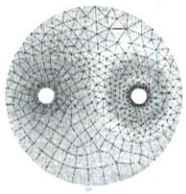
Wydawnictwo
Naukowe
AKAPIT

INFORMATYKA W TECHNOLOGII MATERIAŁÓW

Numer 1, Tom 1, Rok 2001



DOBÓR STRUKTURY SIECI NEURONOWYCH W UKŁADACH STEROWANIA CZASU RZECZYWISTEGO

KRZYSZTOF KOŁEK, WOJCIECH MITKOWSKI

SELECTION OF NEURAL NETWORK STRUCTURE IN REAL TIME CONTROL SYSTEMS

Abstract

In this paper example applications of neural networks for controlling laboratory models are presented. Two structures of neural controllers are considered. The neural networks are applied as neural controllers for three real systems. Finally the remarks on the structure of the neural network are given.

1. WPROWADZENIE

Praca jest kontynuacją badań prezentowanych poprzednio na konferencjach Neuromet'97 (Mitkowski 1997), Neuromet'98 (Kołek i Mitkowski 1998) i Neuromet'99 (Gorczyca, Kołek i Mitkowski 1999). W tym opracowaniu główny nacisk położono na sterowanie w czasie rzeczywistym laboratoryjnych systemów dynamicznych z wykorzystaniem sieci neuronowych oraz na doborze struktury sieci. Specyfika układów sterowania w czasie rzeczywistym wymaga aby reakcja systemu spełniała reżimy czasowe wyznaczone przez sterowany obiekt. Przekroczenie narzuconych przez obiekt ograniczeń czasowych jest niedopuszczalne (może doprowadzić do zniszczenia obiektu) lub nadmiernie kosztowne. Idea zastosowania sztucznych sieci neuronowych w układach sterowania jest niezwykle kusząca w świetle licznych udanych zastosowań (Kołek 1996; Kołek, Tabakowski i Turnau 1993, Obuchowicz 2000, Kaczorek i Nowak 2000, Derbel 2000, Fujarewicz 2000, Grymei i Kiczkowiak 2000, Kempa, Kołek, Tabakowski i Turnau 1997). Zastosowanie sieci do rzeczywistych obiektów sterowanych w czasie rzeczywistym wymaga wyboru minimalnej struktury sieci (a więc wymagającej minimalnych nakładów obliczeniowych) gwarantującej poprawne działanie. Tylko takie podejście zostawia bezpieczny margines czasu obliczeń wymagany w rzeczywistych eksperymentach.

W pracy przedstawiono aplikacje sieci neuronowych do sterowania wybranymi rzeczywistymi obiektami. Przebadano wpływ struktury sieci na czas uczenia oraz jakość interpolacji. Wyniki eksperymentów posłużyły do sformułowania algorytmu doboru struktury sieci neuronowej zapewniającej zadawalającą jakość interpolacji przy minimalnej liczbie kroków obliczeniowych.

2. UKŁADY STETROWANIA WYKORZYSTUJĄCE SIECI NEURONOWE

Istnieją dwie typowe konfiguracje zastosowań sieci neuronowych do celów sterowania:

K. Kołek, W. Mitkowski, Katedra Automatyki, Akademia Górniczo-Hutnicza, Kraków
kko@aquarium.ia.agh.edu.pl, wmi@aquarium.ia.agh.edu.pl

ISSN 1641-8581

- wykorzystanie nabytej innymi metodami wiedzy do nauczenia sieci neuronowej zasad pracy regulatora. W takiej konfiguracji sieć neuronowa pracuje jako interpolator regułowy (Kołek 1996; Kołek, Tabakowski i Turnau 1993),
- nauczenie sieci odwrotnej charakterystyki obiektu (w szczególności odwrotnej dynamiki) i wykorzystanie takiej sieci do sterowania obiektem (Kołek i Mitkowski 1998, Gorczyca, Kołek i Mitkowski 1999).

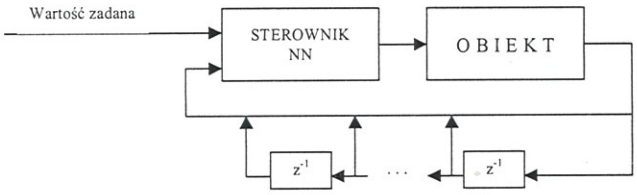
Obie wymienione metody mogą wykorzystywać sieci, które zostały uprzednio nauczone w trybie off-line lub mogą być uczone w trakcie eksperymentów. Nauczanie sieci jest procesem wymagającym dużego nakładu obliczeń w związku z czym zastosowanie w trybie on-line jest możliwe tylko dla wolnych obiektów. Z kolei zaniechanie nauczania w trakcie przeprowadzania eksperymentów uniemożliwia reakcję sieci na zmieniające się warunki eksperymentu (a w szczególności na zmieniające się parametry obiektu).

2.1. Sterownik jako interpolator regułowy

W przypadku złożonych nieliniowych obiektów najczęściej nie istnieją ścisłe matematyczne formuły pozwalające na syntezę regulatora. W takich przypadkach można wykorzystać wiedzę o zachowaniu obiektu w różnych obszarach pracy do określenia reguł sterowania w tych obszarach, a następnie podjąć próbę syntezy regulatora pokrywającego cały zakres pracy obiektu. Inne podejście wykorzystuje matematyczny model sterowanego obiektu. Na podstawie modelu można przeprowadzić szereg eksperymentów symulacyjnych, których wyniki posłużą do określenia zasad sterowania obiektem w konkretnych punktach przestrzeni stanu.

W obu przypadkach posiadana fragmentaryczna wiedza ma posłużyć do syntezy kompletnego regulatora. Jednym z mechanizmów interpolacji posiadanej informacji na całą przestrzeń roboczą obiektu jest zastosowanie sieci neuronowej. Wiedza zdobyta na podstawie doświadczenia lub eksperymentów symulacyjnych służy do nauczenia sieci neuronowej. Sieć po cyklu nauczania interpoluje wiedzę z ciągu uczącego pokrywając całą przestrzeń roboczą obiektu.

Przykładowa konfiguracja układu sterowania przedstawiona została na rysunku 1. Wejściami sieci neuronowej są wartość zadana oraz wyjścia z obiektu. W przypadku obiektów dynamicznych koniecznym może się okazać podanie na wejście sieci neuronowej wyjść obiektu opóźnionych o jeden lub kilka okresów próbkowania.



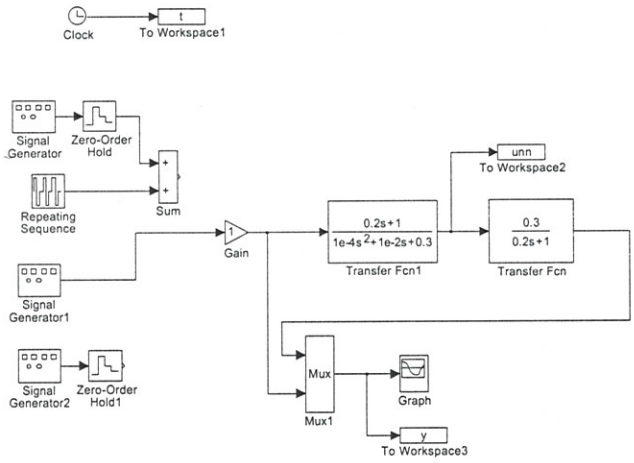
Rysunek 1. Układ sterowania ze sterownikiem neuronowym.

2.2. Sterownik jako model dynamiki odwrotnej

Modelowaniem neuronowym dynamiki odwrotnej zajmowało się wielu autorów, np. Lairi i Bloch (1998), Farrell (1997). Jeżeli modelem matematycznym układu dynamicznego jest transmitancja $G(s)$, to istotą metody jest zamodelowanie transmitancji odwrotnej $G(s)^{-1}$. Połączenie szeregowe członu $G(s)^{-1}$ i $G(s)$ daje transmitancję $G(s)^{-1} * G(s) = 1$, co oznacza, że sygnał wyjściowy może dokładnie odtwarzać zadany przebieg wejściowy (zob. np. Mitkowski 1997b, s. 57; Żurada i inni 1996, s. 304). Problem w tym, że transmitancja $G(s)^{-1}$ może nie być realizowalna fizycznie.

