej, zastosowano kompensację tarcia statycznego wózka i wprowadzono kompensacyjne sprzężenie zwrotne od prędkości wózka (10.1). Kompensacja jest pokazana na rysunku 10.22 – sygnał prędkości wózka „Cart_vel" jest sumowany z sygnałem „CtrlOut" generowanym przez blok „Controller". Ten blok to regulator regułowy przejmujący akcję sterowania od regulatora optymalnoczasowego „CO" pod koniec cyklu sterowania. Dlatego wejściem regulatora „Controller" są dwa sygnały: „CtrlIn" – sterowanie z regulatora „CO" i „State" – zmierzony na bieżąco stan systemu. Regulator regułowy „Controller" uaktywnia algorytmy miękkiego lądowania i stabilizacji końcowej tylko w przypadku spełnienia przez zmienne stanu warunków aktywacji tych algorytmów (zob. tab. 5.1). W pozostałych przypadkach „Controller" jest przezroczysty dla sygnału z regulatora optymalnoczasowego „CO". Jak widać z rysunku 10.22, na wejście „CO" podawany jest sygnał wektorowy. Pierwszą współrzędną jest „U0" – znak sterowania, drugą czas rzeczywisty „t", trzecią, czwartą itd., aż do wyczerpania, są „ts" – czasy kolejnych przełączeń sterowania. Zarezerwowano zmienną „u_ts", o wymiarze szesnaście, na znak sterowania i piętnaście ewentualnych czasów przełączeń. Rysunek 10.22 przedstawia tylko fragment aplikacji czasu rzeczywistego – jej część simulinkową.



Rys. 10.22. Okno sterowania aplikacji czasu rzeczywistego RT-CON w Simulinku

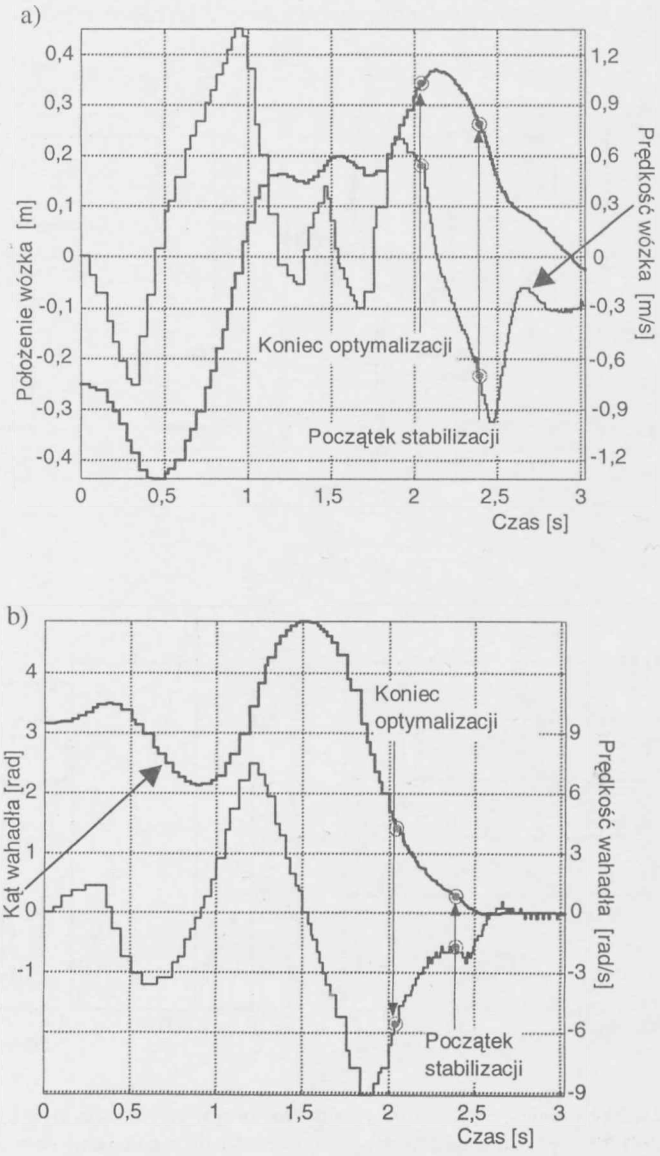
Cała aplikacja to przede wszystkim algorytm napisany w języku C i przetworzony na plik wykonawczy o rozszerzeniu „dll". Komunikację z aplikacją (po załadowaniu jej do pamięci komputera) urzeczywistnia plik MATLAB-a zawierający wywołania algorytmu z przekazaniem bieżącego wektora stanu i zwrotem wartości czasów przełączeń i znaku pierwszego sterowania. Plik korzysta z funkcji przybornika RTW i RT-CON.

Wyniki trzech eksperymentów regulacji optymalnoczasowej w czasie rzeczywistym przedstawiono na rysunkach oznaczonych numerami od 10.23 do 10.35.



Rys. 10.23. Eksperyment regulacji optymalnoczasowej w czasie rzeczywistym dla zadania O2 POST+AKRO: a) sterowanie aplikowane – napięcie sterujące podawane na wejście wzmacniacza silnika napędzającego wózek; b) sterowanie obliczone – (siła sterująca w niutonach) wyliczane przez algorytm optymalnoczasowy w kolejnych krokach optymalizacji; c) czasy optymalizacji; d) czas do przełączenia i liczba przełączeń

Sterowanie wyliczane przez algorytm optymalizacyjny (rys. 10.23b, 10.26a i 10.34b) ma charakter bang-bang. Siła ciągnąca wózek, generowana przez silnik, powinna mieć taki właśnie przebieg w funkcji czasu. W tym celu kompensuje się zwrotnie napięcie wejściowe zasilacza silnika i uzyskuje przebiegi tego napięcia jak na rysunkach 10.23a, 10.25 i 10.34a.



Rys. 10.24. Eksperyment regulacji optymalnoczasowej w czasie rzeczywistym dla zadania O2 POST+AKRO: a) położenie i prędkość wózka; b) kąt i prędkość kątowa wahadła

Przy porównaniu sterowania w eksperymencie w pętli otwartej (rys. 10.3) ze sterowaniem w pętli zamkniętej z rysunku 10.23b widać wiele podobieństw, ale zaznaczają się też różnice. Algorytm optymalnoczasowy (rys. 10.23b) zostaje wyłączony w chwili ostatniego przełączenia. To znaczy, że wtedy procedura ustawia wartość stałą sterowania wyliczoną po ostatnim przełączeniu. Z rysunku 10.23d można odczytać, że regularne ubywanie czasów przełączeń w trakcie sterowania kończy się z chwilą wystąpienia piątego przełączenia. Liczba przełączeń do końca zmienia wartość z 3 na 4, a potem na 2 i znów na 4, by zmaleć za chwilę do 3 i 1. Ruch po powierzchni przełączeń od chwili piątego przełączenia tłumaczy tę nieregularność. Po wyłączeniu optymalizacji, układ do celu doprowadza algorytm stabilizacji końcowej, który rozpoczyna działanie z chwilą, gdy kąt wahadła spełni warunek $|x_2| \leq 0{,}2$ rad (zob. rys. 10.24b, 10.27b i 10.35c). Maksymalna siła stabilizacji $u_{\max}^{STAB}$ wynosi 10 N. Stabilizacja z kompensacją tarcia suchego przebiega jak w algorytmach regułowych, opisanych w rozdziale 5.

W drugim eksperymencie regulatora rzeczywistego użyto w zadaniu POST. Sterowanie stosowane pokazuje rysunek 10.25.

Zadanie optymalnoczasowe O4 POST. Z dowolnego stanu początkowego

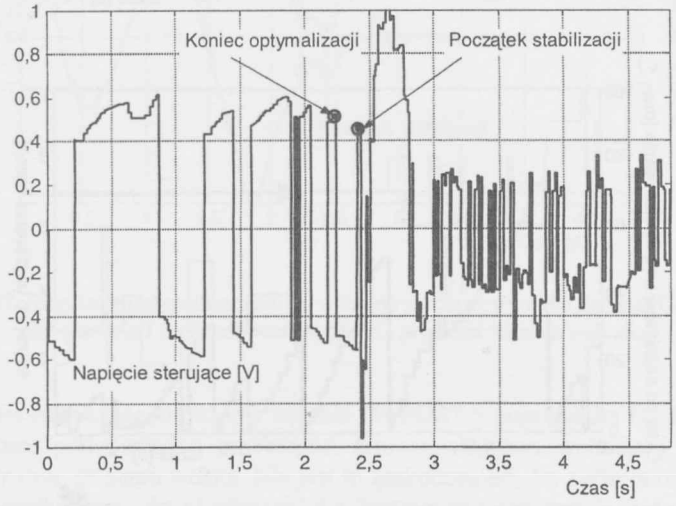
$x^0 \in \boldsymbol{R}^4$ spełniającego $|x_1^0| \leq Z$ (Z jest połową zakresu jezdnego szyny)

przeprowadzić system (model II), w minimalnym czasie, do stanu końcowego

$x^f = \text{col}(0,0,0,0)$,

przy ograniczeniach na sterowanie u,

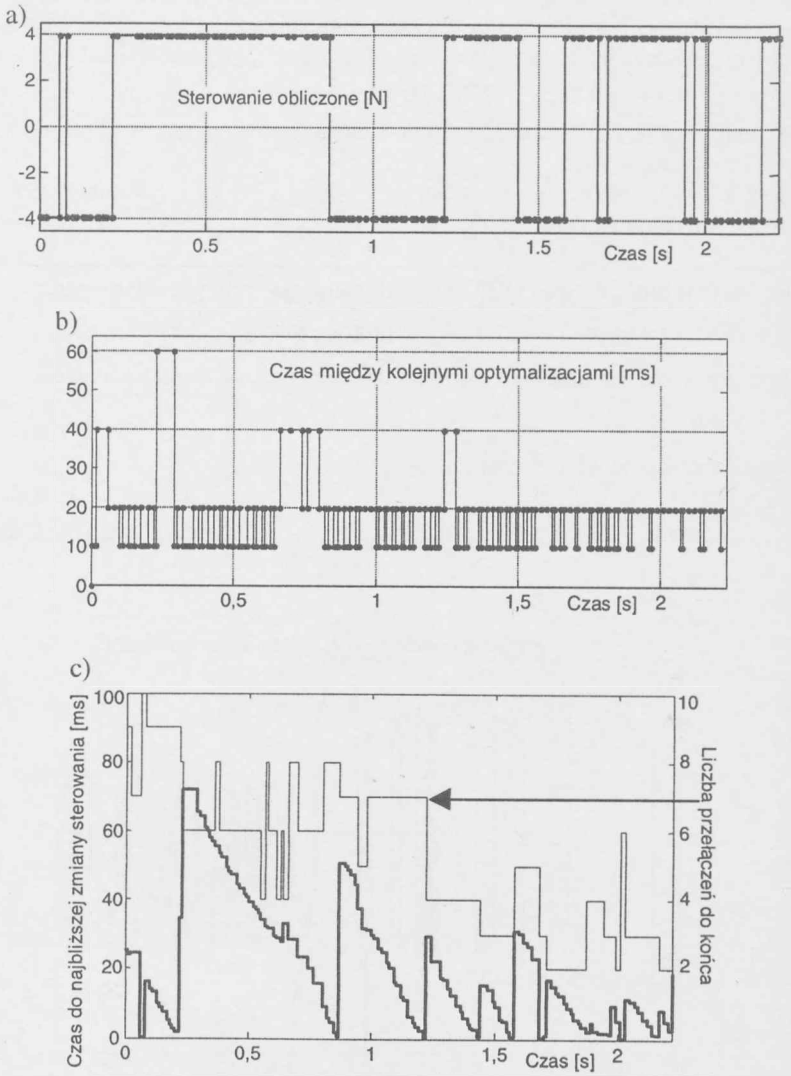
$|u| \leq u_{\max}^{POST}$.



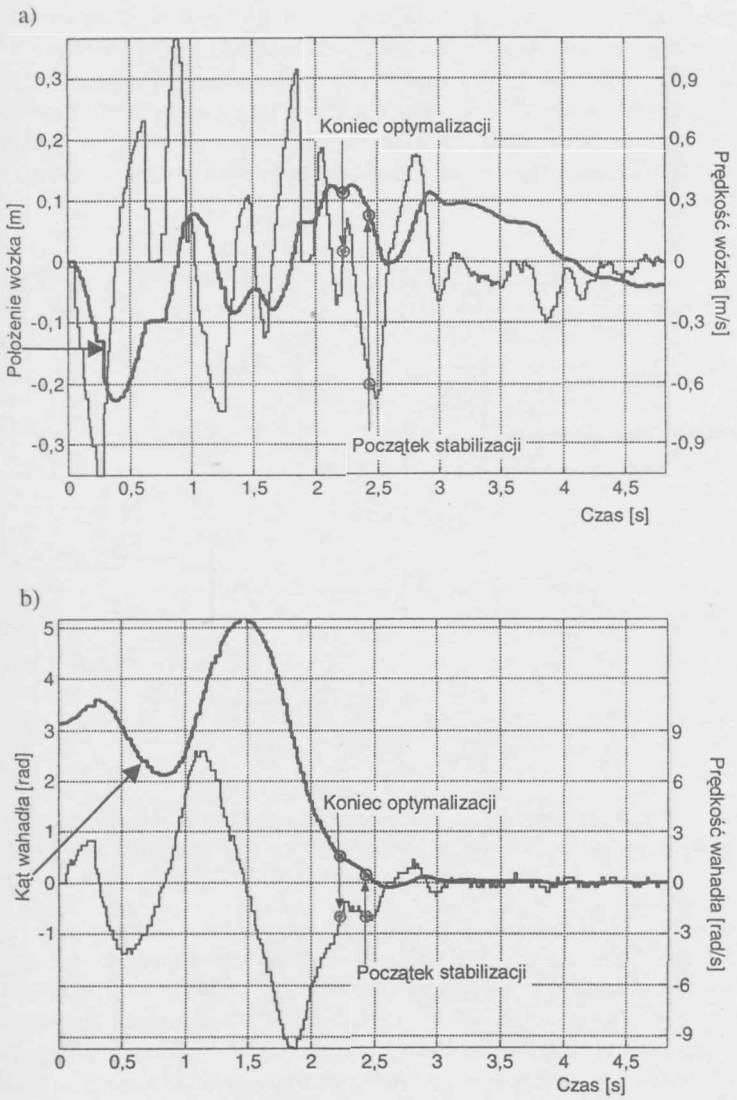
Rys. 10.25. Eksperyment regulacji optymalnoczasowej w czasie rzeczywistym dla zadania O4 POST – sterowanie aplikowane (opis zob. rys. 10.23a)

Widoczna jest zgodność obu sterowań, a także przypadki, gdy sterowanie stosowane gubi przełączenia – na przykład dodatnia szpilka (pierwsze i drugie przełączenie) na rysunku 10.26a nie ma odpowiednika na rysunku 10.25. Podobnie zgubione są przełączenia ósme i dziewiąte na rysunku 10.26a.

Na rysunkach 10.26b i c pokazano czasy optymalizacji, czas do przełączenia z liczbą przełączeń. Na rysunkach 10.27a i b przedstawiono zmienne stanu układu w funkcji czasu.



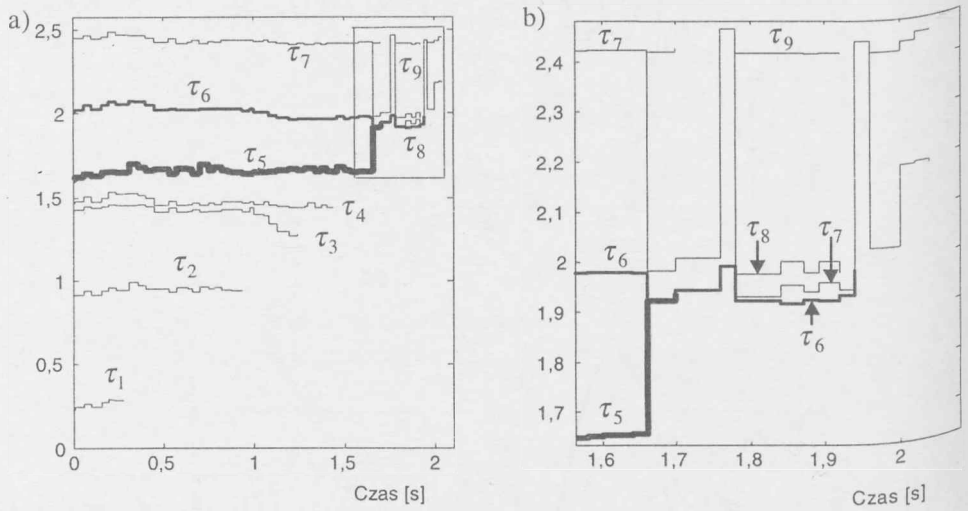
Rys. 10.26. Eksperyment regulacji optymalnoczasowej w czasie rzeczywistym dla zadania O4 POST: a) sterowanie obliczone – (siła sterująca w niutonach) wyliczane przez algorytm optymalnoczasowy; b) czasy optymalizacji; c) czas do przełączenia i liczba przełączeń



Rys. 10.27. Regulacja optymalnoczasowa w czasie rzeczywistym dla zadania O4 POST: a) położenie i prędkość wózka; b) kąt i prędkość kątowa wahadła

Porównanie rozważanego obecnie zadania O4 POST z poprzednim O2 POST+AKRO pokazuje znaczne podobieństwo przebiegów zmiennych stanu wahadła i duże różnice w przebiegu zmiennych stanu wózka. Nie jest to zaskoczeniem, bo zadania różnią się właśnie stanami początkowym i docelowym wózka. Interesującą obserwacją jest natomiast znaczna różnica w przebiegu kolejnych optymalizacji (rys. 10.23c–10.26b). Skróceniu uległ czas średni optymalizacji i waha się między 10 i 20 ms (poprzednio wynosił 30 ms).

Zauważono, że prowadzenie optymalizacji w pętli zamkniętej do chwili ostatniego przełączenia obniża ilość eksperymentów udanych, zakończonych osiągnięciem celu sterowania. Szczególnie niepokojące, lecz zrozumiałe z uwagi na wzrastającą wrażliwość (zob. rozdział 13), są nieregularne zmiany czasów przełączeń po osiągnięciu powierzchni przełączeń w piątym przełączeniu. Zjawisko to wyjaśnia rysunek 10.28, na którym przedstawiono czasy przełączeń, generowane przez regulator optymalnoczasowy w eksperymencie O2 POST+AKRO.



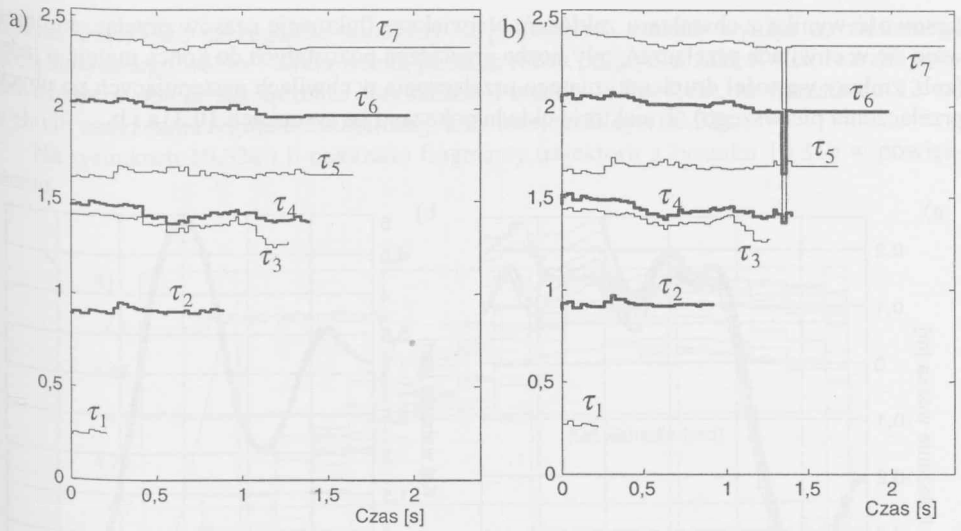
Rys. 10.28. Czasy przełączeń [s] generowane przez regulator optymalnoczasowy: a) całkowity okres działania regulatora; b) zaznaczony fragment rysunku 10.28a w powiększeniu

Na rysunku 10.29 pokazano dwa eksperymenty. Czasy przełączeń są generowane przez regulator optymalnoczasowy tylko do chwili czwartego przełączenia.

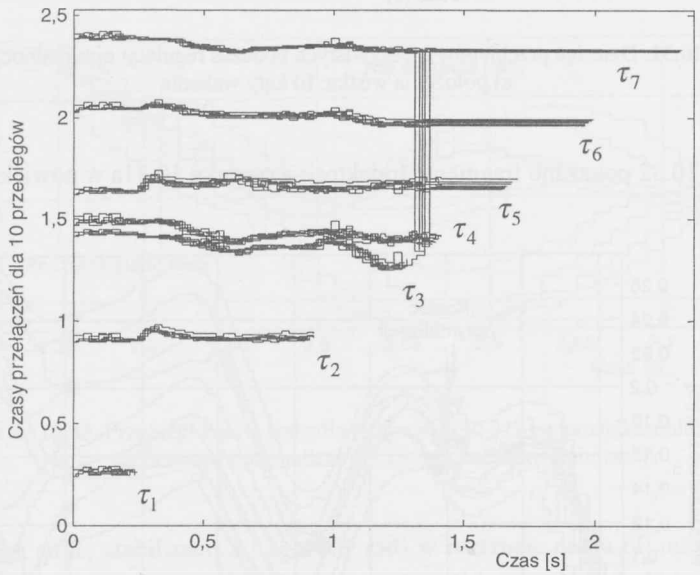
Zdecydowano w dalszych eksperymentach kończyć optymalizację w chwili czwartego przełączenia, pozostawiając dalsze czasy przełączenia na wartościach stałych, takich jakie zostały wygenerowane w chwili zakończenia optymalizacji. Dodatkowo, by wspomóc algorytm optymalnoczasowy, działający od chwili czwartego przełączenia w pętli otwartej, uaktywniono w bloku „Controller" (rys. 10.22) algorytm regułowy miękkiego lądowania wahadła z rozdziału 5.

W ten sposób zintegrowano dwie metody sterowania: klasyczną (używającą algorytmu optymalnoczasowego) z inteligentną (stosującą reguły zbudowane na podstawie zależności energetycznych w układzie). Regulację optymalnoczasową, prowadzoną do końca bez wspomagania, charakteryzowałaby niepewność w działaniu, nawet po zwiększeniu wag Q – zmiennych stanu wahadła – we wskaźniku jakości (7.30).

Wyniki dziesięciu kolejnych eksperymentów pokazano na rysunku 10.30. Zauważmy, że algorytm optymalizacji wyposażony w generację może wyliczyć w pewnym kroku iteracji liczbę przełączeń większą niż 7 (przełączenia nie pokazane na rysunku). W analizowanych zadaniach przełączenia takie pojawiają się, ale szybko ulegają redukcji.



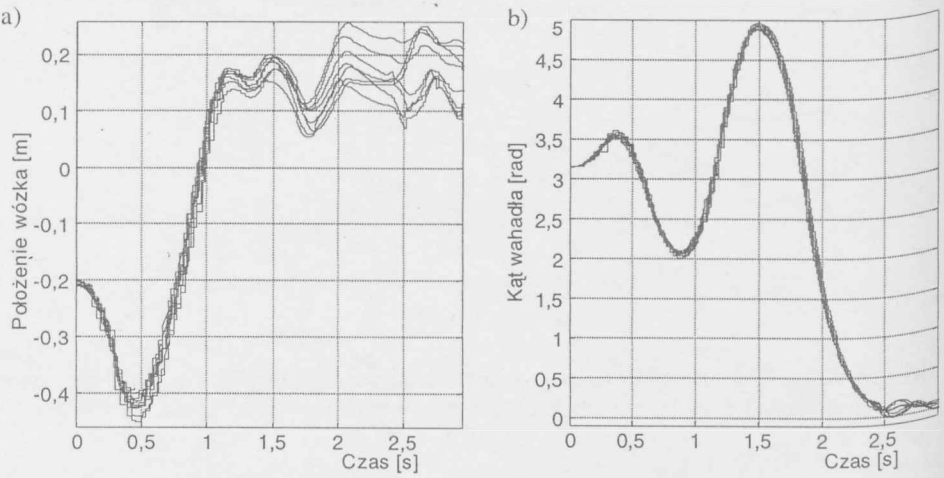
Rys. 10.29. Czasy przełączeń regulatora optymalnoczasowego; od chwili czwartego przełączenia sterowanie zostaje ustalone i układ jest dalej sterowany w pętli otwartej: a) 1 przebieg; b) 2 przebieg



Rys. 10.30. Czasy przełączeń [s] regulatora optymalnoczasowego; od chwili czwartego przełączenia układ jest sterowany w pętli otwartej; na rysunku pokazano dziesięć przebiegów rzeczywistych

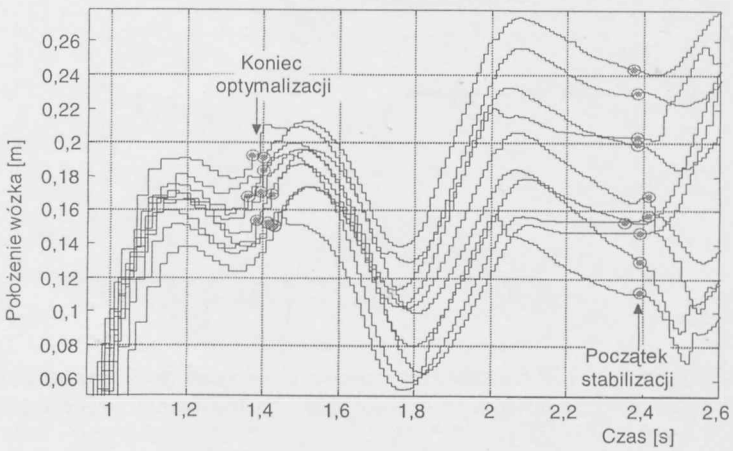
Zmiany czasów przełączeń, widoczne na rysunku 10.30, mają charakter losowy i równocześnie wykazują ukierunkowane tendencje – podobne w 10 kolejnych eksperymentach.

Losowość wynika z charakteru zakłóceń. Największe fluktuacje czasów przełączeń obserwuje się w chwilach przełączeń, gdy liczba przełączeń pozostałych do końca maleje o jeden (zob. zmiany wartości drugiego i piątego przełączenia w chwilach następujących po ubyciu przełączenia pierwszego). Trajektorie układu pokazano na rysunkach 10.31a i b.



Rys. 10.31. Dziesięć przebiegów rzeczywistych podczas regulacji optymalnoczasowej: a) położenia wózka; b) kąty wahadła

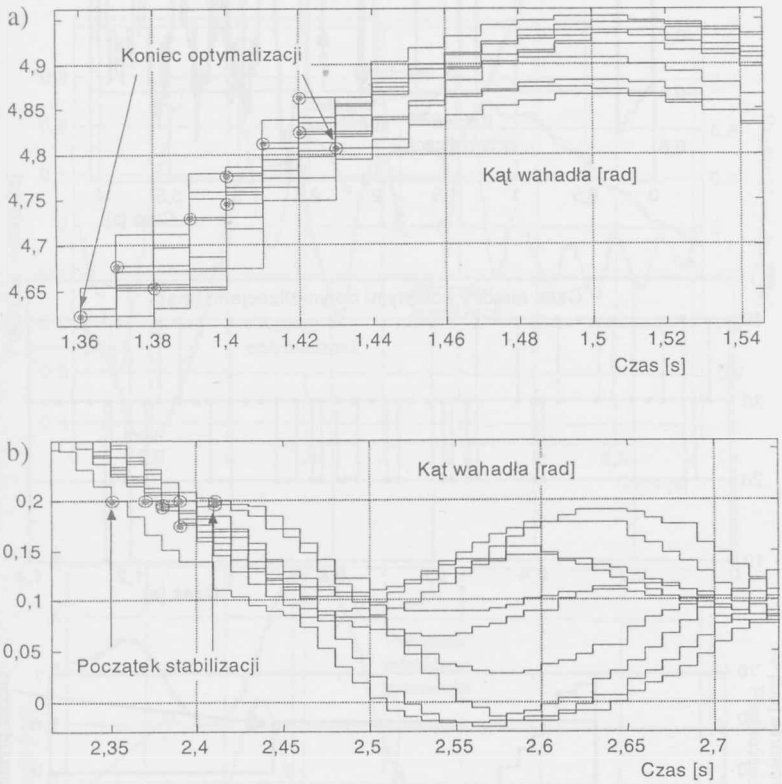
Na rysunku 10.32 pokazano fragmenty trajektorii z rysunku 10.31a w powiększeniu.



Rys. 10.32. Przebiegi położeń wózka (z rysunku 10.31a) w powiększeniu – strefy końca optymalizacji i początku stabilizacji

Rozrzut trajektorii, szczególnie widoczny w położeniach wózka, wynika ze zmiennego tarcia statycznego wózka i tarcia wózka podczas ruchu po szynie. Jakkolwiek rozrzut położeń wózka w chwili zakończenia optymalizacji wynosi tylko 4 cm, to wzrasta do 13 cm w chwili zadziałania regulatora stabilizacji końcowej (zob. rys. 10.32).

