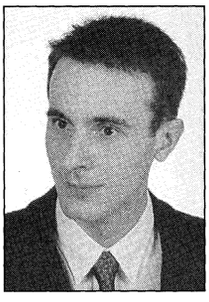
Maciej PETKO
AKADEMIA GÓRNICZO-HUTNICZA, WYDZIAŁ INŻYNIERII MECHANICZNEJ I ROBOTYKI, KATEDRA ROBOTYKI I DYNAMIKI MASZYN

Projektowanie i implementacja inteligentnych czujników w technologii FPGA

dr inż. Maciej Petko



Adiunkt w Katedrze Robotyki i Dynamiki Maszyn Akademii Górniczo-Hutniczej. Ukończył studia na Wydziale Elektrotechniki, Automatyki i Elektroniki w 1991 r. W 1999 r. uzyskał stopień doktora nauk technicznych w dyscyplinie Automatyka i Robotyka, ze specjalnością Mechatronika. Jego zainteresowania skupiają się na mechatronice, robotyce, zagadnieniach prototypowania i implementacji algorytmów przetwarzania sygnałów, głównie w sterowaniu i diagnostyce technicznej.

Streszczenie

Ważną techniką pozwalającą na lokalizację uszkodzeń jest pomiar obciążeń w czasie pracy, jednakże bezpośredni pomiar jest często trudny lub nawet niemożliwy. W artykule omówiono realizację idei, opartego na sieci neuronowej, inteligentnego sensora, estymującego obciążenie w oparciu o odpowiedź struktury: asymetrycznej ramy stalowej i przedniego podwozia samolotu. Opracowana metoda implementacji w układach ASIC/FPGA pozwala na automatyzację najbardziej czasochłonnych i podatnych na błędy zadań. Kod używany do syntezy sprzętu może być również używany w Matlabie/Simulinku, pozwalając na symulację na poziomie systemu. Na zakończenie przedstawiono analizę jakości działania sensora w czasie eksperymentu.

Abstract

An important technique enabling fault localisation is operational load measurement; however direct measurement is often difficult or even impossible. The paper deals with the realisation of the idea of a neural network based „smart sensor", which estimates load based on a response of a structure: an asymmetric steel frame and the front landing gear of an airplane. Developed methodology of implementation in ASIC/FPGA allows for automation of most time-consuming and error prone tasks. The code for hardware synthesis can be also used in Matlab/Simulink, enabling high-level system simulation. Finally, performance of the smart sensor during experiment is analysed.

1 Wstęp

Znajomość wymuszenia (obciążenia) struktury mechanicznej jest istotna dla celów diagnostycznych. Pomaga identyfikować niesprawności maszyn oraz umożliwia monitorowanie zużycia elementów w oparciu o analizę cykli obciążenia [16]. Do celów sterowania niezbędna jest znajomość wymuszenia w czasie rzeczywistym. Jednakże bezpośredni pomiar wymuszenia (siły) jest na ogół skomplikowany i drogi, a często niemożliwy - wymaga ingerencji w strukturę urządzenia. W związku z tym stosuje się metody pośredniej identyfikacji obciążenia na podstawie pomiaru odpowiedzi struktury [3, 14, 4, 17, 6], najczęściej jej drgań. Zadnie to, czasami nazywane problemem identyfikacji odwrotnej [13, 1, 11, 12], może wykorzystywać metody deterministyczne, stochastyczne lub oparte na sztucznej inteligencji.

Dla zastosowań w systemach ciągłego nadzoru diagnostycznego, identyfikacja obciążenia powinna się odbywać w czasie rzeczywistym. Najkorzystniej byłoby wbudować algorytmy identyfikacji w czujniki drgań, osiągając wysoki stopień integracji i hierarchizację systemu monitorującego. Takie czujniki, posiadające możliwość cyfrowego przetwarzania sygnałów i komunikacji z systemem nadrzędnym, nazywane są czujnikami inteligentnymi (ang. smart sensors) [2]. Ponieważ integrują one elementy o różnej naturze fizycznej a sposób działania zależy często od zastosowania, podczas projektowania konieczne jest podejście mechatroniczne.

Poniżej opisano prototyp takiego czujnika oraz projekt i implementację zawartego w nim algorytmu identyfikacji obciążeń. Zadaniem czujnika jest:
- akwizycja sygnałów z lokalnych przetworników drgań (akcelerometrów),
- identyfikacja obciążeń,
- udostępnienie wyestymowanych przebiegów obciążeń systemowi nadrzędnemu.