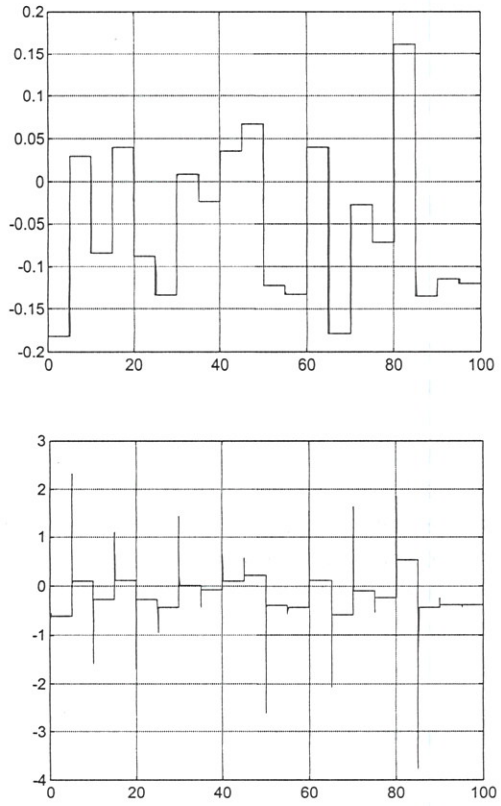
Dla przykładu rozważmy układ dynamiczny rzędu drugiego (Kołek i Mitkowski 1998, s. 49), gdzie $G(s) = \frac{0.3}{0.2s+1}$ i zatem $G(s)^{-1} = \frac{0.2s+1}{0.3}$ jest transmitancją nie realizowalną fizycznie, bo nie jest możliwe idealne różniczkowanie (operatorowi s w dziedzinie czasu odpowiada różniczkowanie po czasie). Praktycznie dobrym przybliżeniem transmitancji odwrotnej może być np. wyrażenie:

$\tilde{G}(s)^{-1} = \frac{0.2s+1}{0.0001s^2 + 0.01s + 0.3}$, które ma praktyczną realizację.

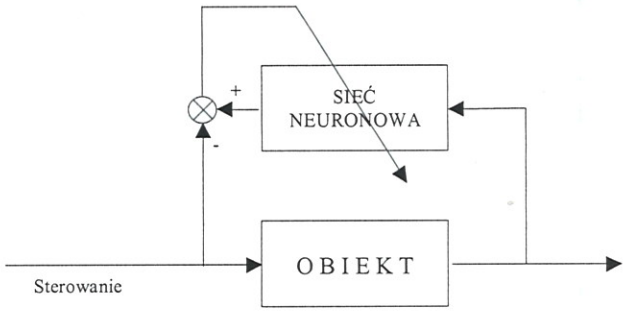


Rysunek 2. Układ z przybliżoną dynamiką odwrotną.

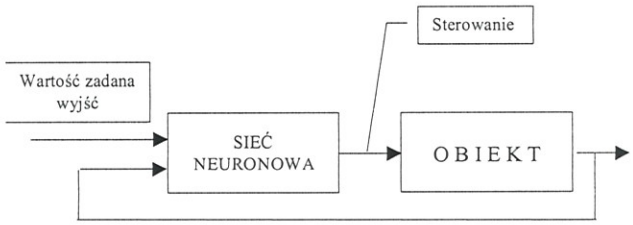
Na rys. 2 przedstawiono układ z przybliżoną dynamiką odwrotną. Na rys. 3 pokazano przebieg wyjściowy nadążający prawie idealnie za odpowiednią falą prostokątną (rys. górny) i sterowanie (przybliżona pochodna fali realizowana przez $\widetilde{G}(s)^{-1}$) na wejściu transmitancji $G(s)$.



Rysunek 3. Odpowiedź na sygnał zadany (rys. górny) i przybliżone różniczkowanie (rys. dolny).

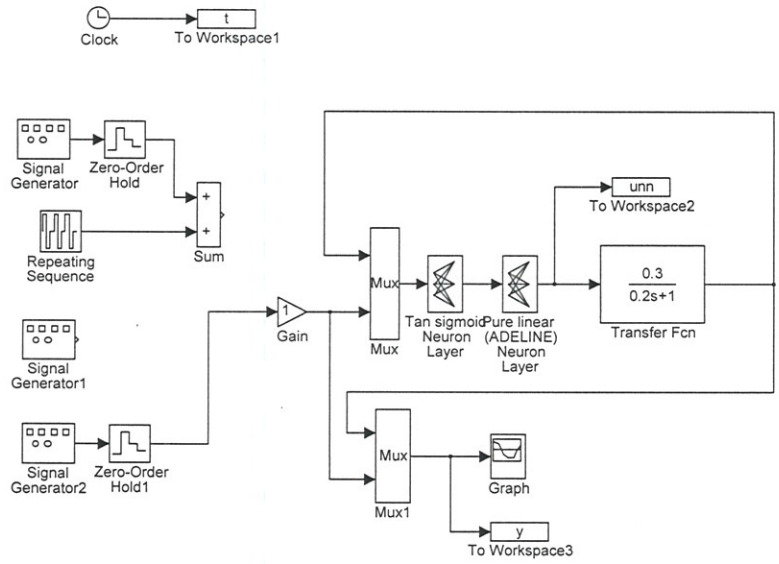


Rysunek 4. Nauczanie dynamiki odwrotnej.



Rysunek 5. Sterowanie oparte o dynamikę odwrotną.

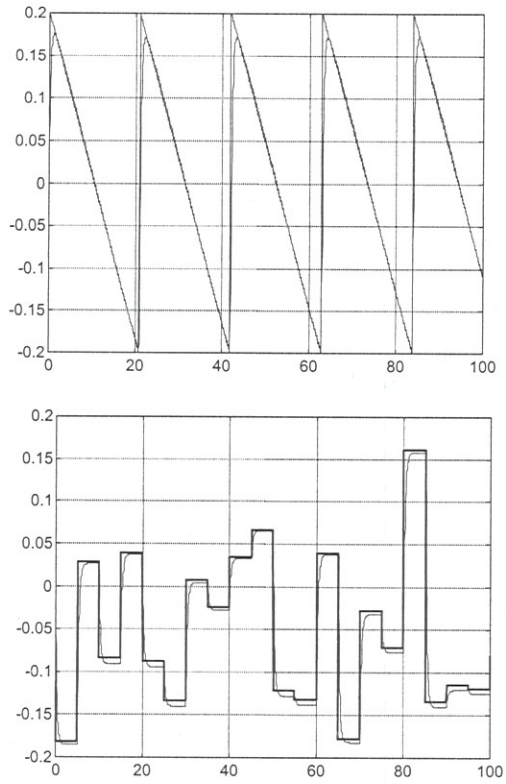
Idea nauczania sieci neuronowej dynamiki odwrotnej przedstawiona została na rysunku 4. Obiekt pobudzany jest znanym ciągiem sterowań. Wyjścia obiektu (wraz z ewentualnymi wyjściami opóźnionymi o jedno lub kilka okresów próbkowania) podawane są na wejście sieci neuronowej. Wyjście sieci neuronowej porównywane jest z wejściowym sterowaniem. Sygnał błędu między wejściowym sterowaniem a wyjściem z sieci neuronowej służy do przestrajania wag sieci. Celem nauczania jest otrzymanie takiej sieci neuronowej, aby w odpowiedzi na zadane wyjścia z obiektu potrafiła odtworzyć sterowanie, którego skutkiem są te wyjścia.



Rysunek 6. Model neuronowy nauczony dynamiki odwrotnej.

Wykorzystanie sieci neuronowej nauczonej charakterystyki odwrotnej obiektu do celów sterowania przedstawia rysunek 5. Na wejścia sieci neuronowej podawana jest wartość zadana wyjść obiektu. Sieć neuronowa generuje na swoim wyjściu sterowanie, które powinno doprowadzić wyjścia obiektu do stanu takiego jak wejścia sieci neuronowej.

Rys. 6 przedstawia model neuronowy nauczony dynamiki odwrotnej przykładowego systemu liniowego.



Rysunek 7. Nadążanie za "piłą" i falą prostokątną "przypadkową".

Natomiast na rys. 7 pokazano odpowiedzi nauczonego dynamiki odwrotnej układu neuronowego na sygnał typu "piła" oraz falę prostokątną "przypadkową". Sieć neuronowa była uczona wyłącznie na podstawie pobudzania obiektu falą prostokątną.