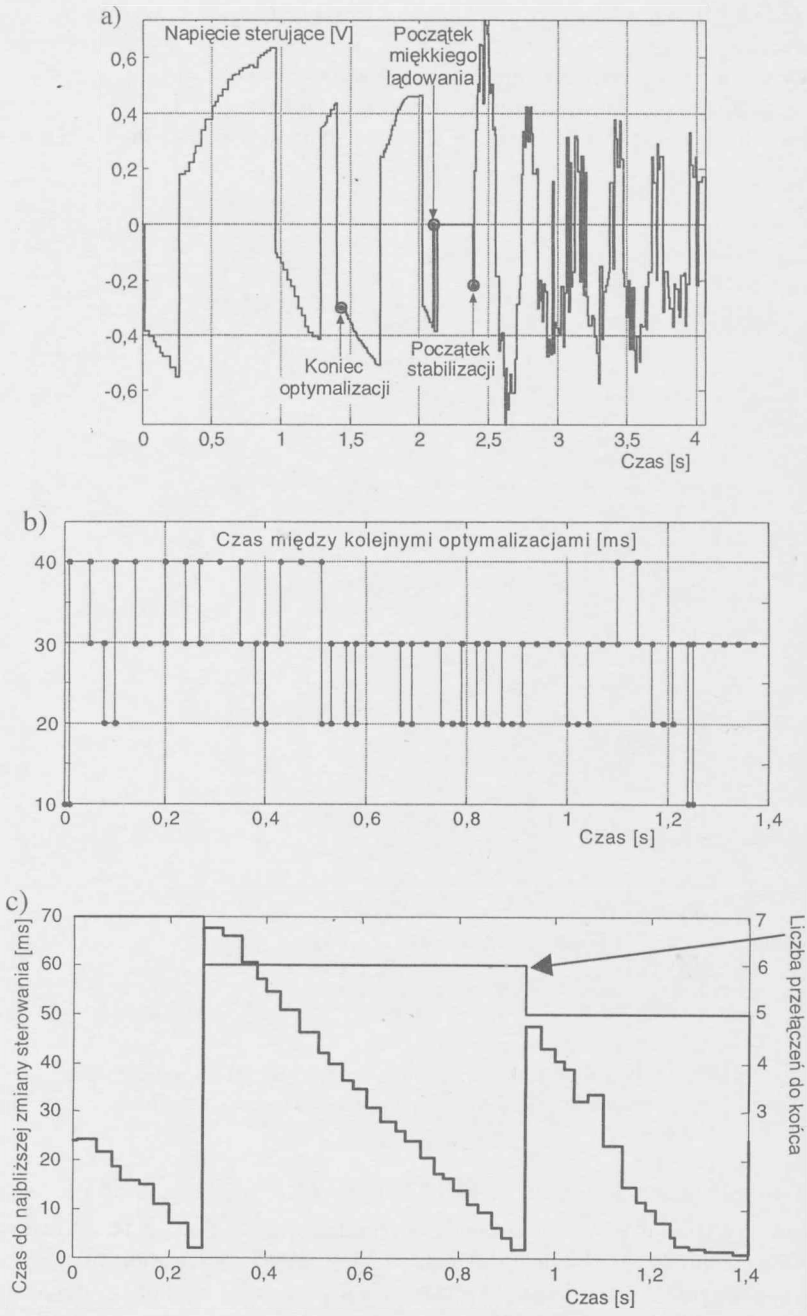
Na rysunkach 10.33a i b pokazano fragmenty trajektorii z rysunku 10.31b w powiększeniu.



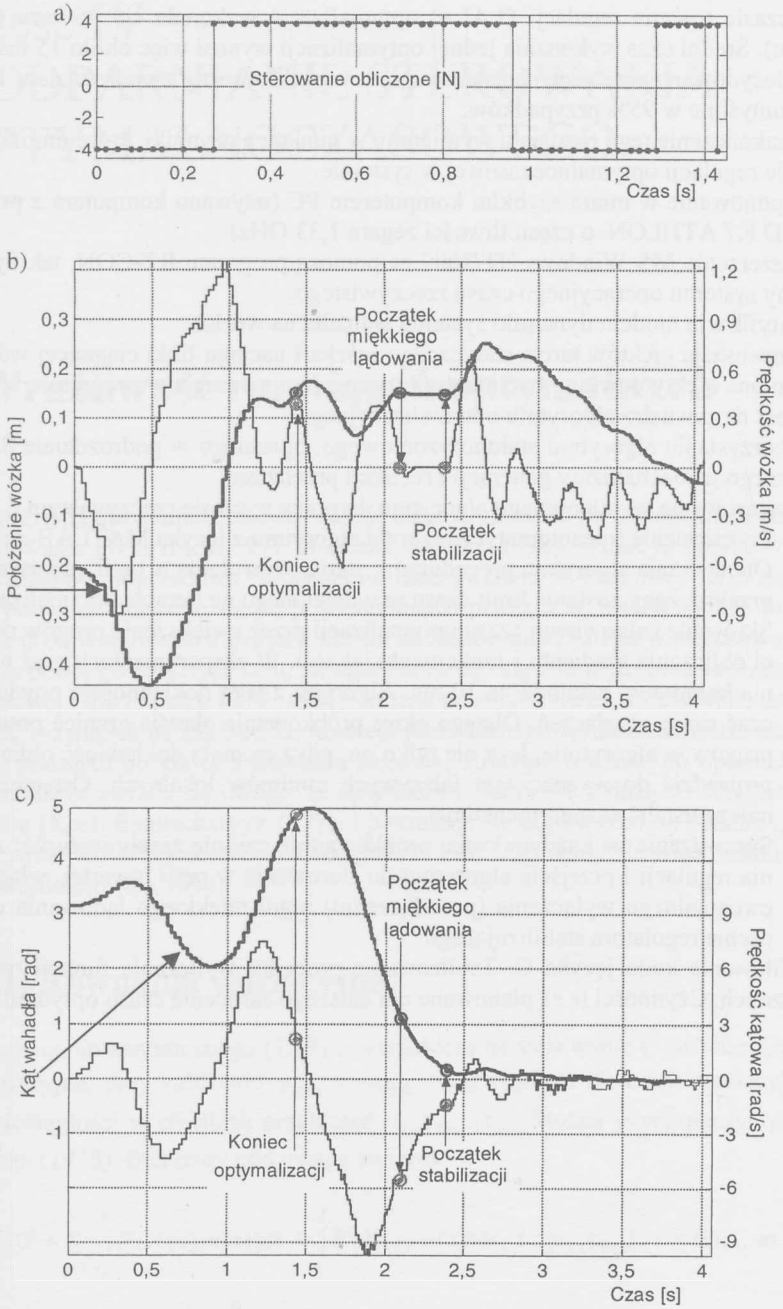
Rys. 10.33. Przebiegi kątów wahadła (z rysunku 10.31b) w powiększeniu: a) strefa końca optymalizacji; b) strefa początku stabilizacji

Wahadło osiąga strefę stabilizacji S ($x_2 \leq 0{,}2$ rad) w rozrzucie czasu 60 ms od 2,35 s do 2,41 s (zob. rys. 10.33b). W chwili końca optymalizacji jego położenia kątowe różnią się między sobą co najwyżej o 0,2 rad (zob. rys. 10.33a); jest to bardzo zadowalający wynik.

Dla porównania z metodami prowadzenia optymalizacji do końca, pokażemy jeden z eksperymentów zadania O4 POST, prowadzony ze skróconą regulacją w pętli zamkniętej i włączającym się algorytmem regułowym miękkiego lądowania (rys. 10.34–10.35). Wyłączenie sterowania przez algorytm miękkiego lądowania (wyraźnie widoczne na rysunku 10.34a) uzasadnia użycie tego algorytmu jako wspomagającego.



Rys. 10.34. Eksperyment regulacji optymalnoczasowej w pętli zamkniętej (przed czwartym przełączeniem) i otwartej (po czwartym przełączeniu), wraz z algorytmami miękkiego lądowania i stabilizacji: a) sterowanie aplikowane; b) czasy optymalizacji; c) czas do przełączenia i liczba przełączeń



Rys. 10.35. Eksperyment regulacji optymalnoczasowej w pętli zamkniętej (przed czwartym przełączeniem) i otwartej (po czwartym przełączeniu), wraz z algorytmami miękkiego lądowania i stabilizacji: a) sterowanie obliczone; b) położenie i prędkość wózka; c) kąt i prędkość kątowa wahadła

W czasie trwania regulacji (1,41 s) optymalizacja wykonała się 96 razy (zob. rys. 10.34b i c). Średni czas wykonania jednej optymalizacji wynosi więc około 15 ms. Regulacja ma zdecydowaną przewagę nad sterowaniem w pętli otwartej z podrozdziału 10.1, kończy się pomyślnie w 95% przypadków.

Na zakończenie tego rozdziału wymieńmy w punktach czynniki, które umożliwiły zrealizowanie regulacji optymalnoczasowej w systemie.

1. Dysponowanie w miarę szybkim komputerem PC (używano komputera z procesorem AMD K7 ATHLON o częstotliwości zegara 1,33 GHz).
2. Rozszerzenie MS Windows NT/2000 za pomocą programu RT-CON, tak by uzyskać cechy systemu operacyjnego czasu rzeczywistego.
3. Identyfikacja modelu dynamiki systemu wahadła na wózku.
4. Kompensacja efektów tarcia statycznego wózka i naciągu linki ciągnącej wózek. Wymienione efekty powinny być identyfikowane. Kompensacja w programie wykonywana jest na zewnątrz algorytmu optymalizacyjnego.
5. Wykorzystanie algorytmu stałohoryzontowego, opisanego w podrozdziale 8.2, wyposażonego w mechanizmy generacji i redukcji przełączeń.
6. Przystosowanie wymienionego algorytmu do pracy w czasie rzeczywistym.
   - 6.1 Przeniesienie (nieautomatyczne) kodu algorytmu z języka MATLAB-a na C.
   - 6.2 Obudowanie algorytmu procedurami: startu, wywołania w pętli, przerwania – gdy przekroczony zostanie limit czasu przydzielonego na iterację optymalizacyjną.
   - 6.3 Skrócenie całkowitego czasu optymalizacji przez zwiększenie progów dokładności obliczania gradientu i funkcjonału jakości. W eksperymentach czas próbkowania komputera ustalono na 10 ms. Algorytm z taką dokładnością powinien wyliczać czasy przełączeń. Dlatego okres próbkowania określa granice powiększania progów w algorytmie, lecz nie tylko on, gdyż za mała dokładność obliczeń może prowadzić do wyznaczania fałszywych minimów lokalnych. Ostatecznie progi należy ustalić eksperymentalnie.
   - 6.4 Sprawdzanie (w każdym kroku próbkowania), czy nie zaszły warunki: zakończenia regulacji i przejścia algorytmu do sterowania w pętli otwartej, włączenia lub ewentualnego wyłączenia (po włączeniu) reguł miękkiego lądowania oraz załączenia regulatora stabilizującego.
7. Profilowanie kodu języka C. Tablicowanie procedur wyliczania funkcji trygonometrycznych. Czynności te są planowane dla dalszego skrócenia czasu optymalizacji.

Część IV
UODPARNIANIE STEROWANIA OPTYMALNOCZASOWEGO

11. Wrażliwość rozwiązań optymalnych

Rozwiązania optymalnoczasowe cechuje duża wrażliwość na zakłócenia, błędy pomiaru i modelowania (Wierzbicki 1977; Bonnans, Shapiro 1998). Widać to dobrze w przykładach z rozdziałów poprzednich. W systemie rzeczywistym w końcowej fazie sterowania, gdy ruch powinien odbywać się po powierzchniach przełączeń, trudno spełnić to żądanie, z uwagi na dużą wrażliwość trajektorii układu na zakłócenia. Dlatego w zastosowaniach, po osiągnięciu pewnego otoczenia celu, na ogół rezygnuje się z kontynuowania strategii optymalnoczasowej (Kurzhansky 1977, 1994). Można postąpić inaczej, mianowicie zmniejszyć wrażliwość rozwiązań na zakłócenia, kosztem nieznacznego wydłużenia czasu sterowania, a więc od początku do końca sterowania stosować strategię zbliżoną do optymalnoczasowej. W literaurze zwraca się uwagę, że optymalność łączy się z dużą wrażliwością i małą odpornością (Keel, Bhattacharyya 1997), i poszukuje się odpowiedzi na pytanie „jak uciec z pułapki wrażliwości" (Kaesbauer, Ackermann 1998). Zajmiemy się więc badaniem wrażliwości (Mordukhowicz 1994).

11.1. Równania wariacyjne

Dla równania kanonicznego (7.39) z warunkiem na sterowanie (7.36) określamy równanie wariacyjne, przy założeniu $u_{\min} = -u_{\max}$. Rozwiązanie równania wariacyjnego ma na ogół nieciągłości w chwilach przełączeń $\tau_1, \tau_2, ..., \tau_m$. Można je wyznaczyć postępując jak Lastman (1978). Bierzemy pod uwagę związek

$$X(t) = X(0) + \sum_{i=1}^{r} \int_{\tau_{i-1}}^{\tau_i} F(X(s),u(s))\mathrm{d}s + \int_{\tau_r}^{t} F(X(s),u(s))\mathrm{d}s,\ t \in]\tau_r, \tau_{r+1}[,\ r = 0,1,...,m \qquad (11.1)$$

gdzie $\tau_0 = 0$, $\tau_{m+1} = T$. Przyrost rozwiązania ΔX wywołuje przyrosty czasów przełączeń $\Delta\tau_1, \Delta\tau_2, ..., \Delta\tau_m$ i sterowania Δu, przy czym wartości sterowania w przedziałach jego stałości nie ulegają zmianie.

Ponieważ $\Delta\tau_0 = 0$ (rozwiązanie zaczyna się w chwili 0), ze związku (11.1) mamy

$$\Delta X(t) = \Delta X(0) + \sum_{i=1}^{r}\left[\int_{\tau_{i-1}+\Delta\tau_{i-1}}^{\tau_i+\Delta\tau_i} F(X(s)+\Delta X(s),u(s)+\Delta u(s))\mathrm{d}s - \int_{\tau_{i-1}}^{\tau_i} F(X(s),u(s))\mathrm{d}s\right] +$$

$$+ \int_{\tau_r+\Delta\tau_r}^{t} F(X(s)+\Delta X(s),u(s)+\Delta u(s))\mathrm{d}s - \int_{\tau_r}^{t} F(X(s),u(s))\mathrm{d}s$$

$$t\in]\tau_r,\tau_{r+1}[,\quad r=0,1,...,m \tag{11.2}$$

Wyliczamy dla $i=1,...,r$

$$\int_{\tau_{i-1}+\Delta\tau_{i-1}}^{\tau_i+\Delta\tau_i} F(X+\Delta X,u+\Delta u)\mathrm{d}s - \int_{\tau_{i-1}}^{\tau_i} F(X,u)\mathrm{d}s = \int_{\tau_{i-1}}^{\tau_i}\big(F(X+\Delta X,u(\tau_i-))-F(X,u(\tau_i-))\big)\mathrm{d}s +$$

$$+ \int_{\tau_{i-1}+\Delta\tau_{i-1}}^{\tau_{i-1}} F(X+\Delta X,u(\tau_{i-1}+))\mathrm{d}s + \int_{\tau_i}^{\tau_i+\Delta\tau_i} F(X+\Delta X,u(\tau_i-))\mathrm{d}s =$$

$$= \int_{\tau_{i-1}}^{\tau_i} \nabla_X F(X,u)^{\mathsf{T}}\Delta X\,\mathrm{d}s + F(X(\tau_i),u(\tau_i-))\Delta\tau_i - F(X(\tau_{i-1}),u(\tau_{i-1}+))\Delta\tau_{i-1} + o.$$

Podobnie

$$\int_{\tau_r+\Delta\tau_r}^{t} F(X+\Delta X,u+\Delta u)\mathrm{d}s - \int_{\tau_r}^{t} F(X,u)\mathrm{d}s = \int_{\tau_r}^{t} \nabla_X F(X,u)^{\mathsf{T}}\Delta X\,\mathrm{d}s - F(X(\tau_r),u(\tau_r+))\Delta\tau_r + o.$$

Po odrzuceniu reszt rzędu wyższego niż pierwszy i zastąpieniu przyrostów wariacjami otrzymujemy z (11.2)

$$\delta X(t) = \delta X(0) + \sum_{i=1}^{r}\int_{\tau_{i-1}}^{\tau_i} \nabla_X F(X(s),u(s))^{\mathsf{T}}\delta X(s)\mathrm{d}s + \int_{\tau_r}^{t} \nabla_X F(X(s),u(s))^{\mathsf{T}}\delta X(s)\mathrm{d}s +$$

$$+ \sum_{i=1}^{r}\big[F(X(\tau_i),u(\tau_i-)) - F(X(\tau_i),u(\tau_i+))\big]\delta\tau_i \tag{11.3}$$

Przez różniczkowanie (11.3) otrzymujemy równanie wariacyjne w postaci

$$\dot{\delta X} = J\,\delta X,\quad t\in]0,T[\setminus\{\tau_1,\tau_2,...,\tau_m\} \tag{11.4}$$

$$J(t) = \nabla_X F(X(t),u(t))^{\mathsf{T}} = \begin{bmatrix} f_x(x,u)^{\mathsf{T}} & 0 \\ -H_{xx}(x,\psi,u) & -f_x(x,u) \end{bmatrix} \tag{11.5}$$

z warunkiem końcowym wynikającym z (7.40)

$$[Q \quad I]\,\delta X(T) = 0 \tag{11.6}$$

Macierz Jacobiego J w równaniu (11.4) jest złożona z czterech podmacierzy

$$J = \begin{bmatrix} -A^{\mathsf{T}} & 0 \\ B & A \end{bmatrix} \tag{11.7}$$

gdzie

$$A = -f_x(x,u),$$

$$B = -\nabla_x^2(\psi^{\mathsf{T}} f(x,u)) = -H_{xx}(x,\psi,u) = -\begin{bmatrix} \psi^{\mathsf{T}}\dfrac{\partial^2 f}{\partial x_1^2} & \psi^{\mathsf{T}}\dfrac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \psi^{\mathsf{T}}\dfrac{\partial^2 f}{\partial x_1 \partial x_n} \\ \psi^{\mathsf{T}}\dfrac{\partial^2 f}{\partial x_2 \partial x_1} & \psi^{\mathsf{T}}\dfrac{\partial^2 f}{\partial x_2^2} & \cdots & \psi^{\mathsf{T}}\dfrac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \vdots & \vdots \\ \psi^{\mathsf{T}}\dfrac{\partial^2 f}{\partial x_n \partial x_1} & \psi^{\mathsf{T}}\dfrac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \psi^{\mathsf{T}}\dfrac{\partial^2 f}{\partial x_n^2} \end{bmatrix}.$$

Zatem wariacje $\delta x, \delta\psi$ dla wszystkich $t \in]0,T[\,\setminus\{\tau_1,\tau_2,\ldots,\tau_m\}$ spełniają układ równań

$$\delta\dot{x} = f_x^{\mathsf{T}}\delta x \tag{11.8}$$

$$\delta\dot{\psi} = -H_{xx}\delta x - f_x\delta\psi, \quad \delta\psi(T) = -Q\delta x(T) \tag{11.9}$$

Z (11.3) wynika, że w punktach $\tau_1,\tau_2,\ldots,\tau_m$ wariacja δX ma na ogół nieciągłości

$$\delta X(\tau_i+) - \delta X(\tau_i-) = \Delta_i F\,\delta\tau_i \tag{11.10}$$

przy czym

$$\Delta_i F = F(X(\tau_i),u(\tau_i-)) - F(X(\tau_i),u(\tau_i+)) =$$

$$= 2u(\tau_i-)\begin{bmatrix} f^1(x(\tau_i)) \\ -(\nabla f^1(x(\tau_i)))\psi(\tau_i) \end{bmatrix} \tag{11.11}$$

Pozostaje do wyznaczenia wariacja czasu przełączenia $\delta\tau_i$ związana z wariacją rozwiązania δX. Z tożsamości dla funkcji przełączającej

$$g(X(\tau_i)) \equiv 0 \tag{11.12}$$

oznaczając przez $\dot{X}(\tau_i\pm)$ i $\delta X(\tau_i\pm)$ odpowiednio granice prawo- i lewostronne pochodnej i wariacji rozwiązania (zgodnie ze znakiem przyrostu czasu przełączenia $\Delta\tau_i$ i jego wariacji $\delta\tau_i$), otrzymujemy

$$g(X(\tau_i+\Delta\tau_i)+\Delta X(\tau_i+\Delta\tau_i)) = g\left(X(\tau_i)+\dot{X}(\tau_i\pm)\delta\tau_i+\delta X(\tau_i\pm)+o\right)=$$

$$= g(X(\tau_i))+\nabla g(X(\tau_i))^{\mathsf{T}}\left(\dot{X}(\tau_i\pm)\delta\tau_i+\delta X(\tau_i\pm)\right)+o.$$

Zatem na mocy (11.12) otrzymujemy

$$\nabla g(X(\tau_i))^{\mathsf{T}}\left(\dot{X}(\tau_i\pm)\delta\tau_i+\delta X(\tau_i\pm)\right)=0 \tag{11.13}$$

oraz

$$\delta\tau_i = -\frac{\nabla g(X(\tau_i))^{\mathsf{T}}\delta X(\tau_i\pm)}{\nabla g(X(\tau_i))^{\mathsf{T}}\dot{X}(\tau_i\pm)} \tag{11.14}$$

Ponieważ

$$\nabla g = \operatorname{col}((\nabla f^1)\psi, f^1) \tag{11.15}$$

i

$$H_x = (\nabla f^0 + u\nabla f^1)\psi,$$

wyliczamy

$$(\nabla g)^{\mathsf{T}}\dot{X} = \psi^{\mathsf{T}}(\nabla f^1)^{\mathsf{T}}(f^0+f^1u)-(f^1)^{\mathsf{T}}(\nabla f^0+u\nabla f^1)\psi =$$

$$= \psi^{\mathsf{T}}(\nabla f^1)^{\mathsf{T}}f^0-\psi^{\mathsf{T}}(\nabla f^0)^{\mathsf{T}}f^1 = -\psi^{\mathsf{T}}[f^0,f^1] \tag{11.16}$$

gdzie $[f^0,f^1]$ oznacza nawias Liego. Zauważmy, że funkcja $t\mapsto\nabla g(X(t))^{\mathsf{T}}\dot{X}(t)$, a zatem także $t\mapsto\nabla g(X(t))^{\mathsf{T}}\delta X(t)$, jest ciągła w chwili przełączenia. Z (11.16) otrzymujemy

$$\delta\tau_i = \frac{\nabla g(X(\tau_i))^{\mathsf{T}}\delta X(\tau_i\pm)}{\psi(\tau_i)^{\mathsf{T}}[f^0,f^1](x(\tau_i))} \tag{11.17}$$

Ze względu na ciągłość licznika jest wszystko jedno, którą granicę weźmiemy. Po rozpisaniu równość (11.17) przybiera postać

$$\delta\tau_i = \frac{\psi(\tau_i)^{\mathsf{T}}\nabla f^1(x(\tau_i))^{\mathsf{T}}\delta x(\tau_i\pm)+f^1(x(\tau_i))^{\mathsf{T}}\delta\psi(\tau_i\pm)}{\psi(\tau_i)^{\mathsf{T}}[f^0,f^1](x(\tau_i))} \tag{11.18}$$

Podstawiając (11.17) do (11.10) otrzymujemy:

$$\delta X(\tau_i \pm) = Z_{i\mp}\delta X(\tau_i \mp) \tag{11.19}$$

$$Z_{i\mp} = I \pm \frac{\Delta_i F \nabla g(X(\tau_i))^{\mathsf{T}}}{\psi(\tau_i)^{\mathsf{T}}[f^0, f^1](x(\tau_i))} \tag{11.20}$$

Oczywiście

$$Z_{i-} = (Z_{i+})^{-1} \tag{11.21}$$

Zauważmy, że

$$Z_{i\mp} = I \pm \alpha \begin{bmatrix} b \\ -c \end{bmatrix} \begin{bmatrix} c \\ b \end{bmatrix}^{\mathsf{T}} = I \pm \alpha \begin{bmatrix} bc^{\mathsf{T}} & bb^{\mathsf{T}} \\ -cc^{\mathsf{T}} & -cb^{\mathsf{T}} \end{bmatrix} \tag{11.22}$$

gdzie:

$$\alpha = \frac{2u(\tau_i -)}{\psi(\tau_i)^{\mathsf{T}}[f^0, f^1](x(\tau_i))}$$

$$b = f^1(x(\tau_i)) \tag{11.23}$$

$$c = \left(\nabla f^1(x(\tau_i))\right)\psi(\tau_i)$$

Uwaga. Założenia o czasach przełączeń: $\tau_1 > 0$, $\tau_m < T$ można osłabić. Niech mianowicie $\tau_1 = 0$ i pochodna prawostronna funkcji $t \mapsto g(x(t), \psi(t))$ w zerze będzie różna od zera. Równość (11.10) dla $i = 1$ należy wtedy zastąpić przez

$$\delta X(0+) - \delta X(0) = \Delta_1 F \delta\tau_1 \tag{11.24}$$

gdzie:

$$\Delta_1 F = F(X(0), -u(0+)) - F(X(0), u(0+)) \tag{11.25}$$

$$\delta\tau_1 = \max\left\{0, \frac{\nabla g(X(0))^{\mathsf{T}} \delta X(0+)}{\psi(0)^{\mathsf{T}}[f^0, f^1](x(0))}\right\} \tag{11.26}$$

Niech teraz $\tau_m = T$ i pochodna lewostronna funkcji $t \mapsto g(x(t), \psi(t))$ w punkcie T będzie różna od zera. Równość (11.10) dla $i = m$ przybiera postać

$$\delta X(T-) = \delta X(T) - \Delta_m F \delta\tau_m \tag{11.27}$$

gdzie:

$$\Delta_m F = F(X(T), u(T-)) - F(X(T), -u(T-)) \tag{11.28}$$

$$\delta\tau_m = \min\left\{0, \frac{\nabla g(X(T))^{\mathsf{T}} \delta X(T-)}{\psi(T)^{\mathsf{T}}[f^0, f^1](x(T))}\right\} \tag{11.29}$$

11.2. Wrażliwość stanu końcowego

Rozpocznijmy od rozwiązania równania w wariacjach i wyznaczenia wrażliwości bez uwzględnienia warunków końcowych. Jeżeli wprowadzimy macierzowe rozwiązanie fundamentalne Φ, to dla dowolnego rozwiązania δX układu wariacyjnego (11.4), (11.6), (11.10) i dowolnych $t, s \in [0,T]$ można napisać związek

$$\delta X(t) = \Phi(t,s)\delta X(s) \tag{11.30}$$

Dwuargumentowa funkcja macierzowa Φ o wymiarach $2n \times 2n$ spełnia warunki

$$\Phi(s,s) = I, \quad \Phi(t,s) = \Phi(s,t)^{-1} \text{ dla } t,s \in]0,T[\setminus\{\tau_1, \tau_2, ..., \tau_m\} \tag{11.31}$$

oraz

$$\frac{\partial}{\partial t}\Phi(t,s) = J(t)\Phi(t,s) \tag{11.32}$$

$$\frac{\partial}{\partial s}\Phi(t,s) = -\Phi(t,s)J(s) \tag{11.33}$$

W chwilach przełączeń funkcja Φ ma nieciągłości, wynikające z (11.19)

$$\Phi(\tau_i \pm, s) = Z_{i\mp}\Phi(\tau_i \mp, s) \tag{11.34}$$

gdzie macierze $Z_{i\mp}$ są określone przez (11.20). Na mocy (11.31) i (11.21) otrzymujemy

$$\Phi(t, \tau_i \pm) = \Phi(t, \tau_i \mp)(Z_{i\mp})^{-1} = \Phi(t, \tau_i \mp)Z_{i\pm} \tag{11.35}$$

Równanie (11.32) z pierwszym warunkiem (11.31) i warunkami skoku pozwala na wyznaczenie wrażliwości rozwiązania równania kanonicznego w dowolnej chwili t na zmiany warunków początkowych.

Z (11.30), (11.32) i (11.34) mamy:

$$\delta X(t) = \Phi(t,0)\delta X(0)$$

$$\frac{\partial}{\partial t}\Phi(t,0) = J(t)\Phi(t,0), \quad t \in]0,T[\setminus\{\tau_1,\tau_2,...,\tau_m\} \tag{11.36}$$

$$\Phi(0,0) = I, \quad \Phi(\tau_i+,0) = Z_{i-}\Phi(\tau_i-,0), \quad i = 1,2,...,m$$

W zastosowaniach potrzebne będzie wyznaczenie wrażliwości rozwiązania równania kanonicznego w chwili końcowej T, na zmiany rozwiązania w dowolnej chwili t. Z (11.30), (11.33) i (11.35) otrzymujemy:

$$\delta X(T) = \Phi(T,t)\delta X(t)$$

$$\frac{\partial}{\partial t}\Phi(T,t) = -\Phi(T,t)J(t), \quad t \in]0,T[\setminus\{\tau_1,\tau_2,...,\tau_m\} \tag{11.37}$$

$$\Phi(T,T) = I, \quad \Phi(T,\tau_i-) = \Phi(T,\tau_i+)Z_{i+}^{-1}, \quad i = 1,2,...,m$$

Na koniec wyznaczmy wrażliwość stanu przy uwzględnieniu warunków końcowych (11.6). Zbadamy najpierw wrażliwość stanu w chwili t na zmiany stanu w chwili końcowej T, przy uwzględnieniu warunku końcowego (11.6) lub (11.9). Macierz Φ dekomponujemy na cztery podmacierze $n \times n$

$$\Phi = \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{21} & \Phi_{22} \end{bmatrix} \tag{11.38}$$

Zatem z (11.30), (11.38) i (11.9) otrzymujemy:

$$\delta x(t) = V_1(t)\delta x(T) \tag{11.39}$$

$$\delta \psi(t) = V_2(t)\delta x(T) \tag{11.40}$$

gdzie:

$$V_1(t) = \Phi_{11}(t,T) - \Phi_{12}(t,T)Q \tag{11.41}$$

$$V_2(t) = \Phi_{21}(t,T) - \Phi_{22}(t,T)Q \tag{11.42}$$

Oznaczając $V = \mathrm{col}(V_1,V_2)$ mamy z (11.4)

$$\dot{V} = JV, \quad t \in]0,T[\setminus\{\tau_1,\tau_2,...,\tau_m\} \tag{11.43}$$

a z (11.9), (11.39) i (11.40)

$$V(T) = \mathrm{col}(I,-Q) \tag{11.44}$$

Nieciągłości w chwilach przełączeń wynikają z (11.19)

$$V(\tau_i-) = Z_{i+}V(\tau_i+), \quad i=1,2,...,m \tag{11.45}$$

Scharakteryzujemy teraz wrażliwość stanu osiąganego w chwili T na zaburzenia stanu w chwili t, przy uwzględnieniu warunku końcowego (11.9). Przy założeniu, że macierz V_1 jest nieosobliwa, definiujemy *macierz wrażliwości*

$$W(t) = V_1(t)^{-1} \tag{11.46}$$

Zgodnie z (11.39) wynika

$$\delta x(T) = W(t)\delta x(t) \tag{11.47}$$

Dzięki temu, że J jest macierzą blokowo trójkątną dolną, macierz $W(t)$ spełnia równanie

$$\dot{W}(t) = W(t)A(t)^{\mathsf{T}}, \quad t \in]0,T[\setminus\{\tau_1,\tau_2,\ldots,\tau_m\}, \quad W(T) = I \tag{11.48}$$

Miarą wrażliwości rozwiązania jest norma spektralna macierzy $W(t)$ wyliczona dla $t \in]0,T[\setminus\{\tau_1,\tau_2,...,\tau_m\}$. Normę obliczamy według wzoru

$$\|W(t)\| = \max_{i=1,...,n} \sqrt{\lambda_i} \tag{11.49}$$

gdzie λ_i, $i=1,...,n$, są wartościami własnymi macierzy $W(t)^{\mathsf{T}}W(t)$ (Turowicz 1995).