W opisywanych przykładach, do rozwiązania problemu identyfikacji odwrotnej zastosowano sieć neuronową.

2 Przykłady realizacji inteligentnego czujnika

Zaproponowane podejście [9] do opracowywania inteligentnych czujników zostało zweryfikowane poprzez zastosowanie do estymacji obciążeń dwóch różnych struktur mechanicznych, stanowiska laboratoryjnego do badania drgań i małego samolotu. Stanowisko laboratoryjne składało się z wymuszanej wzbudnikiem elektrodynamicznym, swobodnie wiszącej, asymetrycznej ramy stalowej z przymocowaną w środku stalową płytą. Zastosowana w tym przypadku perceptronowa sieć neuronowa miała opóźnione wejścia i 22 neurony w jednej warstwie ukrytej. Jakość estymacji obciążenia była dobra w przypadku wymuszeń wąskopasmowych znacznie gorsza dla szerokopasmowych (np. opartych na szumie). Szczegółowy opis badań z ramą stalową można znaleźć w [15].

Drugim przykładem był rzeczywisty problem pomiaru obciążeń w przednim podwoziu małego samolotu transportowego M28-05 „SkyTruck". Pomiary siły były dokonywane za pomocą tensometrów naklejonych na ramionach wahacza podwozia, a odpowiedzi struktury za pomocą akcelerometrów, umieszczonych na wrędze, do której przymocowane było podwozie. Wyniki pomiarów zostały użyte do uczenia i testowania sieci neuronowej.

2.1 Akwizycja zbioru uczącego

Do celów uczenia i oceny działania sieci neuronowej zebrano pewną ilość danych eksperymentalnych. Były to zarejestrowane: siła wymuszająca w kilku miejscach podwozia i wywołane nią przyspieszenia drgań w wręgi, razem dwa sygnały drganiowe i cztery sygnały wymuszenia. Zarejestrowano 126 sekund tych sygnałów, bezpośrednio po wylądowaniu samolotu.

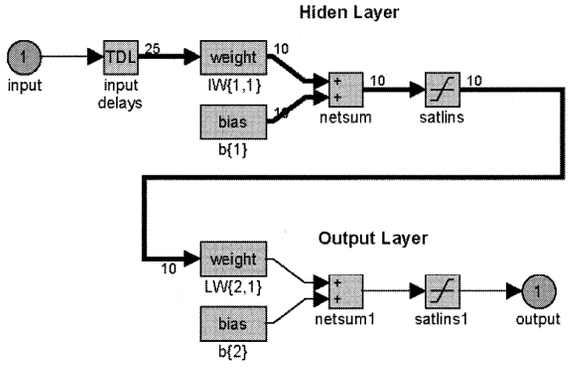
Zarejestrowane sygnały były zaszumione, więc zostały poddane wstępnej obróbce: usunięciu szpilek i filtracji dolnopasmowej filtrem Czebyszewa drugiego rzędu, o częstotliwości granicznej 25 Hz, jako że główne składowe wymuszenia były oczekiwane poniżej tej częstotliwości. Następnie sygnały zostały pozbawione składowej stałej i znormalizowane do pełnego zakresu przetworników analogowo-cyfrowych (±1). Z zarejestrowanych100 000 próbek utworzone zostały trzy podzbiory, każdy zawierający po 6000 punktów czasowych. Pierwszy, zwany dalej zbiorem uczącym, był używany do obliczania gradientów i adaptacji wag sieci neuronowej; drugi, zwany dalej zbiorem kontrolnym (ang. validation set), służył do przerywania procesu uczenia, w celu uniknięcia przeuczenia i dla poprawy zdolności uogólniania przez sieć [5]. Trzeci podzbiór, zwany dalej zbiorem testowym, nie był wykorzystywany w procesie uczenia sieci, ale do celów oceny działania i porównywania różnych ich typów.

2.2 Wybór architektury sieci neuronowej

Mając na względzie ułatwienie procesu implementacji, na architekturę sieci neuronowej narzucono pewne ograniczenia. Wyboru dokonywano pomiędzy sieciami dwuwarstwowymi, z neuronami o symetrycznej, sigmoidalnej funkcji aktywacji w warstwie ukrytej i symetrycznej, liniowej funkcji aktywacji w warstwie wyjściowej. Zadanie wymaga zastosowania sieci dynamicznej i przebadano kilka wariantów: z opóźnionymi wejściami, sprzężeniem od opóźnionych wyjść na wejście oraz inne rodzaje sprzężeń. Sygnały wejściowe zostały wybrane na podstawie analizy korelacji.