Podobne podejście można zastosować w przypadku dyskretnych obiektów nieliniowych. Ogólnie obiekt taki opisywany jest formułą:

$$x(n+1) = f(u(n), x(n), x(n-1), ..., x(n-k))$$

Dla takiej formuły można poszukać przybliżenia funkcji odwrotnej względem zmiennej u(n) w postaci:

$$u(n) = \tilde{f}^{-1}(x(n+1), x(n), x(n-1), ... x(n-k))$$

Przedstawia ona zależność sterowania od stanu obiektu w chwilach poprzednich x(n-1), ... x(n-k), chwili aktualnej x(n) oraz w chwili następnego okresu próbkowania x(n+1). Wartość x(n+1) to właśnie wartość zadana, która jest celem sterowania.

3. APLIKACJE UKŁADÓW STEROWANIA DLA WYBRANYCH RZECZYWISTYCH MODELI LABORATORYJNYCH

Omówione w poprzednim rozdziale sposoby zastosowania sieci neuronowych zostały w praktyce wykorzystane do sterowania w czasie rzeczywistym trzema obiektami laboratoryjnymi znajdującymi się w Katedrze Automatyki Akademii Górniczo-Hutniczej.

Praktyczne eksperymenty z zastosowaniem sieci neuronowych realizujących funkcje dynamiki odwrotnej przeprowadzono z wykorzystaniem pakietu oprogramowania RT-CON (Grega, Kołek i Turnau 1998). Pakiet ten pozwala wykorzystać modele programu SIMULINK do graficznej kompozycji układu regulacji oraz do automatycznej generacji programu czasu rzeczywistego realizującego zaprojektowany sterownik. Program SIMULINK umożliwia wykorzystanie różnorodnych bloków funkcyjnych w projektowanym układzie regulacji. Należą do nich między innymi bloki regulatorów PID oraz bloki realizujące wielowarstwowe sieci neuronowe. Wygenerowane programy czasu rzeczywistego pracują w środowiskach systemów operacyjnych WINDOWS 95/98/NT umożliwiając łatwy transfer danych między zadaniem czasu rzeczywistego a środowiskiem programu MATLAB.

3.1. Wahadło odwrócone

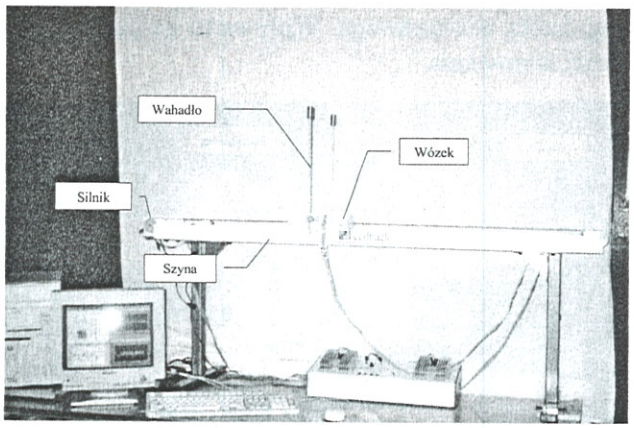
Model wahadła odwróconego przedstawiono na rysunku 8. Zestaw składa się z silnika prądu stałego napędzającego wózek poruszający się po poziomej szynie. Do wózka przymocowane jest swobodnie poruszające się wahadło. Zestaw podłączony jest do komputera w taki sposób, że możliwy jest pomiar położenia wózka, pomiar kąta wahadła oraz sterowanie silnikiem. Celem sterowania jest wyprowadzenie wahadła z dowolnego położenia do położenia górnego punktu niestabilnej równowagi (Rys. 8) i utrzymanie go w tym punkcie. Tak postawiony problem jest nietrywialny, bowiem układ posiada istotne nieliniowości. Jedynie wokół górnego punktu równowagi możliwe jest zastosowanie liniowego przybliżenia i ścisłe matematyczne wyliczenie sterowania jako liniowego regulatora od stanu.

W celu doprowadzenia układu w pobliże górnego punktu równowagi wykorzystano model matematyczny obiektu. Przestrzeń stanu obiektu zawiera cztery zmienne (położenie wózka x_1, prędkość wózka x_3, kąt wahadła x_2 oraz prędkość wahadła x_4). Ponieważ sterowanie wahadłem nie zależy od aktualnego położenia wózka z przestrzeni stanu wybrano kąt i prędkość wahadła. Tak otrzymaną podprzestrzeń podzielono na

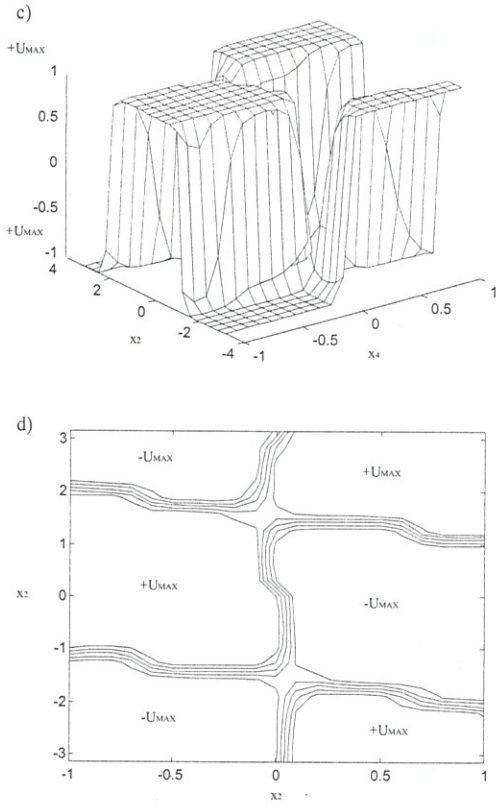
siatkę punktów. W każdym punkcie w sposób symulacyjny obliczono:

- sterowanie maksymalizujące sumę energii kinetycznej i potencjalnej wahadła oraz
- sprawdzono, czy energia wahadła wystarcza do osiągnięcia górnego punktu równowagi.

Maksymalizacja energii ma na celu rozhuśtanie

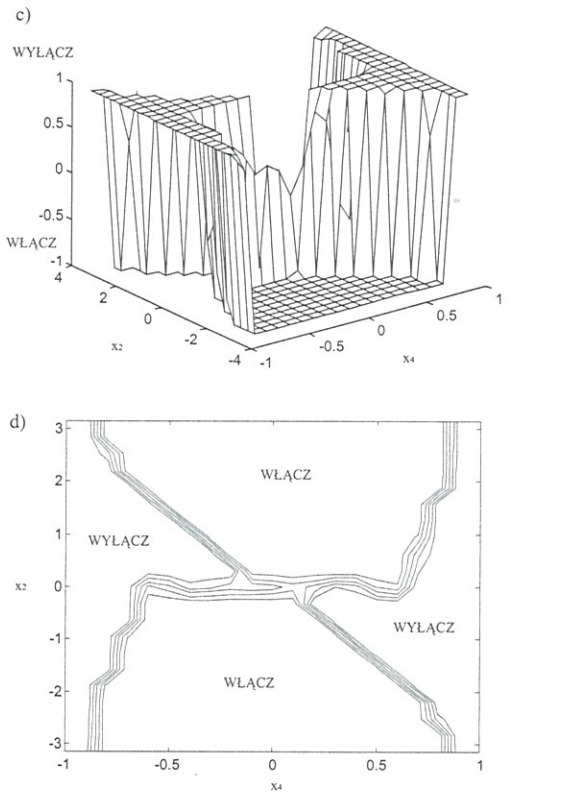


Rysunek 8. Model wahadła odwróconego.



Rysunek 9. Linie przełączeń dla algorytmu rozhuśtywania.

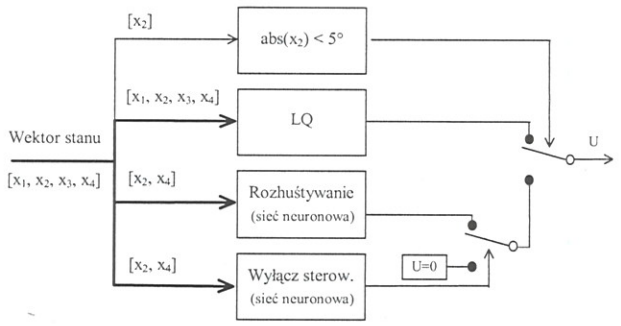
wahadła. Test sumarycznej energii pozwala w łagodny sposób "wylądować" w pobliżu górnego punktu równowagi. Tak przygotowana informacja w punktach siatki stanowiła ciąg uczący dla dwóch sieci neuronowych. Wyniki nauczania algorytmu rozhuśtywania przedstawia rysunek 9. Rysunek 10 przedstawia wyniki nauczania sieci neuronowej testu osiągnięcia su-



Rysunek 10. Linie przełączeń dla algorytmu łagodnego lądowania.