11.3. Równanie Riccatiego

Nieciągłości macierzy wrażliwości w chwilach przełączeń można otrzymać za pomocą rozwiązania odpowiedniego równania Riccatiego. Jeżeli macierz $V_1(t)$ jest wszędzie nieosobliwa, to z (11.39) i (11.40) wynika zależność liniowa

$$\delta\psi(t) = K(t)\delta x(t) \tag{11.50}$$

gdzie $K(t) = V_2(t)V_1(t)^{-1}$ jest macierzą różniczkowalną wszędzie poza chwilami przełączeń. Zatem

$$\delta\dot{\psi} = \dot{K}\delta x + K\delta\dot{x}.$$

Przyjmując $B(t) = -H_{xx}(x(t),\psi(t),u(t))$, z równań (11.8) i (11.9) otrzymujemy równanie Riccatiego na K

$$\dot{K} = AK + KA^{\mathsf{T}} + B, \quad t \in]0,T[\setminus\{\tau_1,\tau_2,...,\tau_m\} \tag{11.51}$$

W rozważanym zadaniu równanie Riccatiego jest liniowe, a więc może być uważane za równanie Lapunowa. Z (11.9) i (11.50) wynika warunek końcowy

$$K(T) = -Q \tag{11.52}$$

Skoki w chwilach przełączeń wyznaczamy na podstawie (11.50) i (11.19). Macierz $Z_{i\mp}$ dekomponujemy na cztery podmacierze o wymiarach $n \times n$

$$Z_{i\mp} = \begin{bmatrix} Z_{i\mp}^{11} & Z_{i\mp}^{12} \\ Z_{i\mp}^{21} & Z_{i\mp}^{22} \end{bmatrix}. \tag{11.53}$$

Z (11.50) wynika

$$\delta\psi(\tau_i \pm) = K(\tau_i \pm)\delta x(\tau_i \pm) \tag{11.54}$$

Z (11.19) wynika:

$$\delta\psi(\tau_i \pm) = Z_{i\mp}^{21}\delta x(\tau_i \mp) + Z_{i\mp}^{22}\delta\psi(\tau_i \mp) = \left(Z_{i\mp}^{21} + Z_{i\mp}^{22}K(\tau_i \mp)\right)\delta x(\tau_i \mp) \tag{11.55}$$

$$\delta x(\tau_i \pm) = Z_{i\mp}^{11}\delta x(\tau_i \mp) + Z_{i\mp}^{12}\delta\psi(\tau_i \mp) = \left(Z_{i\mp}^{11} + Z_{i\mp}^{12}K(\tau_i \mp)\right)\delta x(\tau_i \mp) \tag{11.56}$$

Podstawiając (11.55) i (11.56) do (11.54) otrzymujemy

$$K(\tau_i \pm) = \left(Z_{i\mp}^{21} + Z_{i\mp}^{22}K(\tau_i \mp)\right)\left(Z_{i\mp}^{11} + Z_{i\mp}^{12}K(\tau_i \mp)\right)^{-1}, \quad i = 1,2,...,m \tag{11.57}$$

Zwróćmy uwagę, że w relacjach (11.51) i (11.52) występuje symetria. Aby wykazać, że macierz $K(t)$ jest symetryczna dla każdego t, trzeba udowodnić, że symetria jest zachowana w (11.57). Dla krótkości będziemy pisać

$$\kappa = K(\tau_i \mp), \quad Z_{jl} = Z_{i\mp}^{jl}, \quad j,l = 1,2.$$

Trzeba więc wykazać, że

$$(Z_{21} + Z_{22}\kappa)(Z_{11} + Z_{12}\kappa)^{-1} = \left((Z_{21} + Z_{22}\kappa)(Z_{11} + Z_{12}\kappa)^{-1}\right)^{\mathsf{T}} \tag{11.58}$$

jeśli κ jest macierzą symetryczną. Równość (11.58) zachodzi wtedy i tylko wtedy, gdy

$$(Z_{11} + Z_{12}\kappa)^{\mathsf{T}}(Z_{21} + Z_{22}\kappa) = (Z_{21} + Z_{22}\kappa)^{\mathsf{T}}(Z_{11} + Z_{12}\kappa),$$

czyli

$$Z_{11}^{\mathsf{T}}Z_{21} + Z_{11}^{\mathsf{T}}Z_{22}\kappa + \kappa Z_{12}^{\mathsf{T}}Z_{21} + \kappa Z_{12}^{\mathsf{T}}Z_{22}\kappa =$$

$$= Z_{21}^{\mathsf{T}}Z_{11} + \kappa Z_{22}^{\mathsf{T}}Z_{11} + Z_{21}^{\mathsf{T}}Z_{12}\kappa + \kappa Z_{22}^{\mathsf{T}}Z_{12}\kappa \tag{11.59}$$

Na mocy (11.22) otrzymujemy

$$Z_{11} = I \pm \alpha bc^{\mathsf{T}},\ Z_{12} = \pm \alpha bb^{\mathsf{T}},\ Z_{21} = \mp \alpha cc^{\mathsf{T}},\ Z_{22} = I \mp \alpha cb^{\mathsf{T}} \quad (11.60)$$

Wstawienie (11.60) do (11.59) pokazuje po prostych rachunkach, że (11.58) zachodzi, a więc macierz $K(t)$ jest wszędzie symetryczna.

Rozwiązanie równania Riccatiego wykorzystamy do określenia skoków macierzy $W(t)$ w chwilach przełączeń. Z (11.56) wynika

$$V_1(\tau_i \pm) = \left(Z_{i\mp}^{11} + Z_{i\mp}^{12} K(\tau_i \mp)\right) V_1(\tau_i \mp) \quad (11.61)$$

Ostatecznie otrzymujemy

$$W(\tau_i \pm) = W(\tau_i \mp)\left(Z_{i\mp}^{11} + Z_{i\mp}^{12} K(\tau_i \mp)\right)^{-1} \quad (11.62)$$

11.4. Przykład: wahadło na wózku

Zbadamy wrażliwość rozwiązań optymalnych dla systemu wahadła na wózku. Rozwiązanie optymalnoczasowe cechuje duża wrażliwość na zaburzenia trajektorii optymalnej. Dla zastosowań ma więc pierwszorzędne znaczenie jej określenie i zmniejszenie, kosztem – na przykład – wydłużenia czasu sterowania.

Równania stanu (3.18) i sprzężone (9.1) podajemy w formie kanonicznej (7.39). Dla prostoty zapisu oznaczmy $s = \sin x_2$, $c = \cos x_2$, $S = \sin 2x_2$, $C = \cos 2x_2$. Prawe strony równań kanonicznych przybierają wtedy postać:

$$\begin{aligned}
F_1 &= X_3 \\
F_2 &= X_4 \\
F_3 &= \frac{1}{d}(u - sX_4^2 - b_2 X_3) + \frac{c}{d}(s - b_3 X_4) \\
F_4 &= \frac{c}{d}(u - sX_4^2 - b_2 X_3) + \frac{c_4}{d}(s - b_3 X_4) \\
F_5 &= 0, \qquad F_6 = A_{23} X_7 + A_{24} X_8 \\
F_7 &= -X_5 + A_{33} X_7 + A_{34} X_8, \qquad F_8 = -X_6 + A_{43} X_7 + A_{44} X_8
\end{aligned} \quad (11.63)$$

gdzie $d = c_4 - c^2$, a wartości parametrów są podane w tabeli 3.3.

Wyrażenia A_{ij} są zdefiniowane w (9.2):

$$A_{23} = \frac{1}{d}\left(SF_3 + X_4^2 c - C - b_3 X_4 s\right),$$

$$A_{33} = \frac{b_2}{d},$$

$$A_{43} = \frac{1}{d}\left(2X_4 s + b_3 c\right),$$

$$A_{24} = \frac{1}{d}\left(SF_4 + X_4^2 C + (u - b_2 X_3)s - c_4 c\right),$$

$$A_{34} = \frac{b_2 c}{d},$$

$$A_{44} = \frac{1}{d}\left(X_4 S + b_3 c_4\right).$$

Macierz B ze wzoru (11.7) ma postać

$$B = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & B_{22} & B_{23} & B_{24} \\ 0 & B_{23} & 0 & 0 \\ 0 & B_{24} & 0 & B_{44} \end{bmatrix} \tag{11.64}$$

Jej niezerowe elementy są równe:

$$B_{22} = \frac{X_7}{d}\left(2S(1 - A_{23}) + 2CF_3 - X_4^2 s - b_3 X_4 c\right) +$$

$$+ \frac{X_8}{d}\left(-2S(A_{24} + X_4^2) + 2CF_4 + (u - b_2 X_3)c + c_4 s\right),$$

$$B_{23} = \frac{b_2 X_7 S}{d^2} - \frac{X_8}{d}\left(S A_{34} + b_2 s\right),$$

$$B_{24} = \frac{X_7}{d}\left(-S A_{43} + 2X_4 c - b_3 s\right) + \frac{X_8}{d}\left(-S A_{44} + 2X_4 C\right),$$

$$B_{44} = \frac{2X_7 s + X_8 S}{d}.$$

Równanie wariacyjne (11.7) ma postać:

$$\begin{aligned}
\delta\dot{X}_1 &= \delta X_3 \\
\delta\dot{X}_2 &= \delta X_4 \\
\delta\dot{X}_3 &= -A_{23}\delta X_2 - A_{33}\delta X_3 - A_{43}\delta X_4 \\
\delta\dot{X}_4 &= -A_{24}\delta X_2 - A_{34}\delta X_3 - A_{44}\delta X_4 \\
\delta\dot{X}_5 &= 0 \\
\delta\dot{X}_6 &= B_{22}\delta X_2 + B_{23}\delta X_3 + B_{24}\delta X_4 + A_{23}\delta X_7 + A_{24}\delta X_8 \\
\delta\dot{X}_7 &= B_{23}\delta X_2 - \delta X_5 + A_{33}\delta X_7 + A_{34}\delta X_8 \\
\delta\dot{X}_8 &= B_{24}\delta X_2 + B_{44}\delta X_4 - \delta X_6 + A_{43}\delta X_7 + A_{44}\delta X_8
\end{aligned} \tag{11.65}$$

Nieciągłości rozwiązań równania wariacyjnego w chwilach przełączeń są określone przez macierze Z_{i+} (11.20), które w rozważanym przykładzie scharakteryzowane są następująco. Dla dowolnych $\alpha, \beta \in \boldsymbol{R}^8$, takich że $\alpha = Z_{i+}\beta$, zachodzi $\alpha_j = \beta_j$ dla $j = 1,2,5,7,8$, oraz:

$$\alpha_3 = \beta_3 - M_i(-sX_8\beta_2 + \beta_7 + c\beta_8),$$

$$\alpha_4 = \beta_4 - cM_i(-sX_8\beta_2 + \beta_7 + c\beta_8),$$

$$\alpha_6 = \beta_6 - sX_8M_i(-sX_8\beta_2 + \beta_7 + c\beta_8),$$

$$M_i = \frac{2u(\tau_i -)}{d(X_5 + cX_6 + b_3X_7 + X_4sX_8)},$$

gdzie wszystkie wyrażenia obliczone są w punkcie τ_i.

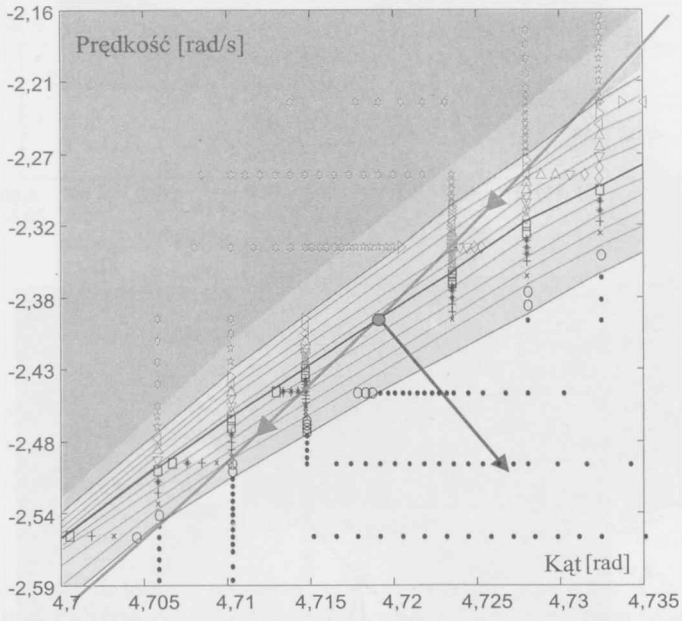
Wyprowadzone wyżej równania wariacyjne dla systemu wahadła na wózku, wraz z warunkami skoków w chwilach przełączeń, posłużą w rozdziałach 12 i 13 do wyznaczenia wrażliwości lokalnej rozwiązań optymalnych, scharakteryzowanej przez macierz $W(t)$.

Dla zadania optymalnoczasowego szczególnie interesująca jest wrażliwość horyzontu optymalnego na zaburzenia stanu na trajektorii. Jak wiadomo, wrażliwość taka, rozumiana w sensie pochodnej lub wariacji, jest źle określona: na różnych kierunkach może przyjmować różne wartości, a w ogólnym przypadku może być nieskończona.

Odnosząc się do zadania O2 POST+AKRO (postawienie wahadła z przesunięciem wózka), opisanego w podrozdziale 10.1, badania wrażliwości uzupełnimy przedstawieniem izochron dotarcia do celu z punktów leżących wokół trajektorii optymalnej.

Przebadano obszar wokół wybranych punktów trajektorii optymalnej (rys. 11.1–11.3). Rysunki pokazują obszary, z których dotarcie do celu trwa krócej, i obszary, z których trwa dłużej – w stosunku do czasu osiągnięcia celu z punktu trajektorii optymalnej zaznaczonego kółkiem w środku rysunku. Im ciemniejsze jest pole w porównaniu z polem, na którym leży zaznaczony punkt trajektorii optymalnej, tym większe jest opóźnienie dotarcia do celu z punktów takiego pola. I na odwrót, dotarcie do celu z pól jaśniejszych jest szybsze. Na każdym z rysunków narysowane są izochrony optymalnoczasowego dotarcia do celu. Prócz trajektorii optymalnej (z zaznaczonym kierunkiem ruchu), na rysunkach pokazano strzałką kierunek, w którym należy przesunąć trajektorię optymalną, by uzyskać trajektorię uodpornioną, mniej wrażliwą na zakłócenia.

Na rysunku 11.1 przedstawiono izochrony dla punktu leżącego mniej więcej w środku pomiędzy czwartym i piątym przełączeniem sterowania optymalnego.



Rys. 11.1. Izochrony na płaszczyźnie zmiennych stanu wahadła; czasy dotarcia do celu z otoczenia punktu na trajektorii optymalnoczasowej wybranego między przełączeniami czwartym i piątym

Znaki na rysunkach 11.1–11.2a i b:

◊ ▽ △ ◁ ▷ ☆ ☼

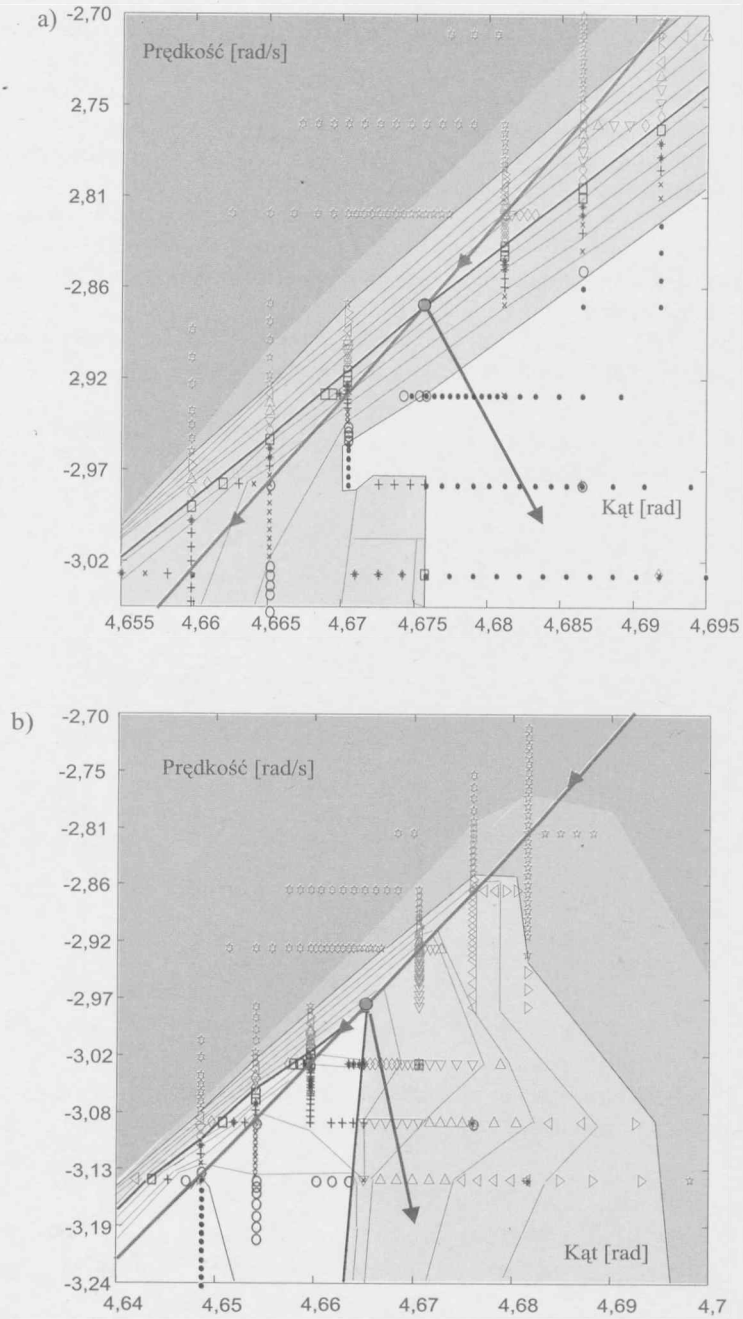
]0 , 1]% **]1, 2]%** **]2, 3]%** **]3, 4]%** **]4, 5]%** **]5, 10]%** **>10%**

oznaczają procentowe wydłużenie czasu dotarcia do celu liczone względem punktu centralnego oznaczonego strzałką, a znaki:

□ ✶ + ‹ ○ •

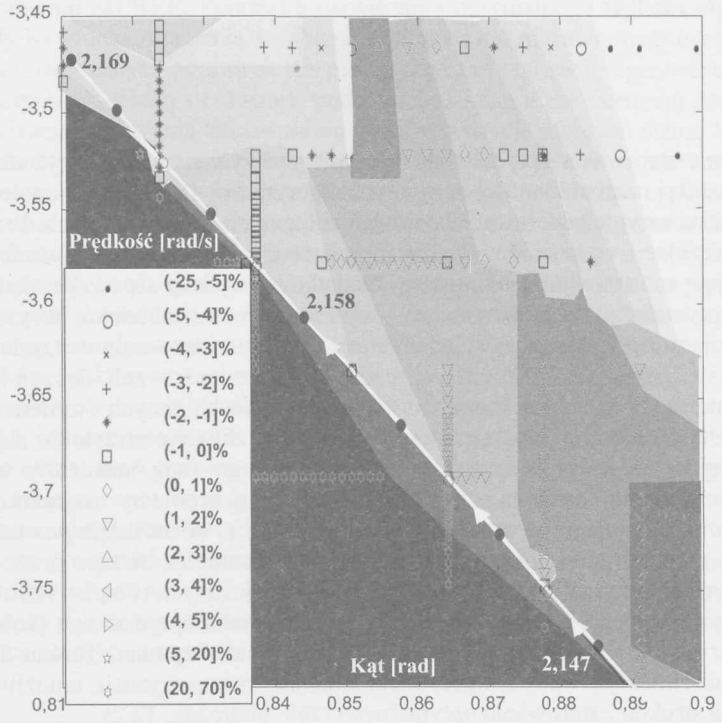
]0,–1[% **]–1,–2[%** **]–2,–3[%** **]–3,–4[%** **]–4,–5[%** **<–5%**

oznaczają procentowe skrócenie czasu.



Rys. 11.2. Izochrony na płaszczyźnie zmiennych stanu wahadła; czasy dotarcia do celu z otoczenia punktu na trajektorii optymalnoczasowej wybranego:
a) tuż przed piątym przełączeniem; b) w chwili piątego przełączenia

Na rysunku 11.2a, 11.2b i 11.3 przedstawiono izochrony dla punktu trajektorii optymalnej wybranego: tuż przed piątym przełączeniem, w chwili piątego przełączenia i pomiędzy szóstym i siódmym przełączeniem. Zauważmy, że procentowe wydłużenia i skrócenia czasów dotarcia do celu, z zaznaczonego punktu leżącego na trajektorii optymalnej w środku rysunku 11.3 (oznaczone w legendzie), różnią się znacznie od procentowych wydłużeń i skróceń na rysunkach 11.1, 11.2a i b. Dane liczbowe do rysunków zebrano w tabeli 11.2.



Rys. 11.3. Izochrony na płaszczyźnie zmiennych stanu wahadła; czasy dotarcia do celu z otoczenia punktu na trajektorii optymalnoczasowej wybranego między przełączeniami szóstym i siódmym

Tabela 11.2
Dane liczbowe do rysunków 11.1, 11.2a, 11.2b i 11.3

Numery rysunków	Współrzędne zakłócanego punktu trajektorii optymalnej stanu	Czas wystąpienia zakłócenia	Czas pozostały do celu	Wrażliwość (w najgorszym kierunku)
Rys. 11.1	(1,3563, 4,7191, −0,5598, −2,3923)	1,584 s	1,004 s	mała
Rys. 11.2a	(1,2597, 4,6757, −0,6111, −2,8706)	1,601 s	0,987 s	średnia
Rys. 11.2b	(1,2371, 4,6650, −0,6220, −2,9759)	1,604 s	0,984 s	duża
Rys. 11.3	(0,8272, 0,8506, 0,0863, −0,6731)	2,158 s	0,429 s	olbrzymia

12. Metody sterowań zbliżonych

Obliczanie sterowania optymalnego w czasie rzeczywistym może być niewykonalne z powodu za dużej czasochłonności używanych algorytmów. Dlatego proponuje się szybkie metody aproksymacyjne generowania sterowań zbliżonych do optymalnych. Przed przystąpieniem do regulacji wylicza się *off line* i zapamiętuje nominalne sterowanie optymalne i odpowiadającą mu trajektorię nominalną. Dodatkowo wylicza się i zapamiętuje gradient sterowania optymalnego względem stanu i – w zadaniach ze swobodnym horyzontem – pochodną horyzontu optymalnego względem stanu. Podczas sterowania w czasie rzeczywistym wylicza się *sterowania zbliżone* (ang. *neighboring controls* – zob. Bryson (1999) i tam podaną literaturę). W tym celu korzysta się z zapamiętanych danych i zmierzonych zaburzeń stanu. Dla problemów bez ograniczeń sterowanie zbliżone otrzymuje się przez rozwiązanie pomocniczego problemu liniowo-kwadratowego (ang. *accessory optimization problem*). Rozszerzenia koncepcji sterowań zbliżonych na problemy z ograniczeniami nierównościowymi znajdujemy w pracach Pescha (1989a, b). W metodzie *powtarzanych poprawek* (ang. *repeated corrections*) uaktualnia się sterowanie na bieżąco przez całkowanie równań dynamiki i optymalizację metodą strzałów wielokrotnych (Pesch 1989b). To podejście stosuje się przy zachowaniu stałej struktury sterowania optymalnego (zob. podrozdz. 12.1). Zlinearyzowany regulator adaptacyjny (Korytowski, Szymkat, Turnau 2001), który łączy regulator zlinearyzowany z algorytmem zmiennej parametryzacji, umożliwia uwzględnienie zmian struktury sterowania optymalnego (zob. podrozdz. 12.2).

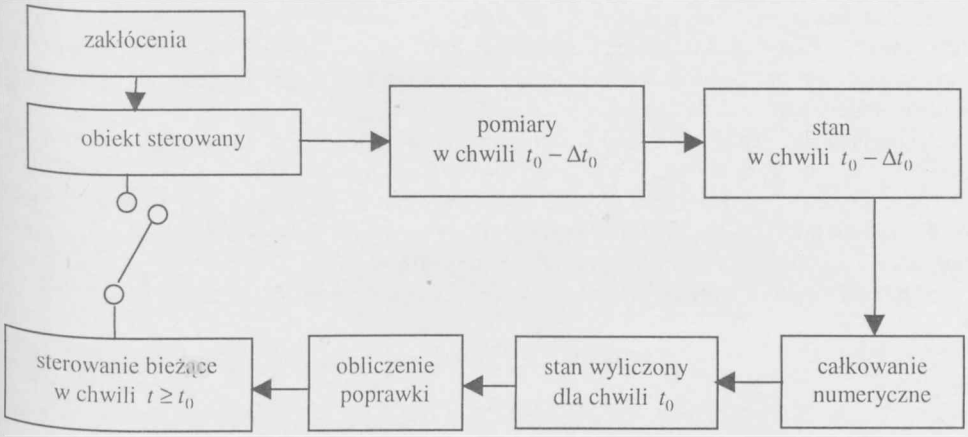
12.1. Metoda powtarzanych poprawek

W pracach Pescha (1989a, b, 1994) rozważa się zagadnienie sterowania optymalnego z ustalonym bądź swobodnym horyzontem, z ograniczeniami nierównościowymi na sterowanie i stan. Warunki konieczne optymalności, wynikające z zasady maksimum, zapisuje się w postaci sparametryzowanego, wielopunktowego problemu brzegowego (MPBVP). Jest to układ równań różniczkowych i algebraicznych wyrażających związki między zmiennymi stanu, zmiennymi sprzężonymi i sterowaniem. Jako parametr najczęściej wybiera się stan początkowy. Rozwiązanie optymalne problemu MPBVP dla ustalonej wartości parametru stanowi nominalny punkt odniesienia. Istotą metody jest linearyzacja zależności rozwiązania optymalnego od parametru w otoczeniu punktu nominalnego. Potrzebne do tego pochodne wyznacza się bądź numerycznie (poprzez aproksymację różnicową), bądź przez

rozwiązanie równań wariacyjnych z uwzględnieniem uwikłanego różniczkowania położenia punktów nieciągłości. Zakres zastosowania tego podejścia jest ograniczony wymaganiem różniczkowalności, które pociąga za sobą między innymi konieczność zachowania stałej struktury sterowań bangbangowych. Znajomość pochodnej rozwiązania optymalnego pozwala wyliczyć (małą) poprawkę rozwiązania optymalnego wywołaną (małą) zmianą parametru.

Podejście Pescha polega na wykorzystaniu metody strzałów wielokrotnych do rozwiązywania problemu MPBVP. Poprawkę rozwiązania optymalnego wylicza się, wykonując krok metody Newtona z macierzą Jacobiego obliczoną dla niezaburzonej wartości parametru. Metoda powtarzanych poprawek stanowi praktyczną realizację opisywanego podejścia w rzeczywistych procesach sterowania, poddanych działaniu zakłóceń. Jej istotą jest okresowa aktualizacja sterowania stosownie do obserwowanych zaburzeń stanu, który traktuje się jako parametr kolejno rozwiązywanych problemów MPBVP. W najprostszym wariancie trajektorie nominalne dla tych problemów otrzymuje się przez obcinanie pierwotnej trajektorii nominalnej do przedziałów czasu pozostałych do zakończenia procesu. Nowe, poprawione rozwiązanie wyznacza się w jednym kroku metody Newtona, a więc nie żądając dużej dokładności. W bardziej złożonym wariancie czas pomiędzy aktualizacjami sterowania przeznacza się na dokładniejsze wyznaczenie nowego rozwiązania nominalnego. Ze specyfiki stosowanej metody strzałów wielokrotnych wynika ograniczenie obu tych wariantów do przypadków, w których zachowywane jest założenie różniczkowalności, zatem przede wszystkim założenie o stałej strukturze sterowania. Środkiem zaradczym, proponowanym przez Pescha, jest przygotowanie z góry zbioru potencjalnych trajektorii i sterowań nominalnych. Do aktualizacji sterowania wybiera się najbliższe rozwiązanie z tego zbioru. Postępowanie takie na ogół wymaga jednak ogromnych nakładów obliczeniowych *off line* i zaangażowania dużych zasobów pamięci sterownika.

Algorytm powtarzanych poprawek jest szybki, dzięki szybkiej zbieżności metody strzałów, ale ma też pewne wady aplikacyjne. Na przykład chwilowy brak danych z pomiaru powoduje, że nie można uzyskać bieżącego stanu. Schemat z rysunku 12.1 pokazuje ideę implementacji algorytmu powtarzanych poprawek.



Rys. 12.1. Schemat działania metody powtarzanych poprawek

Określa się maksymalny czas Δt_0 (wynikający z ograniczeń narzucanych przez dynamikę obiektu), przeznaczony na rozwiązanie równań stanu i wyliczenie sterowania. Przedział $[t_0 - \Delta t_0, t_0]$, w którym całkuje się równania, jest niewielki. Za sterowanie przyjmuje się sterowanie z kroku poprzedniego, lub nominalne, gdy sterowanie z kroku poprzedniego nie jest dostępne. Sterowanie bieżące jest wyznaczane z antycypacją, z uwzględnieniem wyliczonej odchyłki trajektorii od trajektorii nominalnej w chwili przyszłej t_0.

12.2. Zlinearyzowany regulator adaptacyjny

W tym podrozdziale zajmiemy się konstrukcją liniowej aproksymacji regulatora optymalnego dla problemów sterowania optymalnego z ograniczeniami na sterowanie. Sterowanie bangbangowe jest w pełni scharakteryzowane przez swoją wartość początkową i czasy przełączeń. Wartość początkowa i liczba przełączeń określa *strukturę* sterowania. Regulator zlinearyzowany wyliczany na bieżąco adaptuje się do odchyłek trajektorii od trajektorii optymalnej i jest zawsze związany z określoną strukturą. Zakłada się, że w pewnym otoczeniu trajektorii stanu, w którym stosujemy regulator zlinearyzowany, ta struktura jest stała.