Wszystkie sieci były uczone i symulowane w Matlabie z zastosowaniem metody Levenberga-Marquardta [7], z wczesnym zatrzymywaniem procesu uczenia (ang. early stopping) w oparciu o zbiór kontrolny ciągów par wejście - wyjście otrzymanych z eksperymentu. Każda epoka - prezentacja sieci całego zbioru uczącego - składała się z piętnastu niezależnych ciągów czasowych, odpowiadających piętnastu rodzajom wymuszenia.

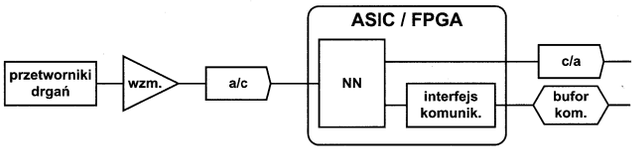
Obiecujące wyniki zostały otrzymane dla sieci neuronowych z opóźnionymi wejściami. W wyniku dalszej selekcji, mającej na celu redukcję ilości sygnałów wejściowych, ilości opóźnień i liczby neuronów w warstwie ukrytej, otrzymano sieć z jednym wejściem, dwudziestoma czterema opóźnieniami, spośród ostatnich stu próbek, i dziesięcioma neuronami w warstwie ukrytej. Następnie, początkowe neurony sigmoidalne były aproksymowane przez symetryczne liniowe z nasyceniem, jako łatwiejsze w implementacji sprzętowej. Analizy, przeprowadzone po krótkim douczeniu sieci, wykazały, że zmiana typu neuronów nie pogorszyła charakterystyki działania sieci oraz, że większość neuronów przekracza w czasie pracy granice liniowości swych funkcji aktywacji, co wskazuje na nieliniowy charakter sieci. Ostatecznie zaakceptowana architektura sieci neuronowej jest przedstawiona na rys. 1.



Rys. 1 Ostateczna architektura sieci neuronowej

3 Warstwa sprzętowa inteligentnego czujnika

Z punktu widzenia sprzętu, inteligentny czujnik zawiera kilka elementów, pokazanych schematycznie na rys. 2. Podstawowe przetworniki przetwarzają wielkości mechaniczne (przyspieszenie drgań) w sygnały elektryczne, wymagające zwykle wzmocnienia i filtracji antyaliasingowej. Są one następnie przekształcane za pomocą przetworników analogowo-cyfrowych w ciąg cyfrowych próbek. Całe cyfrowe przetwarzanie sygnałów odbywa się w jednym specjalizowanym układzie scalonym (ASIC) albo, zwłaszcza w trakcie prototypowania, w jego programowalnym odpowiedniku (FPGA). Wyjście inteligentnego czujnika może być analogowe, poprzez przetwornik cyfrowo-analogowy, albo cyfrowe. W takim przypadku istnieje konieczność zaimplementowania pewnego mechanizmu wymiany danych - interfejsu komunikacyjnego - w tym samym układzie ASIC/FPGA [2].



Rys. 2 Warstwa sprzętowa inteligentnego sensora

Podczas opisywanych badań, podstawowymi przetwornikami były akcelerometry piezoelektryczne, przetwarzanie sygnałów odbywało się w układzie FPGA z rodziny APEX 20KE, a wyjście było typu analogowego.

4 Implementacja w układach ASIC/FPGA

Sieci neuronowe, jak inne algorytmy przetwarzania sygnałów, które mają zostać zaimplementowane w układach FPGA/ASIC są zwykle ciągłe w czasie, wykorzystują ciągłe wartości sygnałów i opisane są za pomocą równań matematycznych, albo za pomocą schematu blokowego [10]. Ponieważ opis taki nie nadaje się do bezpośredniej implementacji, wymaga on przekształceń, modyfikujących nie tylko sposób notacji, ale również sam algorytm. Najważniejszymi i najczęstszymi z nich są dyskretyzacja czasu i amplitudy oraz zastosowanie arytmetyki stałoprzecinkowej. Kolejną znaczącą transformacją jest zmiana sposobu zapisu algorytmu - oprogramowanie używane do syntezy układów FPGA i ASIC akceptuje ich opis w specjalnych językach opisu sprzętu (ang. HDL - Hardware Description Language). Kodowanie w językach HDL jest procesem czasochłonnym i podatnym na błędy. W celu rozwiązania przedstawionych problemów opracowano procedurę implementacji algorytmów przetwarzania sygnałów w układach ASIC/FPGA. Procedura ta, opisana w [8], wymaga jedynie opisu algorytmu w języku C++. W niektórych przypadkach opracowane oprogramowanie pozwala również na automatyczną generację kodu C++ ze schematu Simulinka. Kod C++, będący postacią wyjściową do automatycznej syntezy układu FPGA, może być używany w środowisku Matlab/Simulink, umożliwiając symulację na poziomie systemu, w skład którego wchodzi inteligentny czujnik, oraz weryfikację tej postaci algorytmu.