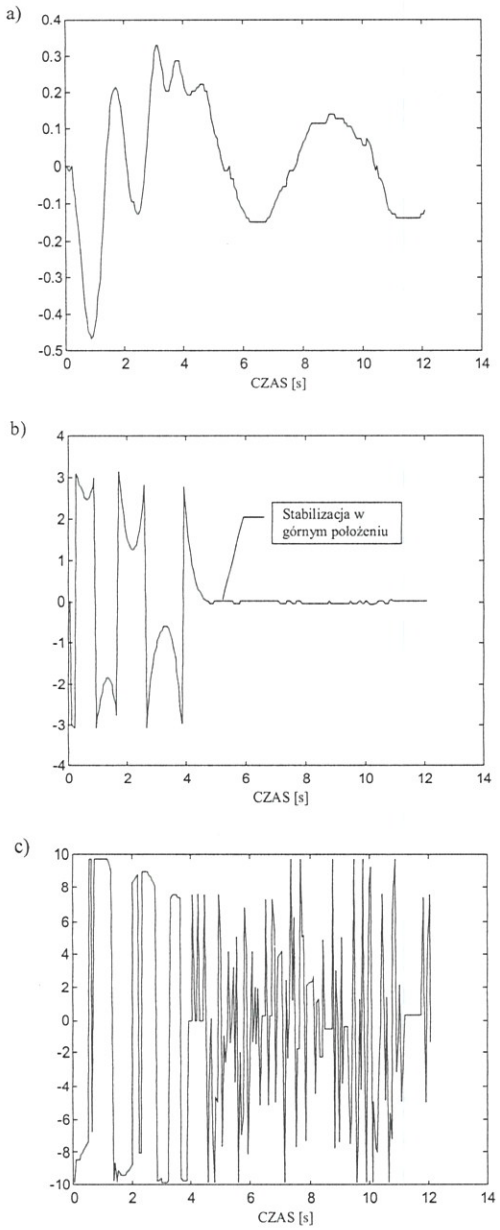
marycznej energii wystarczającej na osiągnięcie górnego punktu równowagi (Kołek 1996).

Struktura regulatora układu przedstawiona jest na rysunku 11. Jeżeli kąt wahadła jest mniejszy od 5° włączany jest liniowy regulator od stanu stabilizujący układ wokół górnego punktu równowagi. Jeżeli nie, włączane jest sterowanie wypracowywane przez dwie sieci neuronowe. Sieć wyłączająca sterowanie spraw-



Rysunek 11. Regulator wahadła odwróconego.

dza, czy energia wahadła osiągnęła poziom wystarczający na osiągnięcie górnego punktu równowagi. Jeżeli tak, sterowanie ustawiane jest na wartość równą zeru. Jeżeli nie, wówczas druga sieć wypracowuje sterowanie rozhuśtujące wahadło. Wyniki pracy algorytmu przedstawia rysunek 12.

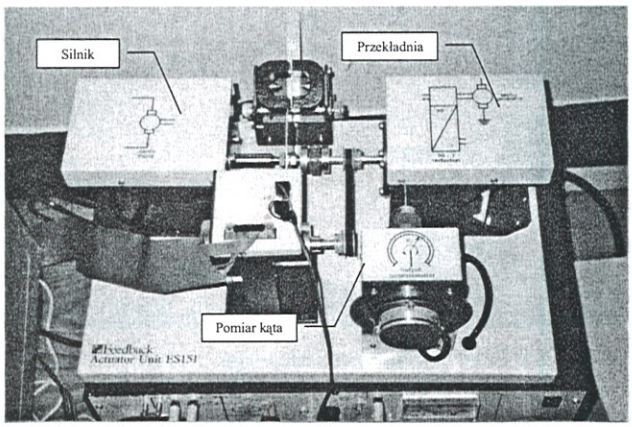


Rysunek 12. Wyniki pracy algorytmu sterowania wahadła odwróconego: a) położenie wózka, b) kąt wahadła, c) wartość sterowania.

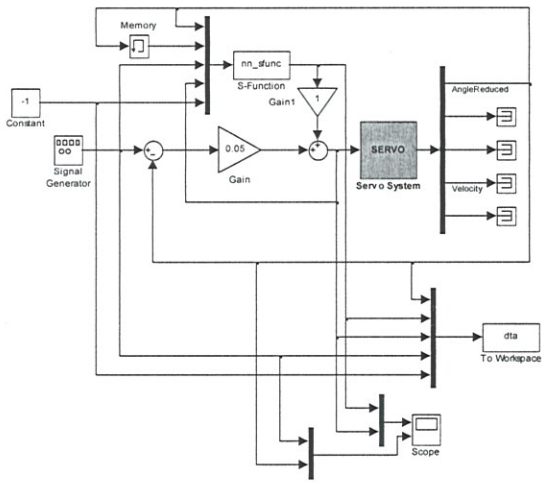
3.2. Sterowanie serwomechanizmem

Przedstawiony na rysunku 13 model serwomechanizmu składa się z silnika prądu stałego, hamulca magnetycznego, przekładni oraz miernika kąta wału po przekładni. Celem starowania jest śledzenie przez wał zadanej trajektorii lub stabilizacja prędkości obrotowej.

Regulator obiektu przedstawia rysunek 14. Regulator składa się z dwóch regulatorów pracujących równolegle. Pierwszy z nich to zwykły regulator proporcjonalny (blok *Gain* na rys. 14). Drugi z nich to regulator neuronowy (blok *nn_sfunc S-function*). Wyjścia z obu regulatorów są sumowane i podawane na obiekt. Na początku eksperymentu waga regulatora proporcjonalnego jest znacząco większa w stosunku do wagi od regulatora neuronowego. Regulator proporcjonalny nie zapewnia dobrej jakości sterowania ale utrzymuje obiekt w bezpiecznym obszarze pracy. Jednocześnie produkuje ciągi uczące służące do nauczenia sieci neuronowej odwrotnej dynamiki serwomechanizmu. W miarę upływu czasu waga od regulatora proporcjonalnego jest zmniejszana na rzecz regulatora neuronowego. Wpływa to na poprawę jakości sterowania.



Rysunek 13. Model serwomechanizmu.



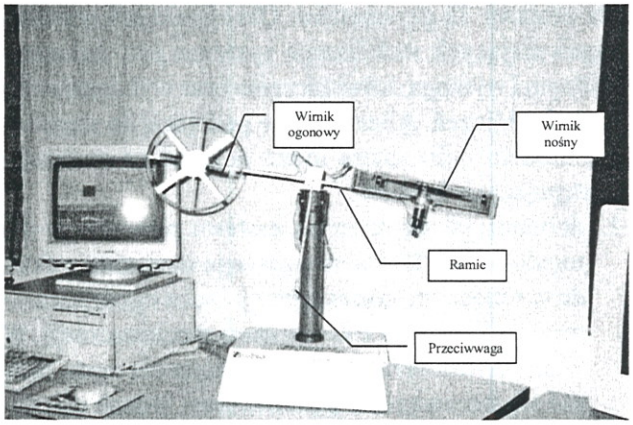
Rysunek 14. Układ sterowania serwomechanizmu.

3.3 Sterowanie modelem helikoptera

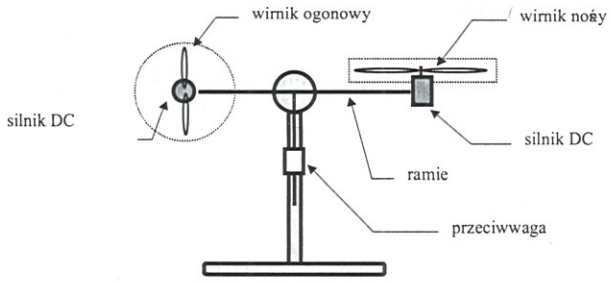
W Laboratorium Katedry Automatyki AGH zbudowano stanowisko badawcze (zob. rys. 15 oraz rys. 16) pozwalające badać pewne aspekty zachowania się śmigłowca w fazie zwisu (Gorczyca, Hajduk i Kołek 1995). Jest to obiekt o złożonej dynamice. Korzystając ze stanowiska badawczego składającego się z aerodynamicznego modelu dynamiki śmigłowca sterowanego w czasie rzeczywistym przez komputer

PC-IBM zostały przeprowadzone różnego typu badania symulacyjne. Zadaniem sterowania było zapewnienie właściwych przebiegów ruchu osi pionowej oraz poziomej.