Podstawą konstrukcji regulatora jest rozwinięcie wyników z dwóch dziedzin teorii sterowania optymalnego: analizy wrażliwości (Malanowski 1987; Orrell, Zeidan 1988; Maurer, Pesch 1994; Malanowski, Maurer 1996) oraz metod optymalizacji dynamicznej, a w szczególności metody sterowań zbliżonych do optymalnych (Bryson 1999; Pesch 1989a, b). Regulator zlinearyzowany jest opisany przez zależność liniową pomiędzy mierzonymi lub wyliczonymi odchyłkami stanu oraz odpowiednimi poprawkami czasów przełączeń.

Jak wspomniano w podrozdziale 12.1, podejście Pescha (1989b) zwane metodą powtarzanych poprawek, w zasadzie opiera się na założeniu stałej struktury sterowania. W podejściu proponowanym poniżej (Korytowski, Szymkat, Turnau 2001), nie wymaga się stałej struktury sterowania. Jest to możliwe dzięki połączeniu schematu regulatora zlinearyzowanego z algorytmem 8.5 zmiennej parametryzacji opisanym w podrozdziale 8.3. W proponowanym schemacie dla aktualnie przyjętej struktury stosuje się adaptacyjny regulator zlinearyzowany. Zmiany struktury przez generację lub redukcję przełączeń wymagają uruchomienia procedur optymalizacji dynamicznej i zwiększają tylko nieznacznie nakład obliczeń, gdyż występują stosunkowo rzadko.

Dla jasności przedstawienia wyniki numeryczne odniesiono do problemu minimalnonormowego z horyzontem ustalonym. Opisana metoda syntezy zlinearyzowanego regulatora adaptacyjnego z powodzeniem może być stosowane do ogólniejszych problemów sterowania optymalnego, ze wskaźnikiem jakości zależnym od stanu końcowego i zmiennego horyzontu, a więc także do problemu optymalnoczasowego.

Przepiszmy równanie (7.1) dla postawionego zadania minimalnonormowego

$$\begin{aligned} &\dot{x}(t) = f(x(t),u(t)) = f^0(x(t)) + f^1(x(t))\,u(t), \quad t \in [0,T] \\ &x(0) = x^0, \quad x(t) \in \boldsymbol{R}^n \end{aligned} \tag{12.1}$$

O funkcjach f^0 i f^1 zakładamy, że są ciągłe i dwukrotnie różniczkowalne. Zbiór sterowań

dopuszczalnych U_{ad} zawiera wszystkie prawostronnie ciągłe funkcje skalarne o postaci $u:[0,\infty[\to[-u_{\max},u_{\max}]$. Dla systemu (12.1) minimalizujemy kwadratowy wskaźnik jakości (7.30). Hamiltonian systemu wylicza się ze wzoru (7.20). Równania sprzężone i warunek końcowy są przedstawione odpowiednio równaniami (7.10) i (7.40), zaś antygradient podaje formuła (7.33). Rzut antygradientu g na zbiór dopuszczalny U w punkcie u definiuje formuła (7.37). Sterowanie stałohoryzontowe (7.36) maksymalizuje hamiltonian (7.20). Wzór (7.36) można podać w postaci

$$u = v(x,\psi) = u_{\max}\ \operatorname{sgn} g(x,\psi) \tag{12.2}$$

Ograniczamy się do przypadku, gdy funkcja $[0,T]\ni t\mapsto g(x(t),\psi(t))$ przyjmuje wartość zerową w skończonej liczbie punktów $\tau_1,\tau_2,...,\tau_m$, $0<\tau_1<\tau_2<...<\tau_m<T$ (czasy przełączeń). Funkcja ta jest różniczkowalna w sposób ciągły. Zakładamy, że pochodna odwzorowania $t\mapsto g(x(t),\psi(t))$ jest różna od zera w punktach τ_i, $i=1,...,m$.

Ze wzorów (11.18) i (11.50) otrzymuje się zależność wariacji czasów przełączeń od wariacji trajektorii stanu:

$$\delta\tau_i = \Lambda_{i\pm}\delta x(\tau_i\pm),\quad i=1,2,...,m \tag{12.3}$$

$$\Lambda_{i\pm} = \frac{\psi(\tau_i)^{\mathrm{T}}\nabla f^1(x(\tau_i))^{\mathrm{T}} + f^1(x(\tau_i))^{\mathrm{T}}K(\tau_i\pm)}{\psi(\tau_i)^{\mathrm{T}}[f^0,f^1](x(\tau_i))} \tag{12.4}$$

Załóżmy najpierw, że przyrost stanu δx, powstały w wyniku zakłócenia stanu początkowego o pewną znaną wartość $\delta x(0)$, spełnia równanie wariacyjne w całym przedziale $[0,T]$. W tym przypadku wartości $\delta x(\tau_i\pm)$ mogą zostać obliczone z góry przez rozwiązanie równań (11.4), (11.19) i (11.6). Odpowiednie poprawki czasów przełączeń $\delta\tau_i$ można wprowadzać podczas procesu sterowania pod warunkiem, że

$$\tau_i+\delta\tau_i<\tau_{i+1}+\delta\tau_{i+1},\quad i=0,1,...,m \tag{12.5}$$

Z definicji $\tau_0+\delta\tau_0=0$, $\tau_{m+1}+\delta\tau_{m+1}=T$.

Podobne rozumowanie odnosi się do przypadku, gdy przyrost stanu δx spełnia równanie wariacyjne w przedziale $[t,T]$ dla pewnego $t\in[0,T[$ i wartość $\delta x(t)$ jest zadana. W oparciu o zasadę optymalności, wystarczające jest wówczas rozwiązanie równań (11.4), (11.19) i (11.6) w przedziale $[t,T]$. Oczywiście, możemy poprawiać tylko te czasy τ_i, dla których warunek (12.5) jest spełniony łącznie z warunkiem

$$\tau_i>t,\quad i=1,2,...,m \tag{12.6}$$

Ze wzoru (11.39), $\delta x(\tau_i\pm)=V_1(\tau_i\pm)V_1(t)^{-1}\delta x(t)$. Wstawiając to wyrażenie do (12.3)

i biorąc pod uwagę definicję macierzy wrażliwości (11.46), $W(t) = V_1(t)^{-1}$, otrzymujemy ogólną postać regulatora zlinearyzowanego

$$\delta\tau_i = \Pi_i W(t)\delta x(t), \quad i = 1,2,...,m \tag{12.7}$$

gdzie

$$\Pi_i = \frac{\nabla g(X(\tau_i))^{\mathrm{T}} V(\tau_i \pm)}{\psi(\tau_i)^{\mathrm{T}}[f^0, f^1](x(\tau_i))}$$

$$\nabla g(X(\tau_i))^{\mathrm{T}} V(\tau_i \pm) = \tag{12.8}$$

$$= \psi(\tau_i)^{\mathrm{T}} \nabla f^1(x(\tau_i))^{\mathrm{T}} V_1(\tau_i \pm) + f^1(x(\tau_i))^{\mathrm{T}} V_2(\tau_i \pm)$$

Wartości Π_i nie są zależne od granic wziętych do obliczeń we wzorach (12.8).

Jeżeli założymy, że w pewnej chwili $t \in]\tau_{i-1} + \delta\tau_{i-1}, \tau_i + \delta\tau_i[$, dla pewnego i, $1 \le i \le m$ znane jest zakłócenie $\delta x(t)$, to poprawkę $\delta\tau_i$ można wyliczyć używając innego sposobu. Określmy mianowicie macierzowe rozwiązanie fundamentalne równania na wariację stanu $\partial\Phi(t,s)/\partial t = -A(t)^{\mathrm{T}}\Phi(t,s)$, $\Phi(s,s) = I$, dla każdego t, s z przedziału $[0,T]$. Równość

$$\delta\tau_i = \Lambda_{i\pm}\Phi(\tau_i, t)\delta x(t) \tag{12.9}$$

jest równoważna (12.7), gdzie Λ_{i-} odpowiada $t < \tau_i$ i Λ_{i+} odpowiada $t > \tau_i$. Decyzja dotycząca wartości poprawki $\delta\tau_i$ winna być powzięta najpóźniej, jak to możliwe. Krytyczna, ostatnia jest chwila t spełniająca warunek $t - \tau_i = \Lambda_{i\pm}\Phi(\tau_i, t)\delta x(t) = \Pi_i W(t)\delta x(t)$.

Schemat obliczeniowy adaptacyjnego algorytmu sterowania jest dwupoziomowy. Na dolnym poziomie działa regulator zlinearyzowany. Regulator ten w każdym kroku aktualizacji sterowania wylicza dodatkowo dwie wielkości, dotyczące odcinka czasu od chwili bieżącej do czasu końcowego. Są to: norma rzutu antygradientu wskaźnika jakości na zbiór dopuszczalny w przestrzeni sterowań i wartość oczekiwana wskaźnika jakości. Po przekroczeniu przez obie wielkości ustalonych progów następuje uaktywnienie algorytmu poziomu górnego. Algorytmem tym jest optymalizacja dynamiczna metodą zmiennej parametryzacji (opisana w podrozdziale 8.3), przystosowana dla obecnego zadania. Zmiennymi decyzyjnymi w tej optymalizacji są czasy przełączeń, a optymalna struktura sterowania jest wyznaczana dzięki mechanizmom generacji i redukcji przełączeń.

Powrót na dolny poziom następuje, gdy norma rzutu spadnie poniżej ustalonego progu. Regulator zlinearyzowany zostaje ponownie wyliczony i staje się aktywnym algorytmem dolnego poziomu sterowania. Charakterystyczne dla tego schematu adaptacyjnego jest, że struktura sterowania podlega adaptacji podczas procesu sterowania. Tym różni się proponowany sposób od metody powtarzanych poprawek Pescha, opisanej w poprzednim podrozdziale.

Adaptacyjny algorytm sterowania

1° Podstaw $t_0 := 0$.

2° Znajdź rozwiązanie optymalne w przedziale $[t_0, T]$ ze sterowaniem optymalnym u i czasami przełączeń $\tau_i \in]t_0, T[$, $1 \le i \le m$; oblicz macierz V i wektory Π_i dla wszystkich czasów przełączeń.

3° Wybierz krok czasowy $\Delta t \le T - t_0$, wykonaj sterowanie u w przedziale $[t_0, t_0+\Delta t]$ i podstaw $t_0 := t_0+\Delta t$.

4° Określ odchyłkę stanu $\Delta x(t_0)$. Dla wszystkich czasów przełączeń $\tau_i > t_0$ wyznacz poprawki $\Delta\tau_i = \Pi_i w$, gdzie w jest wyliczone z równania $V_1(t_0)w = \Delta x(t_0)$. Znajdź poprawione wartości $\tau_i' = \tau_i + \Delta\tau_i$ i w ten sposób określ nowe sterowanie u w przedziale $[t_0, T]$.

5° Uaktualnij stan początkowy. Wylicz normę rzutu antygradientu wskaźnika jakości na zbiór dopuszczalny w przestrzeni sterowań i wartość oczekiwaną wskaźnika jakości. Jeżeli progi ustalone dla obu tych wielkości są przekroczone, to wróć do 2°. W przeciwnym wypadku idź do 3°. ■

Zakres stosowalności algorytmu można poszerzyć o przypadki, w których warunki (12.5) i (12.6) nie są spełnione w przedziale $[t_0, T]$ (zob. krok 4°). Wprowadźmy oznaczenie $i_0 = \min\{i : \tau_i > t_0\}$, i rozważmy wektor przyrostów $s\,\mathrm{col}(\Delta\tau_{i_0}, ..., \Delta\tau_m)$. Dla s rosnącego od 0 do 1, kolejno usuwamy wszystkie przełączenia, dla których odpowiednie wartości $\tau_i'(s) = \tau_i + s\Delta\tau_i$, $i_0 \le i \le m$, osiągają brzeg zbioru dopuszczalnego (12.5), (12.6). Za każdym razem, gdy zostanie naruszone ograniczenie $\tau_i'(s) > t_0$, wartość początkowa sterowania $u(t_0)$ zmienia znak.

12.3. Eksperyment symulacyjny

Rozważmy konstrukcję regulatora zlinearyzowanego dla zadania O2 POST+AKRO z podrozdziału 3.2, z ustalonym horyzontem $T = 13{,}8$. Wartości liczbowe w tym podrozdziale odnoszą się do zmiennych przeskalowanych (zob. (3.11)–(3.17)), są więc bezwymiarowe. Horyzont T jest nieznacznie mniejszy od horyzontu optymalnego $t^* = 13{,}97$. Współczynniki wzmocnienia regulatora zlinearyzowanego podano w tabeli 12.1.

Przedstawimy ciąg eksperymentów symulacyjnych, mających na celu przebadanie działania regulatora w otoczeniu optymalnej trajektorii stanu. Zakłócenia stanu wprowadza się dodając do prawych stron równań stanu szum stochastyczny $e(t)$. Funkcja e jest odcinkami stała, równa e_k w k-tym przedziale dyskretyzacji metody numerycznej RK4 (przy założonej długości kroku 0,005). Ciąg e_k jest pseudolosowym białym szumem gaussowskim o zerowej wartości średniej i odchyleniu standardowym σ.

Tabela 12.1
Współczynniki wzmocnienia regulatora zlinearyzowanego

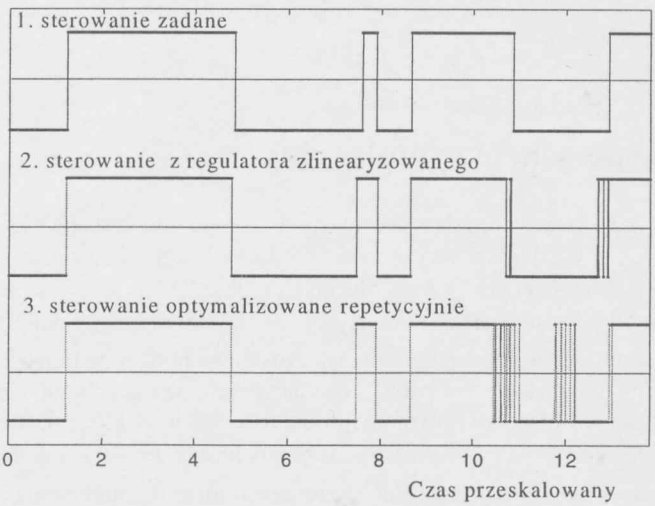
Π_1	541,1700	138,8450	69,9390	154,2580
Π_2	−363,9890	−139,5490	−152,2050	−199,7390
Π_3	997,4820	20,6595	75,4073	−23,7129
Π_4	−1428,9700	−202,4140	−348,4180	−262,9350
Π_5	181,3690	−153,5540	−122,9010	−245,0310
Π_6	−174,6790	−125,2610	−161,1670	−196,2860
Π_7	61,4824	67,9320	70,9232	96,2348

Przeprowadzono dwie serie eksperymentów: z $\sigma = 0{,}05$ i $\sigma = 0{,}1$. W każdej serii wybrano 1000 ciągów zakłóceń e_k.

Dla każdego ciągu zakłóceń porównano efektywność trzech sterowań:

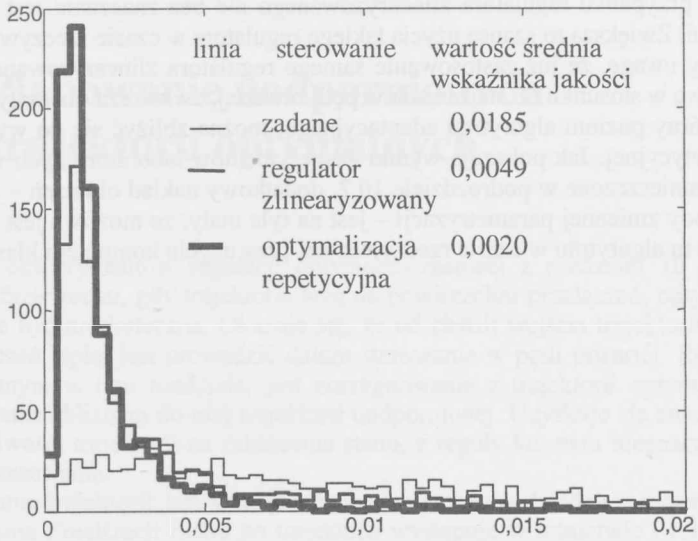
1) sterowania zadanego (jest to sterowanie optymalne dla systemu bez zakłóceń);
2) sterowania z regulatora zlinearyzowanego (jest to sterowanie otrzymywane za pomocą regulatora zlinearyzowanego opisanego powyżej, przy przyjęciu kroku czasowego $\Delta t = 0{,}1$);
3) sterowania optymalizowanego repetycyjnie (jest to sterowanie optymalizowane co krok czasowy $\Delta t = 0{,}1$).

Te trzy sterowania dla wybranego przypadku pokazano na rysunku 12.2.

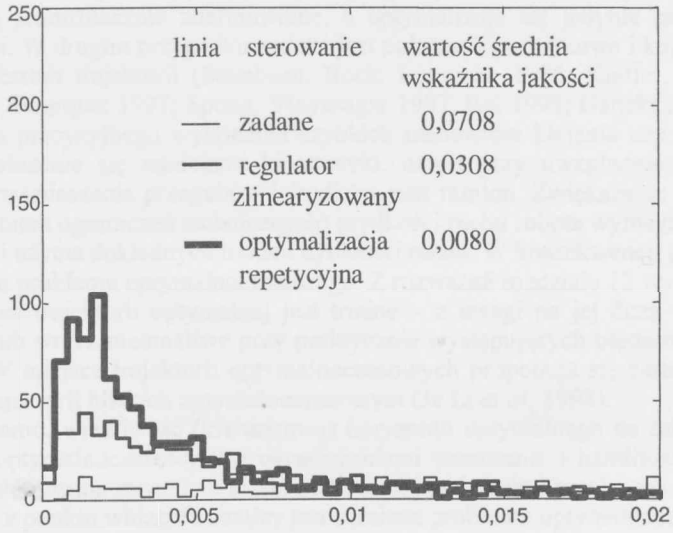


Rys. 12.2. Sterowania dla wybranej sekwencji zakłóceń. Wartości wskaźnika jakości: 1 – 0,23168, 2 – 0,00808, 3 – 0,00165

Znormalizowane histogramy wartości wskaźnika jakości pokazano na rysunku 12.3 (dla $\sigma = 0{,}05$) i na rysunku 12.4 (dla $\sigma = 0{,}1$).



Rys. 12.3. Histogramy wskaźnika jakości ($\sigma = 0{,}05$)



Rys. 12.4. Histogramy wskaźnika jakości ($\sigma = 0{,}1$)

Łatwo zauważyć, że dodatkowe przełączenia generowane przez schemat optymalizacji repetycyjnej mogą pojawiać się zupełnie niezależnie od zadanych, optymalnych przełączeń. To właśnie, w połączeniu z nieliniowością problemu, wyjaśnia przewagę (mierzoną wartościami wskaźnika jakości) algorytmu repetycyjnego nad pozostałymi. Należy jednak podkreślić, że w przypadku regulatora zlinearyzowanego nie bez znaczenia jest niski nakład i czas obliczeń. Zwiększa to szansę użycia takiego regulatora w czasie rzeczywistym.

Zwróćmy uwagę, że już zastosowanie samego regulatora zlinearyzowanego przynosi istotną poprawę w stosunku do sterowania w pętli otwartej, zwłaszcza dla małych zakłóceń. Dołączając górny poziom algorytmu adaptacyjnego, można zbliżyć się do wyników optymalizacji repetycyjnej. Jak pokazują wyniki eksperymentów laboratoryjnych w czasie rzeczywistym, zamieszczone w podrozdziale 10.2, dodatkowy nakład obliczeń – dzięki zastosowaniu metody zmiennej parametryzacji – jest na tyle mały, że możliwa jest implementacja opisanego tu algorytmu w czasie rzeczywistym, przy użyciu komputera klasy PC.

13. Planowanie uodpornionych trajektorii optymalnych

Wyniki eksperymentów regulacji optymalnoczasowej z rozdziału 10 pokazują, że w końcowej fazie ruchu, gdy trajektorie leżą na powierzchni przełączeń, optymalizacja na bieżąco może być nieskuteczna. Okazuje się, że od chwili wejścia trajektorii na powierzchnię przełączeń lepiej jest prowadzić dalsze sterowanie w pętli otwartej. Rozwiązaniem, zaproponowanym w tym rozdziale, jest zrezygnowanie z trajektorii optymalnoczasowej i wygenerowanie zbliżonej do niej trajektorii uodpornionej. Uzyskuje się znaczące zmniejszenie wrażliwości trajektorii na zakłócenia stanu, z reguły kosztem nieznacznego wydłużenia czasu sterowania.

Planowanie trajektorii jest ważnym zadaniem w wielu dziedzinach sterowania. Problem zadawania i realizacji ruchu po trajektorii występuje w lotnictwie i robotyce. Optymalnoczasowe manewry samolotu, czy reorientacja satelity są trudnymi zadaniami aplikacyjnymi teorii sterowania (Seywald 1994; Seywald *et al.* 1994; Seywald, Kumar 1995; Korytowski *et al.* 1999). Charakterystycznym zadaniem dla robota jest podążanie końcówki wykonawczej po zadanej ścieżce lub przejście z punktu do punktu. W pierwszym przypadku ruch robota odbywa się po zadanej krzywej w przestrzeni kartezjańskiej. Położenia przegubów są jednoznacznie zdefiniowane, a optymalizuje się jedynie profil prędkości wzdłuż ścieżki. W drugim przypadku zadane jest położenie początkowe i końcowe, a optymalizuje się kształt trajektorii (Steinbach, Bock, Longman 1995; Kieffer, Cahill, James 1997; Renaud, Fourquet 1997; Spong, Vidyasagar 1997; Rai 1998; Galicki 2000; Tchoń *et al.* 2000). Dla precyzyjnego wykonania szybkich manewrów klejenia czy transportu nie wystarcza posłużenie się modelami kinematyki, nawet przy uwzględnieniu ograniczeń prędkości i przyspieszenia przegubów i środków mas ramion. Zwiększanie do maksimum (z uwzględnieniem ograniczeń technicznych) prędkości ruchu robota wymaga zastosowania optymalizacji i użycia dokładnych modeli dynamiki ruchu. W konsekwencji prowadzi to do rozwiązywania problemu optymalnoczasowego. Z rozważań rozdziału 12 wynika, że utrzymywanie się na trajektorii optymalnej jest trudne – z uwagi na jej dużą wrażliwość na zakłócenia – lub wręcz niemożliwe przy praktycznie występujących błędach modelowania i pomiarów. W miejsce trajektorii optymalnoczasowych proponuje się zastosowanie uodpornionych trajektorii bliskich optymalnoczasowym (Ju Li *et al.* 1998).

Jak wiadomo, wrażliwość (kierunkowa) horyzontu optymalnego na zaburzenia stanu w zadaniach optymalnoczasowych z ograniczeniami sterowania i hamiltonianem liniowo zależnym od sterowania może być nieskończona na powierzchni przełączeń. Najprostszym rozwiązaniem z punktu widzenia analizy jest zamiana problemu optymalnoczasowego (7.7) na stałohoryzontowy (7.5). Trafienie w punkt docelowy zastępuje się osiągnięciem jego małego otoczenia przy horyzoncie ustalonym, nieznacznie krótszym od optymalnego. Rozwiązanie optymalne tego problemu będziemy nazywali *stałohoryzontowym*. Samo prze-

kształcenie trajektorii optymalnoczasowej w stałohoryzontową nie oznacza jej skutecznego uodpornienia. Wprawdzie im większy dopuści się błąd trafienia w cel, tym bardziej zmniejszy się wrażliwość trajektorii, jednak w praktyce powinien on mieścić się w granicach błędów pomiaru wielkości wyjściowych lub założonej dokładności osiągnięcia celu. Dopiero odpowiednie przesunięcie trajektorii stałohoryzontowej prowadzi do istotnego zmniejszenia jej wrażliwości. Metoda planowania takich przesunięć trajektorii, przedstawiona w tym rozdziale, polega na stochastycznym badaniu otoczenia trajektorii optymalnoczasowej, a następnie wybieraniu punktów – kandydatów do przesunięcia – i kierunków przesunięcia tych punktów. Otrzymuje się w ten sposób tak zwane *cele pośrednie*. Dla pierwszego celu pośredniego wyznacza się trajektorię optymalnoczasową, a następnie, po nieznacznym skróceniu horyzontu, optymalną trajektorię stałohoryzontową z pierwszym celem pośrednim jako punktem docelowym. Jej stan końcowy jest pierwszym *punktem pośrednim*. Dalsze cele i punkty pośrednie wyznacza się w ten sam sposób, przyjmując ostatnio otrzymany punkt pośredni jako punkt początkowy. Uodporniona trajektoria jest sklejeniem optymalnych trajektorii stałohoryzontowych. Cel sterowania zostaje osiągnięty w przybliżeniu, w czasie nieco dłuższym od optymalnego. Błąd trafienia w cel, przyczyniający się do zmniejszenia wrażliwości na zakłócenia, powinien być pomijalnie mały z punktu widzenia zastosowań praktycznych i nie przekraczać błędów pomiaru wielkości wyjściowych.

13.1. Badanie otoczenia trajektorii optymalnej metodą stochastyczną

Niech T będzie chwilą nieco wcześniejszą niż horyzont optymalny w zadaniu optymalnoczasowym lub czasem końcowym w zadaniu stałohoryzontowym. Dla ściśle rosnącego ciągu punktów $t_i \in]0,T[$, $i=1,...,N-1$, określamy M ciągów niezależnych zakłóceń stochastycznych $z^{i,l}$, $l=1,...,M$. Dla każdej pary i, l zmienna $z^{i,l}$ jest n-wymiarowym niezależnym wektorem losowym o rozkładzie normalnym. Niech x będzie trajektorią systemu (7.1), wygenerowaną przez sterowanie optymalne u (optymalnoczasowe lub stałohoryzontowe) dla stanu początkowego x^0 i stanu docelowego x^f. Dla tej trajektorii konstruujemy rodzinę trajektorii zakłóconych x^l

$$x^l(t) = x^{i,l}(t), \quad t \in [t_{i-1}, t_i[, \quad i=1,...,N \tag{13.1}$$

gdzie $t_0 = 0$, $t_N = T$, a $x^{i,l}$ spełnia równanie (7.1) ze sterowaniem $u^{i,l}$ i warunkiem początkowym

$$x^{1,l}(0) = x(0),$$

$$x^{i,l}(t_{i-1}) = x^{i-1,l}(t_{i-1}-) + \operatorname{diag}(s_1,...,s_n) z^{i-1,l}(t_{i-1} - t_{i-2}) \text{ dla } i=2,...,N.$$

Sterowanie $u^{i,l}$ jest obcięciem do przedziału $[t_{i-1}, t_i[$ optymalnego sterowania stałohoryzontowego dla problemu z warunkiem początkowym $x^{i,l}(t_{i-1})$, stanem docelowym x^f

i horyzontem $T - t_{i-1}$. Zauważmy, że zaburzenie trajektorii w chwili t_i reprezentuje skumulowany efekt działania zakłóceń o wariancji s_j^2 w przedziale $[t_{i-1}, t_i[$, na j-tą współrzędną stanu.

Niech ε będzie nieujemnym parametrem. Trajektorię zakłóconą x^l nazywamy *dopuszczalną na poziomie* ε, jeśli

$$\| x^l(T) - x(T) \|^2 \leq \varepsilon.$$

Opiszemy na początek postępowanie, gdy zakłócono tylko jeden punkt trajektorii $x(t_1)$. Dla chwili t_1 otrzymuje się zbiór zaburzonych wartości trajektorii $x^l(t_1)$, $l = 1,...,M$. Dla każdego punktu z tego zbioru stosuje się optymalizację stałohoryzontową z horyzontem $T - t_1$ i stanem docelowym x^f. Błąd trafienia w cel można określić, mierząc zaburzenie trajektorii w chwili T

$$\Delta x^l(T) = x^l(T) - x(T) \tag{13.2}$$

i porównać z założoną wartością progową $\varepsilon > 0$. W ten sposób zbiór zaburzeń trajektorii w chwili t_1

$$\Delta x^l(t_1) = x^l(t_1) - x(t_1), \quad l = 1,...,M \tag{13.3}$$

zostaje podzielony na dwa podzbiory: P_D – zaburzeń dopuszczalnych na poziomie ε i P_N – niedopuszczalnych na tym poziomie:

$$\begin{aligned} &\Delta x^l(t_1) \in P_D, \text{ jeżeli } \quad \left\| \Delta x^l(T) \right\|^2 \leq \varepsilon \\ &\Delta x^l(t_1) \in P_N, \text{ jeżeli } \quad \left\| \Delta x^l(T) \right\|^2 > \varepsilon \end{aligned} \tag{13.4}$$

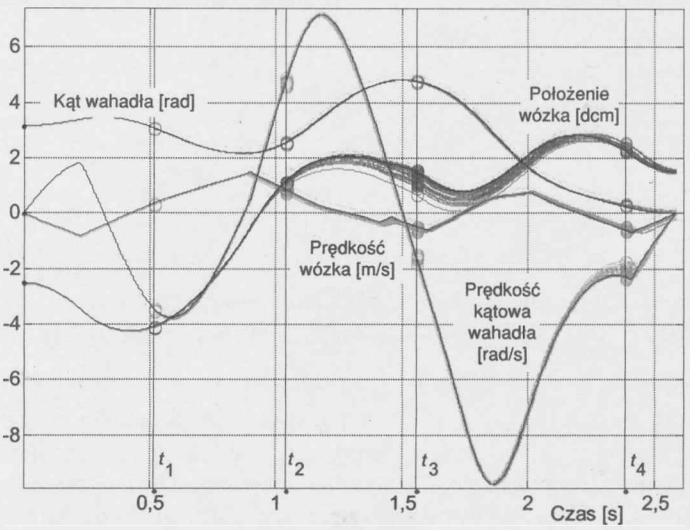
W przypadku zakłócania trajektorii optymalnej w większej liczbie punktów niż jeden, a więc w punktach $x(t_i)$, $i = 1,...,N-1$, postępujemy jak poprzednio, z dwiema modyfikacjami. Po pierwsze, do każdej trajektorii zakłóconej x^l w kolejnych chwilach t_i, dla $i \geq 2$, dodajemy po jednym zaburzeniu. W ten sposób liczba trajektorii zaburzonych jest stała w czasie i wynosi M. Po drugie, zaburzamy końcowy punkt trajektorii zakłóconej, wyliczony w poprzednim etapie, a nie punkt pierwotnej trajektorii optymalnej. Dzięki temu lepiej oddajemy działanie zakłóceń w systemie rzeczywistym. Dla zaburzonego punktu początkowego rozwiązujemy zadanie optymalizacji ze stałym horyzontem, równym $T - t_i$. Oczywiście błędy trafienia w cel odnosimy do celu pierwotnego. Otrzymujemy zbiory za-

burzeń dopuszczalnych P_{D_i} i niedopuszczalnych P_{N_i} dla chwil czasu t_i, $i = 1,...,N-1$:

$$\Delta x^l(t_i) \in P_{D_i}, \text{ jeżeli } \left\|\Delta x^l(T)\right\|^2 \le \varepsilon, \quad l = 1,...,M$$
$$\Delta x^l(t_i) \in P_{N_i}, \text{ jeżeli } \left\|\Delta x^l(T)\right\|^2 > \varepsilon, \quad l = 1,...,M \tag{13.5}$$

Zakłócanie trajektorii optymalnej wahadła na wózku – zadanie POST+AKRO

Na rysunku 13.1 przedstawiono rezultat zaburzania trajektorii optymalnoczasowej. W zadaniu O2 POST+AKRO, opisanym w podrozdziale 10.1, wybrano cztery chwile czasowe, w których zakłócano bieg trajektorii optymalnej. Są to czasy: t_1 = 0,5186 s, t_2 = 1,0370 s, t_3 = 1,5558 s i t_4 = 2,3856 s. Czas optymalny wynosi t^* = 2,5875 s. Przyjęto tu T = 2,5866 s. W chwili t_1 zaburzano trajektorię według formuły (13.1), w której przyjęto dane liczbowe: M = 50 – liczba zaburzeń, s_1 = 0,0025 dcm, s_2 = 0,0025 m/s, s_3 = 0,001 rad, s_4 = 0,01 rad/s – odchylenia standardowe (stałe, przez które mnożone są współrzędne kierunku zaburzenia $z^{i,l}$), $z^{i,l}$ – pseudolosowy wektor czterowymiarowy. Trajektorie zaburzone w chwilach t_i, $i = 1,...,4$ schodzą się do pewnego otoczenia stanu końcowego, gdyż po wystąpieniu zakłócenia stosujemy optymalizację stałohoryzontową. Na rysunku 13.1 widoczny jest pęk tych trajektorii, o szerokim rozrzucie w części środkowej dla zmiennej stanu x_1 – położenia wózka, i w części końcowej dla zmiennej stanu x_4 – prędkości kątowej wahadła.