Implementację przeprowadzono zgodnie z tą procedurą. W pierwszym kroku zmieniono sposób reprezentacji sygnałów i przeprowadzania obliczeń na stałoprzecinkowy. Zastosowanie przybornika Fixed Point Blockset i Simulinka pozwoliło na analizę bardzo istotnej w przypadku sieci neuronowych dynamiki sygnałów oraz dobór i optymalizację typów (zakresu i precyzji) wag sieci i zmiennych występujących w algorytmie.

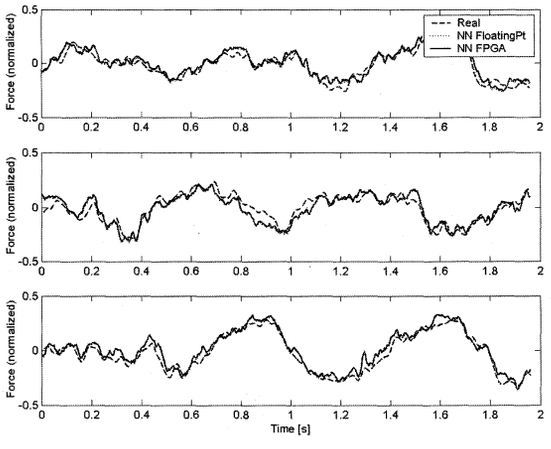
Ze względu na złożoność algorytmu i wynikającą stąd konieczność zastosowania specyficznych technik zmniejszających zużycie zasobów układu scalonego kosztem czasu wykonywania obliczeń, kodowanie w języku C++ z zastosowaniem klas stałoprzecinkowych wykonano ręcznie. Z kodu tego utworzono s-funkcję, co pozwoliło na symulacyjne sprawdzenie w Simulinku poprawności kodowania. Niezbędne interfejsy przetworników analogowo-cyfrowych i cyfrowo-analogowych oraz sposób zarządzania obliczeniami i przepływem danych zostały zaprojektowane, przetestowane i zapisane w C++, w sposób umożliwiający ich łatwe łączenie i kompilację wraz z kodem sieci neuronowej. Następnie ten sam kod, który był wykorzystywany w s-funkcji, wraz opisem interfejsów, został skompilowany do języka VHDL. Otrzymany kod, po kompilacji i alokacji zasobów układu FPGA, pozwolił otrzymać plik binarny służący do zaprogramowania tego układu.

Cała aplikacja (sieć neuronowa, obsługa obliczeń i urządzeń peryferyjnych) zajmuje około 80% elementów logicznych i około 8% pamięci zastosowanego układu EP20K100E, pozostawiając dostatecznie dużo zasobów dla przyszłej realizacji

protokołu i interfejsu komunikacyjnego. Pracując z zegarem 6,25 MHz, system przeprowadza obliczenia sieci neuronowej w czasie 45 μs, co po dodaniu 5 μs potrzebnych na obsługę urządzeń zewnętrznych, daje maksymalną częstotliwość próbkowania (i estymacji obciążeń) 20 kHz. Szybszy zegar (obecnie jest dopuszczalne 20 MHz) pozwoliłby na osiągnięcie częstotliwości próbkowania równej 65 kHz, a nawet większej, po optymalizacji czasowej projektu, która nie była do tej pory przeprowadzana.

4.1 Weryfikacja eksperymentalna

W celu sprawdzenia poprawności implementacji i ocenienia jakości działania inteligentnego sensora został przeprowadzony eksperyment w czasie rzeczywistym. Wyjście czujnika było rejestrowane wraz z mierzoną rzeczywistą siłą w wahaczach podwozia. Trzy fragmenty tych sygnałów przedstawia rys. 3. Dla ilościowego porównania siły rzeczywistej i estymowanej wykorzystane zostały wartości błędu średniokwadratowego (MSE) i współczynnika korelacji jako miary zgodności przebiegów. Tabela 1 przedstawia ich wartości dla przebiegów z rys. 3. Można zaobserwować dobrą zgodność pomiędzy siłą mierzoną bezpośrednio a estymowaną przez inteligentny sensor.