Na rys. 16 przedstawiono schemat aerodynamicz-



Rysunek 15. Model helikoptera.



Rysunek 16. Schemat aerodynamicznego modelu śmigłowca.

nego modelu śmigłowca. Na końcach przegubowo zawieszonej belki znajdują się dwa silniki prądu stałego wyposażone w śmigła. Przegub pozwala na złożony ruch belki w dwóch osiach. W ten sposób końce belki mogą poruszać się po powierzchni kuli w przestrzeni. Do belki przymocowana jest przeciwwaga stabilizująca położenie belki w przestrzeni. Układ jest tak wyważony że, przy zatrzymanych silnikach układ jest pochylony w stronę wirnika głównego. W układzie wielkościami sterującymi są napięcia doprowadzane do silników. Mierzone są położenia belki w przestrzeni oraz prędkości obrotowe wirników. Prędkości belki w płaszczyźnie poziomej i pionowej odtwarzane są programowo poprzez różniczkowanie i filtrację zmierzonych funkcji położeń.

Uproszczony model matematyczny posiada postać równań różniczkowych następującego typu:

$$\frac{dS_v}{dt} =$$

$$\frac{d\alpha_v}{dt} = \Omega_v, \qquad \Omega_v = S_v + \frac{J_{tr}\,\omega_t}{J_v},$$

$$\frac{l_m F_v(\omega_m) - \Omega_v k_v + g\left((A-B)\cos\alpha_v - C\sin\alpha_v\right) - \frac{1}{2}\Omega_h^2 (A+B+C)\sin 2\alpha_v}{J_v}$$

$$\frac{dS_h}{dt} = \frac{l_t F_h(\omega_t)\cos\alpha_v - \Omega_h k_h}{J_h} =$$

$$\frac{l_t F_h(\omega_t)\cos\alpha_v - \Omega_h k_h}{D\,\sin^2\alpha_v + E\cos^2\alpha_v + F}$$

$$\frac{d\alpha_h}{dt} = \Omega_h$$

$$\Omega_h = S_h + \frac{J_{mr}\,\omega_m\cos\alpha_v}{J_h} =$$

$$S_h + \frac{J_{mr}\,\omega_m\cos\alpha_v}{D\sin^2\alpha_v + E\cos^2\alpha_v + F},$$

$$U = \begin{bmatrix} u_h \\ u_v \end{bmatrix} \text{ wejście,}$$

$$X = \begin{bmatrix} S_h \\ \alpha_h \\ u_{hh} \\ S_v \\ \alpha_v \\ u_{vv} \end{bmatrix} \text{ stan układu,} \qquad Y = \begin{bmatrix} \Omega_h \\ \alpha_h \\ \omega_t \\ \Omega_v \\ \alpha_v \\ \omega_m \end{bmatrix} \text{ wyjście}$$

gdzie wybrane wielkości oznaczają:

α_v – kąt podniesienia belki ponad położenie poziome, $\frac{d\alpha_v}{dt} = \Omega_v$,

α_h – położenie horyzontalne belki, $\frac{d\alpha_h}{dt} = \Omega_h$,

ω_m – prędkość obrotowa wirnika głównego,

ω_t – prędkość obrotowa wirnika ogonowego.

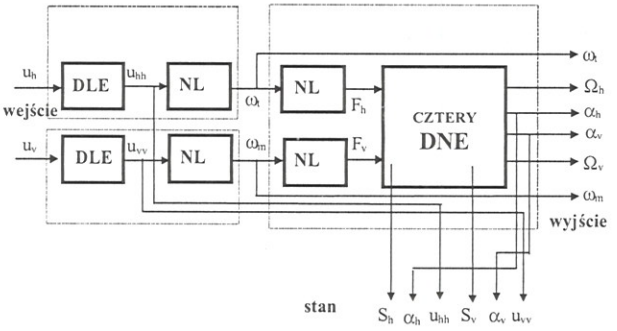
S_v – moment pędu w osi poziomej,

S_h – moment pędu w osi pionowej,

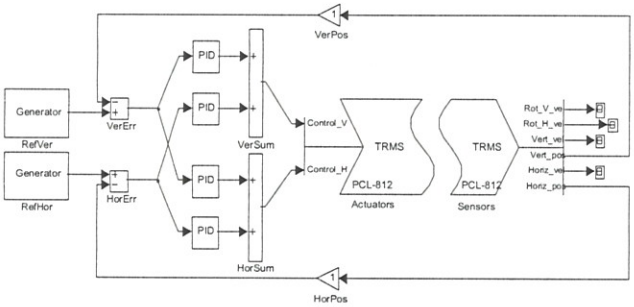
u_{hh}, u_{vv} – stany wewnętrzne związane z dynamiką układu śmigło-silnik.

Na rys. 17 pokazano schemat blokowy modelu matematycznego rozważanego "helikoptera". Bloki DL oznaczają liniowe równania różniczkowe, bloki NL oznaczają różnego typu nieliniowości, a bloki DNE oznaczają nieliniowe równania różniczkowe.

Rysunek 18 przedstawia model Simulink-a wykorzystany do generacji programu czasu rzeczywistego realizującego sterowanie modelem helikoptera za pomocą sprzężonych regulatorów PID. Blok *Actuators* zawiera sterowniki umożliwiające sterowanie silnikami DC. Blok *Sensors* zawiera sterowniki odpowiedzial-



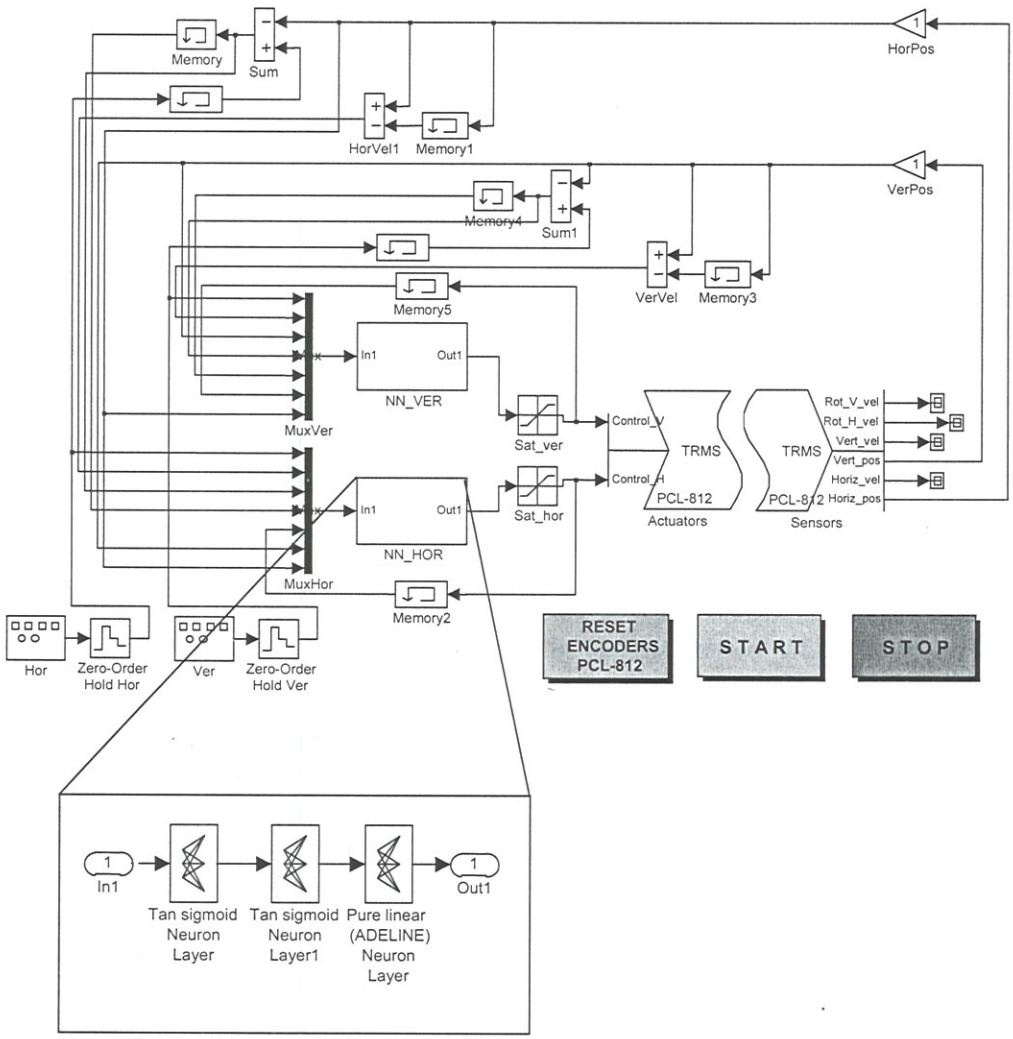
Rysunek 17. Schemat blokowy modelu helikoptera.