Rys. 13.1. Zaburzanie trajektorii optymalnej $x^*_{x^0,x^f}$ w chwilach t_i, $i = 1,...,4$

13.2. Uodparnianie trajektorii optymalnej metodą przesuwania

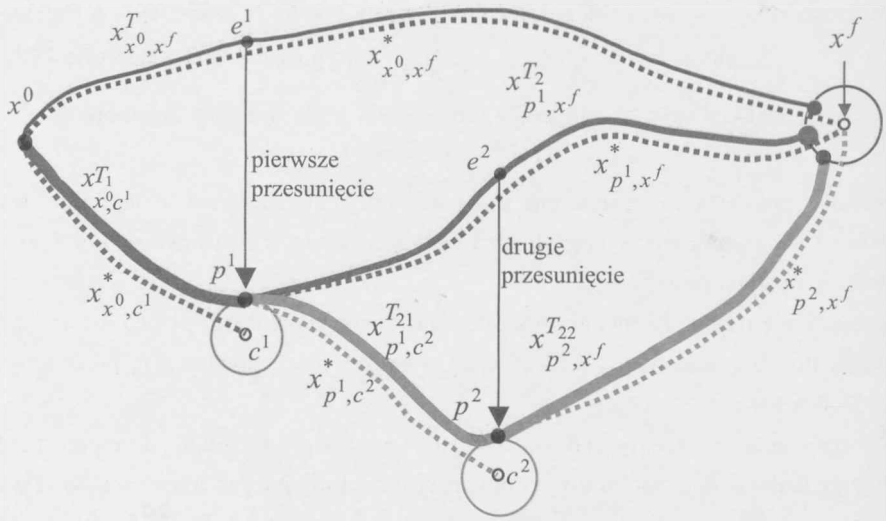
W pierwszym kroku oryginalną trajektorię optymalnoczasową $x^*_{x^0,x^f}$ przekształca się w trajektorię stałohoryzontową $x^T_{x^0,x^f}$, rozwiązując problem optymalizacyjny z ustalonym czasem końcowym (7.5), o horyzoncie T nieznacznie krótszym od optymalnego. Następnie wybiera się cel pośredni c^1, nie leżący na trajektorii $x^T_{x^0,x^f}$ (otrzymanej w kroku pierwszym). W następnym kroku rozwiązuje się problem optymalnoczasowy (7.7) z punktu x^0 do punktu c^1. Z kolei, po nieznacznym skróceniu otrzymanego horyzontu optymalnego do wartości T_1, rozwiązuje się zadanie stałohoryzontowe (7.5) z celem pośrednim c^1 jako punktem docelowym. Osiągnięty stałohoryzontowy stan końcowy nazwiemy pierwszym punktem pośrednim p^1. Uzyskane trajektorie oznaczymy przez: $x^*_{x^0,c^1}$ (optymalnoczasowa) i $x^{T_1}_{x^0,c^1}$ (stałohoryzontowa), zachodzi więc $x^{T_1}_{x^0,c^1}(T_1) = p^1$. W kolejnym kroku rozwiązujemy problem sterowania optymalnoczasowego z punktu p^1 do punktu x^f – i znowu, po niewielkim skróceniu otrzymanego horyzontu do wartości T_2, rozwiązujemy zadanie stałohoryzontowe z x^f jako punktem docelowym. Otrzymujemy w ten sposób trajektorie $x^*_{p^1,x^f}$ i $x^{T_2}_{p^1,x^f}$. Sklejenie trajektorii $x^{T_1}_{x^0,c^1}$ i $x^{T_2}_{p^1,x^f}$ daje poszukiwaną trajektorię uodporniona $x^{T_1+T_2}_{x^0,p^1,x^f}$. Powstałą sytuację ilustrują górna i dwie środkowe pary trajektorii na rysunku 13.2.

Liniami kropkowanymi zaznaczono trajektorie optymalnoczasowe, a ciągłymi – stałohoryzontowe. Otoczenia celów pośrednich i celu końcowego przedstawiono kółkami, dla czytelności rysunku w powiększeniu.

Uodparnianie trajektorii metodą przesuwania z jednym punktem pośrednim może nie wystarczać. Proces uodparniania prowadzi się wówczas dalej, powtarzając procedurę dla kolejnych punktów pośrednich.

Schematycznie pokażemy, jak to wygląda przy wprowadzeniu drugiego punktu pośredniego. Kolejne operacje ilustrują dolne pary trajektorii na rysunku 13.2. Wyznacza się cel pośredni c^2, nie leżący na przesuniętej trajektorii stałohoryzontowej $x^{T_1+T_2}_{x^0,p^1,x^f}$. Dla ustalenia uwagi wybierzmy go na trajektorii $x^{T_2}_{p^1,x^f}$. Następnie rozwiązuje się problem optymalnoczasowy (7.7) z punktu p^1 do punktu c^2 – otrzymuje się trajektorię $x^*_{p^1,c^2}$ i po nieznacznym skróceniu horyzontu do wartości T_{21}, rozwiązuje się zadanie stałohoryzontowe (7.5) z nowym celem pośrednim c^2 jako punktem docelowym. Osiągnięty stan końco-

wy nazwiemy drugim punktem pośrednim p^2. Uzyskaną trajektorię oznaczymy odpowiednio przez $x^{T_{21}}_{p^1,c^2}$, a zatem $x^{T_{21}}_{p^1,c^2}(T_{21}) = p^2$. Jako następny rozwiązuje się problem sterowania optymalnoczasowego z punktu p^2 do punktu x^f – i ponownie, po niewielkim skróceniu horyzontu do wartości T_{22}, rozwiązuje się zadanie stałohoryzontowe z x^f jako punktem docelowym. Otrzymuje się trajektorie $x^*_{p^2,x^f}$ i $x^{T_{22}}_{p^2,x^f}$. Po sklejeniu $x^{T_1}_{x^0,c^1}$, $x^{T_{21}}_{p^1,c^2}$ i $x^{T_{22}}_{p^2,x^f}$ uzyskuje się trajektorię uodpornioną $x^{T_1+T_{21}+T_{22}}_{x^0,p^1,p^2,x^f}$ (podwójnie przesuniętą), złożoną z trzech części. Pierwsza jej część prowadzi z punktu początkowego x^0 do punktu p^1 (otoczenia pierwszego celu pośredniego c^1) – trajektorii tej nie wyznacza się, ponieważ jest ona identyczna z już wyliczoną pierwszą składową trajektorii $x^{T_1+T_2}_{x^0,p^1,x^f}$. Druga składowa (podwójnie przesunięta) prowadzi z p^1 do p^2 (otoczenia drugiego celu pośredniego c^2). Trzecia trajektoria składowa (podwójnie przesunięta) prowadzi z p^2 do punktu końcowego $x^{T_{22}}_{p^2,x^f}(T_{22})$, leżącego w otoczeniu celu x^f.



Rys. 13.2. Schemat uodparniania trajektorii przez podwójne przesunięcie

Do analizy wrażliwości lokalnej nie można posłużyć się trajektorią optymalnoczasową $x^*_{x^0,x^f}$, gdyż z góry wiadomo, że wrażliwość (kierunkowa) horyzontu optymalnego jest na ogół nieskończona. Podobnie nie można składać trajektorii uodpornionej z optymalnoczasowych odcinków przejść między wybranymi punktami.

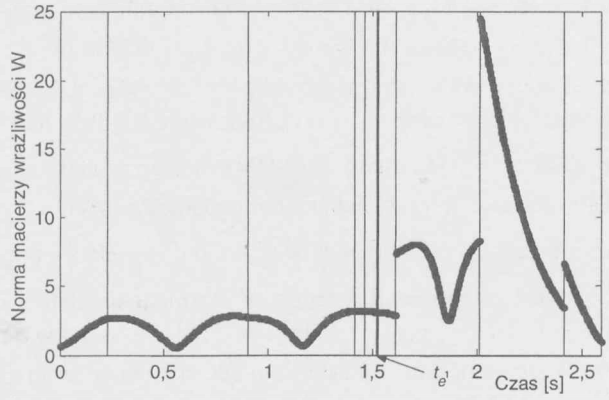
Uodparnianie jest więc procesem generowania odcinków trajektorii stałohoryzontowych i sklejania ich w jedną trajektorię. Szczegółowy opis eksperymentu uodparniania trajektorii optymalnej metodą przesuwania podamy dla przypadku jednego punktu pośredniego p^1. Zgodnie z założeniami z rozdziału 3, obliczenia numeryczne wykonuje się na zmiennych skalowanych. Dla jednolitości dane numeryczne rozwiązań przedstawia się poniżej w postaci nieskalowanej.

Powróćmy do zadania O2 POST+AKRO, rozpatrywanego w podrozdziale 13.1. Dla tego zadania w podrozdziale 10.1 otrzymaliśmy czas optymalny $t^* = 2{,}5875$ s, przy stanie początkowym $x^0 = \text{col}(-2{,}5 \text{ dcm}, 3{,}1416, 0{,}0)$ i stanie docelowym $x^f = \text{col}(1{,}5 \text{ dcm}, 0, 0, 0)$. Skracając nieznacznie horyzont do wartości $T = 2{,}5866$ s rozwiązuje się problem pomocniczy P_T, stosując optymalizację stałohoryzontową (algorytm 8.3). Równania sprzężone rozwiązuje się wstecz z użyciem ruchomej siatki dyskretyzacji w okolicy przełączeń sterowania, tak by zgrać punkty siatki z chwilami przełączeń sterowania. W dalszej optymalizacji za zmienną decyzyjną przyjęto wartość początkową funkcji sprzężonej, a przełączenia sterowania umieszczano w zerach funkcji przełączającej, wyliczanych przy użyciu wzoru (11.14). Ostatecznie otrzymuje się stan końcowy

$$x^T(T) = \text{col}(1{,}4999 \text{ dcm}, 0{,}0007 \text{ rad}, 0{,}0003 \text{ m/s}, -0{,}0032 \text{ rad/s})$$

i wartość wskaźnika jakości $\Sigma_T^* = 5{,}47 \cdot 10^{-6}$ (do wzoru na wskaźnik jakości podstawiamy odpowiednie wielkości w jednostkach jak wyżej). Stan końcowy różni się od x^f, gdyż celu w czasie krótszym od t^* osiągnąć nie można, a jedynie pewne jego otoczenie.

Przeanalizujmy wrażliwość uzyskanej trajektorii $x^T_{x^0,x^f}$. Z równości (11.46) wylicza się macierz wrażliwości $W_{x^0,x^f}(t)$ (dla przejścia z x^0 do otoczenia celu końcowego x^f) i normę (11.49) tej macierzy (rys. 13.3).



Rys. 13.3. Norma $W_{x^0,x^f}(t)$ trajektorii stałohoryzontowej $x^T_{x^0,x^f}$ w zadaniu O2 POST+AKRO

Pionowymi liniami zaznaczono przełączenia sterowania. Norma macierzy wrażliwości jest w chwili końcowej sterowania T równa jeden. Stan końcowy trajektorii $x^T_{x^0,x^1}$ (służącej tu do przybliżonego określenia wrażliwości trajektorii $x^*_{x^0,x^1}$), jest najbardziej podatny na zakłócenia występujące po trzecim przełączeniu od końca. Norma macierzy wrażliwości osiąga maksimum (około 25) w chwili szóstego przełączenia. Na podstawie wzoru (11.47) można wnioskować, że małe zakłócenie trajektorii wprowadzone w chwili szóstego przełączenia, przy niekorzystnym kierunku działania zakłócenia, może zostać wzmocnione dwudziestopięciokrotnie w końcowej chwili sterowania. Tak duża wrażliwość nie jest tu zaskoczeniem, dotyczy bowiem trajektorii stałohoryzontowej, bliskiej optymalnoczasowej, w problemie sterowania do niestabilnego punktu równowagi.

Wybór punktu trajektorii $x^T_{x^0,x^1}$, który zostanie przesunięty, przeprowadza się na podstawie analizy zmian normy macierzy wrażliwości $W_{x^0,x^1}(t)$ w czasie. Wybiera się punkt trajektorii $x^T_{x^0,x^1}$ dla czasu $t_{e^1} = 1{,}5188$ s, przed piątym przełączeniem (pionowa linia pogrubiona na rysunku 13.3). Punkt ten ma współrzędne

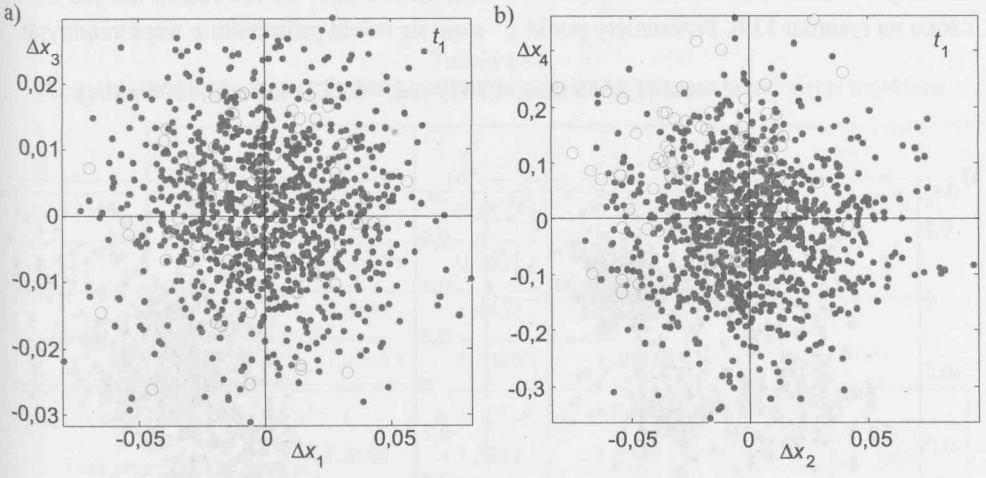
$$e^1 = \text{col}(1{,}6434 \text{ dcm},\ 4{,}8125 \text{ rad},\ -0{,}3425 \text{ m/s},\ -0{,}4527 \text{ rad/s}).$$

Punkt e^1 po przesunięciu da punkt c^1. Analiza zmian normy macierzy wrażliwości w czasie wskazuje, że działanie, mające zmniejszyć wrażliwość trajektorii, powinno wyprzedzać chwilę piątego przełączenia sterowania, w której wrażliwość skokowo wzrasta. Czy wybór chwili t_{e^1} nie jest spóźniony? By odpowiedzieć na to pytanie trzeba przeprowadzić eksperyment stochastycznego zaburzania trajektorii optymalnej i zbadać wzajemne położenie zbiorów zaburzeń dopuszczalnych i niedopuszczalnych w wybranych chwilach poprzedzających t_{e^1}.

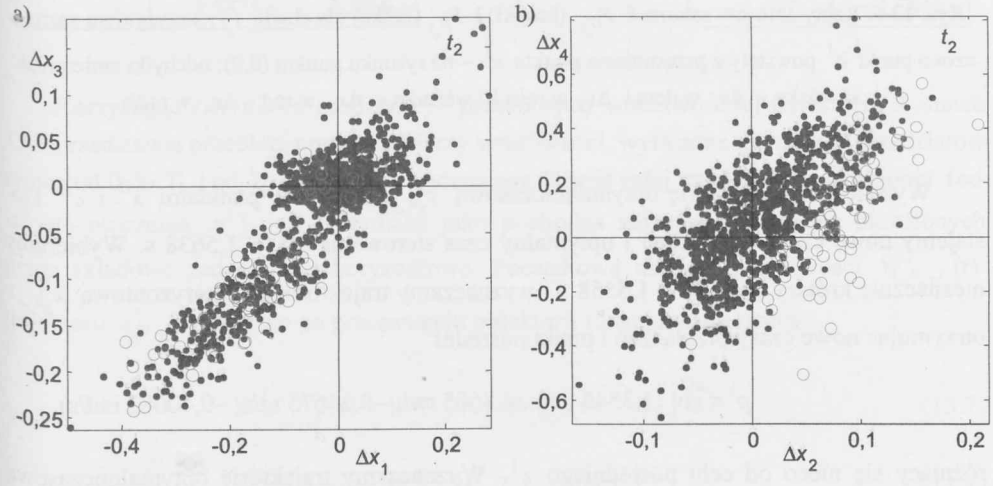
Na rysunkach 13.4, 13.5, 13.6 pokazano rzuty zbiorów zaburzeń dopuszczalnych P_{D_i} i niedopuszczalnych P_{N_i} na płaszczyzny fazowe wózka i wahadła. Zbiory zostały utworzone w analogicznym eksperymencie jak opisany pod koniec podrozdziału 13.1, z tą różnicą, że chwila t_3 została zastąpiona przez t_{e^1}. Przyjęto nastepujące dane liczbowe we wzorach (13.1)–(13.5): $M = 2000$, $s_1 = 0{,}005$ dcm, $s_2 = 0{,}005$ rad, $s_3 = 0{,}002$ m/s, $s_4 = 0{,}02$ rad/s oraz $\varepsilon = 0{,}5$. Kółkami oznaczono rzuty zaburzeń niedopuszczalnych P_{N_i}, a kropkami rzuty zaburzeń dopuszczalnych P_{D_i}. Zaburzenia odniesiono do trajektorii optymalnej.

Jak widać z rysunku 13.4, w chwili t_1 trudno wyróżnić jakiś kierunek, w którym należałoby przesunąć punkt trajektorii stanu, tak by statystycznie zwiększyć liczbę punktów dopuszczalnych P_{D_1} w jego otoczeniu i odsunąć go jak najdalej od zbioru punktów niedopuszczalnych P_{N_1}. W chwili t_2 (rys. 13.5) kierunek jest wyraźnie widoczny tylko w rzucie na płaszczyznę $\Delta x_2 \Delta x_4$. Można go określić jako kierunek w lewo. Początek układu współ-

rzędnych na płaszczyźnie $\Delta x_1 \Delta x_3$ można odsunąć niewiele od punktów niedopuszczalnych. Tak więc chwila t_2 jest za wczesna na przesuwanie trajektorii.



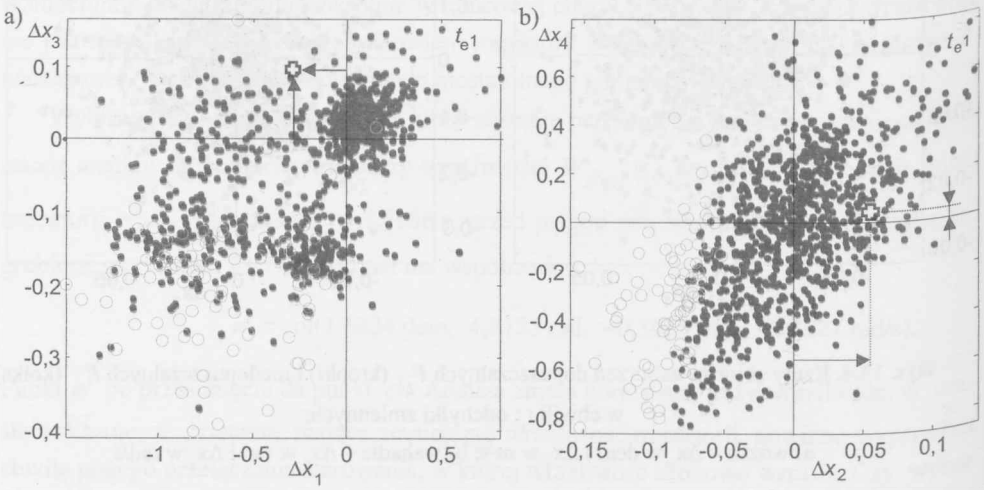
Rys. 13.4. Rzuty zbiorów zaburzeń dopuszczalnych P_{D_1} (kropki) i niedopuszczalnych P_{N_1} (kółka) w chwili t_1; odchyłki zmiennych:
a) wózka – Δx_1 w dcm i Δx_3 w m/s; b) wahadła – Δx_2 w rad i Δx_4 w rad/s



Rys. 13.5. Rzuty zbiorów zaburzeń P_{D_2} (kropki) i P_{N_2} (kółka) w chwili t_2; odchyłki zmiennych:
a) wózka – Δx_1 w dcm i Δx_3 w m/s; b) wahadła – Δx_2 w rad i Δx_4 w rad/s

Inaczej sytuacja wygląda w chwili t_3. Obrazuje to rysunek 13.6. Rzuty punktów niedopuszczalnych są zgrupowane w części dolnej płaszczyzny $\Delta x_1 \Delta x_3$ i lewej części płaszczyzny $\Delta x_2 \Delta x_4$. Wybór chwili czasu $t_{e^1} = t_3 = 1{,}5188$ s jest więc właściwy. Punkt e^1 przesuwamy o wektor col(−0,2875 dcm, 0,0555 rad, 0,0953 m/s, 0,0418 rad/s), tak jak zaznaczono na rysunku 13.6. Przesunięty punkt c^1 staje się celem pośrednim o współrzędnych

$$c^1 = \text{col}\,(1{,}3558 \text{ dcm}, 4{,}8679 \text{ rad}, -0{,}2471 \text{ m/s}, -0{,}4108 \text{ rad/s}).$$



Rys. 13.6. Rzuty zbiorów zaburzeń $P_{D_{e^1}}$ (kropki) i $P_{N_{e^1}}$ (kółka) dla chwili t_{e^1}; krzyżykiem zaznaczono punkt c^1 powstały z przesunięcia punktu e^1 – na rysunku punktu (0,0); odchyłki zmiennych: a) wózka – Δx_1 w dcm i Δx_3 w m/s; b) wahadła – Δx_2 w rad i Δx_4 w rad/s

Wyznaczamy trajektorię optymalnoczasową $x^*_{x^0,c^1}$ pomiędzy punktami x^0 i c^1. Dostajemy nowe czasy przełączeń i optymalny czas sterowania $t_1^* = 1{,}5638$ s. Wybieramy nieznacznie krótszy czas $T_1 = 1{,}5558$ s i wyznaczamy trajektorię stałohoryzontową $x^{T_1}_{x^0,c^1}$, otrzymując nowe czasy przełączeń i punkt pośredni

$$p^1 = \text{col}\,(1{,}3545 \text{ dcm}, 4{,}8605 \text{ rad}, -0{,}24675 \text{ m/s}, -0{,}40082 \text{ rad/s}),$$

różniący się nieco od celu pośredniego c^1. Wyznaczamy trajektorię optymalnoczasową $x^*_{p^1,x^f}$ pomiędzy punktami p^1 i x^f. Dostajemy nowe czasy przełączeń i czas optymalny sterowania $t_2^* = 1{,}0493$ s. Skrócony nieznacznie czas wynosi $T_2 = 1{,}0308$ s. Wyznaczamy

trajektorię stałohoryzontową $x^{T_2}_{p^1,x^f}$. Czasy przełączeń i horyzonty sterowania dla kolejno otrzymywanych trajektorii zebrano porównawczo w tabeli 13.1. Zaznaczono pogrubioną czcionką czasy przełączeń i czas sterowania trajektorii uodpornionej.

Tabela 13.1

Czasy przełączeń i horyzonty dla otrzymanych trajektorii – pojedyncze przesunięcie trajektorii

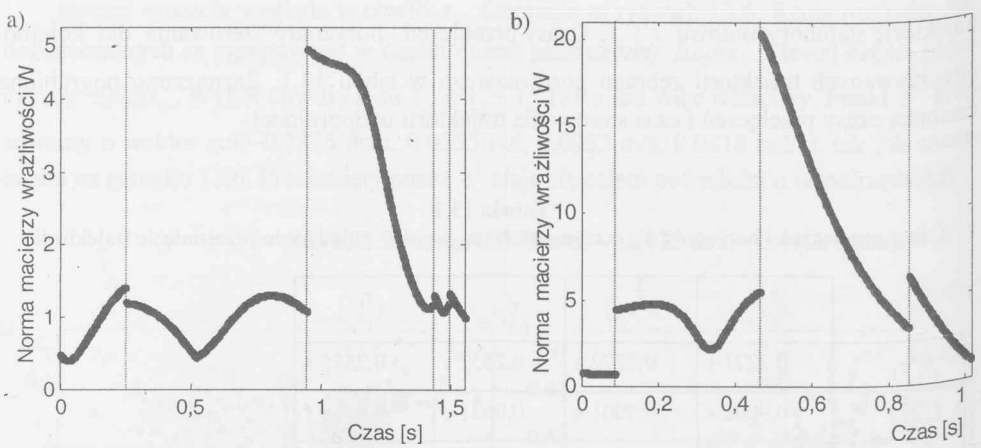
	$x^*_{x^0,x^f}$	$x^T_{x^0,x^f}$	$x^*_{x^0,c^1}$	$x^{T_1}_{x^0,c^1}$		
τ_1	0,2271 s	0,2272 s	0,2557 s	**0,2525** s		
τ_2	0,9002 s	0,9001 s	0,9631 s	**0,9469** s		
τ_3	1,4103 s	1,4103 s	1,4458 s	**1,4360** s		
τ_4	1,4595 s	1,4595 s	1,4969 s	**1,4954** s		
punkty pośrednie e^1, p^1 i cel pośredni c^1		e^1; t_{e^1} = = 1,5188 s	p^1; t_1* = = 1,5649 s	c^1; T_1 = = 1,5558 s	$x^*_{p^1,x^f}$	$x^{T_2}_{p^1,x^f}$
τ_5	1,6041 s	1,6041 s			1,6406 s	**1,6415** s
τ_6	2,0098 s	2,0099 s			2,0138 s	**2,0192** s
τ_7	2,4098 s	2,4093 s			2,4179 s	**2,4103** s
horyzonty	t^* = 2,5875 s	T = 2,5866 s			$T_1 + t_2^*$ = = 2,6051 s	$T_1 + T_2$ = = **2,5866** s

Korzystając z wyników rozdziału 11 przebadajmy wrażliwość tej trajektorii. Rysunek 13.7 przedstawia przebiegi normy macierzy wrażliwości, wyliczone dla dwóch przedziałów czasu: od 0 do T_1 i od T_1 do $T_1 + T_2$. Łączna wrażliwość całej trajektorii uodpornionej (od x^0 do otoczenia x^f) jest rozumiana jako pochodna złożenia operatorów określonych przez składowe zadania stałohoryzontowe. Początkową macierz wrażliwości $W^T_{x^0,x^f}(t)$ trajektorii $x^T_{x^0,x^f}$ zastępuje po przesunięciu trajektorii funkcja macierzowa:

$$W^{T_1+T_2}_{x^0,p^1,x^f}(t) = W^{T_2}_{p^1,x^f}(0) \cdot W^{T_1}_{x^0,c^1}(t), \quad \text{dla } t < T_1 \tag{13.7}$$

$$W^{T_1+T_2}_{x^0,p^1,x^f}(t) = W^{T_2}_{p^1,x^f}(t - T_1), \quad \text{dla } t \geq T_1 \tag{13.8}$$

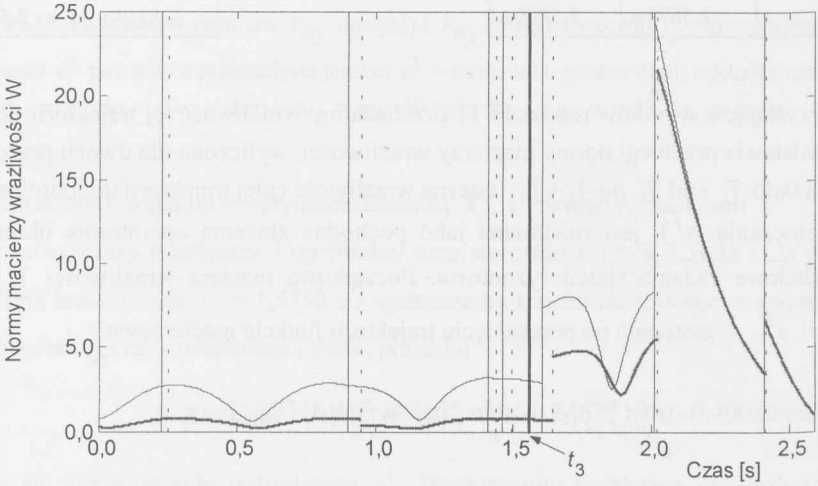
która jest macierzą wrażliwości dla trajektorii uodpornionej $x^{T_1+T_2}_{x^0,p^1,x^f}$.



Rys. 13.7. Normy wrażliwości trajektorii stałohoryzontowych: a) $x^{T_1}_{x^0,c^1}$; b) $x^{T_2}_{p^1,x^f}$

Na rysunku 13.8 pokazano normy wrażliwości: trajektorii stałohoryzontowej od 0 do T i trajektorii stałohoryzontowej przesuniętej od 0 do T_1+T_2.