Rys. 3 Porównanie sił w spodniej części lewego wahacza przedniego podwozia samolotu SkyTruck po lądowaniu: rzeczywistej i wyestymowanej w czasie rzeczywistym, przez realizowaną sprzętowo sieć neuronową (NN FPGA); obliczone później wyjście sieci zmiennoprzecinkowej (NN FloatingPt) zostało pokazane dla porównania

Lp.	MSE znormalizowany do zakresu wyjścia	Współczynnik korelacji
1	0,00051	0,943
2	0,00073	0,913
3	0,00062	0,975

Tabela 1 Porównanie rzeczywistej i wyestymowanej siły wymuszającej

5 Podsumowanie

Zaprezentowana w artykule idea inteligentnego sensora daje nowe możliwości zastosowania pomiarów obciążeń podczas pracy w diagnostyce i monitorowaniu stanu maszyn. Zmiany obciążenia wydają się być jednym z najbardziej czułych symptomów uszkodzeń struktury. Opracowana metodologia projektowania, oparta na podejściu mechatronicznym, ułatwia projektowanie inteligentnych czujników dla identyfikacji obciążeń w czasie rzeczywistym w złożonych strukturach mechanicznych.

Zaproponowana metodologia została z powodzeniem zastosowana do problemu opracowania inteligentnego sensora opartego na sieci neuronowej. Pozwala ona na automatyzację najbardziej czasochłonnych, sprzyjających powstawaniu błędów faz implementacji, pozostawiając pełną kontrolę nad każdą operacją. Otrzymane wyniki pokazują możliwość zastosowania zbudowanego czujnika do pomiarów obciążenia w oparciu o mierzone sygnały odpowiedzi struktury.

Literatura

[1] Busby H.R., Trujillo D.M., Solution of an invesrse dynamics problem using an eigenvalue reduction technique, Computer & Structures, vol.25, no.1.

[2] Frank R., Understanding Smart Sensors, Artech House, Norwood, 2000

[3] Giergiel J., Uhl T., Identification of the impact force in mechanical systems, Archives of Machine Design, tom XXXVI, no. 2-3, 1989.

[4] Haas D.J., Milano J., Flitter L., Prediction of Helicopter Component Loads Using Neural Networks, Journal of the American Helicopter Society, no.1, 1995.

[5] Jang J.-S. R., Sun C.-T., Mizutani E., Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence, Prentice Hall, Upper Saddle River, 1997

[6] Lisowski W, Mendrok K, Uhl T., Identidfication of loads basing on output signal measurement, Mat. V Konferencji Naukowej nt. Metody doświadczalne w budowie i eksploatacji maszyn, Wrocław- Szklarska Poreba, 2001.(in Polish)

[7] Norgaard, M., Ravn, O., Poulsen, N. K., Hansen, L. K. Neural Networks for Modelling and Control of Dynamic Systems. Springer-Verlag, London, 2000

[8] Petko M., Implementacja algorytmów sterowania w układach ASIC/FPGA, PAK, nr 1, s.18-21, Warszawa, 2002

[9] Petko M., Implementacja w układach ASIC/FPGA algorytmów wykorzystywanych w sterowaniu i monitorowaniu stanu maszyn, w: T. Uhl [ed.], Projektowanie mechatroniczne: zagadnienia wybrane, Wyd. KRiDM AGH, Krakow, 2002

[10] Petko M., Realizacja produktów mechatronicznych, w: T. Uhl [ed.], Projektowanie mechatroniczne, Wyd. KRiDM AGH, Krakow, 1999

[11] Simonian S.S., Inverse problems in structural dynamics, Int. Journal on Numerical Methods in Engineering, vol. 17, pp. 357-365, 1981.

[12] Trujillo D.M., Application of Dynamic programming to the general inverse problem, International Journal on Numerical Methods in Engineerring, vol.23, pp.613-624, 1987.

[13] Uhl T., Computer assisted identification of mechanical structures, WNT, Warszawa, 1998. (in Polish)

[14] Uhl T., Identification of loading forces in mechanical systems using genetic algorithms, Proc. of AIMECH 01, Gliwice, 2001.

[15] Uhl T., Petko M., Smart Sensor for Operational Load Measurements. In J. of Theoretical and Applied Mechanics. No. 3, vol. 40, pp. 797-815, 2002.

[16] Uhl T., Trends and progress in monitoring and diagnostic systems, PAK, no.4, 1999. (in Polish)

[17] Zion L, Predicting fatigue loads using regression diagnostics, Proceedings of The American Helicopter Society Annual Formu, 1994, Washington D.C.

Title: Design and implementation of smart sensors in FPGA

Artykuł recenzowany