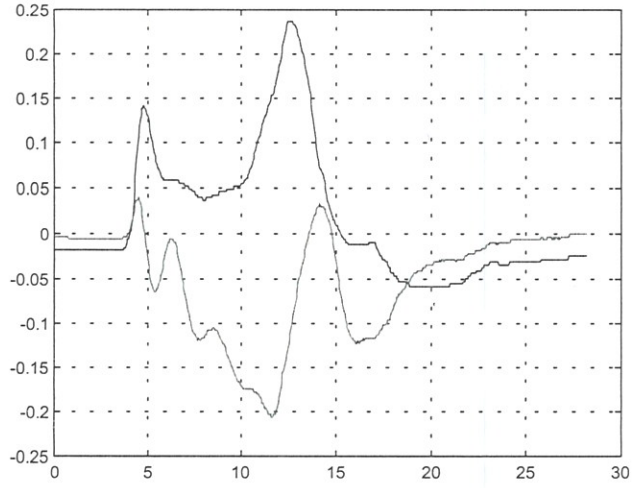
Rysunek 18. Sterowanie modelem "helikoptera" za pomocą regulatorów PID.

ne za komunikacje z układem pomiarowym. Zmierzone pozycje w osi pionowej i poziomej porównywane są z sygnałami referencyjnymi tworząc sygnał błędu stanowiący wejścia dla regulatorów PID. Wykorzystano cztery regulatory PID realizujące sterowania w dwóch osiach. Struktura regulatorów wymaga podania zestawu 16 parametrów. Określenie ich wartości zapewniających stabilizację systemu jest zadaniem trudnym i wymaga doświadczenia oraz wykonania złożonych obliczeń. Alternatywne podejście do sterowania modelu helikoptera może polegać na realizacji następujących kroków:

1. dobraniu jakiegokolwiek zestawu parametrów dla regulatorów PID zapewniającego utrzymanie układu w roboczym zakresie pracy (poza ograniczeniami wynikającymi z konstrukcji mechanicznej). W wielu zastosowaniach wystarczającym może okazać się sterowanie w pętli otwartej, jednak mechaniczna konstrukcja modelu helikoptera wyklucza sterowanie w pętli otwartej,
2. przeprowadzeniu eksperymentu i zebraniu informacji o wejściach i wyjściach systemu. W trakcie eksperymentu zakres wejść i wyjść powinien zmie-



Rysunek 19. Model programu Simulink wykorzystany do generacji regulatora neuronowego.



Rysunek 20. Położenie pionowe i poziome - jedno ręczne zaburzenie. Zadaniem sterowania było utrzymanie obu kątów na wartości zero.

niać się w jak najszerszym zakresie, jednak nie jest istotne aby układ nadążał za wartością zadaną,

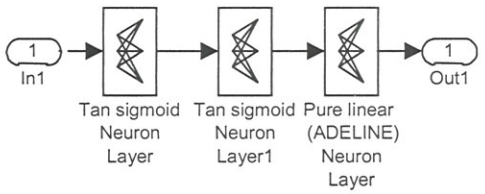
3. wybranie struktury sieci neuronowej oraz przeprowadzenie nauczania dynamiki odwrotnej układu (Lairi i Bloch 1998),
4. zastosowanie nauczonej sieci neuronowej do realizacji praktycznych eksperymentów stabilizacji położenia belki modelu helikoptera.

Rys. 19 przedstawia model programu Simulink wykorzystany do automatycznej generacji zadania czasu rzeczywistego realizującego sterowanie neuronowe. Zastosowano dwa niezalezne regulatory neuronowe po jednym dla każdego stopnia swobody.

Rys. 20 przedstawia odpowiedź układu z neurosterownikiem na ręczne zaburzenie położenia belki. Układ sterowania doprowadza model do wartości zadanej.

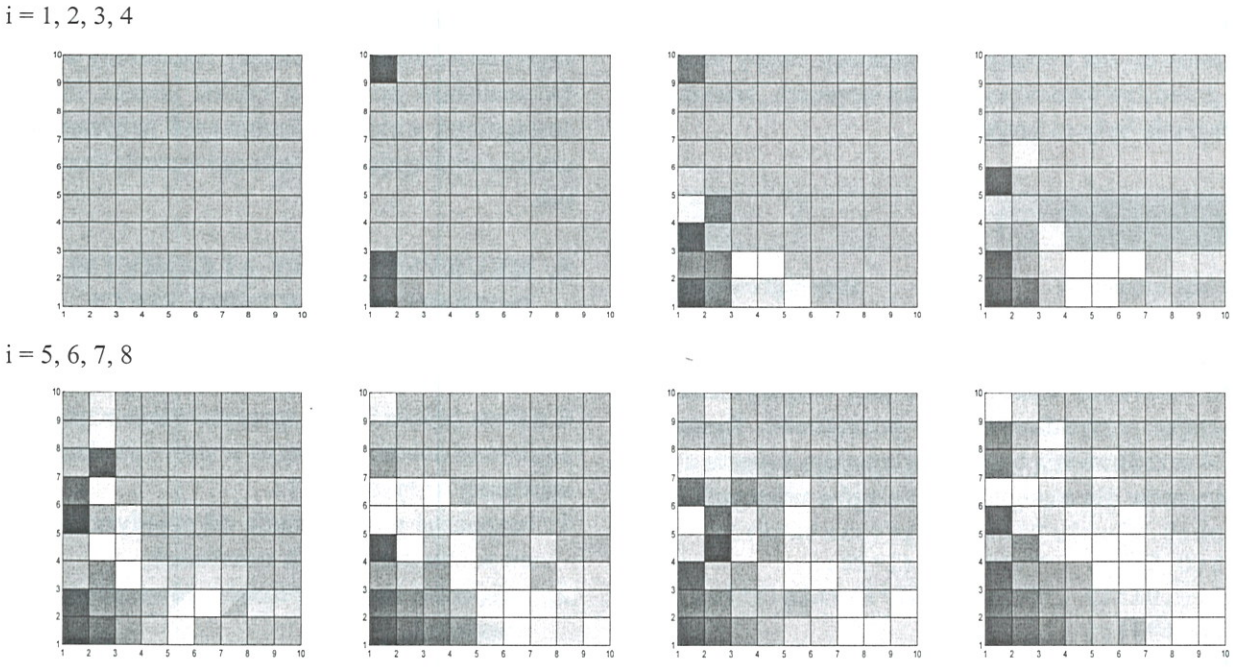
4. STRUKTURA SIECI NEURONOWEJ A DOKŁADNOŚĆ INTERPOLACJI ORAZ CZAS NAUCZANIA

W zadaniach sterowania w czasie rzeczywistym



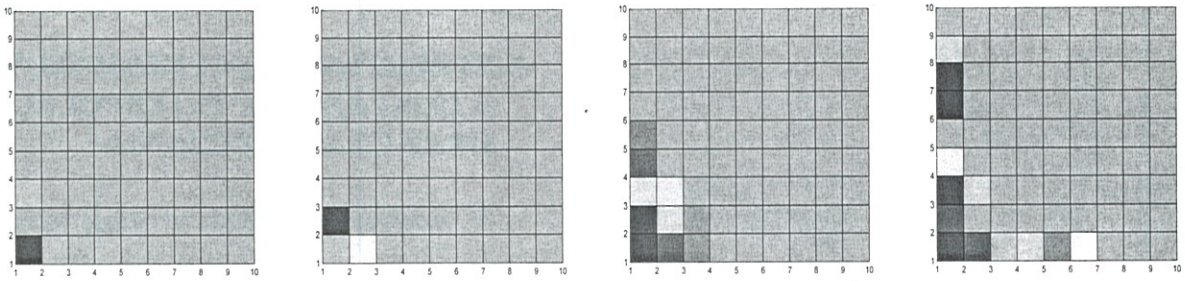
Rysunek 21. Struktura sieci wykorzystywana w trakcie obliczeń.