Istotny wpływ na obniżenie wrażliwości tej trajektorii w przedziale od 0 do T_1 ma wartość $W^{T_2}_{p^1,x^f}(0)$ – normę tej macierzy nazwiemy *wrażliwością dolną* W_d (rys. 13.7b).



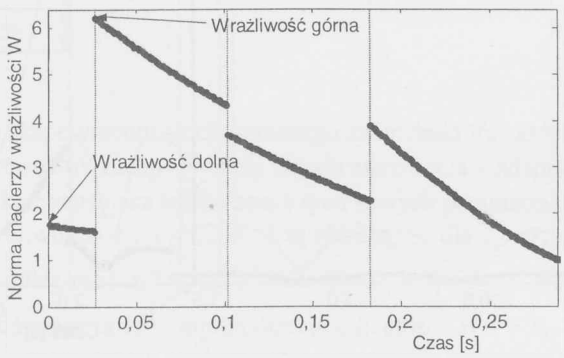
Rys. 13.8. Normy wrażliwości: $W^{T}_{x^0,x^f}$ – trajektorii stałohoryzontowej od 0 do T, i $W^{T_1+T_2}_{x^0,p^1,x^f}(t)$ – trajektorii stałohoryzontowej przesuniętej (linia pogrubiona) od 0 do T_1+T_2

Norma $\| W^{T_1+T_2}_{x^0,p^1,x^f} \|$ z rysunku 13.8 w przedziale od 0 do T_1 zostanie zmniejszona co najmniej $\| W^{T_2}_{p^1,x^f}(0) \|^{-1}$ razy, jeżeli jest spełniona nierówność

$$\| W^{T_2}_{p^1,x^f}(0) \| < 1 \tag{13.9}$$

Zaznaczono siedem czasów przełączeń sterowania dla trajektorii: $x^{T}_{x^0,x^f}$ – linie ciągłe, i $x^{T_1+T_2}_{x^0,p^1,x^f}$ – linie przerywane. Trajektoria $x^{T_1+T_2}_{x^0,p^1,x^f}$ uległa wyraźnemu odwrażliwieniu w przedziale od początku sterowania do szóstego przełączenia. Wykresy czasowe norm wrażliwości są podobne, poczynając od chwil szóstych przełączeń. Oznacza to, że trzeba poszukiwać następnego punktu pośredniego, tym razem leżącego na trajektorii $x^{T_1+T_2}_{x^0,p^1,x^f}$, i kontynuować procedurę uodparniania.

Wykonajmy zatem drugie przesunięcie trajektorii. Punkt e^2 na trajektorii $x^{T_1+T_2}_{x^0,p^1,x^f}$ wybiera się dla chwili $t_{e^2} = 2{,}3856$ s poprzedzającej ostatnie, siódme przełączenie $\tau_7 = 2{,}4103$ s. Ustalenie kierunku przesuwania punktu będzie wynikiem analizy symulacyjnej. Wybiera się r punktów próbnych p^2_i, $i = 1,...,r$. Testuje się wrażliwości trajektorii stałohoryzontowych, łączących kolejno punkty x^0, p^1, p^2_i, $i = 1,...,r$, z punktem z otoczenia x^f, wyliczając odpowiednie macierze wrażliwości. Na rysunku 13.9 pokazano wykres normy macierzy wrażliwości, charakteryzującej ostatecznie wybrany punkt p^2. Parametrami tego wykresu są *wrażliwość górna* i *wrażliwość dolna*, zaznaczone strzałkami. Pierwsza określa maksymalną wrażliwość trajektorii w badanym przedziale, druga pozwala oszacować wpływ zaburzeń z wcześniejszych odcinków trajektorii (zob. wzory (13.7) i (13.8)).



Rys. 13.9. Norma wrażliwości końcowego fragmentu trajektorii podwójnie przesuniętej $x^{T_{22}}_{p^2,x^f}$

Wybrano punkt pośredni p^2, dla którego przebieg normy wrażliwości przedstawia wykres zaznaczony pogrubioną linią na rysunku 13.10. Dla zobrazowania wielkości drugiego przesunięcia podano poniżej współrzędne liczbowe punktów: e^2 – przed przesunięciem, p^2 – po przesunięciu i współrzędne punktu celu c^2.

Ostateczny wynik zmniejszania wrażliwości trajektorii przez podwójne przesunięcie pokazano na rysunku 13.10. Przesunięcie jest w chwili t_4, a współrzędne punktów są:

$$e^2 = \text{col}(2{,}2275 \text{ dcm}, 0{,}2957 \text{ rad}, -0{,}5917 \text{ m/s}, -2{,}5184 \text{ rad/s}),$$

$$c^2 = \text{col}(2{,}2238 \text{ dcm}, 0{,}2659 \text{ rad}, -0{,}3406 \text{ m/s}, -1{,}7818 \text{ rad/s}),$$

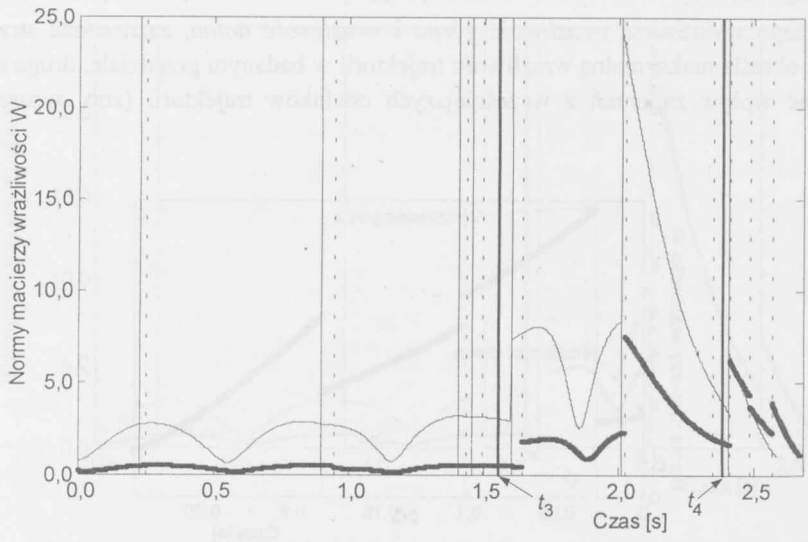
$$p^2 = \text{col}(2{,}2128 \text{ dcm}, 0{,}2901 \text{ rad}, -0{,}3180 \text{ m/s}, -1{,}8742 \text{ rad/s}).$$

Podobnie jak dla pojedynczego przesunięcia, zob. (13.7) i (13.8), wyrazimy macierz wrażliwości $W^{T_1+T_{21}+T_{22}}_{x^0,p^1,p^2,x^f}$ trajektorii przez macierze wrażliwości trajektorii składowych:

$$W^{T_1+T_{21}+T_{22}}_{x^0,p^1,p^2,x^f}(t) = W^{T_{22}}_{p^2,x^f}(0) \cdot W^{T_{21}}_{p^1,c^2}(0) \cdot W^{T_1}_{x^0,c^1}(t), \quad \text{dla } t < T_1 \tag{13.10}$$

$$W^{T_1+T_{21}+T_{22}}_{x^0,p^1,p^2,x^f}(t) = W^{T_{22}}_{p^2,x^f}(0)\, W^{T_{21}}_{p^1,c^2}(t-T_1), \quad \text{dla } T_1 + T_{21} > t \geq T_1 \tag{13.11}$$

$$W^{T_1+T_{21}+T_{22}}_{x^0,p^1,p^2,x^f}(t) = W^{T_{22}}_{p^2,x^f}(t-T_1-T_{21}), \quad \text{dla } t \geq T_1 + T_{21} \tag{13.12}$$



Rys. 13.10. Normy wrażliwości $W^{T}_{x^0,x^f}(t)$ – linia cienka, i $W^{T_1+T_{21}+T_{22}}_{x^0,p^1,p^2,x^f}$ – linia pogrubiona

Pogrubione linie oznaczają punkty przesunięcia trajektorii. Dla porównania przedstawiono normy macierzy wrażliwości przed przesunięciem i po przesunięciu. Czasy przełączeń sterowania przed odwrażliwieniem zaznaczono liniami ciągłymi. Czasy przełączeń po odwrażliwieniu zaznaczono liniami przerywanymi (zauważmy, że linie odpowiadające czasom przełączeń τ_8 i τ_9 pokrywają się z liniami dla chwil t_4 i τ_7 przed odwrażliwieniem). Czas sterowania $T_1 + T_{21} + T_{22} = 1{,}5558 + 0{,}8298 + 0{,}2880 = 2{,}6736$ s dla trajektorii podwójnie uodpornionej jest dłuższy od czasu sterowania $T_1 + T_2 = 2{,}5866$ s dla trajektorii stałohoryzontowej. Czas sterowania uodpornionego $T_1 + T_{21} + T_{22} = 2{,}6736$ s jest dłuższy od optymalnego $t^* = 2{,}5875$ s o 3,3%. Wrażliwość górna maleje około trzykrotnie.

Analiza wrażliwości trajektorii metodami różniczkowymi ma zawsze charakter lokalny, dlatego uzyskaną obniżoną wrażliwość należy zweryfikować badając trajektorie leżące w większym otoczeniu trajektorii odwrażliwionej. Czasy przełączeń sterowania i horyzont dla trajektorii uodpornionej – podwójnie przesuniętej – zebrano w tabeli 13.2. Należy zwrócić uwagę, że w punkcie p^2 osiąganym w chwili $T_1 + T_{21}$ ma miejsce ósme przełączenie sterowania (w punkcie p^1 osiąganym w chwili T_1 nie występuje przełączenie).

Tabela 13.2
Czasy przełączeń i horyzont dla trajektorii podwójnie przesuniętej (uodpornionej)

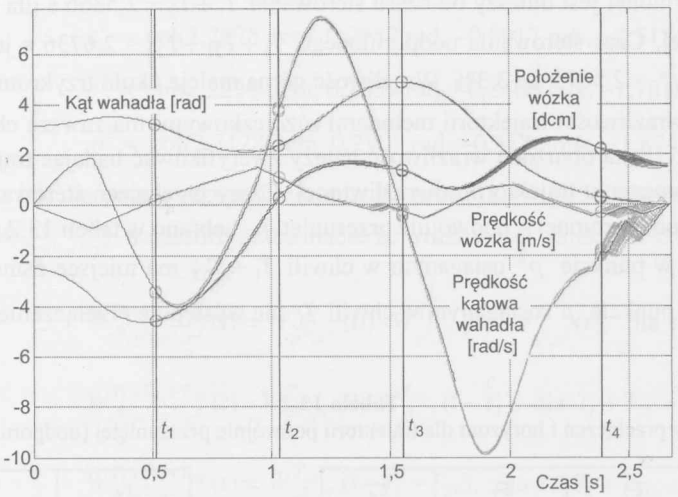
τ_1	τ_2	τ_3	τ_4		τ_5	τ_6
0,2525 s	0,9469 s	1,4360 s	1,4954 s		1,6453 s	2,0242 s
				T_1		
				1,5558 s		
τ_7	τ_8	τ_9	τ_{10}		τ_{11}	horyzont
2,3499 s	2,3856 s	2,4124 s	2,4872 s		2,5681 s	2,6736 s
	$T_1 + T_{21}$					$T_1 + T_{21} + T_{22}$
	2,3856 s					2,6736 s

Powróćmy do eksperymentu stochastycznego zaburzania trajektorii, opisanego w podrozdziale 13.1. Porównamy zachowanie się układu sterowania z adaptacyjną optymalizacją stałohoryzontową, kompensującą zaburzenia o tych samych parametrach dodawane do stanu w tych samych chwilach t_i, $i = 1,...,4$, z tą różnicą, że dla $t_i < T_1$ celem optymalizacji jest minimalizacja odległości $x(T_1)$ od c^1, dla $T_1 \le t_i < T_1 + T_{21}$ – odległości $x(T_1 + T_{21})$ od c^2, a tylko w końcowej fazie – minimalizacja odległości $x(T_1 + T_{21} + T_{22})$ od x^f.

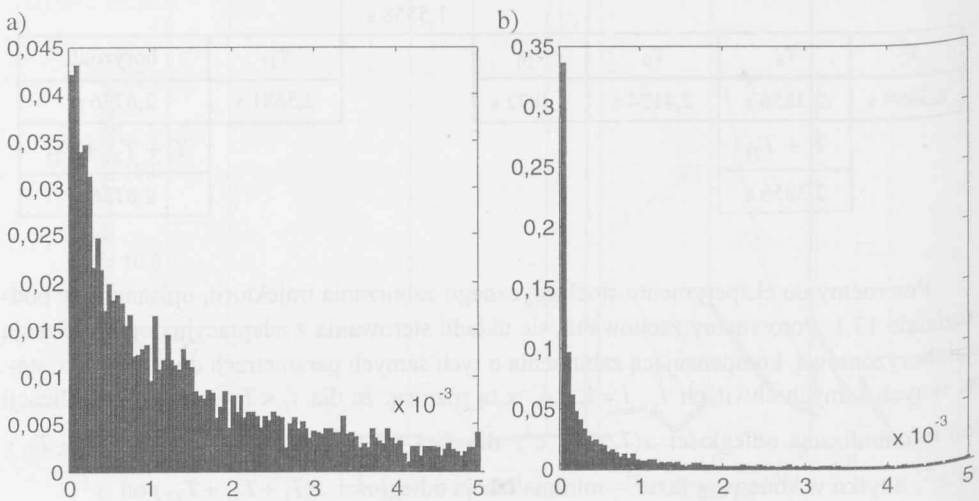
W eksperymencie zaburzania trajektorii przyjęto nastepujące dane liczbowe:
$M = 5000$, $s_1 = 0{,}0025$ dcm, $s_2 = 0{,}0025$ rad, $s_3 = 0{,}001$ m/s, $s_4 = 0{,}01$ rad/s i $\varepsilon = 0{,}015$.

Przykładowy pęk 50 otrzymanych trajektorii przedstawia rysunek 13.11, zmienne stanu podano w tych samych jednostkach co na rysunku 13.1.

Rysunki 13.12a i b przedstawiają histogramy wartości wskaźników jakości Σ_T i $\Sigma_{odp} = \frac{1}{2} \| x^{T_1+T_{21}+T_{22}}_{x^0,p^1,p^2,x^f}(T_1+T_{21}+T_{22}) - x^f \|^2$, z jednostkową macierzą wagową, dla rodzin trajektorii liczących po 5000 elementów.



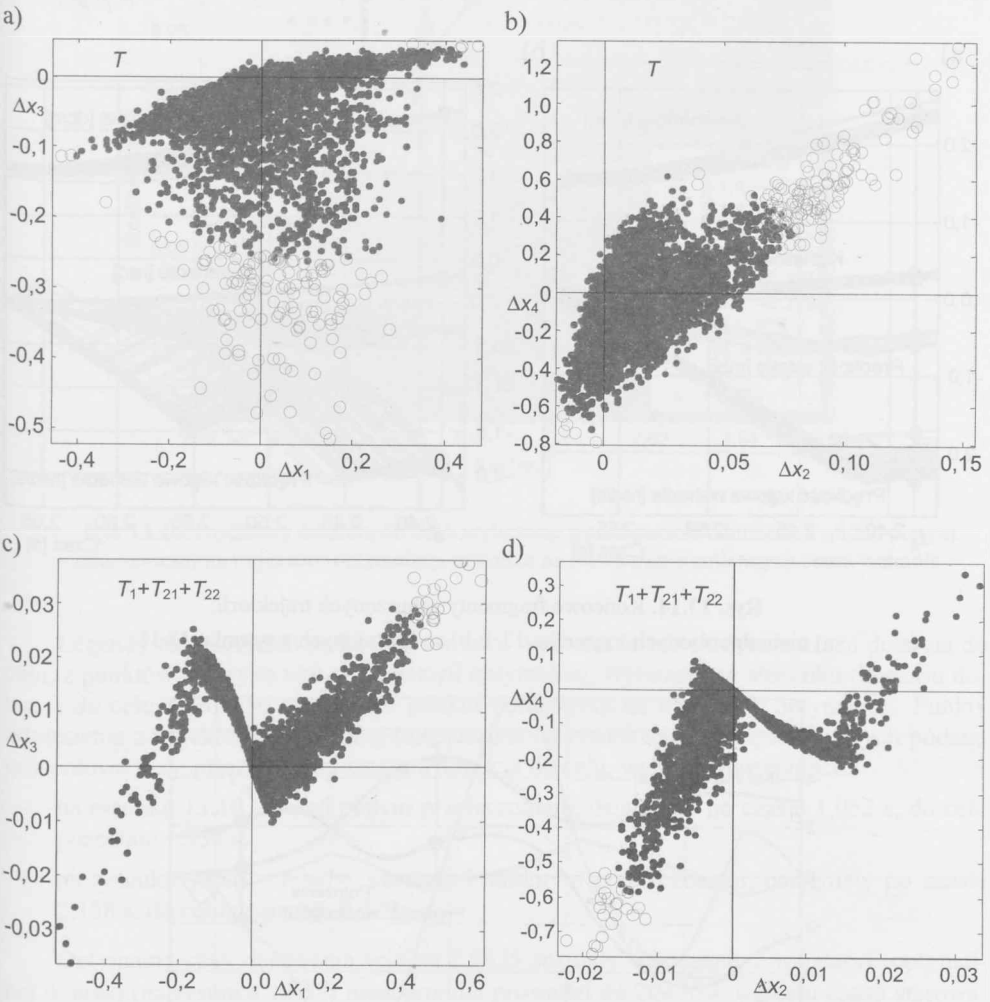
Rys. 13.11. Zaburzanie trajektorii $x^{T_1+T_{21}+T_{22}}_{x^0,p^1,p^2,x^f}$; w chwilach czasu t_i, $i = 1,...,4$



Rys. 13.12. Histogramy wskaźników jakości dla sterowania:
a) nominalnego (optymalnoczasowego bez zakłóceń); b) uodpornionego

Wartości średnie tych wskaźników wyniosły: $E(\Sigma_T) = 28 \cdot 10^{-4}$ i $E(\Sigma_{odp}) = 8{,}2 \cdot 10^{-4}$.

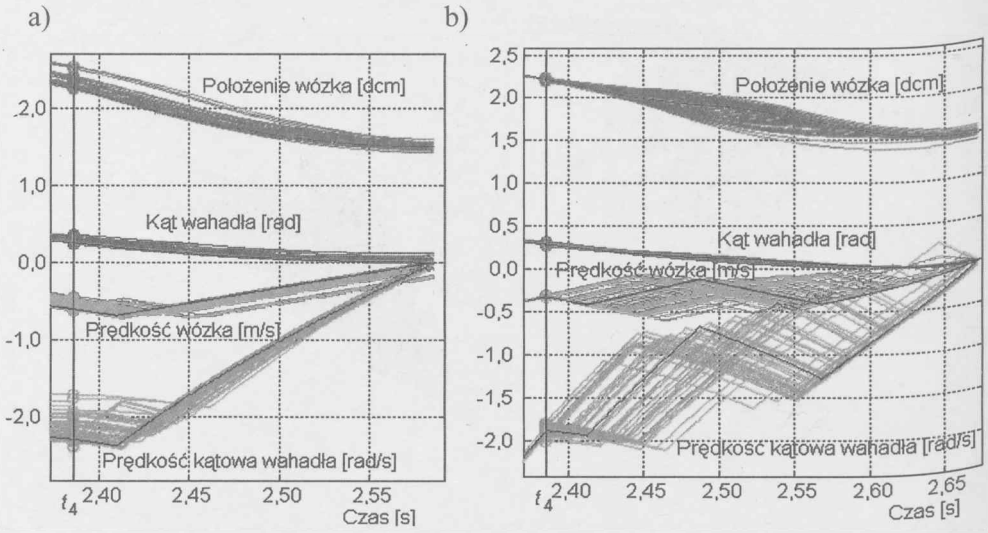
Dla uzupełnienia przedstawimy rzuty zbiorów zaburzeń dopuszczalnych P_D i niedopuszczalnych P_N trajektorii $x^T_{x^0,x^f}$ (rys. 13.13a, b) i $x^{T_1+T_{21}+T_{22}}_{x^0,p^1,p^2,x^f}$ (rys. 13.13c, d), dla chwil końcowych. Początki układów współrzędnych umieszczono w punkcie docelowym x^f.



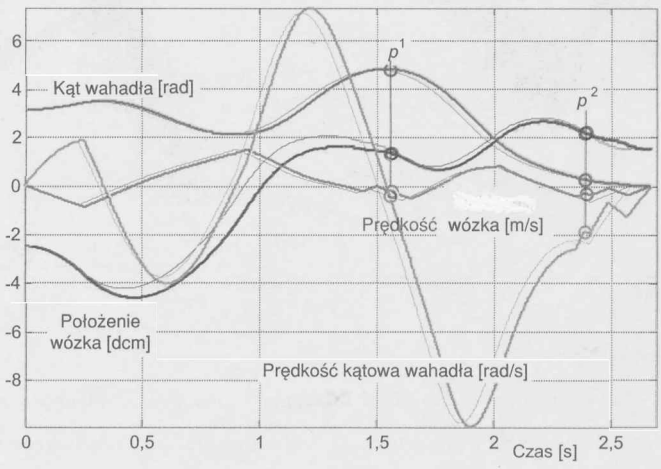
Rys. 13.13. Rzuty zaburzeń dopuszczalnych P_D (kropki) i niedopuszczalnych P_N (kółka) w chwilach końcowych dla trajektorii: a) i b) $x^T_{x^0,x^f}$; c) i d) $x^{T_1+T_{21}+T_{22}}_{x^0,p^1,p^2,x^f}$; odchyłki zmiennych: a) i c) wózka – Δx_1 w dcm i Δx_3 w m/s; b) i d) wahadła – Δx_2 w rad i Δx_4 w rad/s

Wyraźnie widoczna jest różnica w rozkładzie i ilości zaburzeń niedopuszczalnych w chwilach końcowych dla obu eksperymentów.

Analizę zagadnienia odwrażliwiania trajektorii optymalnoczasowych rozszerzymy o pewne aspekty geometryczne. Porównano końcowe fragmenty trajektorii z rysunku 13.1, powstałych przez zaburzanie trajektorii optymalnoczasowej (zob. rys. 13.14a) i końcowe fragmenty 50 wybranych trajektorii uodpornionych z rysunku 13.11 (zob. rys. 13.14b). Na rysunku 13.15 pokazano trajektorię optymalnoczasową $x^{*}_{x^0.x^f}$ i uodpornioną $x^{T_1+T_{21}+T_{22}}_{x^0.p^1.p^2.x^f}$.

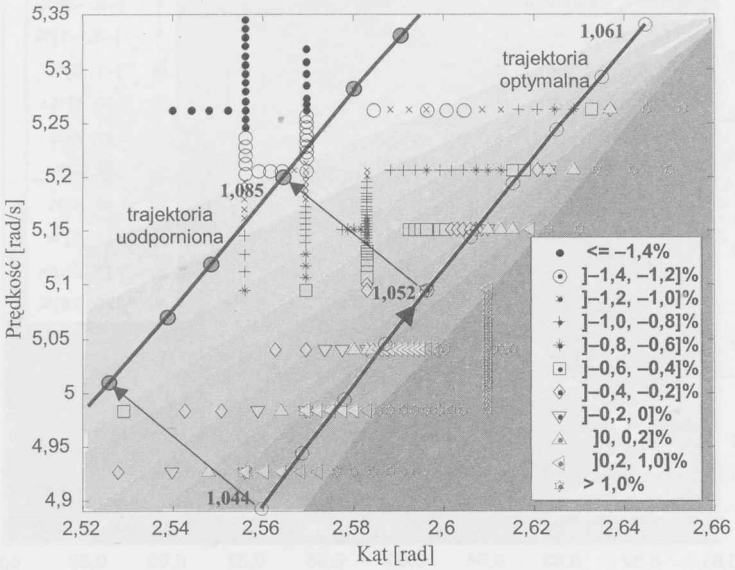


Rys. 13.14. Końcowe fragmenty zaburzonych trajektorii:
a) nieuodpornionych z rysunku 13.1; b) uodpornionych z rysunku 13.11



Rys. 13.15. Trajektorie optymalne i uodpornione (pogrubiona linia) w zadaniu O2 POST+AKRO

Na zakończenie tego podrozdziału pokażmy, jak wyglądają rzuty trajektorii uodpornionej i optymalnoczasowej na płaszczyznę zmiennych stanu wahadła. Na rysunkach 13.16 i 13.17, prócz trajektorii, zaznaczono izochrony dotarcia do celu.



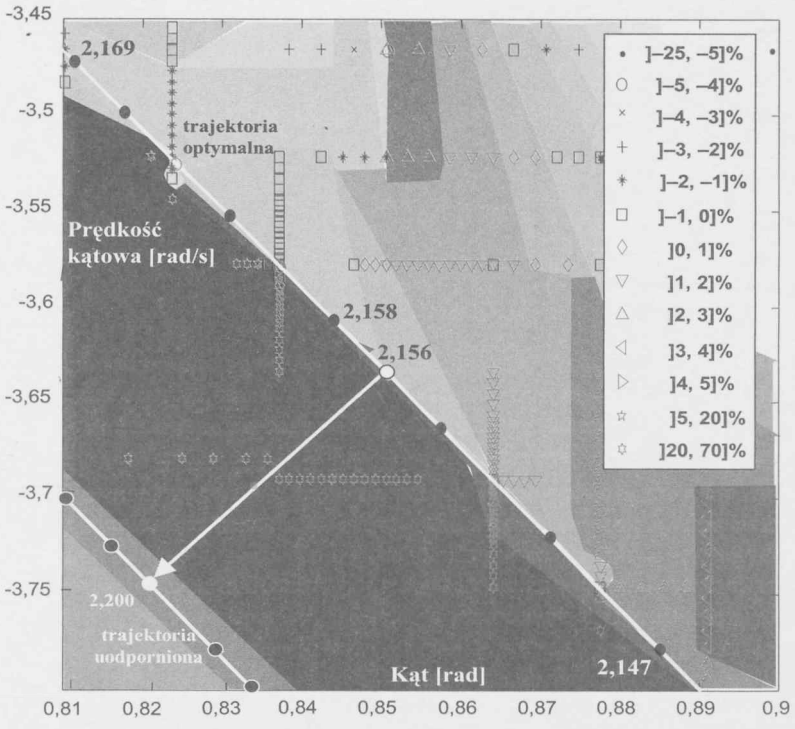
Rys 13.16. Izochrony dotarcia do celu, wyliczone wokół punktu odniesienia (przed piątym przełączeniem) na trajektorii optymalnej, w rzucie na płaszczyznę zmiennych stanu wahadła

Legendy na rysunkach podają procentowe wydłużenie lub skrócenie czasu dotarcia do celu, z punktów leżących wokół trajektorii optymalnej, wyliczone w stosunku do czasu dotarcia do celu z jednego ustalonego punktu odniesienia na trajektorii optymalnej. Punkty odniesienia z trajektorii optymalnej (zaznaczone na rysunkach), względem których podano procentowe przyspieszenia i opóźnienia dotarcia do celu, wybrano następująco:

- na rysunku 13.16 – przed piątym przełączeniem, osiągnięty po czasie 1,052 s, do celu pozostaje 1,535 s;
- na rysunku 13.17 – między szóstym i siódmym przełączeniem, osiągnięty po czasie 2,158 s, do celu pozostaje 0,429 s.

Optymalny czas sterowania wynosi 2,5875 sekundy. Zboczenie z trajektorii optymalnej w lewo (na rysunku 13.17) nieuchronnie prowadzi do 20÷70% wzrostu czasu sterowania. Zjawisko to nie ma miejsca na rysunku 13.16, gdzie trajektoria nie leży na powierzchni przełączeń.

Zauważmy, że uodparnianie trajektorii wiąże się ze zwiększeniem liczby przełączeń sterowania w końcowej fazie ruchu. Wówczas to przesunięcie trajektorii, tak by znalazła się po właściwej stronie optymalnej powierzchni przełączeń, i prowadzenie jej w zbliżeniu do oryginalnej trajektorii optymalnej jest warunkiem uodpornienia.



Rys 13.17. Izochrony dotarcia do celu, wyliczone wokół punktu odniesienia (między szóstym i siódmym przełączeniem) na trajektorii optymalnej, w rzucie na płaszczyznę zmiennych wahadła

13.3. Dalsze przykłady i wnioski o uodparnianiu

Przeprowadzimy eksperyment uodparniania trajektorii wahadła na wózku dla zadania AKRO (ruch akrobaty).

Zadanie optymalnoczasowe O5 AKRO. Ze stanu początkowego

$$x^0 = \mathrm{col}\,(x_1^0,\ 0,\ 0,\ 0),\quad x_1^0 = -0{,}25\ \mathrm{m}$$

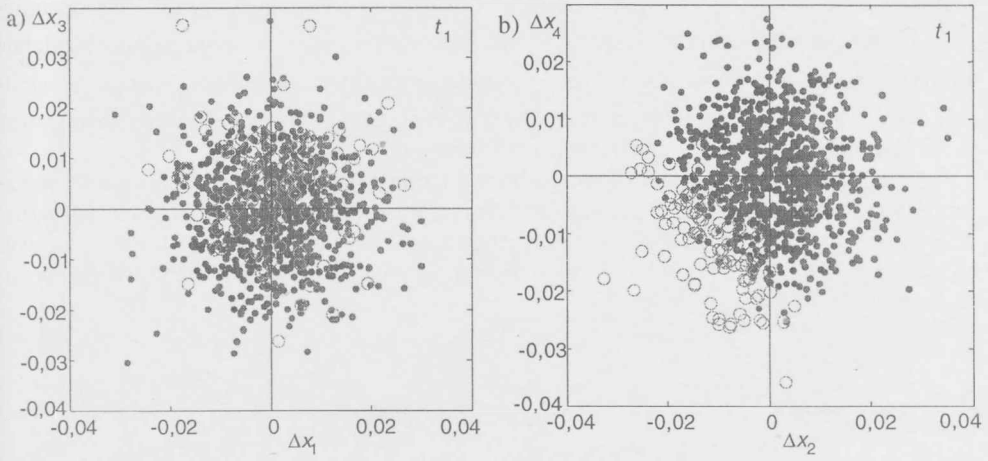
przeprowadzić w minimalnym czasie system (model II) do stanu końcowego

$$x^f = \mathrm{col}\,(x_1^f,\ 0,\ 0,\ 0),\quad x_1^f = 0{,}15\ \mathrm{m}$$

przy ograniczeniach na sterowanie u, $|u| \le u_{\max} = 4\ \mathrm{N}$.

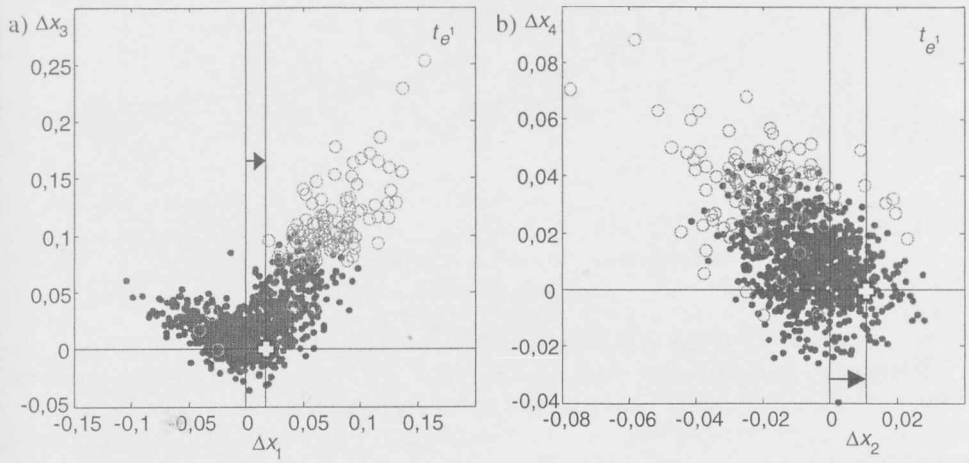
Zbadajmy otoczenie trajektorii optymalnej ruchu akrobaty metodą stochastyczną opisaną w podrozdziale 13.1. Optymalny horyzont t^* wynosi 1,19 s. Rzuty zbiorów zaburzeń dopuszczalnych P_{D_i} i niedopuszczalnych P_{N_i} bada się w chwilach: $t_1 = 0{,}37$ s, $t_2 = 0{,}74$ s i $t^* = 1{,}19$ s.

Z rzutów zaburzeń na płaszczyznę x_1x_3 w chwili t_1 (zob. rys. 13.18) nie można wywnioskować, w którym kierunku przesuwać trajektorię celem uodpornienia.



Rys. 13.18. Rzuty zbiorów zaburzeń dopuszczalnych P_{D_1} (kropki) i niedopuszczalnych P_{N_1} (kółka) w chwili t_1: a) zaburzenia wózka; b) zaburzenia wahadła

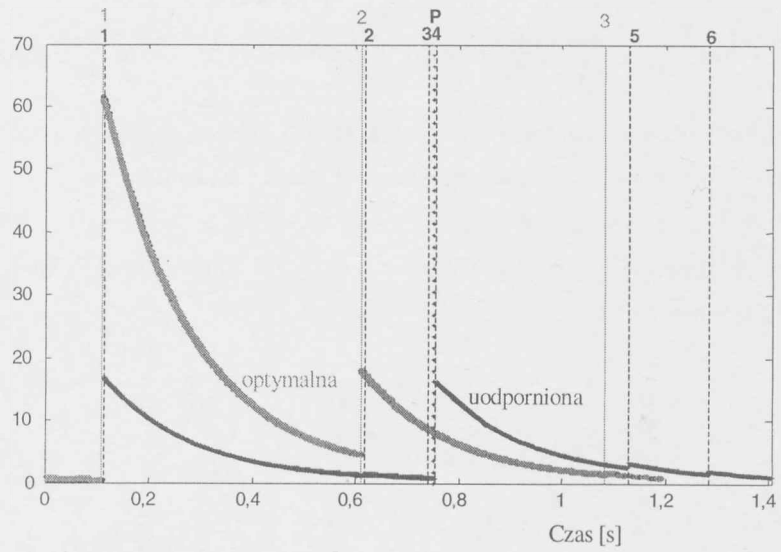
Do operacji przesuwania wybiera się chwilę $t_{e^1} = t_2 = 2t_1$. Otrzymuje się rozkład rzutów pokazany na rysunku 13.19.



Rys. 13.19. Rzuty zbiorów zaburzeń dopuszczalnych P_{D_2} (kropki) i niedopuszczalnych P_{N_2} (kółka) dla chwili t_2: a) zaburzenia wózka; b) zaburzenia wahadła

Trajektorię przesuwa się tak, by możliwie „rozsunąć" zbiory P_{D_2} i P_{N_2} kosztem nieznacznego wzrostu sumarycznego optymalnego czasu przejść od x^0 do p^1 i od p^1 do x^f, w stosunku do optymalnego czasu przejścia od x^0 do x^f. Kierunek i wielkość przesunięcia zaznaczono na rysunku 13.19. Stosując procedurę opisaną w poprzednim rozdziale uodparnia się trajektorię na zakłócenia. Przesunięcie trajektorii w jednym punkcie p^1 skutkuje około trzykrotnym zmniejszeniem wrażliwości górnej. Wygenerowane zostają trzy nowe przełączenia i wzrasta nieznacznie czas sterowania.

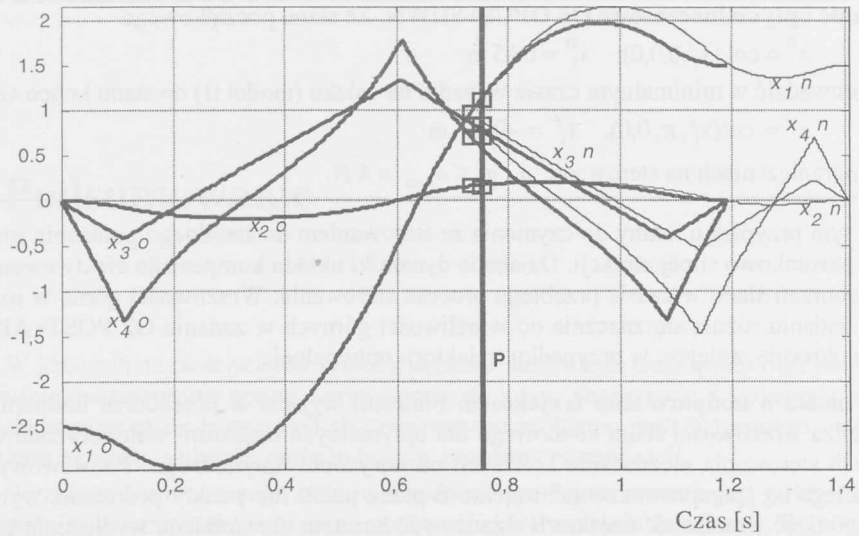
Wykresy norm wrażliwości trajektorii oryginalnej – optymalnej i uodpornionej pokazano na rysunku 13.20. Przełączenia sterowania optymalnoczasowego (linie ciągłe) oznaczono liczbami cienkimi 1, 2 i 3. Przełączenia sterowania uodpornionego (linie przerywane) oznaczono liczbami pogrubionymi od **1** do **6**. Chwilę przesunięcia trajektorii zaznaczono literą P.



Rys. 13.20. Wykresy norm wrażliwości oryginalnej trajektorii optymalnej i trajektorii uodpornionej; literą „P" zaznaczono chwilę przesunięcia trajektorii

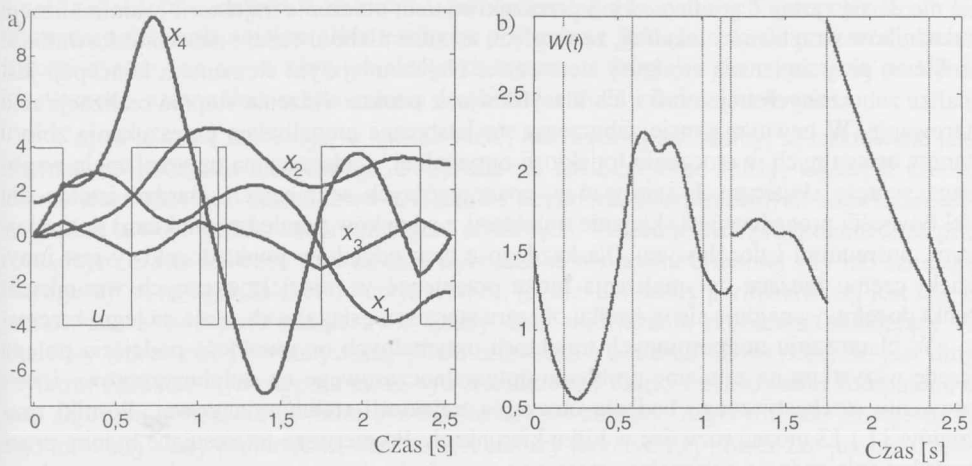
Na rysunku 13.21 przedstawiono przebiegi czasowe trajektorii optymalnych (linia gruba – litera „o") i uodpornionych (linia cienka – litera „n"). Czas sterowania uodpornionego wynosi 1,41 s i jest dłuższy o około 16% od czasu optymalnego. Kwadracikami zaznaczono punkty trajektorii optymalnej (przed przesunięciem) i punkty trajektorii uodpornionej (po przesunięciu).

Dalsze odwrażliwianie można by uzyskać wprowadzając następny punkt przesunięcia w przedziale $[T_1, T_1+T_2]$ między piątym a szóstym przełączeniem, uzyskując nowy odcinek trajektorii z trzema przełączeniami, podobnie jak w zadaniu O2 POST+AKRO.



Rys. 13.21. Przebiegi czasowe trajektorii optymalnych (linie grube – litera „o") i uodpornionych (linie cienkie – litera „n"); x_1 podano w dcm, x_2 w rad, x_3 w m/s i x_4 w rad/s

Na zakończenie przedstawimy zadanie OPUS+SUWN (opuszczenie wahadła z przemieszczeniem wózka – zob. zadanie O6 na następnej stronie), cechujące się niską wrażliwością, w którym uodparnianie trajektorii okazuje się niepotrzebne. Przebieg trajektorii optymalnej i wykres normy wrażliwości w funkcji czasu przedstawiono na rysunku 13.22.



Rys. 13.22. Zadanie O6 OPUS+SUWN: a) przebieg sterowania i trajektorii optymalnoczasowych (sterowanie w N, pozostałe jednostki jak na rysunku 13.21);
b) wykres normy wrażliwości dla zadania stałohoryzontowego z $T = 2{,}525$ s

Zadanie optymalnoczasowe O6 OPUS+SUWN. Ze stanu początkowego

$$x^0 = \mathrm{col}\,(x_1^0, 0, 0, 0), \quad x_1^0 = 0{,}15 \text{ m}$$

przeprowadzić w minimalnym czasie wahadło na wózku (model II) do stanu końcowego

$$x^f = \mathrm{col}(x_1^f, \pi, 0, 0), \quad x_1^f = -0{,}25 \text{ m}$$

przy ograniczeniach na sterowanie u, $|u| \le u_{\max} = 4$ N.

W tym przypadku mamy do czynienia ze sterowaniem do stabilnego położenia równowagi o stosunkowo silnej atrakcji. Działanie dynamiki układu kompensuje efekty ewentualnych zaburzeń stanu w czasie przebiegu procesu sterowania. Wrażliwości górne w rozważanym zadaniu różnią się znacznie od wrażliwości górnych w zadaniu O2 POST+AKRO. Są ośmiokrotnie mniejsze w przypadku trajektorii optymalnej.

Wnioski o uodparnianiu trajektorii. Punktem wyjścia w procedurze uodparniania jest analiza wrażliwości stanu końcowego dla optymalnych trajektorii stałohoryzontowych o czasach sterowania nieznacznie krótszych od horyzontu optymalnego. Podstawowy pomysł polega na „przeprowadzeniu" trajektorii przez punkt lub punkty pośrednie, wybrane tak by obniżyć wrażliwość trajektorii składowych kosztem niewielkiego wydłużenia całkowitego czasu sterowania.

W pierwszym etapie poszukuje się obszaru przestrzeni stanu, z którego zapoczątkowane trajektorie zmierzają do otoczenia celu końcowego i są odporne na zakłócenia. Punkty tego obszaru powinny być osiągalne ze stanu początkowego z małym nadłożeniem całkowitego czasu dotarcia do celu. To właśnie zakłócenia sprzyjające kierują stan do tego obszaru. Wystarczy więc zakłócenia sprzyjające zastąpić „naśladującym" je sterowaniem. W drugim etapie trzeba ustalić, w jaki sposób zakłócenia trajektorii z wybranego obszaru są skutecznie kompensowane na drodze do celu i wybrać właściwe sterowanie kompensujące.

Uodparnianie może wymagać iteracji zarówno pierwszego, jak i drugiego etapu. W zadaniu AKRO charakterystyczne jest połączenie obu etapów w jeden. Opisanego postępowania nie da się zastąpić gradientowym przeszukiwaniem otoczenia trajektorii i minimalizacją wskaźników wrażliwości lokalnej, ze względu na silne nieliniowości i nieróżniczkowalność problemu przy zmianach struktury sterowania. Najistotniejszym elementem koncepcji jest analiza zaburzonych trajektorii i ich klasyfikacja, z punktu widzenia stopnia realizacji celu sterowania. W pewnym sensie zaburzenia stochastyczne umożliwiają przeszukanie zbioru stanów osiągalnych w otoczeniu trajektorii optymalnej, a adaptacyjna optymalizacja wspomaga syntezę skutecznych sterowań w poszczególnych segmentach. Bardzo istotne dla efektywności procedury jest sklejenie trajektorii z odcinków pomiędzy punktami początkowym, pośrednimi i docelowym. Dla każdego z tych odcinków punkt docelowy jest inny, dzięki czemu bieżąca optymalizacja może przebiegać w mniej krytycznych warunkach. Punkt docelowy znajduje się w środku obszaru sterowań osiągalnych, a nie na jego brzegu.

W planowaniu uodpornionych trajektorii optymalnych oryginalność podejścia polega przede wszystkim na zamianie problemu optymalnoczasowego na stałohoryzontowy i zastosowaniu stochastycznego badania otoczenia trajektorii stałohoryzontowej. Wyniki rozdziałów 11 i 13 można rozwijać w kilku kierunkach. Po pierwsze interesujące byłoby przeformułowanie zadania optymalnoczasowego na zadanie sterowania optymalnego ze wskaźnikiem jakości uwzględniającym odporność na zakłócenia, w taki sposób, żeby rozwiązanie tego nowego zadania było zbliżone do rozwiązania problemu pierwotnego. Gdyby okazało się to nieskuteczne, to wydaje się, że można by zalgorytmizować procedurę uodparniania.

Podsumowanie

W klasycznym podejściu do syntezy układów sterowania istotną rolę odgrywają: modelowanie matematyczne oparte o prawa przyrody i dane empiryczne, i wykorzystanie modelu do rozwiązania odpowiednio sformułowanego problemu matematycznego, w którym wyrażone są zadania układu; może to być np. problem optymalizacji.

Wyznacznikiem obecnego stanu w dziedzinie syntezy sterowania jest gwałtowny rozwój techniki komputerowej i rosnące możliwości przetwarzania informacji. Wynikają stąd dwie charakterystyczne tendencje. Pierwsza z nich to stopniowy wzrost roli metod typu *brute force*, w których dominuje przegląd rozwiązań, oparty na symulacjach komputerowych i bezpośrednim wykorzystaniu obserwacji i pomiarów w systemie rzeczywistym. Druga tendencja, powiązana z pierwszą, charakteryzuje się rozwojem metod, które wspólnie określa się jako *soft computing*. Należą tu różnorodne metody sterowania inteligentnego. Kierunek ten wywodzi się z dążenia do uproszczenia procesu syntezy sterowania, uczynienia go bardziej bezpośrednim, przez częściowe przynajmniej uwolnienie projektanta od trudnego zadania konstrukcji modelu matematycznego, jego analizy i stosowania zaawansowanych technik matematycznych. W zamian za to wysiłek projektanta jest kierowany na decyzje wyższego poziomu, o znaczeniu strategicznym, a także na bezpośrednie konstruowanie regulatora w oparciu o ilościowe i jakościowe wyniki symulacji komputerowych lub obserwacji działania obiektu. Obie tendencje prowadzą z jednej strony do poszerzania możliwości syntezy na nowe klasy obiektów złożonych, z drugiej – do wypierania podejścia klasycznego, szczególnie w odniesieniu do obiektów trudnych do modelowania.

Często te nowe tendencje, w sposób mniej lub bardziej świadomy, są traktowane jako alternatywa podejścia klasycznego. Podkreśla się ich bezsporne zalety, takie jak szeroki zakres stosowalności, stosunkowo niski koszt uzyskania akceptowalnych rozwiązań, czy względna łatwość generowania rozwiązań odpornych. Trzeba jednak przypomnieć, że przy rozsądnym czasie obliczeń rozwiązania wyznaczone metodami brutalnej siły lub sztucznej inteligencji są na ogół dalekie od optymalnych. Istotne dla oceny porównawczej jest też, że jakkolwiek w podejściu klasycznym łączny czas rozwiązania problemu, składający się z analizy teoretycznej wykonywanej przez człowieka i obliczeń numerycznych, jest długi i tylko w ograniczonym stopniu może być zredukowany dzięki zastosowaniu komputera, to sam algorytm obliczeniowy może być bardzo szybko zbieżny. Algorytm taki stosuje się na ogół do całej klasy problemów, dla których analizy teoretycznej powtarzać już nie trzeba. Należy przy tym zauważyć, że siła podejścia klasycznego także szybko wzrasta dzięki rozwojowi techniki; obliczenia, które dzisiaj prowadzi się w czasie rzeczywistym, były w ogóle niemożliwe jeszcze kilka lat temu. Rozwój techniki pomiarowej powiązany z analitycznymi metodami uodparniania pozwala implementować rozwiązania bliskie optymalnym.

Celem ogólnym niniejszej monografii było porównanie metod sztucznej inteligencji z metodami klasycznymi, w zastosowaniu do syntezy sterowania dla wybranej klasy układów nieliniowych. Jako reprezentatywne dla pierwszej grupy wybrano metody heurystyczne, nazwane regułowymi, oraz metody oparte na logice rozmytej i sieciach neuralnych. Podejście klasyczne było reprezentowane przez sterowanie optymalnoczasowe i jego różne aproksymacje stałohoryzontowe. Taki wybór uznano za najbardziej odpowiedni, ponieważ rozwiązania optymalnoczasowe mają wśród praktyków opinię szczególnie trudnych w implementacji (głównie z powodu wysokiej wrażliwości), a jednocześnie w wielu dziedzinach zastosowań są bardzo atrakcyjne ze względu na jakość. W ramach podejścia klasycznego analizowano stosowalność lokalnej linearyzacji – pewnej wersji wywodzącej się od Brysona i Pescha metodyki zbliżonych sterowań optymalnych.

Porównanie wymagało stworzenia środowiska programowo-sprzętowego realizującego technologie szybkiego prototypowania regulatorów z wykorzystaniem komputerów klasy PC. Środowisko to umożliwiło przede wszystkim przeprowadzenie eksperymentów weryfikacyjnych, polegających na sterowaniu dynamicznymi układami laboratoryjnymi w czasie rzeczywistym.

Metody „inteligentne", w tym regulatory regułowe, rozmyte i neuralne, okazały się dla wybranej klasy systemów efektywne i przydatne do implementacji w czasie rzeczywistym przy stosunkowo niskim nakładzie obliczeniowym i umiarkowanej jakości sterowania, ocenianej na podstawie czasu osiągnięcia celu.

Doświadczalnie wykazano, że regulacja optymalnoczasowa w układzie nieliniowym jest zupełnie realna, nawet przy użyciu niezbyt zaawansowanego środowiska programowo-sprzętowego (komputer PC). Osiągnięto to dzięki sterowaniu w pętli zamkniętej z repetycyjną optymalizacją. Jakość takiego rozwiązania jest znacznie wyższa niż uzyskiwanego za pomocą metod „inteligentnych". Pokazano również, że trajektorie optymalnoczasowe można skutecznie uodpornić na wpływ zakłóceń, niedokładności modelowania itp. Wyniki wskazują, że rozwój środków sprzętowych i metod obliczeniowych stwarza praktyczne możliwości ponownego zwrócenia się ku technikom analitycznym syntezy sterowania optymalnoczasowego. Nadzieje takie, żywe w latach 60., osłabły po nieudanych próbach implementacji.

Realizacja celu ogólnego pracy była możliwa dzięki oryginalnym rezultatom uzyskanym dla kilku problemów bardziej szczegółowych. W dziedzinie sterowania inteligentnego oryginalny jest algorytm regułowy sterowania systemem wahadła na wózku oraz zastosowanie regulatora Mamdaniego i neuralnego do sterowania tym systemem. Najciekawsze nowe propozycje, zdaniem autora, zawarte są w części pracy dotyczącej sterowania optymalnego. W zakresie algorytmów obliczeniowych optymalizacji dynamicznej należy tu przede wszystkim wymienić metodę zmiennej parametryzacji z generacją i redukcją przełączeń. W dziedzinie syntezy opartej na koncepcji trajektorii zbliżonych oryginalnymi rezultatami są: jawna reprezentacja regulatora zlinearyzowanego, sformułowanie równania Riccatiego i koncepcja regulacji adaptacyjnej ze zmienną strukturą, łącząca metodę zmiennej parametryzacji z lokalną linearyzacją. Szczególną uwagę poświęcono w pracy syntezie sterowań odpornych, bliskich optymalnym, połączonej z analizą wrażliwości. Zaproponowano tu oryginalną metodę stochastycznego badania otoczenia trajektorii optymalnej i syntezy trajektorii uodpornionych metodą punktów pośrednich. Nowa jest również koncepcja syntezy regulatora optymalnoczasowego metodą zbiorów reprezentatywnych.

Wyniki przedstawione w pracy pozwalają ustalić kierunki dalszych badań. Wysiłki powinny koncentrować się na kilku kierunkach badawczych. Po pierwsze, należy rozważyć połączenie zadania uodparniania z zadaniem optymalizacji przez konstrukcję syntetycznych wskaźników jakości. Po drugie, obiecujące doświadczenia w zakresie optymalizacji repetycyjnej skłaniają do podjęcia próby włączenia elementów identyfikacji na bieżąco do warstwy obliczania sterowania w czasie rzeczywistym. Ważnym kierunkiem dalszych prac będzie również rozszerzenie zakresu stosowalności zaproponowanych rozwiązań na inne klasy obiektów. Powinno to być łatwiejsze dzięki syntezie metod *soft computing* i podejścia analitycznego.

Literatura

Adamski A., Turnau A. 1998: *Simulation support tool for real-time dispatching control in public transport.* Transportation Res.-A, vol. 32, No 2, pp. 73–87.

Alamir M. 2000: *Solutions of nonlinear optimal and robust control problems via a mixed collocation/DAE's based algorithm.* 39th IEEE Conf. Decision Contr., Sydney, Australia, December 12–15 2000, 6 p.

Anderson C.W. 1989: *Learning to control an inverted pendulum using neural networks.* IEEE Contr. Syst. Mag., vol. 9, pp. 31–37.

Andrews G. 1996: *Shooting method for numerical solution of optimal control problems with bounded state variable.* J. Optim. Theory Appl., vol. 89, No 2, pp. 351–372.

Aoustin Y., Chevallereau A., Glumineau A., Moog C.H. 1994: *Experimental Results for the End-Effector Control of a Single Flexible Robotic Arm.* IEEE Trans. Contr. Syst. Technol., vol. 2, pp. 371–381.

Arzen K-E. 1995: *AI in the Feedback Loop: a Survey of Alternative Approaches.* [in:] Kocijan J. (ed.), Int. Workshop on Artificial Intelligence in Real-time Control, Springer-Verlag, Berlin, pp. 207–218.

Athans M., Falb P.L. 1966: *Optimal Control.* McGraw-Hill, New York.

Åström K.J., Furuta K. 2000: *Swinging up a pendulum by energy control.* Automatica, vol. 36, No 2, pp. 287–295.

Barclay A., Gill P.E., Rosen J.B. 1998: *SQP methods and their application to numerical optimal control.* Int. Series of Numerical Mathematics 124, Birkhäuser, Basel, pp. 207–222.

Barto A.G., Sutton R.S., Anderson C.W. 1983: *Neuron-like adaptive elements that can solve difficult control problems.* IEEE Trans. Syst. Man. Cybern., vol. 13, pp. 834–846.

Bless R.R., Hodges D.H., Seywald H. 1995: *Finite element method for the solution of state-constrained optimal control problems.* J. Guid., Contr. Dyn., vol. 18, No 5, pp. 1036–1043.

Bloch A.M., Leonard N.E., Marsden J.E. 2000: *Controlled lagrangians and the stabilization of mechanical systems I: The first matching theorem.* IEEE Trans. Automat. Contr., vol. 45, No 12, pp. 2253–2270.

Bock H.G., Plitt K.J. 1984: *A multiple shooting algorithm for direct solution of optimal control problems.* IFAC 9th World Congress, Budapest, July 2–6 1984, pp. 242–247.

Bołtianski W.G. 1971: *Matematyczne metody sterowania optymalnego.* WNT, Warszawa.

Bonnans F., Shapiro A. 1998: *Optimization problems with perturbations: A guided tour.* SIAM Rev., vol. 40, No 2, pp. 228–264.

Boone G. 1997: *Minimum-time control of the acrobot.* IEEE Int. Conf. Robotics and Automation, Albuquerque, New Mexico, April 1997, pp. 3281–3287.

Branch M.A., Grace A. 1996: *MATLAB Optimization Toolbox. User's Guide.* MathWorks, Inc., Natick.

Brown R.H., Schneider S. C., Mulligan M. G. 1992: *Analysis of algorithms for velocity estimation from discrete position versus time data.* IEEE Trans. Industr. Electron., vol. 39, No 1, pp. 11–19.

Bryson A.E. Jr. 1999: *Dynamic Optimization.* Addison – Wesley.

Bryson A. E., Ho Y.C. 1975: *Applied Optimal Control.* Hemisphere, Washington DC.

Byrski W., Duda J., Turnau A. 1988: *Problemy sterowania on-line procesem wielostopniowej destylacji.* Materiały X Krajowej Konferencji Automatyki, t. 2, Lublin, 21–24 czerwca 1988, s. 69–77.

Canudas de Wit C., Olsson H., Åström K.J., Lischinsky P. 1995: *A new model for control of systems with friction.* IEEE Trans. Automat. Contr., vol. 40, No 3, pp. 419–425.

Carew W.D., Prince K.J. 1997: *Windows NT Workstation, the VMEbus and Real-Time control.* IEEE Contr. Syst., pp. 77–88.

Carli A. De, Cong S., Matacchioni D. 1994: *Dynamic Friction Compensation in Servo drives,* Proc. IEEE Int. Conf. Contr. Appl., pp. 193–198.

Chen L., Smith R. 1998: *Closed-loop model validation; an application to an unstable experimental system.* Proc. Amer. Contr. Conf., Philadelphia, Pensylvania, June 24–26 1998, 5 p.

Chen Yung-Yaw, Lin Kao-Zong. 1992: *Learning Behaviour of Fuzzy Controllers with Neuron Adaptive Elements.* ACC/TM3, pp. 1878–1882.

Chuang C.-H., Lee T.S. 1996: *A hybrid fuzzy neural network and its control applications.* Proc. IEEE Int. Symp. Intell. Contr., Dearbon, USA, September 15–18 1996, pp. 175–180.

Chung Choo Chung, Hauser J. 1995: *Nonlinear control of a swinging pendulum.* Automatica, vol. 31, No 6, pp. 851–862.

Conway B.A., Larson K.M. 1998: *Collocation versus differential inclusion in direct optimization.* J. Guid., Contr. Dyn., vol. 21, No 5, pp. 780–785.

Cribb J. 1989: *Comparison and analysis of algorithms for reinforcement learning.* Department of Computer Science, University of New South Wales (honours thesis).

Cypkin J.Z. 1973: *Podstawy teorii układów uczących się.* WNT, Warszawa.

Doyle J.C., Glover K., Khargonekar P.P., Francis B.A. 1989: *State-space solutions to standard H_2 / H_∞ control problems.* IEEE Trans. Automat. Contr., vol. AC-34, No 8, pp. 831–847.

Driankov D., Hellendoorn H., Reinfrank M. 1996: *Wprowadzenie do sterowania rozmytego.* WNT, Warszawa.

Fattorini H.O. 1999: *Infinite dimensional optimization and control theory.* Cambridge University Press, United Kingdom.

Fiedorenko R.P. 1978: *Priblizennoje reszenije zadacz optimalnowo uprawlenija.* Nauka, Moskwa.

Findeisen W., Szymanowski J., Wierzbicki A. 1980: *Teoria i metody obliczeniowe optymalizacji.* PWN, Warszawa.

Fuksa S., Turnau A. 1996: *Stereo-vision system for robot configuration and motion detection.* 27th Int. Symp. „Industr. Robots ROBOTICS towards 2000", Milano, Italy, October 6–8 1996, pp. 643–645.

Furuta K., Ochai T., Ono N. 1984: *Altitude control of triple inverted pendulum.* Int. J. Contr., vol. 39, pp. 1351–1365.

Furuta K., Kaiwara H., Kosuga K. 1988: *Digital control of a double inverted pendulum on an inclined rail.* Int. J. Contr., vol. 32, pp. 907–924.

Gabasow R., Kiriłowa F.M. 1973: *Osobyje optimalnyje uprawlenija.* Nauka, Moskwa.

Galicki M. 2000: *Wybrane metody planowania optymalnych trajektorii robotów manipulacyjnych.* WNT, Warszawa.

Geva S., Hang C.C., Zhang T. 1998: *A direct approach to adaptive controller design and its application to inverted pendulum tracking.* Proc. Amer. Contr. Conf., Philadelphia, Pensylvania, June 24–26 1998, pp. 1043–1047.

Geva S., Sitte J. 1993: *A cartpole experiment benchmark for trainable controllers.* IEEE Contr. Syst., pp. 40–51.

Ghalia M.B. 1997: *Modelling and robust control of uncertain dynamical systems using fuzzy set theory.* Int. J. Contr., vol. 68, No 6, pp. 1367–1395.

Goodrich M.A., Stirling W.C., Frost R.L. 1996: *A satisficing approach to intelligent control of nonlinear systems.* Proc. IEEE Int. Symp. Intell. Contr., Dearbon, USA, September 15–18 1996, pp. 248–252.

Gorczyca P., Turnau A. 1998: *Wielowymiarowy nieliniowy system MIMO.* [w:] Szymkat M. (red.), „Komputerowe wspomaganie w obliczeniach naukowo-technicznych – przykłady zastosowań pakietów MATLAB i Maple V", Kraków, CCATIE Krakowskie Centrum Informatyki Stosowanej, s. 37–60.

Górecki H., Turowicz A. 1968: *On switching instants in minimum-time control problem.* Bull. Acad. Polon. Sci. – Ser. Sci. Techn., vol. 16, No 2, pp. 205–218.