szczególnie istotny jest nakład obliczeniowy niezbędny do obliczenia wyjścia sterownika. Czas jest zasobem, który w systemach czasu rzeczywistego jest limitowany. W przedstawionych w rozdziale 3 przykładach przekroczenie zadanego okresu próbkowania wynoszącego około 10 milisekund powoduje, że cel sterowania nie jest możliwy do osiągnięcia. Wykorzystane sieci neuronowe powinny charakteryzować się minimalnym rozmiarem gwarantującym osiągnięcie zadanego poziomu dokładności interpolacji. Rodzi się pytanie jak liczba neuronów w strukturze cieci przedstawionej na rysunku 21 wpływa na dokładność oraz na czas nauki. Do badań wykorzystano sieć neuronową o jednym wejściu i jednym wyjściu. W pierw-

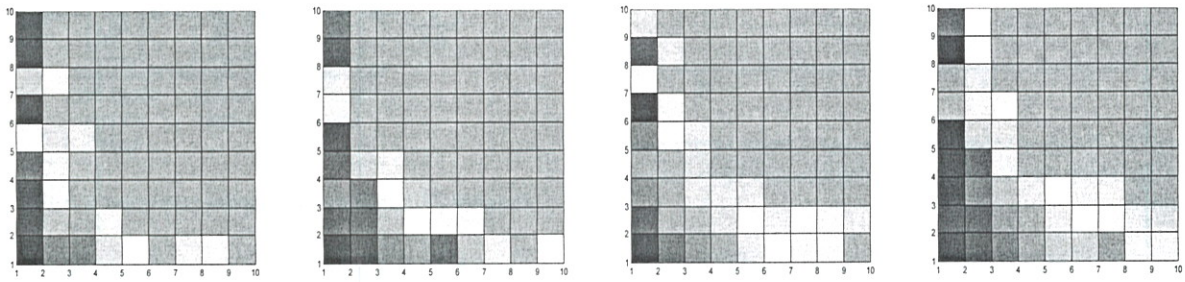


Rysunek 22. Błąd interpolacji ciągu funkcji; `P = -pi:0.01:pi; T(i) = sign(sin(i*P));`

i = 1, 4, 7, 10



i = 16, 19, 22, 25



Rysunek 23. Błąd interpolacji ciągu funkcji `P = -1:0.01:1; T(i) = sin(i*P);`

szych dwóch warstwach funkcja progowa to tangens hiperboliczny. W trzeciej warstwie funkcja progowa jest liniowa. Sieć zastosowano do odtwarzania wartości ciągu funkcji jednej zmiennej. Pojedyncza seria eksperymentów polegała na próbie odtwarzania funkcji o zwiększającym się stopniu skomplikowania. Jednocześnie zmieniano liczby neuronów w pierwszych dwóch warstwach.

Przykładowe wyniki przedstawiono na rysunkach 22–24. Na osiach przedstawiono liczby neuronów w dwóch pierwszych warstwach. Jakość interpolacji (błąd między wartościami funkcji a wyjściem z sieci neuronowej) dla kolejnych kombinacji liczby neuronów przedstawiono odpowiednim stopniem „szarości" (w orginale były to kolory).

Duży błąd interpolacji — Mały błąd interpolacji

Dla każdej kombinacji liczby neuronów w pierwszych warstwach wykonano po pięć eksperymentów dobierając przypadkowe wagi początkowe. Rysunki 22–24 przedstawiają wyniki najlepszego z pięciu eksperymentów dla następujących ciągów uczących:

```
P=-pi:0.01:pi;T(i)= sign(sin(i*P));
P=-1:0.01:1;  T(i)=sin(i*P);
P=-1:0.01:1;  T(i)=sin(2*P)+sin(3*P)+ ..
+ sin(i*P);
```

Zgodnie z intuicją wyniki eksperymentów pokazują, że wzrost liczby neuronów powoduje poprawę jakości interpolacji. Dodatkowo jeżeli rozpocząć poszukiwania od pojedynczego neuronu w warstwie pierwszej i drugiej to równomierny wzrost liczby neuronów w obu warstwach powoduje osiągnięcie zadanego poziomu jakości interpolacji w najmniejszej liczbie kroków. Dodatkowo zauważono, że wraz ze wzrostem liczby neuronów staje się coraz trudniejsze wygenerowanie warunków początkowych gwarantujących zadany poziom jakości nauczania w zadanej liczbie epok.

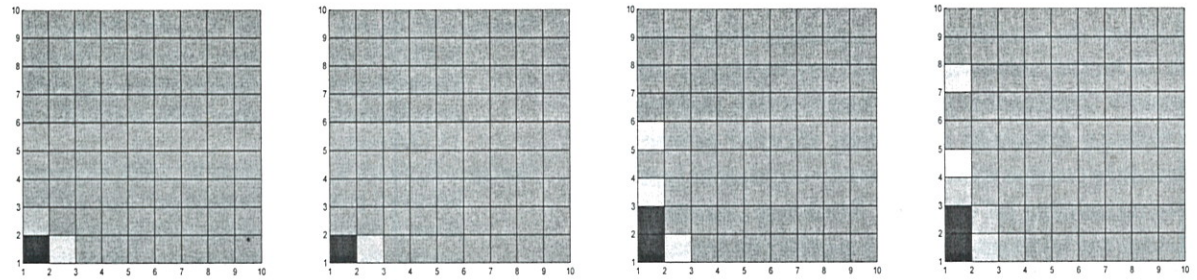
Rysunek 25 przedstawia jakość nauczania w funkcji liczby neuronów w drugiej warstwie oraz czas nauczania. Jakość nauczania przedstawiona została jako logarytm z funkcji kryterialnej. Nadmierny wzrost liczby neuronów powoduje spadek jakości interpolacji (osiągniętej w zadanej liczbie epok nauczania) jednocześnie wymagając radykalnego wzrostu czasu nauczania.

5. HEURYSTYCZNY ALGORYTM DOBORU STRUKTURY SIECI

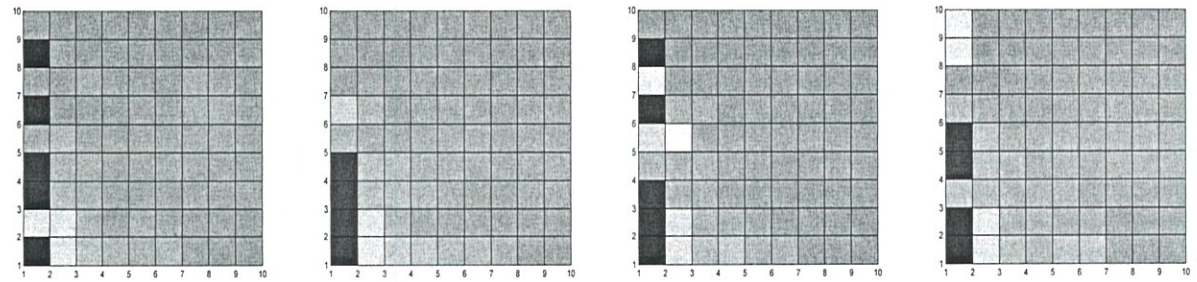
W świetle wyników przedstawionych w rozdziale 4 można pokusić się o sprecyzowanie następującego algorytmu doboru neuronów w sieci o strukturze przedstawionej na rysunku 21:

1. rozpocząć poszukiwania od pojedynczego neuronu w obu pierwszych warstwach,
2. zwiększać liczby neuronów równomiernie w obu warstwach,
3. po osiągnięciu satysfakcjonującego wskaźnika pracy sieci bezwzględnie zaprzestać zwiększania liczby neuronów w warstwach sieci.

i = 2, 4, 7, 10



i = 16, 19, 22, 25



Rysunek 24. Błąd interpolacji ciągu funkcji `P = -1:0.01:1; T(i) = sin(2*P)+sin(3*P)+..+sin(i*P);`

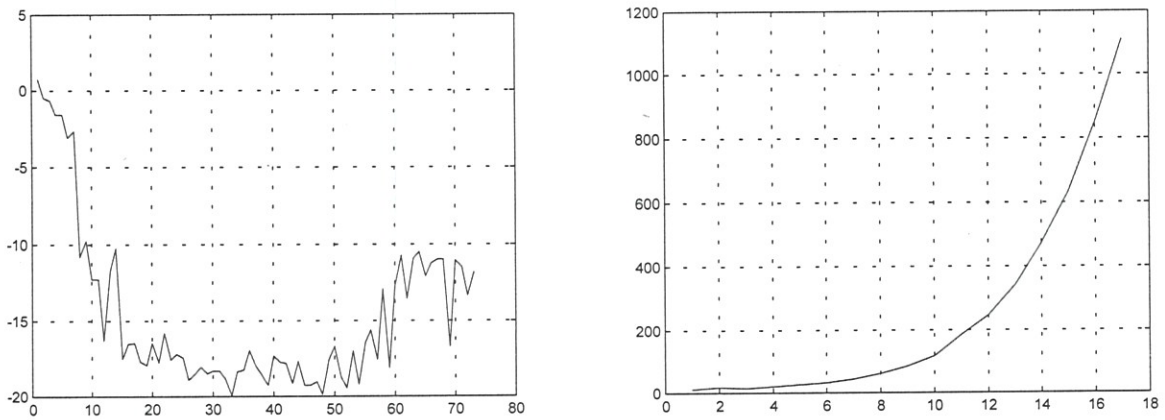
Start poszukiwań od minimalnej liczby neuronów w warstwach wymaga niewielkiego nakładu obliczeń, a sieć może już zachowywać się w sposób satysfakcjonujący. Z kolei nadmierny wzrost liczby neuronów wymaga nie tylko znacznego nakładu obliczeń ale może również prowadzić do pogorszenia jakości pracy sieci.