Górecki H., Fuksa S., Korytowski A., Mitkowski W. 1983: *Sterowanie optymalne w systemach liniowych z kwadratowym wskaźnikiem jakości.* PWN, Warszawa.

Green M. 1993: *H_∞ controller synthesis by J-Lssless coprime factorization.* SIAM J. Contr. Optim., vol. 30, pp. 522–547.

Grega W. 1996: *Integrated environment for real-time control and simulation.* Comput. Industry, vol. 31, pp. 3–14.

Grega W., Kołek K., Turnau A. 1997: *Sterowanie w zintegrowanym środowisku czasu rzeczywistego.* I Krajowa Konferencja CCATIE „Metody i systemy komputerowe w badaniach naukowych i projektowaniu inżynierskim", Kraków, 25–26 listopada 1997, s. 695–7021.

Grega W., Kołek K., Turnau A. 1998a: *Real Time Kernel dedicated to fast mechatronic systems.* Proc. Mediterr. Conf. "Electron. Automat. Contr.", Marrakech, Maroc, September 17–19 1998, pp. 480–483.

Grega W., Kołek K., Turnau A. 1998b: *Rapid environment for real-time control in education.* Proc. 3rd IEEE Real-time System Education Workshop, Poznań, November 21 1998, pp. 85–92.

Grega W., Turnau A. 1998: *Development of intelligent control algorithms in open architecture environment.* Proc. Summer School '98, Kraków, July 7–11 1998, pp. 233–262.

Grega W., Turnau A. 1999: *Time-optimal control of nonlinear systems: new methods and applications.* Proc. Summer School '99, Integrated Control Systems and Intelligent Control, Rzeszów University of Technology, Wetlina, June 8–12 1999, pp. 49–75 .

Grega W., Pauluk M., Turnau A. 2001: *Projektowanie algorytmów sterowania dla suwnicy ramowej.* Zeszyty Naukowe Politechniki Opolskiej, z. 64, nr 265, XIV Konferencja Naukowa Problemy rozwoju maszyn roboczych, Zakopane, 2001, s. 129–136.

Guez A., Selinsky J. 1988: *A trainable neuromorphic controller.* J. Robotics Syst., vol. 5, pp. 363–388.

Gupta M.M., Sinha K. 1996: *Intelligent control systems. Theory and applications.* IEEE Press, New York.

Hajduk K., Pauluk M., Turnau A. 1995: *Sterowanie serwomechanizmem cyfrowym w środowisku MATLAB-a.* Materiały I Krajowej Konferencji Użytkowników MATLAB-a, Kraków, 14–15 listopada 1995, s. 203–208.

Hanselmann H. 1992: *Motion control with DSP.* JSME Seminar on "Control for Motion and Vibration Based on Advanced Theory with DSP", Chuo University, Tokyo, November 16–17.

Hanselman H. 1993: *Hardware-In-The-Loop Simulation for Development and Test of Electronic Control Units and Mechanical Components Real-time Systems 93.* Conf. and Exhibition, Paris, 1993, pp. VII–5 – VII–19.

Hargraves C.R., Paris S.W. 1987: *Direct trajectory optimization using nonlinear programming and collocation.* AIAA J. Guid., vol. 10, No 4, pp. 338–342.

Hauser J., Meyer D.G. 1998: *Trajectory morphing for nonlinear systems.* Proc. Amer. Contr. Conf., Philadelphia, Pensylvania, June 24–26 1998, pp. 2065–2070.

Hull D.G. 1997: *Conversion of optimal control problems into parameter optimization problems.* J. Guid., Contr. Dyn., vol. 20, No 1, pp. 57–60.

Hunt K.J., Sbarbaro D., Żbikowski R., Gawthrop P.J. 1992: *Neural networks for control systems – a survey.* Automatica, vol. 28, No 6, pp. 1083–1112.

Isidori A., Byrnes C.I. 1990: *Output regulation of nonlinear systems.* IEEE Trans. Automat. Contr., AC-35, pp.131–140.

Jakubczyk B., Respondek W. (eds) 1997: *Geometry of Feedback and Optimal Control.* Marcel Dekker, New York – Basel – Hong Kong.

Johansson M. 1994: *The nonlinear nature of fuzzy control.* [in:] Reusch B. (ed.), Fuzzy Logic, Springer-Verlag, Dortmund, pp. 350–356.

Johnson D., Brown G.V., Inman D.J. 1998: *Adaptive variable bias magnetic bearing control.* Amer. Contr. Conf., Philadelphia, Pensylvania, June 24–26 1998, 7 p.

Ju Li, Longman W., Schulz V.H., Bock H.G. 1998: *Implementing time optimal robot maneuvers using realistic actuator constraints and learning control.* www.space-flight.org/AAS meetings/ winter.

Kaesbauer D., Ackermann J. 1998: *How to escape from the fragility trap.* Proc. Amer. Contr. Conf., Philadelphia, Pensylvania, June 24–26, pp. 2832–2836.

Kajiwara H., Apkarian P., Gahinet P. 1999: *LPV techniques for control of an inverted pendulum.* IEEE Contr. Syst., pp. 44–54.

Keel L.H., Bhattacharyya S.P. 1997: *Robust, fragile, or optimal.* IEEE Trans. Automat. Contr., vol. 42, No 8, pp. 1098–1105.

Kempa A., Kołek K., Korytowski A., Tabakowski P., Turnau A. 1997: *Neural time-optimal real-time controller.* Proc. 16th IASTED Int. Conf. "Modelling, Identification and Control", Innsbruck, Austria, February 17–19 1997, pp. 214–219.

Kieffer J., Cahill A. J., James M. R. 1997: *Robust and accurate time-optimal path-tracking control for robot manipulators.* IEEE Trans. Robotics Automat., vol. 13, No 6, pp. 880–890.

Kierzenka J., Shampine L.F. 1999: *A BVP solver based on residual control and the MATLAB PSE.* Preprint.

Kołek K., Turnau A. 1990: *Sprzężenie IBM'a z wahadłem odwróconym sterowanym on-line.* Materiały Konferencji Resortowej Teorii Sterowania i Optymalizacji, Kozubnik, 20–22 września 1990, 10 s.

Kołek K., Tabakowski P., Turnau A. 1993: *Control of nonlinear system, simulation, real-time control, application of neural networks.* Proc. Int. Conf. "Modelling, Identification and Control", Innsbruck, Austria, 1993, pp. 126–127.

Kołek K., Pauluk M., Rosół M., Turnau A. 1995: *Real time simulation environment for control of high-speed unstable mechanical systems.* Proc. of 2nd Int. Symp. "Methods and Models in Automation and Robotics", Międzyzdroje, Poland, 30 August – 2 September 1995, vol. 1, pp. 413–418.

Korytowski A., Szymkat M., Turnau A. 1998: *Optymalnoczasowe sterowanie wahadłem na wózku.* [w:] Szymkat M. (red.), „Komputerowe wspomaganie w obliczeniach naukowo-technicznych – przykłady zastosowań pakietów MATLAB i Maple V". Kraków, CCATIE Krakowskie Centrum Informatyki Stosowanej, s. 177–270.

Korytowski A., Szymkat M., Turnau A., Miller J. 1999: *Optymalnoczasowe wyprowadzenie samolotu F-15 do lotu poziomego z maksymalną prędkością.* II Krajowa Konferencja „Metody i systemy komputerowe w badaniach naukowych i projektowaniu inżynierskim", Kraków, 25–27 października 1999, s. 237–242.

Korytowski A., Szymkat M., Turnau A. 2001: *Adaptive linearized switching controller for constrained optimal control problems.* Proc. VII IEEE Int. Conf. "Methods and Models in Automation and Robotics", Międzyzdroje, August 28–31 2001, pp. 187–192.

Kosko B. 1992: *Neural networks and fuzzy systems.* Prentice Hall, New York.

Krafka P., Kunze P. 1994: *DSP Controller for Synchronous Drives.* PCIM Europe, pp. 20–21.

Kraft D. 1994: *TOMP – Fortran modules for optimal control calculations.* ACM Trans. Math. Software, vol. 20, No 3, pp. 262–281.

Kratochwil K., Engelbrecht R., Jörgl H.P. 1993: *A reward punishment learning method to swing up a pendulum into its upright position.* Proc. IFAC, vol. 9, pp. 477–482.

Kristic M., Canellacopoulos J., Kokotovic P. 1995: *Nonlinear and adaptive control design.* John Wiley & Son, New York.

Kumar R. R., Seywald H. 1996: *Should controls be eliminated while solving optimal control problems via direct methods?.* J. Guid., Contr. Dyn., vol. 19, No 2, pp. 418–423.

Kurzhansky A. B. 1977: *Control and observation under conditions of uncertainty.* Nauka, Moscow.

Kurzhansky A. B. 1994: *On the stabilization of uncertain differential systems.* Lect. Not. Appl. Math., vol. 162, pp. 217–225.

Kwakernaak H., Sivan R. 1972: *Linear Optimal Control Systems.* Wiley – Interscience, New York.

Lairi M., Bloch G. 1998: *Neural control of maglev.* Proc. Mediterr. Conf. Electron. Automat. Contr., Marrakech, Maroc, September 17–19 1998, pp. 472–475.

Lairi M., Bloch G., Millerioux G. 1999: *Real time feedforward neural control of a MagLev system.* Kluwer Academic Publisher, Netherlands.

Lambrechts P.F. 1993: *Application of a DSP-based control implementation environment on mechanical servo systems.* INFO, No 7, pp. 69–75.

Lastman G.J. 1978: *A shooting method for solving two-point boundary-value problems arising from non-singular bang-bang optimal control problems.* Int. J. Contr., vol. 27, No 4, pp. 513–524.

Law J.K.C. 1992: *Adaptive rule-based control.* Master Thesis, School of Computer Science and Engineering, University of New South Wales.

Layne J.R, Passino K.M., Yurkovitch S. 1993: *Fuzzy learning control for antiskid braking systems.* IEEE Tran. Contr. Sys. Tech., vol. 1, No 2, pp. 122–129.

Lee A.Y., Bryson Jr. A.E. 1989: *Neighbouring extremals of dynamic optimization problems with parameter variations.* Optimal Contr. Appl. Meth., vol. 10, pp. 39–521.

Lee A.Y., Smyth P. 1994: *Synthesis of minimum-time feedback laws for dynamic systems using neural networks.* J. Guid., Contr. Dyn., vol. 17, No 4, pp. 868–870.

Lee E.B., Markus L. 1967: *Foundation of optimal control theory.* SIAM Ser. in Applied Math., John Wiley, New York.

Lemkin M., Yang P.H., Huang A.C., Jones J., Auslander D.M. 1995: *Velocity estimation from widely spaced encoder pulses.* lemkin@euler.me.berkeley.edu.

Lewis F.L., Kai Liu 1996: *Towards a paradigm for fuzzy logic control.* Automatica, vol. 32, No 2, pp. 167–181.

Li F., Bainum P.M. 1994: *Analytic time-optimal control synthesis of fourth-order system and maneuvers of flexible structures.* J. Guid., Contr. Dyn., vol. 17, No 6, pp. 1171–1178.

Lin Z., Saberi A., Gutmann M., Shmash Y.A. 1996: *Linear controller for an inverted pendulum having restricted travel: a high-and-low gain approach.* Automatica, vol. 32, pp. 933–937.

Linden G.W., Lambrechts P.F. 1993: H_∞ *control of an experimental inverted pendulum with dry friction.* IEEE Contr. Syst., pp.44–50.

Lischinsky P., Canudas de Wit C., Morel G. 1999: *Friction compensation for an industrial hydraulic robot.* IEEE Contr. Syst., pp. 25–32.

Longman R.W., Li J., Steinbach M.C., Bock H.G. 1997: *Issues in the implementation of time-optimal robot path planning.* Proc. *AIAA* "Nonlinear Dynamical Systems Symposium", Reno, NV, Jan.

Maiers J., Sherif Y.S. 1985: *Applications of fuzzy set theory.* IEEE Trans. Syst., Man. Cybern., vol. SMC-15, No 1, pp. 175–178.

Malanowski K. 1987: *Stability and sensitivity of solutions to optimal control problems for systems with control appearing linearly.* Appl. Math. Optimiz., vol. 16, pp. 73–91.

Malanowski K., Maurer H. 1996: *Sensitivity analysis for parametric control problems with control-state constraints.* Computational Optimization and Applications, vol. 5, pp. 253–283.

Maurer H., Pesch H.J. 1994: *Solution differentiability for nonlinear parametric control problems.* SIAM J. Contr. Optimiz., vol. 32, No 6, pp. 1542–1556.

Medrano-Cerda G.A. 1999: *Robust computer control of an inverted pendulum.* IEEE Contr. Syst., vol. 19, No 3, pp. 58–67.

Meier H., Farwig Z.U., Unbehauen H., 1990: *Discrete computer control of a triple-inverted pendulum.* Optimal Contr. Appl. Meth., vol. 11, pp. 151–171.

Michie D., Chambers R.A. 1968: *BOXES: an experiment in adaptive control.* [in:] Dale E., Michie D. (eds), Machine Intelligence 2, Oliver and Boyd, Edinburgh, pp. 137–152.

Milam M.B., Mushambi K., Murray R.R. M. 2000: *A new computational approach to real-time trajectory generation for constrained mechanical systems.* 39th IEEE Conf. Decision Contr., Sydney, Australia, December 12–15 2000, 6 p.

Mitkowski W., Turnau A. 1976: *Modelowanie obiektów cieplnych z punktu widzenia sterowania docelowego.* Zeszyty Naukowe AGH, Automatyka, z. 13, pp. 143–151.

Mitkowski W., Turnau A. 1982: *Approximate solution of minimum-time problem for heat equation.* Elektrotechnika, vol. 1, No 3, pp. 81–95.

Mitkowski W. 1991: *Stabilizacja systemów dynamicznych.* WNT, Warszawa.

Mori S., Nishihara H., Furuta K. 1976: *Control of unstable mechanical system: Control of pendulum.* Int. J. Contr., vol. 23, No 5, pp. 673–692.

Motet G., Szmuc T. 1998: *Programowanie systemów czasu rzeczywistego.* CCATIE Krakowskie Centrum Informatyki Stosowanej, Kraków, vol. 11.

Mordukhowich B. 1994: *Sensitivity analysis for constraint and variational systems by means of set-valued differentiation.* Optimization, vol. 31, pp. 13–45.

Napoleon, Hoshino T., Furuta K. 2000: *Hand over control of unstable object using manipulators – an approach of continuously switching of controllers.* 39th IEEE Conf. Decision Contr., Sydney, Australia, December 12–15 2000, 6 p.

Narendra K.S., Parthasarathy K. 1990: *Identification and control of dynamical systems using neural networks.* IEEE Trans. on Neural Networks, vol. 1, No 1, pp. 4–26.

Nguen S., Widrow B. 1990: *Neural networks for self-learning control systems.* IEEE Control Syst. Mag., pp. 18–23.

Orrell D., Zeidan V. 1988: *Another Jacobi sufficiency criterion for optimal control with smooth constraints.* J. Optimization Theory and Applications, vol. 58, pp. 283–300.

Palm R., Driankov D., Hellendoorn H. 1996: *Model Based Fuzzy Control.* Springer-Verlag, Berlin.

Pao L.Y. 1996: *Minimum-time control characteristics of flexible structures.* J. Guid., Contr. Dyn., vol. 19, No 1, pp. 123–129.

Pao Y., Franklin G.F. 1993: *Proximate time-optimal control of third-order servomechanisms.* IEEE Trans. Automat. Contr., vol. 38, No 4, pp. 560–580.

Pauluk M., Turnau A. 1997: *Stateflow jako narzędzie prototypowania.* I Krajowa Konferencja CCATIE „Metody i systemy komputerowe w badaniach naukowych i projektowaniu inżynierskim", Kraków, 25–26 listopada 1997, s. 507–514.

Pauluk M., Korytowski A., Turnau A., Szymkat M. 2001: *Time optimal control of 3d crane.* Proc. VII IEEE Int. Conf. „Methods and Models in Automation and Robotics", Międzyzdroje, August 28–31 2001, pp. 927–932.

Pesch H.J. 1989a: *Real-time computation of feedback controls for constrained optimal control problems, Part 1: Neighboring extremals.* Optimal Control Applications and Methods, vol. 10, No 2, pp. 129–145.

Pesch H.J. 1989b: *Real-time computation of feedback controls for constrained optimal control problems, Part 2: A correction method based on multiple shooting.* Optimal Control Applications and Methods, vol. 10, No 2, pp. 147–171.

Pesch H.J. 1994: *Off-line and on-line computations of optimal trajectories in the aerospace field.* [in:] Miele A., Salvetti A. (eds), Applied Mathematics in Aerospace Science and Engineering. Mathematical Concepts in Science and Engineering 44, Plenum Press, New York, pp. 165–220.

Pontriagin L.S., Bołtianskij W.G., Gamkrelidze R.W, Miszczenko E.F. 1976: *Matematiczeskaja teorija optimalnych processow.* Fizmatgiz, Moskwa.

Qifeng Wei, Dayawansa W.P., Levine W.S. 1995: *Nonlinear controller for an inverted pendulum having restricted travel.* Automatica, vol. 31, No 6, pp. 841–850.

Rai S. 1998: *Design optimization of robots based on time optimal control.* Proc. IEEE Int. Conf. Contr. Appl., Trieste, Italy, September 1–4 1998, pp. 913–919.

Ravn O., Szymkat M. 1995: *Mechatronic approach to robotic modelling-software engineering perspective.* Proc. 2nd Int. Symp. "Methods and Models in Automatics and Robotics", Banka S. (ed.), Szczecin, 1995, pp. 455–466.

Reif K., Weinzierl K., Zell A., Unbehauen R. 1997: *Nonlinear feedback stabilization by tangential linearization.* Int. J. Contr., vol. 68, No 3, pp. 673–687.

Renaud M., Fourquet J.Y. 1997: *Minimum time motion of a mobile robot with two independent, acceleration-driven wheels.* IEEE Int. Conf. Robotics and Automation, Albuquerque, New Mexico, April, 1997, pp. 2608–2613.

Rolewicz S. 1974: *Analiza funkcjonalna i teoria sterowania.* PWN, Warszawa.

Sammut C. 1994: *Recent progress with BOXES.* In: Furukawa K., Micchie D., Muggleton S. (eds), Machine Intelligence 13, pp. 363–383, Clarendon Press, Oxford.

Sammut C., Michie D. 1991: *Controlling a "Black-Box" simulation of a spacecraft.* AI Magazine, vol. 12, No 1, pp. 56–63.

Schmid Chr. 1992: *Real-Time Control with CADACS-PC.* [in:] Jamshidi M. (ed.), Recent Advances in Computer-Aided Control Systems Engineering, Elsevier, pp. 337–355.

Schwartz A.L. 1995: *RIOTS. A Matlab Toolbox for Solving Optimal Control Problems.* The MathWorks, Inc. Natick, Mass.

Seywald H. 1994: *Trajectory optimization based on differential inclusion.* J. Guid., Contr. Dyn., vol. 17, No 3, pp. 480–487.

Seywald H., Kumar R. HR., Deshpande S.S., Heck M.L. 1994: *Minimum fuel spacecraft reorientation.* J. Guid., Contr. Dyn., vol. 17, No 1, pp. 21–29.

Seywald H., Kumar R. 1995: *Genetic algorithm approach for optimal control problems with linearly appearing.* J. Guid., Contr. Dyn., vol. 18, No 1, pp. 177–182.

Shr-Shiung Hu, Bor-Chin Chang 1998: *Design of a nonlinear H∞ controller for the inverted pendulum system.* Proc. Int. Conf. Contr. Appl., Trieste, Italy, 1–4 September 1998, pp. 699–703.

Siciliano B., Villani L. 1996: *A passivity-based approach to force regulation and motion control of robot manipulators.* Automatica, vol. 32, No 3, pp. 443–447.

Sirisena H.R. 1974: *A gradient method for computing optimal bang-bang control.* Int. J. Contr., vol. 19, No 2, pp. 257–264.

Sirisena H.R., Chou F. S. 1979: *Convergence of Control Parametrization Ritz Method for nonlinear optimal control problems.* J. Optim. Theory Appl., vol. 29, pp. 369–382.

Smith M.H., Elbs M. 1999: *Towards a more efficient approach to automotive embeded control system development.* Proc. IEEE Int. Symp. Comput. Aided Contr. Design, Kohala Coast-Island of Hawai'i, USA, August 22–27 1999, pp. 219–224.

Smith R.S., Doyle J.C. 1992: *Model validation: a connection between robust control and identification.* IEEE Trans. Automat. Contr., vol. 39, No 5, pp. 951–959.

Sontag E.D. 1993: *Neural Networks for Control.* [in:] Trentelman H.L., Willems J.C. (eds), Essays on Control: Perspectives in the Theory and its Applications. Birkhäuser, Basel, pp. 339–380.

Sontag E.D., Sussmann H.J. 1993: *Time-optimal control of manipulators.* IEEE Int. Conf. Robotics and Automation, San Francisco, April 1986, pp.1692–1697; przedruk [w:] Spong M.W., Lewis F.L., Abdallah C.T. (eds), Robot Control, IEEE Press, New York.

Spong M.W. 1994: *Swing up control of the acrobot.* Proc. IEEE Int. Conf. Robotics and Automation, San Diego, CA, 1994, pp. 2356–2361.

Spong M.W., Vidyasagar M. 1997: *Dynamika i sterowanie robotów.* WNT, Warszawa; przekład. *Robot Dynamics and Control*, John Wiley & Sons, New York.

Steinbach M.C., Bock H.G., Longman R.W. 1995: *Time-optimal extension and retraction of robots: Numerical analysis of the switching structure.* J. Optim. Theory and Appl., vol. 84, No 3, pp. 589–616.

Stoer J., Bulirsch R. 1987: *Wstęp do analizy numerycznej.* PWN, Warszawa.

Stryk O. von 1993: *Numerical solution of optimal control problems by direct collocation.* [in:] Bulirsch R., Miele A., Stoer J., Well K.-H.(eds), Optimal Control – Calculus of Variations, Optimal Control Theory and Numerical Methods, Int. Ser. Num. Math., Birkhäuser, Basel, vol. 111, pp. 129–143.

Stryk O. von, Bulirsch R. 1992: *Direct and indirect methods for trajectory optimization.* Ann. Operations Res., vol. 37, pp. 357–373.

Sussmann H. 1979: *A Bang-bang theorem with bounds on the number of switchings.* SIAM J. Ctrl. & Opt., vol. 17. pp. 629–651.

Szmuc T. 1998: *Zaawansowane metody tworzenia oprogramowania systemów czasu rzeczywistego.* CCATIE Krakowskie Centrum Informatyki Stosowanej, vol. 15.

Szymkat M., Ravn O., Turnau A., Kołek K., Pjetursson A. 1995: *Integrated mechatronic modelling environments.* Proc. Int. Conf. Recent Advances in Mechatronics, ICRAM'95, Istanbul, Turkey, August 14–16 1995, pp. 767–772.

Szymkat M., Turnau A., Uhl T. 1995: *Parameter estimation problem in mechatronic modelling – neural network approach.* Proc. IECPD Int. Conf. Advanced Robotics and Intelligent Automation. Athens, Greece, September 6–8 1995, pp. 203–208.

Szymkat M., Turnau A., Korytowski A. 1997: *Aspekty numeryczne rozwiązywania problemu optymalnoczasowego dla układów nieliniowych.* [w:] Szmuc T., Szymkat M., Tadeusiewicz R., Uhl T. (red.), I Krajowa Konferencja „Metody i systemy komputerowe w badaniach naukowych i projektowaniu inżynierskim". CCATIE Krakowskie Centrum Informatyki Stosowanej, Kraków, 25–26 listopada 1997, pp. 397–404.

Szymkat M., Korytowski A., Turnau A. 1999a: *Computation of time optimal controls by gradient matching.* Proc. IEEE Int. Conf. Contr. Appl., Kohala Coast-Island of Hawai'i, USA, August 22–27 1999, pp. 363–368.

Szymkat M., Korytowski A., Turnau A. 1999b: *Przegląd metod numerycznych rozwiązywania zadań sterowania optymalnego.* II Krajowa Konferencja „Metody i systemy komputerowe w badaniach naukowych i projektowaniu inżynierskim", referat plenarny, Kraków, 25–27 października 1999.

Szymkat M., Korytowski A., Turnau A. 2000: *Variable control parameterization for time-optimal problems.* IFAC CACSD Conf., Salford, September, 2000, 6 p.

Tadeusiewicz R. 1993: *Sieci neuronowe.* Akademicka Oficyna Wydawnicza RM, Warszawa.

Takagi T., Sugeno M. 1985: *Fuzzy identification of systems and its applications to modelling and control.* IEEE Trans. Syst., Man. Cybernet., vol. SMC-15, No 1, pp. 116–132.

Tchoń K., Mazur A., Dulęba I., Hossa R., Muszyński R. 2000: *Manipulatory i roboty mobilne, modele, planowanie ruchu, sterowanie.* Akademicka Oficyna Wydawnicza PLJ, Warszawa.

Timmerman M., Van Beneden B., Uhres L. 1998a: *RTOS evaluations kick off!.* Real-Time Magazine, vol. 3, pp. 6–10.

Timmerman M., Van Beneden B., Uhres L. 1998b: Windows *NT Real-Time Extensions better or worse?.* Real-Time Magazine, vol. 3, pp. 11–19.

Tolat V.V., Widrow B. 1998: *An adaptive 'broom balancer' with visual inputs.* Proc. IEEE Int. Conf. "Neural Net", San Diego, CA, 1988, vol. 2, pp. 641–647.

Trybus L. 1992: *Regulatory wielofunkcyjne.* WNT, Warszawa.

Tsachouridis V.A. 1999: *Robust control of a triple inverted pendulum.* Proc. IEEE Int. Conf. Contr. Appl., Kohala Coast-Island of Hawai'i, USA, August 22–27 1999, pp. 1235–1240.

Turnau A. 1993: *Przykład projektowania z wykorzystaniem pakietu Simulink.* [w:] Szymkat M. (red.), Komputerowe wspomaganie w projektowaniu układów regulacji. WNT, Warszawa.

Turnau A. 1994: *Fuzzy and rule-based controller for cart-pole system on finite rail.* Proc. European Conf. "Modelling and Simulation", Barcelona, June 1–3 1994, pp. 460–464.

Turnau A. 1995: *From a rule-based to a time-suboptimal controller.* Proc. 14th IASTED Int. Conf. "Modelling, Identification and Control", Igls, Austria, February 20–22 1995, pp. 246–249.

Turnau A. 1997: *Aproksymacja regulatora czasooptymalnego i stabilizująca procedura Sontaga.* Prace z Automatyki pod redakcją W. Mitkowskiego, dedykowane Prof. H. Góreckiemu, Wydawnictwa AGH, Kraków, s. 139–142.

Turnau A. 1998: *Prototyping of conventional and intelligent controllers.* EAEEIE 98 Lisbon – Enhancement of Education in Electrical and Information Engineering through Industry Co-operation and Research, Lisbon, Portugal, May 1998, pp. 93–98.

Turnau A., Kołek K., Pauluk M., Rosół M. 1995: *The self learning algorithms. Motion and force control.* Proc. IECPD Int. Conf. "Advanced Robotics and Intelligent Automation", Athens, Greece, September 6–8 1995, pp. 197–202.

Turnau A., Korytowski A. 1996: *Time optimal control of pendulum-cart system.* Sivasundaram S. (ed.), Proc. 1st Int. Conf. "Nonlinear Problems in Aviation and Aerospace", Embry Riddle Aeronautical University Press, Daytona Beach, Florida, USA, May 9–11 1996, pp. 649–654.

Turnau A., Korytowski A. 1997: *Synthesis of time optimal controller for a laboratory pendulum-cart model.* Proc. 16th IASTED Int. Conf. "Modelling, Identification and Control", Innsbruck, February 17–19 1997, pp. 366–371.

Turnau A., Kołek K. 1998: *Time-optimal and variable structure controller.* Proc. Mediterr. Conf. "Electron. Automat. Contr.", Marrakech, Maroc, September 17–19 1998, pp. 476–479.

Turnau A., Korytowski A., Szymkat M. 1999: *Time optimal control for the pendulum cart system in real-time.* Proc. IEEE Int. Conf. Contr. Appl., Kohala Coast–Island of Hawai'i, USA, August 22–27 1999, pp. 1249–1254.

Turnau A., Rosół M., Piątek P. 2001: *Sterowanie serwomechanizmem z generatora PWM skonfigurowanego w logice XILINX z kompensacją zniekształceń.* [w:] Tadeusiewicz R., Ligęza A., Szymkat M., (red.), III Krajowa Konferencja „Metody i systemy komputerowe w badaniach naukowych i projektowaniu inżynierskim", Kraków, 19–21 listopada 2001, s. 285–290.

Turowicz A. 1995: *Teoria macierzy.* Wykłady na Studium Doktoranckim w Zakresie Automatyki i Elektrotechniki w roku akademickim 1970/71, spisał Wojciech Mitkowski, wydanie piąte, Wydawnictwa AGH, Kraków.

Tzes A., Pei-Yuan Peng, Guthy J. 1998: *Genetic-based fuzzy clustering for DC-Motor friction identification and compensation.* IEEE Trans. Contr. Syst. Techn., vol. 6, No 4, pp. 462–472.

Uhl T., Szymkat M. 1992: *A comparison of the classical and neural-based approach to control of manipulation robots.* Conf. Experimental and Numerical Methods in Structural Dynamics, Leuven, Belgium.

Wei Q., Dayawansa W.P., Levine W.S. 1995: *Nonlinear controller for an inverted pendulum having restricted travel.* Automatica, vol. 31, No 6, pp. 841–850.

Widrow N.K., Gupta M.M., Maitra S. 1973: *Punish/reward: Learning with a critic in adaptive threshold systems.* IEEE Trans. Syst., Man, Cybern., vol. 3, pp. 455–465.

Whittle P. 1996: *Optimal Control. Basics and Beyond.* John Wiley & Sons, New York.

Wierzbicki A. 1977: *Modele i wrażliwość układów sterowania.* WNT, Warszawa.

Yen J., Langari R., Zadeh L. 1995: *Industrial applications of fuzzy logic and intelligent systems.* IEEE Press, Inc., New York.

Yurkovich S., Widjaja M. 1996: *Fuzzy controller synthesis for an inverted pendulum system.* Contr. Eng. Practice, vol. 4, No 4, pp. 455–469.

Zadeh L.A. 1965: *Fuzzy sets.* Inform. Contr., vol. 8, No 3, pp. 338–353.

Zhang B., Edmunds J. 1992: *A state learning controller.* Proc. Amer. Contr. Conf., pp. 3057–3061.