6. UWAGI KOŃCOWE

W przypadkach gdy nie są znane ścisłe reguły sterowania obiektem zastosowanie sieci neuronowych umożliwia osiągnięcie celu sterowania. Sieć neuronowa pracując jako interpolator reguł heurystycznych lub jako układ realizujący odwrotną charakterystykę obiektu potrafi doprowadzić wyjścia układu do wartości zadanej. Sterownik neuronowy, posiadając względnie prostą strukturę, może zostać zaimplementowany do pracy w czasie rzeczywistym i sterowania obiektów wymagających czasów próbkowania na poziomie pojedynczych milisekund.

Systemy sterowania w czasie rzeczywistym narzucają wymaganie minimalnej liczby obliczeń wykonywanej przez sterownik. Wymaganie to wymusza stosowanie sterowników neuronowych o minimalnej liczbie neuronów, zapewniających zadawalającą jakość interpolacji. Nadmierna liczba neuronów wymaga radykalnego wzrostu mocy obliczeniowej oraz dodatkowo może doprowadzić do pogorszenia jakości pracy sieci.

Podsumowując, wydaje się, że sieci neuronowe mogą praktycznie dobrze pracować jako sterowniki



Rysunek 25. Logarytm z funkcji kryterialnej w funkcji liczby neuronów w drugiej warstwie oraz czas nauczania w funkcji liczby neuronów.

układów dynamicznych. MATLAB/Simulink/RTW/ RT-CON/Neural Network Toolbox są efektywnymi narzędziami do projektowania i praktycznego testowania układów sterowania wykorzystujących sieci neuronowe. Zastosowanie neurosterownika realizującego funkcję dynamiki odwrotnej stanowi pewnego rodzaju uniwersalną metodę syntezy regulatora (praktycznie niezależną od obiektu). Struktura i parametry regulatora nie wymagają wiedzy a-priori na temat sterowanego obiektu. Wiedza ta jest odtwarzana przez, nauczoną dynamiki odwrotnej, sieć neuronową. Jedyną niezbędną informacją do generacji regulatora jest "historia" zachowania się obiektu na pewien zestaw sterowań (np. zachowania obiektu sterowanego w pętli otwartej, czyli bez sprzężenia zwrotnego).

Jednak z pewnością lepszym modelem systemu dynamicznego jest model matematyczny w postaci odpowiedniego równania różniczkowego (np. Kołek i Mitkowski 1998, s. 56), umożliwiający nie tylko syntezę sterowania, ale i głębokie zrozumienie działania układu regulacji. Dobre oprogramowanie (z łatwo dostępnymi przyjaznymi programami użytkowymi) modelu neuronowego powoduje nadal duże zainteresowanie stosowaniem różnych technik sieci neuronowych, zwłaszcza przy bardzo dużej liczbie danych pomiarowych. W systemach sterowania być może pozwoli to upraszczać proces adaptacji, zwłaszcza w układach o niepewnych parametrach.

Przedstawiony tekst z niewielkimi zmianami był referowany w dniu 27.04.2000 w ramach IV Seminarium NeroMet' 2000 - AGH, Kraków, "Zastosowanie Sztucznych Sieci Neuronowych w Symulacji i Sterowaniu Procesami Metalurgicznymi". Praca była wykonywana w ramach kontynuacji działalności statutowej "Stabilizacja systemów dynamicznych", grant AGH nr 11 11 120 230 oraz częściowo grantu KBN Nr 7T08B 062 20.

LITERATURA

Derbel N., 2000, On the optimal control of nonlinear systems by neural networks, *Proceedings of the First Conference Neural networks and Soft Computing, Zakopane,* Poland, June 6-10, 55-462.

Farrell J.A., 1997, *The Control Handbook. Neural Control*, IEEE Press, 1017-1030.

Fujarewicz K., 2000, Identification and suboptimal control of heat exchanger using neural networks, *Proceedings of the First Conference Neural networks and Soft Computing,* Zakopane, Poland, June 6-10, 469-476.

Gorczyca P., Hajduk K., Kołek K., 1995, Zastosowanie pakietu MATLAB do sterowania i zbierania danych on-line dla nieliniowego obiektu wielowymiarowego (MIMO), *I Krajowa Konferencja Uzytkowników MATLAB-a,* Kraków, 14-15 Listopad.

Gorczyca P., Kołek K., Mitkowski W., 1999, Neurosterownik stabilizujący położenie "helikoptera". *Mat. VII Krajowej Konferencji Naukowo-Dydaktycznej Automatyzacja i Eksploatacja Systemów Sterowania,* Akademia Marynarki Wojennej, Gdynia, 13-15.10, 357-366.

Gorczyca P., Kołek K., Mitkowski W., 1999, O modelowaniu dynamiki sieciami neuronowymi. *Mat. Seminarium NEUROMET'99,* Ed. J. Kusiak, "Akapit", Kraków, 7-20.

Grega W., Kołek K., Turnau A., 1998, Rapid Prototyping Environment for Real-Time Control Education, *Proceedings Real-Time Systems Education III, IEEE Computer Society,* Poznań, 21 November, 85-92.

Grymek S., Kiczkowiak T., 2000, Neural models of dynamic plants in optimisation continuation, *Proceedings of the First Conference Neural networks and Soft Computing,* Zakopane, Poland, June 6-10, 407-412.

Kaczorek T., Nowak S., 2000, Application of neural networks to synthesis of state-feedback pole placement regulator, *Proceedings of the First Conference Neural networks and Soft Computing,* Zakopane, Poland, June 6-10, 494-502.

Kempa A., Kołek K., Korytowski A., Tabakowski P., Turnau A., 1997, Neural Time-Optimal Real-Time Crane Controller, *Proceedings of the 16th IASTED International Conference: Modelling, Identification and Control,* Innsbruck, Austria, February 17019, 214-219.

Kołek K., Mitkowski W., 1998, Przykłady zastosowania sieci neuronowych. *Mat. Seminarium NEUROMET'98,* Ed. J. Kusiak, "Akapit", Kraków, 49-57.

Kołek K., Tabakowski P., Turnau A., 1993, Control of a Nonlinear System: Simulation, Real-Time Control, Application of Neural Networks, *Proceedings of the IASTED Conference,* Innsbruck, February 14-17, Austria.

Kołek K., 1996, Systemy operacyjne w sterowaniu systemami dynamicznymi w czasie rzeczywistym. *Praca doktorska,* Katedra Automatyki, Akademia Górniczo-Hutnicza, Kraków.

Korbicz J., Obuchowicz A., Uciński D., 1994, *Sztuczne sieci neuronowe. Podstawy i zastosowania.* Akademicka Oficyna Wydawnicza PLJ, Warszawa.

Lairi M. Bloch G., 1998, Neural Control for MagLev System, *Proceedings of the Mediterranean Conference on Electronics and Automatic Control, MCEA'98,* Marrakech, 17-19 September, 472-475.

Mitkowski W., 1997, Zastosowania sztucznych sieci neuronowych w sterowaniu. *Mat. Seminarium NEUROMET'97,* Ed. M. Pietrzyk, "Akapit", Kraków, 49-59.

Obuchowicz A., 2000, Optymalizacja architektury sieci neuronowych, *Biocybernetyka i Inżynieria Biomedyczna,* Tom 6: Sieci Neuronowe, Akademicka Oficyna Wydawnicza Exit, Warszawa, 323-368.

Żurada J., Barski M., Jędruch W., 1996, *Sztuczne sieci neuronowe,* PWN, Warszawa.